



SAS[®] Visual Data Mining and Machine Learning 8.5: Reference Help

The correct bibliographic citation for this manual is as follows: SAS Institute Inc. 2019. *SAS® Visual Data Mining and Machine Learning 8.5: Reference Help*. Cary, NC: SAS Institute Inc.

SAS® Visual Data Mining and Machine Learning 8.5: Reference Help

Copyright © 2019, SAS Institute Inc., Cary, NC, USA

All Rights Reserved. Produced in the United States of America.

For a hard copy book: No part of this publication may be reproduced, stored in a retrieval system, or transmitted, in any form or by any means, electronic, mechanical, photocopying, or otherwise, without the prior written permission of the publisher, SAS Institute Inc.

For a web download or e-book: Your use of this publication shall be governed by the terms established by the vendor at the time you acquire this publication.

The scanning, uploading, and distribution of this book via the Internet or any other means without the permission of the publisher is illegal and punishable by law. Please purchase only authorized electronic editions and do not participate in or encourage electronic piracy of copyrighted materials. Your support of others' rights is appreciated.

U.S. Government License Rights; Restricted Rights: The Software and its documentation is commercial computer software developed at private expense and is provided with RESTRICTED RIGHTS to the United States Government. Use, duplication, or disclosure of the Software by the United States Government is subject to the license terms of this Agreement pursuant to, as applicable, FAR 12.212, DFAR 227.7202-1(a), DFAR 227.7202-3(a), and DFAR 227.7202-4, and, to the extent required under U.S. federal law, the minimum restricted rights as set out in FAR 52.227-19 (DEC 2007). If FAR 52.227-19 is applicable, this provision serves as notice under clause (c) thereof and no other notice is required to be affixed to the Software or documentation. The Government's rights in Software and documentation shall be only those set forth in this Agreement.

SAS Institute Inc., SAS Campus Drive, Cary, NC 27513-2414

February 2023

SAS® and all other SAS Institute Inc. product or service names are registered trademarks or trademarks of SAS Institute Inc. in the USA and other countries. ® indicates USA registration.

Other brand and product names are trademarks of their respective companies.

8.5-P1:vdmmref

Contents

PART 1 SAS Visual Data Mining and Machine Learning 1

Chapter 1 / Anomaly Detection	5
Overview of Anomaly Detection	5
Anomaly Detection Properties	6
Anomaly Detection Results	8
Chapter 2 / Batch Code	11
Overview of Batch Code	11
Using the Batch Code Node	12
Batch Code Properties	12
Batch Code Results	14
Chapter 3 / Bayesian Network	19
Overview of Bayesian Network	19
Bayesian Network Properties	20
Bayesian Network Results	26
Chapter 4 / Clustering	31
Overview of Clustering	31
Clustering Properties	32
Clustering Results	35
Chapter 5 / Data Exploration	37
Overview of Data Exploration	37
Data Exploration Properties	38
Data Exploration Results	39
Chapter 6 / Decision Tree	41
Overview of Decision Tree	41
Decision Tree Properties	42
Decision Tree Results	50
Chapter 7 / Ensemble	55
Overview of Ensemble	55
Ensemble Properties	56
Ensemble Results	58
Chapter 8 / Feature Extraction	63
Overview of Feature Extraction	63
Feature Extraction Properties	65
Feature Extraction Results	69
Chapter 9 / Feature Machine	71
Overview of Feature Machine	71

Feature Machine Properties	72
Feature Machine Results	74
Chapter 10 / Filtering	75
Overview of Filtering	75
Filtering Properties	76
Filtering Results	77
Chapter 11 / Forest	79
Overview of Forest	79
Forest Properties	80
Forest Results	86
Chapter 12 / GLM	91
Overview of GLM	91
GLM Properties	92
GLM Results	101
Chapter 13 / Gradient Boosting	105
Overview of Gradient Boosting	105
Gradient Boosting Properties	106
Gradient Boosting Results	113
Chapter 14 / Imputation	117
Overview of Imputation	117
Imputation Properties	118
Imputation Results	121
Chapter 15 / Linear Regression	125
Overview of Linear Regression	125
Linear Regression Properties	126
Linear Regression Results	131
Chapter 16 / Logistic Regression	135
Overview of Logistic Regression	135
Logistic Regression Properties	136
Logistic Regression Results	144
Chapter 17 / Manage Variables	149
Overview of Manage Variables	149
Manage Variables Properties	149
Manage Variables Results	150
Chapter 18 / Model Comparison	151
Overview of Model Comparison	151
Model Comparison Properties	152
Model Comparison Results	154
Chapter 19 / Model Composer	157
Overview of Model Composer	157
Model Composer Properties	158
Model Composer Results	163
Chapter 20 / Neural Network	167
Overview of Neural Network	167
Neural Network Properties	168

Neural Network Results	178
Chapter 21 / Open Source Code	183
Overview of Open Source Code	183
Open Source Code Properties	184
Open Source Code Editor User Interface	185
Open Source Code Results	187
Using the Open Source Code Node	188
Installation of Python or R Software	189
Common Questions	189
Chapter 22 / Quantile Regression	193
Overview of Quantile Regression	193
Quantile Regression Properties	194
Quantile Regression Results	199
Chapter 23 / Replacement	203
Overview of Replacement	203
Replacement Properties	203
Replacement Results	205
Chapter 24 / SAS Code	207
Overview of SAS Code	207
SAS Code Properties	208
Code Editor User Interface	210
SAS Code Results	220
Using the SAS Code Node	223
Chapter 25 / Save Data	225
Overview of Save Data	225
Save Data Properties	226
Save Data Results	226
Chapter 26 / Score Code Import	229
Overview of Score Code Import	229
Using the Score Code Import Node	230
Score Code Import Properties	230
Score Code Import Results	232
Chapter 27 / Score Data	237
Overview of Score Data	237
Score Data Properties	238
Score Data Results	239
Chapter 28 / Segment Profile	241
Overview of Segment Profile	241
Segment Profile Properties	242
Segment Profile Results	243
Chapter 29 / SVM	245
Overview of SVM	245
SVM Properties	246
SVM Results	251
Chapter 30 / Text Mining	255
Overview of Text Mining	255

Text Mining Properties	256
Text Mining Results	257
Chapter 31 / Transformations	259
Overview of Transformations	259
Transformations Properties	260
Transformations Results	263
Chapter 32 / Variable Clustering	265
Overview of Variable Clustering	265
Variable Clustering Properties	266
Variable Clustering Results	267
Chapter 33 / Variable Selection	269
Overview of Variable Selection	269
Variable Selection Properties	270
Variable Selection Results	281

PART 2 Risk Modeling

Chapter 34 / Interactive Grouping	285
Overview of Interactive Grouping	285
Interactive Grouping Properties	286
Interactive Grouping Results	290
Using the Interactive Grouping Editor	291
Using a Special Codes Data Set	292
Chapter 35 / Reject Inference	295
Overview of Reject Inference	295
Reject Inference Properties	296
Reject Inference Results	298
Chapter 36 / Scorecard	299
Overview of Scorecard	299
Scorecard Properties	300
Scorecard Results	308
Using the Scorecard Node	312

SAS Visual Data Mining and Machine Learning

Chapter 1		
	Anomaly Detection	5
Chapter 2		
	Batch Code	11
Chapter 3		
	Bayesian Network	19
Chapter 4		
	Clustering	31
Chapter 5		
	Data Exploration	37
Chapter 6		
	Decision Tree	41
Chapter 7		
	Ensemble	55
Chapter 8		
	Feature Extraction	63
Chapter 9		
	Feature Machine	71
Chapter 10		
	Filtering	75
Chapter 11		
	Forest	79
Chapter 12		
	GLM	91
Chapter 13		
	Gradient Boosting	105

Chapter 14	
Imputation	117
Chapter 15	
Linear Regression	125
Chapter 16	
Logistic Regression	135
Chapter 17	
Manage Variables	149
Chapter 18	
Model Comparison	151
Chapter 19	
Model Composer	157
Chapter 20	
Neural Network	167
Chapter 21	
Open Source Code	183
Chapter 22	
Quantile Regression	193
Chapter 23	
Replacement	203
Chapter 24	
SAS Code	207
Chapter 25	
Save Data	225
Chapter 26	
Score Code Import	229
Chapter 27	
Score Data	237
Chapter 28	
Segment Profile	241
Chapter 29	
SVM	245
Chapter 30	
Text Mining	255
Chapter 31	
Transformations	259
Chapter 32	
Variable Clustering	265

Chapter 33

Variable Selection 269

Anomaly Detection

<i>Overview of Anomaly Detection</i>	5
<i>Anomaly Detection Properties</i>	6
General Properties	6
Gaussian Kernel Bandwidth	6
Solver Options	7
Stochastic Subset Options	7
Scored Output Roles	8
<i>Anomaly Detection Results</i>	8

Overview of Anomaly Detection

The **Anomaly Detection** node is a Data Mining Preprocessing node that identifies and excludes anomalies (observations) using the Support Vector Data Description (SVDD). Briefly, the SVDD formulation identifies outliers by determining the smallest possible hypersphere (built using support vectors) that encapsulates the training data points. The SVDD then excludes those data points that lie outside the sphere that is built from the training data. It is a one-class classification technique that is useful in applications where data that belongs to one class is abundant, but data about any other class is scarce or missing. You can use SVDD to model such one-class data and subsequently use the model to perform anomaly detection. Some of the applications include fraud detection, equipment health monitoring, and hyperspectral image analysis.

The Gaussian kernel in the SVDD algorithm provides a more flexible description of training data. That is, the Gaussian kernel can encapsulate non-spherical regions that closely identify the actual geometry of the data. It is highly recommended that you try different values for the Gaussian kernel bandwidth property to determine what is best for the data.

Training can be performed using the **Active set** solver, the **Fast incremental** solver, or the **Stochastic subset** solver.

The number and percentage of anomalies can be viewed in the Anomaly Counts table in the Results. Specify **Include counts and distance histogram reports** for a

histogram that displays the distribution of observations with respect to their distances from the boundary. This report is not generated by default as it requires an additional step of scoring the train data.

The anomaly observations are filtered out in the subsequent node or nodes by default because the **Anomaly indicator role** is set to **Filter** and **Include anomaly observations** is not selected. Select the **Include anomaly observations** property to keep anomalies in the subsequent node or nodes and to filter out normal observations in the subsequent node or nodes. Observations with missing input values are always passed to the subsequent node or nodes since they are ignored during training.

Anomaly Detection Properties

General Properties

- **Standardize interval inputs** — Specifies whether to standardize interval input variables.
- **Include class inputs** — Specifies whether to include class input variables during training.

Gaussian Kernel Bandwidth

- **Bandwidth computation method** — Specifies the method to use to compute the bandwidth value. Here are the possible values:
 - Mean**
 - Modified mean**
 - Trace**
 - User Specify**

The default value is **User Specify**. If **Include class inputs** is selected, then **User Specify** is the only option available.
- **Bandwidth value** — Specifies the bandwidth parameter value that is used. If a value is not specified, one is computed based on empirical evidence. It is highly recommended that you try different values to determine the best bandwidth value. This property is available when **User Specify** is selected for the **Bandwidth computation method** property.
- **Delta** — Specifies the delta (tolerance) value that is used if **Mean** is selected for the **Bandwidth computation method** property. The default value is 0.000001.
- **Number of representative points** — Specifies the number of representative points that is used if **Trace** is selected for the **Bandwidth computation method** property. The default value is 5.

If you do not specify any settings for the **Gaussian Kernel Bandwidth**, the following defaults are used to calculate the bandwidth value:

- If only interval input variables are used and the number of input variables is less than or equal to ten, then the **Mean** method is used.
- If only interval input variables are used and the number of input variables is greater than or equal to ten, then the **Trace** method with the number of representative points set to 5 is used.
- If both interval input variables and class input variables are used, then the bandwidth value is the square root of the total number of inputs.

Solver Options

- **Solver** — Specifies the type of optimization solver that is used for SVDD training. Possible values are **Active set**, **Fast incremental**, and **Stochastic subset**. The default value is **Active set**.
- **Anomaly fraction** — Specifies the expected fraction of the training data that consists of anomalies. The default value is 0.01. If you specify **Stochastic subset** or **Fast incremental** as the solver, this value is not used.
- **Solver tolerance** — Specifies the tolerance value for the solver. The default value is 0.001. If you specify **Fast incremental** as the solver, this value is not used.
- **Maximum number of iterations** — Specifies the maximum number of iterations for the solver. If you specify **Fast incremental** as the solver, this value is not used.
- **Maximum time (minutes)** — Specifies the maximum amount of time (in minutes) that the solver runs before stopping. The default value is 30 minutes.
- **Maximum support vectors** — Specifies the maximum number of support vectors for the solver. The default value is 350. If you specify **Active set** as the solver, this value is not used.
- **Include counts and distance histogram reports** — Specifies whether to generate a counts table, which displays the number of observations marked as anomalies during training. The contents table is not generated because that would require an additional step of scoring the train data.

Stochastic Subset Options

These configurations are available only when you select **Stochastic subset** as the solver.

- **Sample size** — Specifies the number of observations that are sampled in each iteration of the stochastic solver. You must specify a value greater than 0. The default value is 5.
- **Threshold tolerance** — Specifies the tolerance that is used to detect convergence of the threshold value. The default value is 0.01.


- **Center tolerance** — Specifies the tolerance that is used to detect convergence of the center. The default value is 0.01.
- **Convergence criterion** — Specifies a convergence criterion for the stochastic solver. If the radius and center values converge for this number of consecutive iterations, then convergence is declared. The default value is 3.
- **Random seed** — Specifies the seed value that is used for selecting a random sample. The default value is 12345.

Scored Output Roles

- **Anomaly indicator role** — Specifies how the anomaly indicator (the variable that indicates whether an observation is an anomaly) is used. Possible values are **Input** and **Filter**. Select **Input** to use the indicator as an input in the subsequent node. Select **Filter** to apply the indicator to the scoring data to exclude these rows. The default value is **Filter**.
- **Include anomaly observations** — Specifies whether to include anomaly observations in subsequent nodes when **Filter** is selected for the **Anomaly indicator role** property. If the **Include anomaly observations** property is selected, then anomaly observations are included in subsequent nodes and normal observations are filtered out in subsequent nodes.
- **Anomaly distance role** — Specifies the role that is assigned to the SVDD distance variable. Possible values are **Rejected** and **Input**. Select **Input** to use the distance variable as an input in the subsequent node. Select **Rejected** to set the variable to Rejected.

Anomaly Detection Results

After running the node, open the Results window by right-clicking the node and selecting **Results**.

You can download the data that is used to build the charts and tables in the Results window by clicking . A CSV file with the chart or table data will immediately begin downloading.

In the **Output Data** tab, you can view, save, and explore the data that is generated by the node. For more information, see [Explore and Visualize](#).

Summary

- **Anomaly Counts** — Displays the anomaly counts after scoring the train data. The anomaly counts include number and percentage of anomalies, observations used in training, observations with missing inputs, and total observations. If you do not select **Include counts and distance histogram reports**, the table is not displayed.
- **SVDD Distance Histogram** — Displays a bar chart that shows the distribution of observations using their respective SVDD distance (after scoring the train data). The distance is divided into 20 bins of equal size. Bins are color-coded to

indicate whether the bars are less than or greater than the threshold that marks the boundary when identifying anomalies. If you do not select **Include counts and distance histogram reports**, this chart is not selected.

- **Training Results** — Displays the results of the training procedure, which include the following:
 - Number of support vectors
 - Number of support vectors on boundary
 - Number of outliers
 - Number of dropped observations
 - Threshold R^2 value
 - Constant value
 - Run time
 - Bandwidth calculation time
- **Optimization Summary** — Displays a summary of the anomaly detection run itself, including the number of iterations, objective value, infeasibility, optimization status, and the degenerate indicator variable.
- **Properties** — Specifies the various properties that you selected before running the node.
- **Output** — Displays the SAS output of the anomaly detection run.

Batch Code

<i>Overview of Batch Code</i>	11
<i>Using the Batch Code Node</i>	12
<i>Batch Code Properties</i>	12
Post Training Properties	13
<i>Batch Code Results</i>	14

Overview of Batch Code


The **Batch Code** node enables you to import batch code models that were created in SAS Enterprise Miner. This node does not perform batch retraining of models that were created in Model Studio.

When the **Batch Code** node runs, a sample of the input data is provided to the SAS client and the batch code that represents the SAS Enterprise Miner process flow diagram is run on that sample. After the batch code runs, Model Studio retrieves either the model score code or the analytic store and EP score code. The retrieved code is used to score the original input data in Model Studio. This scored data is used to produce the assessment results. The batch code is dynamic and re-creates the score code based on your data or a sample of your data.

Note: Any references to CAS libraries, URIs, or analytic stores are not saved or set if you save the node or pipeline to the exchange, duplicate or modify the node in the exchange, or import a project with a **Batch Code** node. This avoids potential issues that can arise when running the node outside the initial environment. For example, the CAS library might not exist in the new environment or a different user who runs the node might not have access to the CAS library or table.

Using the Batch Code Node

To use the **Batch Code** node, perform the following steps:

- 1 Build a process flow diagram in SAS Enterprise Miner. Your process flow diagram must end in a **Score** node.
- 2 Right-click the **Score** node in your diagram and select **Export Path as SAS Program**. Ensure that **Run this path** is enabled in the Export Path as SAS Program window. Specify a location to save the batch code and click **OK**.
- 3 In a Model Studio pipeline, add a **Batch Code** node to your pipeline.
- 4 In the **Batch Code** node properties pane, click **Open Code Editor** and navigate to the file that you saved earlier. The batch code that you created in SAS Enterprise Miner is opened in the code editor.
- 5 Click  to save this code. Then click **Close**.
- 6 Continue building your pipeline. When you run your pipeline, the **Batch Code** node runs alongside all other nodes.

Batch Code Properties

- **Open Code editor** — invokes the Batch Code Editor. For more information about the SAS Code Editor see [Code Editor User Interface on page 210](#).
- **Data Sample** — Specifies the data sampling strategy.
 - **Sampling method** — Specifies whether to perform **Simple random** sampling, **Stratified** sampling, or no data sampling.
 - **Sample using** — Specifies whether you want to include a fixed number of observations or a percentage of the total observations in the sample.
 - **Number of Observations** — Specifies the fixed number of observations to create the data sample. The default value is 10,000.
 - **Percentage of Observations** — Specifies the percentage of total observations to create the data sample. The default value is 10%.

Note: For class targets, all levels of the target variable must be in the data sample. Otherwise, the node run fails. You can specify **Stratified** sampling for class target variables to ensure that all levels of the target variable are in the sample.

- **Use the exact percentile method for lift calculations** — Specifies whether to use the exact percentile method for calculating lift and related assessment measures. When this property is deselected, the iterative method is used. The

exact method should be used if there are convergence errors when using the iterative method.

Post Training Properties

Changing the following properties will not retrain the model.

Model Interpretability — Specifies the method of model interpretation in order to show how the model responds to changing inputs.

Note: If you are importing a model from SAS Enterprise Miner that uses Date variables as input variables, you must change the variable role to **ID** in order to generate model interpretability reports.

■ Global Interpretability

- **Variable importance** — Specifies whether to display a table of the relative importance values for the input variables. The surrogate variable importance values are calculated using a one-level decision tree for each input to predict the predicted value as a global surrogate model. For tree-based supervised learning models, the model variable importance table is displayed as well.
- **PD plots** — Specifies whether to display partial dependency (PD) plots that show the relationship between model input variables and their predictors. PD plots are generated for only up to the top ten variables in the relative importance table. Here is how to determine the variables that are displayed in the PD plot: for tree-based models (forest, decision tree, and gradient boosting), the model variable importance table is used. For other supervised learning models, the surrogate variable importance table is used.

■ Local Interpretability


- **ICE plots** — Specifies whether to display Individual Conditional Expectation (ICE) plots. ICE plots can reveal interesting subgroups and interactions between model variables. ICE plots are generated for only up to the top ten variables in the relative importance table. Here is how to determine the variables that are displayed in the ICE plot: for tree-based models (forest, decision tree, and gradient boosting), the model variable importance table is used. For other supervised learning models, the surrogate variable importance table is used.
- **LIME** — Specifies whether to use the Locally Interpretable Model-Agnostic Explanation (LIME) method to explain the model predictions.
- **Kernel SHAP** — Specifies whether to use the Kernel Shapley additive explanations (Kernel SHAP) method to explain the model predictions.
- **Maximum number of Kernel SHAP variables** — Specifies the maximum number of Kernel SHAP variables to plot. Possible values range from 1 to 100. The default value is 20.
- **Specify instances to explain** — Specifies the individual observations that you want to explain. The **Random** observations option provides explanations for five randomly selected observations. The **Specify up to 5** observations option enables you to provide IDs for the specific observations in the data. The IDs should be unique values of the variable that has the role **Key** in the

metadata. If no variable is assigned the role **Key**, then you can use the variable `_dmIndex_` to specify the individual observations.

- **LIME/Kernel SHAP Tables**
 - **Explainer Information table** — Specifies whether the Explainer Information tables for the LIME and Kernel SHAP methods should be included in the node results. The Explainer Information table contains information about what settings were used to perform the LIME and Kernel SHAP methods.
 - **Explainer Fidelity table** — Specifies whether the Explainer Fidelity tables for the LIME and Kernel SHAP methods should be included in the node results. The Explainer Fidelity table contains information about the accuracy of the local explanation predictions, compared to the black-box machine learning model.
- **PD/ICE Options**
 - **Maximum number of variables** — Specifies the maximum number of input variables to use to generate the PD and ICE plots. The input variables are selected based on their relative importance. Possible values range from 1 to 10. The default value is 5.
 - **Number of observations** — Specifies the maximum number of observations to sample for the PD and ICE plots. The default value is 1000.
 - **Number of tick points** — Specifies the number of tick points for an interval analysis variable to generate for the PD and ICE plots. Possible values range from 3 to 100. The default value is 50.
 - **Truncate lower tail** — Specifies under what percentage to discard the values of the interval analysis variables for the PD and ICE plots. Possible values range from 0 to 99. The default value is 5%.
 - **Truncate upper tail** — Specifies above what percentage to discard the values of the interval analysis variables for the PD and ICE plots. Possible values range from 0 to 99. The default value is 95%.
- **Seed** — Specifies the seed for generating random numbers. The seed is used for selecting individual observations to explain, selecting observations to generate PD and ICE plots, and simulating data for LIME and Kernel SHAP.

Batch Code Results

After running the node, open the Results window by right-clicking the node and selecting **Results**.

You can download the data that is used to build the charts and tables in the Results window by clicking . A CSV file with the chart or table data will immediately begin downloading.

In the **Output Data** tab, you can view, save, and explore the data that is generated by the node. For more information, see [Explore and Visualize](#).

Several charts have detailed descriptions available that explain how the charts are generated. The descriptions also provide information about how to interpret the

charts that is custom to your particular data and models. To access these detailed descriptions:

- Expand an individual chart by clicking ↕. The detailed description is in the right panel.
- Select ⓘ on an individual chart. A window appears with the detailed description.

Note: The detailed descriptions are available only in the English language.

Node

- **Path Score Code** — Provides the SAS code from SAS Enterprise Miner when the model creates DATA step score code.
- **Path EP Score Code** — Provides the SAS code from SAS Enterprise Miner when the model creates an analytic store for scoring.
- **DS2 Package Code** — Displays the SAS DS2 package code that was created by the node when the model creates an analytic store for scoring. DS2 package code is used when publishing models to SAS Micro Analytic Service.
- **Properties** — Specifies the various properties that you selected before running the node.
- **Score Inputs** — Displays a table of data on the input variables for scoring calculations. The table includes the variables' name, role, level, type, label, format, and length.
- **Score Outputs** — Displays a table of data on the predicted response variable for scoring calculations. The table includes the variables' name, role, type, format and length, as well as the creator, creator GUID, and function.
- **EM Batch Code** — Provides the batch code that was created by SAS Enterprise Miner.

Assessment

- **Predicted Reports** — Displays the predicted and target mean as a function of the depth for the model. The predicted and target mean are given for each of the data roles. To examine the predicted mean as a function of the target mean, use the drop-down menu in the upper right corner. This result is displayed only if the target is an interval variable.
- **Lift Reports** — Displays the cumulative lift as a function of the depth for the model. The cumulative lift is given for each of the data roles. To examine other statistics as a function of depth, use the drop-down menu in the upper right corner. Other statistics include lift, gain, captured response percentage, cumulative captured response percentage, response percentage, and cumulative response percentage. This result is displayed only if the target is a class variable.
- **ROC Reports** — Displays the ROC (receiver operating characteristic) chart for a model, giving the sensitivity as a function of 1-specificity. The sensitivity is given for each of the data roles. To examine other statistics, use the drop-down menu in the upper right corner. Other statistics include accuracy and F1 score. This result is displayed only if the target is a class variable.
- **Fit Statistics** — Displays a table of the fit statistics for the model, broken down by data role.
- **Event Classification** — Displays the confusion matrix at various cutoff values for each partition. The confusion matrix contains true positives (events that are

correctly classified as events), false positives (non-events that are classified as events), false negatives (events that are classified as non-events), and true negatives (non-events that are correctly classified as non-events). The classification cutoffs that are used are the default value of 0.5, the Kolmogorov-Smirnov cutoff value for each partition, and the **Node binary classification cutoff** that is specified for binary targets. Use the drop-down menu to view the information in the chart as percentages or counts. You can also view the information summarized in a table. This result is displayed only if the target is a class variable.

- **Nominal Classification** — Displays either the percentage of observations or the number of observations that predict each level of the target variable. The plot is segmented by target level and partition level. The target level with the greatest predicted probability is the predicted outcome. This result is displayed only if the target is a nominal variable.

Model Interpretability

- **Surrogate Model Variable Importance** — Displays a table of the relative importance of the input variables, including variable level and label. Relative importance is calculated using a one-level decision tree for each input variable to predict the predicted value as a global surrogate model.
- **PD Plot** — Displays the functional relationship between the input variable and the model prediction. For interval input variables, the 95% confidence interval for the average target prediction is indicated by the shaded band around the line. The X axis includes a heat map that shows the distribution of the input variable. The extreme values of the input variables are removed by truncating the lower and upper tails of its distribution. The amount of truncation can be controlled by specifying values for the **Truncate lower tail** and the **Truncate upper tail** properties.
- **PD and ICE Overlay Plot** — Displays the functional relationship between the input variable and the model prediction as well as the functional relationship between individual observations and the model prediction. For each individual observation, the corresponding ICE curve displays values of the input variable on the X axis and the corresponding predicted probability or prediction of the target variable on the Y axis, holding the other input variable values constant at that observation. The X axis includes a heat map that shows the distribution of the input variable. The extreme values of the input variables are removed by truncating the lower and upper tails of its distribution. The amount of truncation can be controlled by specifying values for the **Truncate lower tail** and the **Truncate upper tail** properties.
- **LIME Explanations** — The LIME algorithm calculates a more easily interpretable linear model around an individual observation. The chart displays the regression estimates for the input variables that are selected in the local surrogate linear regression model. The input variables are ordered by significance such that the most important variable for the local regression model is at the bottom of the chart. The LASSO technique is used to select the most significant predictors from the set of input variables that are used to train the model. Each nominal input variable is binary-encoded based on whether it matches the level of the individual observation. A positive estimate indicates that the observed value of the input variable increases the predicted probability of the event.
- **Kernel SHAP Values** — For each individual observation, an input variable's Kernel SHAP value is the contribution of the observed value of the input variable to the predicted probability or prediction of the target variable. The Kernel SHAP values are the regression coefficients that are obtained by fitting a weighted least

squares regression. The Kernel SHAP values of all input variables sum to one. The input variables are ordered by significance such that the most important variable according to the absolute Kernel SHAP values is at the top of the chart. Each nominal input is binary-encoded based on whether it matches the individual observation. Interval inputs are binary encoded based on their proximity to the individual observation.

- **LIME Information Table** — Displays information about the settings used to perform the LIME method.
- **Kernel SHAP Information Table** — Displays information about the settings used to perform the Kernel SHAP method.
- **LIME Fidelity Table** — Displays information about the accuracy of the local explanation predictions that are calculated using the LIME method, compared to the black-box machine learning model. The model prediction, the explainer prediction, and the explainer root mean square error (RMSE) are displayed.
- **Kernel SHAP Fidelity Table** — Displays information about the accuracy of the local explanation predictions that are calculated using the Kernel SHAP method, compared to the black-box machine learning model. The model prediction, the explainer prediction, and the explainer root mean square error (RMSE) are displayed.

Bayesian Network

<i>Overview of Bayesian Network</i>	19
<i>Bayesian Network Properties</i>	20
General Properties	20
Variable Selection Options	20
Network Structure Options	21
Perform Autotuning	21
Lift Calculation Property	24
Binary Classification Cutoff	24
Post Training Properties	24
<i>Bayesian Network Results</i>	26

Overview of Bayesian Network

The **Bayesian Network** node is a Supervised Learning node that fits a Bayesian network model for a nominal target. A Bayesian network is a directed, acyclic graphical model in which the nodes represent random variables, and the links between the nodes represent conditional dependency between two random variables. The structure of a Bayesian network is based on the conditional dependency between the variables. As a result, a conditional probability table is created for each node in the network. Because of these features, Bayesian networks can be effective predictive models during supervised data mining. The following Bayesian network structures are available:

- Naive — A naive Bayesian network connects the target variable to each input variable. There are no other connections between the variables because the input variables are assumed to be conditionally independent of each other.
- Tree-Augmented — A Tree-Augmented Bayesian network connects the target variable to each input variable and connects the input variables in a tree structure. The tree that connects the input variables is based on the maximum spanning tree algorithm. In a Tree-Augmented network, each node can have at most two parents, one of which must be the target variable.

- **Parent-Child (PC)** — A PC Bayesian network connects the target variable to each input variable. However, input variables can be either a parent of the target variable or a child of the target variable. The Bayesian Information Criterion is used to determine whether an input variable is a parent of the target variable or a child of the target variable.
- **Markov Blanket** — A Markov Blanket Bayesian network creates a set of connections between the target variable and the input variables. But it also permits connections between certain input variables. Only the children of the target variable can have an additional parent node. All other variables are conditionally independent of the target variable and thus do not affect the classification model. The Bayesian Information Criterion is used to determine whether an input variable is a parent of the target variable or a child of the target variable.

To choose the best structure among multiple structures, then select multiple values for the **Network structure** property.

Bayesian Network Properties

General Properties

- **Missing class inputs** — Specifies how to handle missing values for class inputs. Possible values are **Exclude**, **Impute with mode**, and **Impute with level**. Choose **Exclude** to ignore the observations that have missing values in any of the class input variables. Choose **Impute with mode** to replace the missing values in any class input variable by the mode of the variable. Finally, choose **Impute with level** to treat the missing values in any class input variable as a separate level of the variable. The default value for this option is **Exclude**.
- **Missing interval inputs** — Specifies how to handle missing values for interval inputs. Possible values are **Exclude** and **Impute with mean**. Choose **Exclude** to ignore the observations that have missing values in any of the interval variables. Choose **Impute with mean** to replace the missing values in any interval variable by the mean of the variable. The default value for this option is **Exclude**.
- **Number of bins** — Specifies the number of bins for interval variables. Possible values range from 2 to 100. The default value is 10.

Variable Selection Options

- **Prescreen variables** — Specifies whether to prescreen variables using independence tests between the target and each input variable. By default, this option is selected.

- **Variable selection** — Specifies whether to select variables using conditional independence tests between the target and each input variable given the network. By default, this option is deselected.
- **Independence test statistics** — Specifies the choice of statistics used for the independence test. Possible values are **Chi-Square**, **G-Square**, and **Chi-Square and G-Square**. Select **Chi-Square and G-Square** to use both independence test statistics. Both statistics must be satisfied for the independence test. The default value is **G-Square**.
- **Significance level** — Specifies the significance level (p -value) for independence testing using Chi-Square and G-Square. The smaller the value you specify, the fewer that variables that are selected. The default value is 0.2.

Network Structure Options

- **Network structure** — Specifies the network structure. This option enables you to select multiple structures. The following options are available:
 - Naive**
 - Tree-Augmented**
 - Parent-Child**
 - Markov Blanket**

To choose the best structure among several structures, select multiple values in any combination. For more information about these networks, see the [Overview on page 19](#). By default, the **Naive**, **Tree-Augmented**, and **Parent-Child** options are selected.

- **Maximum parents** — Specifies the maximum number of parents that is allowed for each node in the network structure. Possible values range from 1 to 16. The default value is 5.
- **Choose best number of parents** — Specifies whether to choose the best number of parents by trying from 1 to the value of the **Maximum parents** property above. By default, this option is selected.
- **Parenting method** — Specifies the parenting method at each node. Possible values are **One parent** and **Set of parents**. Select **One parent** to add the best possible candidate as a parent of the node. Select **Set of parents** to test a set of variables among possible candidates to be the parents of each node. Then, the best set of variables are added as the parents of the node. The default value is **Set of parents**.

Perform Autotuning

This feature specifies whether to perform autotuning of any Bayesian network parameters. Warning: Performing autotuning can substantially increase run time. If **Perform Autotuning** is selected, the following options are available:

- **Network Structure** — Specifies whether to autotune Network Structure. If this option is selected, the following values are available:

- Naive**
- Tree-Augmented**
- Parent-Child**
- Markov blanket**
- **Maximum Parents** — Specifies whether to autotune Maximum Parents. If this option is selected, then the following option is available:
 - Initial value** — Specifies the initial value for autotuning Maximum Parents. The default value is 5. Use the **From** and **To** options to specify the range. The default **From** value is 1, and the default **To** value is 5.
- **Parenting Method** — Specifies whether to autotune Parenting Method. If this option is selected, the following values are available:
 - One parent**
 - Set of parents**
- **Number Of Bins** — Specifies whether to autotune Number Of Bins. If this option is selected, then the following option is available:
 - Initial value** — Specifies the initial value for autotuning Number Of Bins. The default value is 10. Use the **From** and **To** options to specify the range. The default **From** value is 2, and the default **To** value is 20.
- **Search Options** — Specifies the options for autotuning searching. The following options are available:
 - Search method** — Specifies the tuning search method. Here are the possible values:
 - **Bayesian** — Specifies the Bayesian method. This method uses priors to seed the iterative optimization.
 - **Genetic algorithm** — Specifies the genetic algorithm method. This method uses an initial Latin Hypercube sample that seeds a genetic algorithm to generate a new population of alternative configurations at each iteration.
 - **Grid** — Specifies the grid method. This method performs an exhaustive search of all configurations. Each hyperparameter of interest is discretized into a desired set of values to be studied. Models are trained and assessed for all combinations of values across all the hyperparameters (thus forming a multi-dimensional “grid”).
 - **Latin hypercube sample** — Specifies the Latin Hypercube method. This method performs an optimized grid search that is uniform in each tuning parameter, but random in combinations.
 - **Random** — Specifies the Random method. This method generates a single sample of purely random configurations.

The default value is **Genetic algorithm**.
 - Number of evaluations per iteration** — Specifies the number of tuning evaluations in one iteration. This option is available only if the **Search method** is **Genetic algorithm** or **Bayesian**. The default value is 10.
 - Maximum number of evaluations** — Specifies the maximum number of tuning evaluations. This option is available only if the **Search method** is **Genetic algorithm** or **Bayesian**. The default value is 50.

- **Maximum number of iterations** — Specifies the maximum number of tuning iterations. This option is available only if the **Search method** is **Genetic algorithm** or **Bayesian**. The default value is 5.
- **Maximum number of points in model** — Specifies the maximum number of points in the model. This option is available only if the **Search method** is **Bayesian**. The default value is 100.
- **Sample size** — Specifies the sample size. This option is available only if the **Search method** is **Random** or **Latin hypercube sample**. The default value is 50.
- **General Options** — Specifies the general properties for autotuning. The following options are available:
 - **Validation method** — Specifies how to partition the data for assessing the models. Note that if your data is partitioned, then that partition is used and **Validation method**, **Validation data proportion**, and **Cross validation number of folds** are all ignored. Here are the possible values:
 - **Partition** — Specifies using a single partition of a training set. With partition, you specify proportions to use for randomly assigning observations to each role.
 - **K-fold cross validation** — Specifies using the k -fold cross validation method. In k -fold cross validation, each model evaluation requires k training executions (on $k-1$ data folds) and k scoring executions (on one holdout fold). This increases the evaluation time by approximately a factor of k .

For small to medium data tables, cross validation provides, on average, a better representation of error across the whole data table. **Partition** is the default value.
 - **Training data proportion** — Specifies the proportion of training data to be used for the **Partition** validation method. The default value is 0.7.
 - **Validation data proportion** — Specifies the proportion of data to be used for the **Partition** validation method. The default value is 0.3.
 - **Cross validation number of folds** — Specifies the number of partition folds in the cross validation process (the k defined above). Possible values range from 2 to 20. The default value is 5.
 - **Class target objective function** — Specifies the objective function to optimize for tuning parameters for a class target. Here are the possible values:
 - **Area under the curve**
 - **Average squared error**
 - **F0.5 score**
 - **F1 score**
 - **Gamma**
 - **Gini coefficient**
 - **Kolmogorov-Smirnov statistic**
 - **Misclassification rate**
 - **Multi-class log loss**
 - **Root average squared error**

- **Tau**

The default value is **Kolmogorov-Smirnov statistic**.

- **Maximum time (minutes)** — Specifies the maximum time (in minutes) for the optimization tuner. The default value is 60.
- **Maximum training time for single model (minutes)** — Specifies the maximum training time (in minutes) for the single model.

Lift Calculation Property

- **Use the exact percentile method for lift calculations** — Specifies whether to use the exact percentile method for calculating lift and related assessment measures. When this property is deselected, the iterative method is used. The exact method should be used if there are convergence errors when using the iterative method.

Binary Classification Cutoff

- **Specify node binary classification cutoff** — Specifies whether to use the binary classification cutoff specified below for the node. If this property is deselected, the project binary classification cutoff is used for determining the predicted value for a binary target based on the posterior probabilities. By default, this property is deselected.
- **Node binary classification cutoff** — Specifies the cutoff to use in the node for determining the predicted value for a binary target based on the posterior probabilities. This option is available only if you select **Specify node binary classification cutoff**. The default value is 0.5.

.....

Note: The **Rules** settings in the Project Settings window enable you to specify a new cutoff value by selecting **Override the default classification cutoff**. If **Specify node binary classification cutoff** is deselected, then the value that you specified in the Project Settings window is used. If **Specify node binary classification cutoff** is selected, then the value that you specified in **Node binary classification cutoff** is used.

.....

Post Training Properties

Changing the following properties will not retrain the model.

Model Interpretability — Specifies the method of model interpretation in order to show how the model responds to changing inputs.

- **Global Interpretability**


- **Variable importance** — Specifies whether to display a table of the relative importance values for the input variables. The surrogate variable importance values are calculated using a one-level decision tree for each input to predict the predicted value as a global surrogate model. For tree-based supervised learning models, the model variable importance table is displayed as well.
- **PD plots** — Specifies whether to display partial dependency (PD) plots that show the relationship between model input variables and their predictors. PD plots are generated for only up to the top ten variables in the relative importance table. Here is how to determine the variables that are displayed in the PD plot: for tree-based models (forest, decision tree, and gradient boosting), the model variable importance table is used. For other supervised learning models, the surrogate variable importance table is used.
- **Local Interpretability**
 - **ICE plots** — Specifies whether to display Individual Conditional Expectation (ICE) plots. ICE plots can reveal interesting subgroups and interactions between model variables. ICE plots are generated for only up to the top ten variables in the relative importance table. Here is how to determine the variables that are displayed in the ICE plot: for tree-based models (forest, decision tree, and gradient boosting), the model variable importance table is used. For other supervised learning models, the surrogate variable importance table is used.
 - **LIME** — Specifies whether to use the Locally Interpretable Model-Agnostic Explanation (LIME) method to explain the model predictions.
 - **Kernel SHAP** — Specifies whether to use the Kernel Shapley additive explanations (Kernel SHAP) method to explain the model predictions.
 - **Maximum number of Kernel SHAP variables** — Specifies the maximum number of Kernel SHAP variables to plot. Possible values range from 1 to 100. The default value is 20.
 - **Specify instances to explain** — Specifies the individual observations that you want to explain. The **Random** observations option provides explanations for five randomly selected observations. The **Specify up to 5** observations option enables you to provide IDs for the specific observations in the data. The IDs should be unique values of the variable that has the role **Key** in the metadata. If no variable is assigned the role **Key**, then you can use the variable `_dmIndex_` to specify the individual observations.
 - **LIME/Kernel SHAP Tables**
 - **Explainer Information table** — Specifies whether the Explainer Information tables for the LIME and Kernel SHAP methods should be included in the node results. The Explainer Information table contains information about what settings were used to perform the LIME and Kernel SHAP methods.
 - **Explainer Fidelity table** — Specifies whether the Explainer Fidelity tables for the LIME and Kernel SHAP methods should be included in the node results. The Explainer Fidelity table contains information about the accuracy of the local explanation predictions, compared to the black-box machine learning model.
- **PD/ICE Options**
 - **Maximum number of variables** — Specifies the maximum number of input variables to use to generate the PD and ICE plots. The input variables are

selected based on their relative importance. Possible values range from 1 to 10. The default value is 5.

- **Number of observations** — Specifies the maximum number of observations to sample for the PD and ICE plots. The default value is 1000.
- **Number of tick points** — Specifies the number of tick points for an interval analysis variable to generate for the PD and ICE plots. Possible values range from 3 to 100. The default value is 50.
- **Truncate lower tail** — Specifies under what percentage to discard the values of the interval analysis variables for the PD and ICE plots. Possible values range from 0 to 99. The default value is 5%.
- **Truncate upper tail** — Specifies above what percentage to discard the values of the interval analysis variables for the PD and ICE plots. Possible values range from 0 to 99. The default value is 95%.
- **Seed** — Specifies the seed for generating random numbers. The seed is used for selecting individual observations to explain, selecting observations to generate PD and ICE plots, and simulating data for LIME and Kernel SHAP.



Bayesian Network Results

After running the node, open the Results window by right-clicking the node and selecting **Results** from the pop-up menu.

You can download the data that is used to build the charts and tables in the Results window by clicking . A CSV file with the chart or table data will immediately begin downloading.

In the **Output Data** tab, you can view, save, and explore the data that is generated by the node. For more information, see [Explore and Visualize](#).

Several charts have detailed descriptions available that explain how the charts are generated. The descriptions also provide information about how to interpret the charts that is custom to your particular data and models. To access these detailed descriptions:

- Expand an individual chart by clicking . The detailed description is in the right panel.
- Select  on an individual chart. A window appears with the detailed description.

Note: The detailed descriptions are available only in the English language.

Node

- **Bayesian Network** — Displays the Bayesian network that the node created.
- **Autotune Best Configuration** — Displays a table of the resulting values of parameters that were autotuned. This table is not displayed if autotuning is not selected.

- **Autotune Results** — Displays the iterative results of the autotuning process, showing how each of the parameter values evolve as the model iterates. This table is not displayed if autotuning is not selected.
- **Validation Results** — Displays a table of statistics for the automatic selection process. These statistics include best model, misclassification errors, significance threshold, prescreening, variable selection, structure, parenting method, and maximum number of parents.
- **Variable Selection** — Displays a table of various computed statistics for each variable. These statistics include whether the variable was selected, chi-square, g-square statistics, degrees of freedom, *p*-value of chi-square, *p*-value of g-square statistics, mutual information, and conditional variables.
- **Order of Input Variables** — Displays a table of input variables ranked by their score.
- **Path EP Score Code** — displays the SAS score code that was created by the node. The score code can be used outside the Model Studio environment to score new data.
- **DS2 Package Code** — Displays the SAS DS2 package code that was created by the node. The score code can be used outside the Model Studio environment to score new data. DS2 package code is used when publishing models to SAS Micro Analytic Service.
- **Score Inputs** — Displays a table of input variables used in scoring calculations. The table includes the variables' name, role, level, type, label, format, and length.
- **Score Outputs** — Displays a table of predicted response variables that are generated during scoring calculations. The table includes the variables' name, role, type, format and length, as well as the creator, creator GUID, and function.
- **Training Code** — Displays the SAS code that Model Studio used to train the node.
- **Properties** — Specifies the various properties that you selected before running the node.
- **Output** — Displays the SAS output of the Bayesian network run.

Assessment

- **Lift Reports** — Displays the cumulative lift as a function of the depth for the model. The cumulative lift is given for each of the data roles. To examine other statistics as a function of depth, use the drop-down menu in the upper right corner. Other statistics include lift, gain, captured response percentage, cumulative captured response percentage, response percentage, and cumulative response percentage. This result is displayed only if the target is a class variable.
- **ROC Reports** — Displays the ROC (receiver operating characteristic) chart for a model, giving the sensitivity as a function of 1-specificity. The sensitivity is given for each of the data roles. To examine other statistics, use the drop-down menu in the upper right corner. Other statistics include accuracy and F1 score. This result is displayed only if the target is a class variable.
- **Fit Statistics** — Displays a table of the fit statistics for the model, broken down by data role.
- **Event Classification** — Displays the confusion matrix at various cutoff values for each partition. The confusion matrix contains true positives (events that are correctly classified as events), false positives (non-events that are classified as

events), false negatives (events that are classified as non-events), and true negatives (non-events that are correctly classified as non-events). The classification cutoffs that are used are the default value of 0.5, the Kolmogorov-Smirnov cutoff value for each partition, and the **Node binary classification cutoff** that is specified for binary targets. Use the drop-down menu to view the information in the chart as percentages or counts. You can also view the information summarized in a table. This result is displayed only if the target is a class variable.

- **Nominal Classification** — Displays either the percentage of observations or the number of observations that predict each level of the target variable. The plot is segmented by target level and partition level. The target level with the greatest predicted probability is the predicted outcome. This result is displayed only if the target is a nominal variable.

Model Interpretability

- **Surrogate Model Variable Importance** — Displays a table of the relative importance of the input variables, including variable level and label. Relative importance is calculated using a one-level decision tree for each input variable to predict the predicted value as a global surrogate model.
- **PD Plot** — Displays the functional relationship between the input variable and the model prediction. For interval input variables, the 95% confidence interval for the average target prediction is indicated by the shaded band around the line. The X axis includes a heat map that shows the distribution of the input variable. The extreme values of the input variables are removed by truncating the lower and upper tails of its distribution. The amount of truncation can be controlled by specifying values for the **Truncate lower tail** and the **Truncate upper tail** properties.
- **PD and ICE Overlay Plot** — Displays the functional relationship between the input variable and the model prediction as well as the functional relationship between individual observations and the model prediction. For each individual observation, the corresponding ICE curve displays values of the input variable on the X axis and the corresponding predicted probability or prediction of the target variable on the Y axis, holding the other input variable values constant at that observation. The X axis includes a heat map that shows the distribution of the input variable. The extreme values of the input variables are removed by truncating the lower and upper tails of its distribution. The amount of truncation can be controlled by specifying values for the **Truncate lower tail** and the **Truncate upper tail** properties.
- **LIME Explanations** — The LIME algorithm calculates a more easily interpretable linear model around an individual observation. The chart displays the regression estimates for the input variables that are selected in the local surrogate linear regression model. The input variables are ordered by significance such that the most important variable for the local regression model is at the bottom of the chart. The LASSO technique is used to select the most significant predictors from the set of input variables that are used to train the model. Each nominal input variable is binary-encoded based on whether it matches the level of the individual observation. A positive estimate indicates that the observed value of the input variable increases the predicted probability of the event.
- **Kernel SHAP Values** — For each individual observation, an input variable's Kernel SHAP value is the contribution of the observed value of the input variable to the predicted probability or prediction of the target variable. The Kernel SHAP values are the regression coefficients that are obtained by fitting a weighted least squares regression. The Kernel SHAP values of all input variables sum to one.

The input variables are ordered by significance such that the most important variable according to the absolute Kernel SHAP values is at the top of the chart. Each nominal input is binary-encoded based on whether it matches the individual observation. Interval inputs are binary encoded based on their proximity to the individual observation.

- **LIME Information Table** — Displays information about the settings used to perform the LIME method.
- **Kernel SHAP Information Table** — Displays information about the settings used to perform the Kernel SHAP method.
- **LIME Fidelity Table** — Displays information about the accuracy of the local explanation predictions that are calculated using the LIME method, compared to the black-box machine learning model. The model prediction, the explainer prediction, and the explainer root mean square error (RMSE) are displayed.
- **Kernel SHAP Fidelity Table** — Displays information about the accuracy of the local explanation predictions that are calculated using the Kernel SHAP method, compared to the black-box machine learning model. The model prediction, the explainer prediction, and the explainer root mean square error (RMSE) are displayed.

Clustering

<i>Overview of Clustering</i>	31
<i>Clustering Properties</i>	32
General Properties	32
Interval Inputs	32
Class Inputs	33
Number of Clusters Estimation	33
Stop Criterion	34
Scored Output Roles	34
<i>Clustering Results</i>	35

Overview of Clustering

The **Clustering** node is a Data Mining Preprocessing node. Use the **Clustering** node to perform observation clustering based on distances that are computed from quantitative or qualitative variables (or both). The observations are divided into clusters such that every observation belongs to exactly one cluster. Clustering places observations into groups or clusters suggested by the data, such that observations within a cluster are similar and observations from different clusters are dissimilar.

This node uses the following algorithms:

- the *k*-means algorithm for clustering interval (quantitative) input variables
- the *k*-modes algorithm for clustering nominal (qualitative) input variables
- the *k*-prototypes algorithm for clustering mixed input that contains both interval and nominal variables

Clustering is accomplished by updating the cluster centroids and the cluster membership of the data iteratively until the convergence criterion is satisfied or until the maximum number of iterations is reached. The aligned box criterion (ABC) technique is used to estimate the number of clusters needed when you do not specify the desired number of clusters.

Observations with missing input values are not used during training. Therefore, you might need to impute your data. It is recommended that you standardize interval variables because variables that have large variances tend to affect the distance measurements more than variables with small variances.

After clustering is performed, the characteristics of the clusters can be examined graphically using the Clustering Results. The consistency of clusters across variables is of particular interest. The multidimensional charts and plots enable you to graphically compare the clusters.

Note that the cluster identifier for each observation can be passed to subsequent nodes for use as an input, ID, or segment variable. The default is a segment variable.

Clustering Properties

General Properties

- **Cluster initialization** — Specifies the method used to determine initial cluster membership. Here are the possible values:
 - **Forgy** — Assigns k data points as the k initial clusters.
 - **Random** — Randomly assigns each observation to a cluster.
- **Automatic gamma** — Specifies whether to automatically compute the gamma coefficient in the mixed distance computation.
- **User specified gamma** — Specifies the value for the coefficient gamma in the mixed distance computation.
- **Random seed** — Specifies a positive integer to be used to start the pseudo-random number generator. The default value is 12345.

Interval Inputs

- **Missing interval inputs** — Specifies the imputation method for interval input variables. Possible values are **Exclude** or **Impute with mean**. The default setting is **Impute with mean**.
- **Standardization method** — Specifies the method for standardizing interval input variables. Here are the possible values:
 - **(none)**
 - **Range**
 - **Z Score**

Specify **Range** to standardize on the range of the input variable, or specify **Z Score** to standardize on the calculated Z Score. Select **(none)** to disable standardization. The default setting is **(none)**.

- **Similarity distance** — Specifies the distance measure for similarity measurement for interval input variables. Possible values are **Euclidean distance** and **Manhattan distance**. **Euclidean distance** calculates the line segment distance between two clusters. **Manhattan distance**, or taxicab distance, calculates the distance between two clusters using only axis-aligned movements. The default setting is **Euclidean distance**.

Class Inputs

- **Missing class inputs** — Specifies the imputation method for class input variables. Possible values are **Exclude** and **Impute with mode**. The default setting is **Exclude**.
- **Similarity distance** — Specifies the distance measure for similarity measurement of class input variables. Here are the possible values:
 - Binary**
 - Global frequency**
 - Relative frequency**

Binary calculates a simple matching distance. **Global frequency** and **Relative frequency** calculate distances based on the frequency of class input variables in the input data table or in each cluster respectively. The default setting is **Binary**.

Number of Clusters Estimation

- **Number of clusters method** — Specifies the method used to estimate the number of clusters. Possible values are **Aligned box criterion** and **User specify**. Select **User specify** to enter the number of clusters in the **Number of clusters** field.

Select the **Aligned box criterion** property to generate the number of clusters based on the following parameters:

- Number of reference data sets** — Specifies the number of reference data sets to be created for each cluster candidate. The default value is 1.
- Maximum number of clusters** — Specifies the maximum number of clusters. If the number of observations is less than the number specified, then the number of clusters will be set to the number of observations. If the number of observations in a cluster is zero, then this cluster will not be displayed in the results. The default value is 6.
- Minimum number of clusters** — Specifies the minimum number of clusters. The default value is 2.
- Estimation criterion** — Specifies the criterion to use to estimate the number of clusters that use the statistics obtained in the ABC method. Here are the possible values:

- **All criteria** — Specifies that the number of clusters is determined by a combination of the other three available property settings.
- **First peak value** — Specifies that the number of clusters is determined by the first peak among the peak values in gap statistics.
- **First peak with one-standard-error** — Specifies that the number of clusters is determined by the smallest k such that the gap value for that k is greater than the one-standard-error adjusted gap value for $k+1$.
- **Global peak value** — Specifies that the number of clusters is determined by the maximum value among all peak values in gap statistics.

The default value is **Global peak value**.

- **Alignment method** — Specifies the method for aligning the reference data set based on the input data. Possible values are **(none)** and **PCA**. If set to **(none)**, the node generates the reference data set from a uniform distribution over the range of values for each subset of the input data set. If set to **PCA**, the node generates the reference data set from a uniform distribution over a box aligned with the principal components of each subset of the input data set.

Stop Criterion

- **Stop method** — Specifies the stop criterion method. The possible values are **Cluster change** and **Within cluster distance**. **Cluster change** is the default.
- **Cluster change parameter** — Specifies the percentile of observations that do not change their cluster for the iteration. The range is between 0 and 100, and the default value is 20. If you select **Cluster change** as the **Stop method**, the cluster calculation will conclude when the number of observations that change clusters is smaller than the chosen parameter value.
- **Within cluster distance parameter** — Specifies the difference of within-cluster distance change between iterations. If you select **Within cluster distance** for the **Stop method**, the cluster calculation will conclude if the cluster distance changes by less than the given parameter after an iteration.
- **Maximum number of iterations** — Specifies the maximum number of iterations for the algorithm to perform. The default value of 10. In each iteration, each observation is assigned to the nearest cluster centroid, and the centroids are recomputed.

Scored Output Roles

- **Cluster variable role** — Specifies the role that you want to assign to the cluster variable. By default, the cluster variable (segment identifier) is assigned a role of **Segment**. Here are the available roles:
 - **ID**
 - **Input**
 - **Rejected**


- **Segment**

The role of segment is useful for BY-group processing. Note that the segment identifier retains the selected variable role when it is passed to subsequent nodes in the pipeline.

- **Cluster distance role** — Specifies the role that is assigned to the cluster distance variable. The default role is **Rejected**.

Clustering Results

After running the node, open the Results window by right-clicking the node and selecting **Results**.

You can download the data that is used to build the charts and tables in the Results window by clicking . A CSV file with the chart or table data will immediately begin downloading.

In the **Output Data** tab, you can view, save, and explore the data that is generated by the node. For more information, see [Explore and Visualize](#).

Summary

- **Segment Plot** — Displays clusters by segment number. The size of each pie slice reflects each segment's percentage of all the clustered data. Each segment ID appears as you click a given pie slice.
- **ABC Statistics** — Displays the relationship between the number of clusters and the gap statistic using ABC. It also displays the selected number of clusters based on ABC. Therefore, it is available only when that method is chosen to estimate the number of clusters. Click on the graph to see the exact gap statistic values at each point.
- **Cluster Centroids** — Displays the centroid coordinates for each of the clusters. Use the **Options** tab to select which coordinates appear in the table.
- **Node Score Code** — Displays SAS score code that was created by the node. The SAS score code can be used outside the Model Studio environment to score new data.
- **Properties** — Specifies the various properties that you selected before running the node.
- **Output** — Displays the SAS output of the cluster run.

Data Exploration

<i>Overview of Data Exploration</i>	37
<i>Data Exploration Properties</i>	38
Variable Selection	38
Interval Variables	39
<i>Data Exploration Results</i>	39

Overview of Data Exploration

You will frequently find it useful to profile a data set before continuing analysis. Much like PROC CONTENTS, the **Data Exploration** node enables you to profile a data set. The **Data Exploration** node is a Miscellaneous node that selects a subset of variables to provide a representative snapshot of the data. Variables can be selected to show most important inputs, or to indicate “suspicious variables”—that is, variables with anomalous statistics. The **Data Exploration** node can be placed anywhere in a pipeline, except after the **Model Comparison** node.

The **Data Exploration** node helps you understand the variables in your data table, including the target variable and potential predictors of your target variable. The node can identify good candidates for inclusion in predictive models and variables that you might want to exclude. It can suggest variables that might require transformation (for example, skewed variables) or imputation. If you connect the **Data Exploration** node to a Supervised Learning node, you can use the node to explore the model's predicted values.

The node's two modes display either the most important inputs or the suspicious variables. By default, the most important inputs are shown, and you can control the maximum number of variables that are displayed. If you run the node in **Importance** mode, the results display a bar chart of input variables, ranked by importance. When you specify **Screening** mode, you control the selection of suspicious variables by specifying cutoffs that flag variables with a high percentage of missing values, high cardinality class variables, class variables with dominant levels, rare modes, skewed or peaky interval variables, and interval variables with heavy tails. In **Screening**

mode, the results display tables of suspicious variables and the reasons why they were deemed suspicious.

You can display a three-dimensional nonlinear projection of interval inputs using t-Distributed Stochastic Neighbor Embedding (t-SNE). The t-SNE plot might help identify naturally occurring clusters in the data.

Data Exploration Properties

Variable Selection


- **Input data partition** — Specifies the input data partition to analyze. Possible values are **All Data**, **Train**, **Validation**, and **Test**. The default value is **All data**.
- **Variable selection criterion** — Specifies whether to display the most important input variables or suspicious variables. You control the selection of suspicious variables by specifying screening criteria. Possible values are **Importance** and **Screening**. The default value is **Importance**.
- **Maximum number of variables to select** — Specifies the maximum number of important inputs to display. The default value is 50. This option is unavailable if the **Variable selection criterion** is **Screening**.
- **Screening Cutoffs** — Specifies the screening criteria for suspicious variables. These options are available only if the **Variable selection criterion** is **Screening**.
 - **Missing values** — Specifies the cutoff for flagging variables with a high percentage of missing values. Specify a percentage. The default value is 25.
 - **Class levels** — Specifies the cutoff for flagging high cardinality class variables. Specify the number of class levels. The default value is 32.
 - **Dominant mode** — Specifies the cutoff for flagging class variables with dominant levels. Specify a percentage. The default value is 80.
 - **Rare mode** — Specifies the cutoff for flagging class variables with rare modes. Specify a percentage. The default value is 20.
 - **Skewness** — Specifies the cutoff for flagging skewed interval variables. Specify an absolute skewness value. (For example, a value of 1 flags variables with skewness greater than 1 or less than -1.) The default value is 1.
 - **Peakiness** — Specifies the cutoff for flagging peaky (leptokurtic) interval variables. Specify a positive kurtosis value. The default value is 2.
 - **Flatness** — Specifies the cutoff for flagging interval variables with thick tails (that is, platykurtic distributions). Specify a negative kurtosis value. The default value is -1.

Interval Variables

- **Number of bins** — Specifies the number of bins for histograms of interval variables. Possible values range from 4 to 32. The default value is 8.
- **Cluster plot** — Specifies whether to display clustering of interval variable distributions. By default, this option is deselected.
- **Minimum number of interval variables** — Specifies the minimum number of interval variables needed to produce a cluster plot. If there are fewer interval variables, the plot is not produced. This option is available only if **Cluster plot** is selected. The default value is 20.
- **t-SNE projection** — Specifies whether to perform t-SNE projections of input variables. This option is deselected by default.
- **t-SNE perplexity** — Specifies the t-SNE perplexity that controls the separation of points in the projected space. Possible values range from 1 to 100. The default value is 30.

Data Exploration Results

After running the node, open the Results window by right-clicking the node and selecting **Results**.

You can download the data that is used to build the charts and tables in the Results window by clicking . A CSV file with the chart or table data will immediately begin downloading.

In the **Output Data** tab, you can view, save, and explore the data that is generated by the node. For more information, see [Explore and Visualize](#).

Summary

- **Data Partition Summary** — Displays information about the data partition used for the analysis and the number of observations in that data partition.
- **Important Inputs** — Displays a bar chart of the input variables, ranked by importance. This bar chart is available only if the **Variable selection criterion** is **Importance**.
- **Class Variable Summaries** — Displays a bar chart of the number of levels for each of the class variables. Hidden categories are displayed by blue bars, and non-hidden categories are displayed by orange bars. To examine the percent of values equal to the mode for each class variable, use the drop-down menu in the upper right corner.
- **Suspicious Class Variables** — Displays a table of class variables that have been flagged for having suspicious characteristics. These characteristics include too many levels, rare mode, dominant mode, and too many missing values. This table is available only if the **Variable selection criterion** is **Screening**.

- **Suspicious Interval Variables** — Displays a table of interval variables that have been flagged for having suspicious characteristics. These characteristics include skewness, peakiness, flat (thick tails), and too many missing values. This table is available only if the **Variable selection criterion** is **Screening**.
- **Class Variable Distributions** — Displays a bar chart of the frequency percentage of each value for a given class variable. To switch the displayed class variable, use the drop-down menu in the upper right corner.
- **Interval Variable Moments** — Displays a table of the interval variables with their associated statistics. These statistics include minimum, maximum, mean, standard deviation, skewness, kurtosis, relative variability, and the mean plus or minus two standard deviations.
- **Interval Variable Summaries** — Displays a scatter plot that represents the deviation from normality (that is, kurtosis on the Y axis and skewness on the X axis) for each interval variable. To examine a bar chart of the relative variability for each interval variable, use the drop-down menu in the upper right corner.

Note: Relative variability is useful for comparing variables with similar scales, such as several income variables.

- **Interval Variable Distributions** — Displays a bar chart of the frequency percentage of each bin for a given interval variable. The X axis displays the minimum variable value in the bin. To switch the displayed interval variable, use the drop-down menu in the upper right corner.
- **Missing Values** — Displays a bar chart of the percentage of missing values for each variable.
- **Target by Input Crosstabulations** — Displays a bar chart of the relative frequency percentage of each level of the target variable for a given input variable. For interval variables, the X axis displays the minimum variable value in the bin. To switch the displayed variable, use the drop-down menu in the upper right corner. The variables that are available in the drop-down menu are determined by the **Variable selection criterion** that you specify. If you specify **Importance** as the **Variable selection criterion**, then you can also specify **Maximum number of variables to select** to limit the number of variables that are displayed. This bar chart is available only if the target is a class variable.
- **Target Variable Means** — Displays a bar chart of the mean value of the target variable for each level or bin of the input variable. Hover over a bar to see the number of observations in each level of the input variable. To switch the displayed variable, use the drop-down menu in the upper right corner. This bar chart is available only if the target is an interval variable.
- **t-SNE Projection of Interval Inputs** — Displays a 3-D nonlinear projection of interval inputs using t-Distributed Stochastic Neighbor Embedding (t-SNE). The t-SNE plot can help identify naturally occurring clusters in the data.
- **Properties** — Specifies the various properties that you selected before running the node.
- **Output** — Displays the SAS output of the data exploration run.

Decision Tree

<i>Overview of Decision Tree</i>	41
<i>Decision Tree Properties</i>	42
Splitting Options	42
Pruning Options	43
Seed Property	44
Tree Diagram Options	44
Perform Autotuning	45
Lift Calculation Property	48
Binary Classification Cutoff	48
Post Training Properties	48
<i>Decision Tree Results</i>	50

Overview of Decision Tree

The **Decision Tree** node is a Supervised Learning node. An empirical tree represents a segmentation of the data that is created by applying a series of simple rules. Each rule assigns an observation to a segment based on the value of one input. One rule is applied after another, resulting in a hierarchy of segments within segments. The hierarchy is called a tree, and each segment is called a node. The original segment contains the entire data set and is called the root node of the tree. A node with all its successors forms a branch of the node that created it. The final nodes are called leaves. For each leaf, a decision is made and applied to all observations in the leaf. The type of decision depends on the context. In predictive modeling, the decision is the predicted value.

You can use the **Decision Tree** node to create decision trees that do one of the following tasks:

- classify observations based on the values of nominal or interval inputs
- predict responses based on the values of nominal or interval inputs

Decision trees produce a set of rules that can be used to generate predictions for a new data set. This information can then be used to drive business decisions. For

example, in database marketing, decision trees can be used to develop customer profiles that help marketers target promotional mailings in order to generate a higher response rate.

Decision Tree Properties

Splitting Options

- **Grow criterion** — Specifies the criterion for splitting a parent node into child nodes. The following options are available:
 - **Class target criterion** — Specifies the splitting criterion to use for determining best splits on inputs that are given a class target. Here are the possible values:
 - **CHAID**
 - **Chi-square**
 - **Entropy**
 - **Gini**
 - **Information gain ratio**The default value is **Information gain ratio**.
 - **Interval target criterion** — Specifies the splitting criterion to use for determining the best splits on inputs that are given an interval target. Here are the possible values:
 - **CHAID**
 - **F test**
 - **Variance**The default value is **Variance**.
 - **Significance level** — Specifies the significance level for the splitting criteria CHAID, Chi-Square, and F Test. The default value is 0.2.
 - **Bonferroni** — Specifies whether to apply a Bonferroni adjustment to the top p-values for the splitting criteria CHAID, Chi-Square, and F Test. By default, this option is deselected.
- **Maximum number of branches** — Specifies the maximum number of branches that a splitting rule produces. The default value of 2 results in binary splits. Possible values range from 2 to 10.
- **Maximum depth** — Specifies the maximum number of generations in nodes. The original node, generation 0, is called the root node. The children of the root node are the first generation. Possible values range from 1 to 50. The default value is 10.
- **Minimum leaf size** — Specifies the smallest number of training observations that a leaf can have. The default value is 5.

- **Missing values** — Specifies how a splitting rule handles an observation with missing values. Here are the possible values:
 - **Ignore**
 - **Largest branch**
 - **Most correlated branch**
 - **Separate branch**
 - **Use as machine smallest**
 - **Use in search**

Missing values are either used as a value during the split search or assigned to a node based on this selection. The default value is **Use in search**.

- **Minimum missing use in search** — Specifies the minimum number of missing values needed in a splitting variable for missing to be treated a separate level. This property is available only when **Use in search** is specified for **Missing values**. The default value is 1.
- **Number of interval bins** — Specifies the number of bins used for interval inputs. Bin size is $(\text{maximum value} - \text{minimum value}) / (\text{Number of interval bins})$. The default value is 50.
- **Interval bin method** — Specifies the method used to bin the interval input variables. Select **Bucket** to divide input variables into evenly spaced intervals based on the difference between maximum and minimum values. Select **Quantile** to divide input variables into approximately equal sized groups. The default value is **Quantile**.
- **Surrogate rules** — Specifies the number of surrogate rules. The default value is 0.
- **Use input once** — Specifies that no splitting rule will be based on an input variable that has already been used in a splitting rule of an ancestor node. By default, this is deselected.
- **Perform clustering-based split search** — Specifies that a clustering-based search algorithm, instead of an exhaustive search, be used for determining the best split for each input for each tree node. When this option is selected, the order of bins is ignored for interval inputs. By default, this is deselected.

Pruning Options

- **Subtree method** — Specifies how to construct the subtree in terms of subtree methods. Possible values include the following:
 - **C4.5** — The pruning is done with a C4.5 algorithm.
 - **Cost complexity** — The subtree with a minimum leaf-penalized ASE is chosen.
 - **Reduced error** — The smallest subtree with the best assessment value is chosen.

C4.5 is available only for class targets. With **Reduced error** pruning, the assessment measure for class targets is misclassification rate, and the

assessment measure for interval targets is ASE. The default value is **Cost complexity**.

- **Selection method** — Specifies how to construct the subtree in terms of selection methods. Possible values include the following:
 - **Automatic** — Specifies the appropriate subtree for the specified subtree pruning method.
 - **Cost-complexity alpha** — Specifies the subtree by cost-complexity. The default value is 0.
 - **Largest** — Specifies the full tree.
 - **N** — Specifies the largest subtree with at most N leaves.

The default value is **Automatic**.
- **Number of leaves** — Specifies the number of leaves that are used in creating the subtree when the subtree selection method is set to **N**. The default value is 1.
- **Cost-complexity alpha** — Specifies the alpha value for selecting the tree when the subtree selection method is set to **Cost-complexity alpha**. The default value is 0.
- **Confidence** — Specifies the binomial distribution confidence level to use to determine the error rates of merged and split nodes. The default value is 0.25. This option is available only when **C4.5** is the pruning method.
- **Cross validation folds** — Specifies the number of cross validation folds to use for cost-complexity pruning when there is no validation data. Possible values range from 2 to 20. The default value is 10.
- **1-SE rule** — Specifies whether to perform the one standard error rule when performing cross validated cost-complexity pruning. By default, this is deselected.

Seed Property

- **Seed** — Specifies the random number seed used for cross validation cost complexity or for autotuning. The default value is 12345.

Tree Diagram Options

- **Display embedded bar charts in tree diagram** — Specifies whether to display an embedded bar chart of the target levels in each node of a tree diagram for a class target. By default, this is selected.
- **Class target node color** — Specifies the value by which to color the nodes in the tree diagram and the treemap for a class target. Here are the possible values:
 - **Probability of event**
 - **Proportion correctly classified**

- **Single color**

The default value is **Probability of event**.

- **Interval target node color** — Specifies the value by which to color the nodes in the tree diagram and the treemap for an interval target. Possible values are **Average** and **Single color**. The default value is **Average**.

Perform Autotuning

This property selects whether to perform autotuning of any decision tree parameters. Warning: Performing autotuning can substantially increase run time. If **Perform Autotuning** is selected, the following options are available:

- **Maximum Depth** — Specifies whether to autotune the maximum depth parameter. If selected, the following option is available:
 - **Initial value** — Specifies the initial value for autotuning maximum depth. The default value is 10. Use the **From** and **To** options to specify the range. The default **From** value is 1, and the default **To** value is 19.
- **Minimum Leaf Size** — Specifies whether to autotune the minimum leaf size parameter. The following option is available:
 - **Initial value** — Specifies the initial value for autotuning the minimum leaf size. The default value is 5. Use the **From** and **To** options to specify the range. The default **From** value is 1, and the default **To** value is 100.
- **Interval Input Bins** — Specifies whether to autotune the number of interval input bins. If selected, the following option is available:
 - **Initial value** — Specifies the initial value for autotuning the number of interval input bins. The default value is 50. Use the **From** and **To** options to specify the range. The default **From** value is 20, and the default **To** value is 200.
- **Grow Criterion** — Specifies whether to autotune the grow criterion. If selected, the following options are available:
 - **Class target** — Specifies the list of values for autotuning the grow criterion for a class target. Select a subset of the following values:
 - **Entropy**
 - **CHAID**
 - **Information gain ratio**
 - **Gini**
 - **Chi-square**
 - **Interval target** — Specifies the list of values for autotuning the grow criterion for an interval target. Select a subset of the following values:
 - **Variance**
 - **F test**
 - **CHAID**
- **Search Options** — Specifies the options for autotuning searching. The following options are available:

- **Search method** — Specifies the tuning search method. Here are the possible values:
 - **Bayesian** — Specifies the Bayesian method. This method uses priors to seed the iterative optimization.
 - **Genetic algorithm** — Specifies the genetic algorithm method. This method uses an initial Latin Hypercube sample that seeds a genetic algorithm. The genetic algorithm generates a new population of alternative configurations at each iteration.
 - **Grid** — Specifies the grid method. This method performs an exhaustive search of all configurations. Each hyperparameter of interest is discretized into a desired set of values to be studied. Models are trained and assessed for all combinations of values across all the hyperparameters (thus forming a multi-dimensional “grid”).
 - **Latin hypercube sample** — Specifies the Latin Hypercube method. This method performs an optimized grid search that is uniform in each tuning parameter, but random in combinations.
 - **Random** — Specifies the Random method. This method generates a single sample of purely random configurations.

The default value is **Genetic algorithm**.

- **Number of evaluations per iteration** — Specifies the number of tuning evaluations in one iteration. This option is available only if the **Search method** is **Genetic algorithm** or **Bayesian**. The default value is 10.
- **Maximum number of evaluations** — Specifies the maximum number of tuning evaluations. This option is available only if the **Search method** is **Genetic algorithm** or **Bayesian**. The default value is 50.
- **Maximum number of iterations** — Specifies the maximum number of tuning iterations. This option is available only if the **Search method** is **Genetic algorithm** or **Bayesian**. The default value is 5.
- **Maximum number of points in model** — Specifies the maximum number of points in the model. This option is available only if the **Search method** is **Bayesian**. The default value is 100.
- **Sample size** — Specifies the sample size. This option is available only if the **Search method** is **Random** or **Latin hypercube sample**. The default value is 50.
- **General Options** — Specifies the general properties for autotuning. The following options are available:
 - **Validation method** — Specifies the validation method for finding the objective value. If your data is partitioned, then that partition is used. **Validation method**, **Validation data proportion**, and **Cross validation number of folds** are all ignored. Here are the possible values:
 - **Partition** — Specifies using a single partition of a training set. With partition, you specify proportions to use for randomly assigning observations to each role.
 - **K-fold cross validation** — Specifies using the cross validation method. In cross validation, each model evaluation requires k training executions (on $k-1$ data folds) and k scoring executions (on one holdout fold). This increases the evaluation time by approximately a factor of k .

For small to medium data tables, cross validation provides, on average, a better representation of error across the whole data table. **Partition** is the default value.

- **Training data proportion** — Specifies the proportion of data to be used for training the **Partition** validation method. The default value is 0.7.
- **Validation data proportion** — Specifies the proportion of data to be used for the **Partition** validation method. The default value is 0.3.
- **Cross validation number of folds** — Specifies the number of partition folds in the cross validation process (the k defined above). Possible values range from 2 to 20. The default value is 5.
- **Class target objective function** — Specifies the objective function to optimize for tuning parameters for a nominal target. Here are the possible values:
 - **Area under the curve**
 - **Average squared error**
 - **F0.5 score**
 - **F1 score**
 - **Gamma**
 - **Gini coefficient**
 - **Kolmogorov-Smirnov statistic**
 - **Misclassification rate**
 - **Multi-class log loss**
 - **Root average squared error**
 - **Tau**

The default value is **Kolmogorov-Smirnov statistic**.

- **Interval target objective function** — Specifies the objective function to optimize for tuning parameters for an interval target. Here are the possible values:
 - **Average squared error**
 - **Mean absolute error**
 - **Mean squared logarithmic error**
 - **Root average squared error**
 - **Root mean absolute error**
 - **Root mean squared logarithmic error**

The default value is **Average squared error**.

- **Maximum time (minutes)** — Specifies the maximum time (in minutes) for the optimization tuner. The default value is 60.
- **Maximum training time for single model (minutes)** — Specifies the maximum time (in minutes) for training a single model.

Lift Calculation Property

- **Use the exact percentile method for lift calculations** — Specifies whether to use the exact percentile method for calculating lift and related assessment measures. When this property is deselected, the iterative method is used. The exact method should be used if there are convergence errors when using the iterative method.

Binary Classification Cutoff

- **Specify node binary classification cutoff** — Specifies whether to use the binary classification cutoff specified below for the node. If this property is deselected, the project binary classification cutoff is used for determining the predicted value for a binary target based on the posterior probabilities. By default, this property is deselected.
- **Node binary classification cutoff** — Specifies the cutoff to use in the node for determining the predicted value for a binary target based on the posterior probabilities. This option is available only if you select **Specify node binary classification cutoff**. The default value is 0.5.

Note: The **Rules** settings in the Project Settings window enable you to specify a new cutoff value by selecting **Override the default classification cutoff**. If **Specify node binary classification cutoff** is deselected, then the value that you specified in the Project Settings window is used. If **Specify node binary classification cutoff** is selected, then the value that you specified in **Node binary classification cutoff** is used.

Post Training Properties

Changing the following properties will not retrain the model.

Model Interpretability — Specifies the method of model interpretation in order to show how the model responds to changing inputs.

- **Global Interpretability**
 - **Variable importance** — Specifies whether to display a table of the relative importance values for the input variables. The surrogate variable importance values are calculated using a one-level decision tree for each input to predict the predicted value as a global surrogate model. For tree-based supervised learning models, the model variable importance table is displayed as well.
 - **PD plots** — Specifies whether to display partial dependency (PD) plots that show the relationship between model input variables and their predictors. PD plots are generated for only up to the top ten variables in the relative

importance table. Here is how to determine the variables that are displayed in the PD plot: for tree-based models (forest, decision tree, and gradient boosting), the model variable importance table is used. For other supervised learning models, the surrogate variable importance table is used.

■ Local Interpretability

- **ICE plots** — Specifies whether to display Individual Conditional Expectation (ICE) plots. ICE plots can reveal interesting subgroups and interactions between model variables. ICE plots are generated for only up to the top ten variables in the relative importance table. Here is how to determine the variables that are displayed in the ICE plot: for tree-based models (forest, decision tree, and gradient boosting), the model variable importance table is used. For other supervised learning models, the surrogate variable importance table is used.
- **LIME** — Specifies whether to use the Locally Interpretable Model-Agnostic Explanation (LIME) method to explain the model predictions.
- **Kernel SHAP** — Specifies whether to use the Kernel Shapley additive explanations (Kernel SHAP) method to explain the model predictions.
- **Maximum number of Kernel SHAP variables** — Specifies the maximum number of Kernel SHAP variables to plot. Possible values range from 1 to 100. The default value is 20.
- **Specify instances to explain** — Specifies the individual observations that you want to explain. The **Random** observations option provides explanations for five randomly selected observations. The **Specify up to 5** observations option enables you to provide IDs for the specific observations in the data. The IDs should be unique values of the variable that has the role **Key** in the metadata. If no variable is assigned the role **Key**, then you can use the variable `_dmIndex_` to specify the individual observations.
- **LIME/Kernel SHAP Tables**
 - **Explainer Information table** — Specifies whether the Explainer Information tables for the LIME and Kernel SHAP methods should be included in the node results. The Explainer Information table contains information about what settings were used to perform the LIME and Kernel SHAP methods.
 - **Explainer Fidelity table** — Specifies whether the Explainer Fidelity tables for the LIME and Kernel SHAP methods should be included in the node results. The Explainer Fidelity table contains information about the accuracy of the local explanation predictions, compared to the black-box machine learning model.


■ PD/ICE Options

- **Maximum number of variables** — Specifies the maximum number of input variables to use to generate the PD and ICE plots. The input variables are selected based on their relative importance. Possible values range from 1 to 10. The default value is 5.
- **Number of observations** — Specifies the maximum number of observations to sample for the PD and ICE plots. The default value is 1000.
- **Number of tick points** — Specifies the number of tick points for an interval analysis variable to generate for the PD and ICE plots. Possible values range from 3 to 100. The default value is 50.

- **Truncate lower tail** — Specifies under what percentage to discard the values of the interval analysis variables for the PD and ICE plots. Possible values range from 0 to 99. The default value is 5%.
- **Truncate upper tail** — Specifies above what percentage to discard the values of the interval analysis variables for the PD and ICE plots. Possible values range from 0 to 99. The default value is 95%.
- **Seed** — Specifies the seed for generating random numbers. The seed is used for selecting individual observations to explain, selecting observations to generate PD and ICE plots, and simulating data for LIME and Kernel SHAP.



Decision Tree Results

After running the node, open the Results window by right-clicking the node and selecting **Results**.

You can download the data that is used to build the charts and tables in the Results window by clicking . A CSV file with the chart or table data will immediately begin downloading.

In the **Output Data** tab, you can view, save, and explore the data that is generated by the node. For more information, see [Explore and Visualize](#).

Several charts have detailed descriptions available that explain how the charts are generated. The descriptions also provide information about how to interpret the charts that is custom to your particular data and models. To access these detailed descriptions:

- Expand an individual chart by clicking . The detailed description is in the right panel.
- Select  on an individual chart. A window appears with the detailed description.

Note: The detailed descriptions are available only in the English language.

Node

- **Tree Diagram** — Displays the decision tree. The line width of the tree is proportionally given by the ratio of the number of observations in the branch to the number of observations in the root node. When you move your mouse pointer over a node of a tree, a text box displays information about the node. If **Display embedded bar charts in tree diagram** was selected, a bar chart is displayed. This bar contains the counts of each target level in each node. In addition, you can color the nodes by probability of event or proportion correctly classified. For an interval target, you can color the nodes by the target average.
- **Treemap** — Displays a compact graphical display of the tree, which is similar to the tree in the **Tree Diagram** window. The rectangular regions that represent the tree nodes are colored by probability of event or proportion correctly classified for a class target or the average of an interval target. The node width is proportional to the number of observations in the node. Selecting a node in the **Treemap** window displays information about the node.

- **Pruning Error Plot** — Displays a graph of the misclassification rate for a class target or ASE for an interval target. These values are given as a function of the number of leaves in a subtree for each data partition when pruning data exists. A reference line is drawn to indicate which subtree was selected as the final model.
- **Cross Validation Cost-Complexity** — Displays a plot with data on cross validation cost-complexity. This plot is displayed only when cross validation cost-complexity pruning is performed. For a class target, the plot displays the misclassification rate. For an interval target, this plot displays ASE as a function of the number of leaves in a subtree. Error bars are also given, indicating the average error rate plus or minus one standard error. A vertical reference line is drawn to indicate which subtree was selected as the final model. A horizontal reference line is drawn to represent the “1-SE Rule” even if it is not used for subtree selection.
- **Variable Importance** — Displays a table of importance data for each variable. These statistics include validation importance, importance standard deviation, relative importance, and count.
- **Autotune Best Configuration** — Displays a table of the resulting values of parameters that were autotuned. This table is not displayed if autotuning is not selected.
- **Autotune Results** — Displays the iterative results of the autotuning process, showing how each of the parameter values evolve as the model iterates. This table is not displayed if autotuning is not selected.
- **Node Score Code** — Displays the SAS score code that was created by the node. The SAS score code can be used outside the Model Studio environment in custom user applications.
- **Path Score Code** — Displays the SAS score code that was created by the node, as well as all preceding nodes. The SAS model score code can be used outside the Model Studio environment in custom user applications.
- **DS2 Package Code** — Displays the SAS DS2 package code that was created by the node. The SAS score code can be used outside the Model Studio environment in custom user applications. DS2 package code is used when publishing models to SAS Micro Analytic Service.
- **Score Inputs** — Displays a table of data on the input variables for scoring calculations. The table includes the variables’ name, role, level, type, label, format, and length.
- **Score Outputs** — Displays a table of data on the predicted response variable for scoring calculations. The table includes the variables’ name, role, type, format and length, as well as the creator, creator GUID, and function.
- **Training Code** — Displays the SAS code that Model Studio used to train the node.
- **Properties** — Specifies the various properties that you selected before running the node.
- **Output** — Displays the SAS output of the decision tree run.

Assessment

- **Predicted Reports** — Displays the predicted and target mean as a function of the depth for the model. The predicted and target mean are given for each of the data roles. To examine the predicted mean as a function of the target mean, use the drop-down menu in the upper right corner. This result is displayed only if the target is an interval variable.

- **Lift Reports** — Displays the cumulative lift as a function of the depth for the model. The cumulative lift is given for each of the data roles. To examine other statistics as a function of depth, use the drop-down menu in the upper right corner. Other statistics include lift, gain, captured response percentage, cumulative captured response percentage, response percentage, and cumulative response percentage. This result is displayed only if the target is a class variable.
- **ROC Reports** — Displays the ROC (receiver operating characteristic) chart for a model, giving the sensitivity as a function of 1-specificity. The sensitivity is given for each of the data roles. To examine other statistics, use the drop-down menu in the upper right corner. Other statistics include accuracy and F1 score. This result is displayed only if the target is a class variable.
- **Fit Statistics** — Displays a table of the fit statistics for the model, broken down by data role.
- **Event Classification** — Displays the confusion matrix at various cutoff values for each partition. The confusion matrix contains true positives (events that are correctly classified as events), false positives (non-events that are classified as events), false negatives (events that are classified as non-events), and true negatives (non-events that are correctly classified as non-events). The classification cutoffs that are used are the default value of 0.5, the Kolmogorov-Smirnov cutoff value for each partition, and the **Node binary classification cutoff** that is specified for binary targets. Use the drop-down menu to view the information in the chart as percentages or counts. You can also view the information summarized in a table. This result is displayed only if the target is a class variable.
- **Nominal Classification** — Displays either the percentage of observations or the number of observations that predict each level of the target variable. The plot is segmented by target level and partition level. The target level with the greatest predicted probability is the predicted outcome. This result is displayed only if the target is a nominal variable.

Model Interpretability

- **Surrogate Model Variable Importance** — Displays a table of the relative importance of the input variables, including variable level and label. Relative importance is calculated using a one-level decision tree for each input variable to predict the predicted value as a global surrogate model.
- **Model Variable Importance** — Displays a table of the relative importance of the input variables, including variable level and label. This is the variable importance table that is generated when training the original decision tree model.
- **PD Plot** — Displays the functional relationship between the input variable and the model prediction. For interval input variables, the 95% confidence interval for the average target prediction is indicated by the shaded band around the line. The X axis includes a heat map that shows the distribution of the input variable. The extreme values of the input variables are removed by truncating the lower and upper tails of its distribution. The amount of truncation can be controlled by specifying values for the **Truncate lower tail** and the **Truncate upper tail** properties.
- **PD and ICE Overlay Plot** — Displays the functional relationship between the input variable and the model prediction as well as the functional relationship between individual observations and the model prediction. For each individual observation, the corresponding ICE curve displays values of the input variable on the X axis and the corresponding predicted probability or prediction of the target variable on the Y axis, holding the other input variable values constant at

that observation. The X axis includes a heat map that shows the distribution of the input variable. The extreme values of the input variables are removed by truncating the lower and upper tails of its distribution. The amount of truncation can be controlled by specifying values for the **Truncate lower tail** and the **Truncate upper tail** properties.

- **LIME Explanations** — The LIME algorithm calculates a more easily interpretable linear model around an individual observation. The chart displays the regression estimates for the input variables that are selected in the local surrogate linear regression model. The input variables are ordered by significance such that the most important variable for the local regression model is at the bottom of the chart. The LASSO technique is used to select the most significant predictors from the set of input variables that are used to train the model. Each nominal input variable is binary-encoded based on whether it matches the level of the individual observation. A positive estimate indicates that the observed value of the input variable increases the predicted probability of the event.
- **Kernel SHAP Values** — For each individual observation, an input variable's Kernel SHAP value is the contribution of the observed value of the input variable to the predicted probability or prediction of the target variable. The Kernel SHAP values are the regression coefficients that are obtained by fitting a weighted least squares regression. The Kernel SHAP values of all input variables sum to one. The input variables are ordered by significance such that the most important variable according to the absolute Kernel SHAP values is at the top of the chart. Each nominal input is binary-encoded based on whether it matches the individual observation. Interval inputs are binary encoded based on their proximity to the individual observation.
- **LIME Information Table** — Displays information about the settings used to perform the LIME method.
- **Kernel SHAP Information Table** — Displays information about the settings used to perform the Kernel SHAP method.
- **LIME Fidelity Table** — Displays information about the accuracy of the local explanation predictions that are calculated using the LIME method, compared to the black-box machine learning model. The model prediction, the explainer prediction, and the explainer root mean square error (RMSE) are displayed.
- **Kernel SHAP Fidelity Table** — Displays information about the accuracy of the local explanation predictions that are calculated using the Kernel SHAP method, compared to the black-box machine learning model. The model prediction, the explainer prediction, and the explainer root mean square error (RMSE) are displayed.

Ensemble

<i>Overview of Ensemble</i>	55
<i>Ensemble Properties</i>	56
Interval Target	56
Class Target	56
Lift Calculation Property	56
Binary Classification Cutoff	56
Post Training Properties	57
<i>Ensemble Results</i>	58

Overview of Ensemble

The **Ensemble** node is a Postprocessing node. The **Ensemble** node creates new models by combining the posterior probabilities (for class targets) or the predicted values (for interval targets) from multiple predecessor models. The new model is then used to score the project and assess the new model.

One common ensemble approach is to use multiple modeling methods, such as a neural network and a decision tree, to obtain separate models from the same training data set. The component models from the two complementary modeling methods are integrated by the **Ensemble** node to form the final model solution.

The **Ensemble** node produces score code by combining the score code of the various models. The score code consists of a single score code file that contains DATA step code if all models produce DATA step score code. If any of the models have score code in the form of ecode and analytic stores, then the ensemble code consists of an ecode file and one or more analytic stores.

Note: The ensemble model can be more accurate than the individual models only if the individual models disagree with one another. You should always compare the model performance of the ensemble model with the individual models. You can compare models in a **Model Comparison** node. One useful feature of this

component is the ability to assemble complex models such as forest or gradient boosting models.

Ensemble Properties

Interval Target

- **Predicted values** — Specifies the function to combine models for interval targets. Possible values are **Average** and **Maximum**. The default value is **Average**.

Class Target

- **Posterior probabilities** — Specifies the function to combine models for class targets. Here are the possible values:
 - Average**
 - Geometric Mean**
 - Maximum**
 - Voting**The default value is **Average**.

Lift Calculation Property

- **Use the exact percentile method for lift calculations** — Specifies whether to use the exact percentile method for calculating lift and related assessment measures. When this property is deselected, the iterative method is used. The exact method should be used if there are convergence errors when using the iterative method.

Binary Classification Cutoff

- **Specify node binary classification cutoff** — Specifies whether to use the binary classification cutoff specified below for the node. If this property is deselected, the project binary classification cutoff is used for determining the

predicted value for a binary target based on the posterior probabilities. By default, this property is deselected.

- **Node binary classification cutoff** — Specifies the cutoff to use in the node for determining the predicted value for a binary target based on the posterior probabilities. This option is available only if **Specify node binary classification cutoff** is selected. The default value is 0.5.

Note: The **Rules** settings in the Project Settings window enable you to specify a new cutoff value by selecting **Override the default classification cutoff**. If **Specify node binary classification cutoff** is deselected, then the value that you specified in the Project Settings window is used. If **Specify node binary classification cutoff** is selected, then the value that you specified in **Node binary classification cutoff** is used.

Post Training Properties

Changing the following properties will not retrain the model.

Model Interpretability — Specifies the method of model interpretation in order to show how the model responds to changing inputs.

- **Global Interpretability**

- **Variable importance** — Specifies whether to display a table of the relative importance values for the input variables. The surrogate variable importance values are calculated using a one-level decision tree for each input to predict the predicted value as a global surrogate model. For tree-based supervised learning models, the model variable importance table is displayed as well.
- **PD plots** — Specifies whether to display partial dependency (PD) plots that show the relationship between model input variables and their predictors. PD plots are generated for only up to the top ten variables in the relative importance table. Here is how to determine the variables that are displayed in the PD plot: for tree-based models (forest, decision tree, and gradient boosting), the model variable importance table is used. For other supervised learning models, the surrogate variable importance table is used.


- **Local Interpretability**

- **ICE plots** — Specifies whether to display Individual Conditional Expectation (ICE) plots. ICE plots can reveal interesting subgroups and interactions between model variables. ICE plots are generated for only up to the top ten variables in the relative importance table. Here is how to determine the variables that are displayed in the ICE plot: for tree-based models (forest, decision tree, and gradient boosting), the model variable importance table is used. For other supervised learning models, the surrogate variable importance table is used.
- **LIME** — Specifies whether to use the Locally Interpretable Model-Agnostic Explanation (LIME) method to explain the model predictions.
- **Kernel SHAP** — Specifies whether to use the Kernel Shapley additive explanations (Kernel SHAP) method to explain the model predictions.

- **Maximum number of Kernel SHAP variables** — Specifies the maximum number of Kernel SHAP variables to plot. Possible values range from 1 to 100. The default value is 20.
- **Specify instances to explain** — Specifies the individual observations that you want to explain. The **Random** observations option provides explanations for five randomly selected observations. The **Specify up to 5** observations option enables you to provide IDs for the specific observations in the data. The IDs should be unique values of the variable that has the role **Key** in the metadata. If no variable is assigned the role **Key**, then you can use the variable `_dmIndex_` to specify the individual observations.
- **LIME/Kernel SHAP Tables**
 - **Explainer Information table** — Specifies whether the Explainer Information tables for the LIME and Kernel SHAP methods should be included in the node results. The Explainer Information table contains information about what settings were used to perform the LIME and Kernel SHAP methods.
 - **Explainer Fidelity table** — Specifies whether the Explainer Fidelity tables for the LIME and Kernel SHAP methods should be included in the node results. The Explainer Fidelity table contains information about the accuracy of the local explanation predictions, compared to the black-box machine learning model.
- **PD/ICE Options**
 - **Maximum number of variables** — Specifies the maximum number of input variables to use to generate the PD and ICE plots. The input variables are selected based on their relative importance. Possible values range from 1 to 10. The default value is 5.
 - **Number of observations** — Specifies the maximum number of observations to sample for the PD and ICE plots. The default value is 1000.
 - **Number of tick points** — Specifies the number of tick points for an interval analysis variable to generate for the PD and ICE plots. Possible values range from 3 to 100. The default value is 50.
 - **Truncate lower tail** — Specifies under what percentage to discard the values of the interval analysis variables for the PD and ICE plots. Possible values range from 0 to 99. The default value is 5%.
 - **Truncate upper tail** — Specifies above what percentage to discard the values of the interval analysis variables for the PD and ICE plots. Possible values range from 0 to 99. The default value is 95%.
- **Seed** — Specifies the seed for generating random numbers. The seed is used for selecting individual observations to explain, selecting observations to generate PD and ICE plots, and simulating data for LIME and Kernel SHAP.



Ensemble Results

After running the node, open the Results window by right-clicking the node and selecting **Results**.

You can download the data that is used to build the charts and tables in the Results window by clicking . A CSV file with the chart or table data will immediately begin downloading.

In the **Output Data** tab, you can view, save, and explore the data that is generated by the node. For more information, see [Explore and Visualize](#).

Several charts have detailed descriptions available that explain how the charts are generated. The descriptions also provide information about how to interpret the charts that is custom to your particular data and models. To access these detailed descriptions:

- Expand an individual chart by clicking . The detailed description is in the right panel.
- Select  on an individual chart. A window appears with the detailed description.

Note: The detailed descriptions are available only in the English language.

Node

- **Path EP Score Code** — Displays the SAS score code that was created by the node. The EP score code is available only if an analytic store is generated by a node in the pipeline.
- **DS2 Package Code** — Displays the SAS DS2 package code that was created by the node. The score code can be used outside the Model Studio environment to score new data. DS2 package code is used when publishing models to SAS Micro Analytic Service.
- **Path Score Code** — Displays the SAS score code that was created by the node, as well as all preceding nodes. The SAS model score code can be used outside the Model Studio environment in custom user applications.
- **Score Inputs** — Displays a table of data on the input variables for scoring calculations. The table includes the variables' name, role, level, type, label, format, and length.
- **Score Outputs** — Displays a table of data on the predicted response variable for scoring calculations. The table includes the variables' name, role, type, format and length, as well as the creator, creator GUID, and function.
- **Properties** — Specifies the various properties that you selected before running the node.
- **Models** — Displays a table of the statistical models that were combined by the **Ensemble** node.

Assessment

- **Predicted Reports** — Displays the predicted and target mean as a function of the depth for the model. The predicted and target mean are given for each of the data roles. To examine the predicted mean as a function of the target mean, use the drop-down menu in the upper right corner. This result is displayed only if the target is an interval variable.
- **Lift Reports** — Displays the cumulative lift as a function of the depth for the model. The cumulative lift is given for each of the data roles. To examine other statistics as a function of depth, use the drop-down menu in the upper right corner. Other statistics include lift, gain, captured response percentage, cumulative captured response percentage, response percentage, and

cumulative response percentage. This result is displayed only if the target is a class variable.

- **ROC Reports** — Displays the ROC (receiver operating characteristic) chart for a model, giving the sensitivity as a function of 1-specificity. The sensitivity is given for each of the data roles. To examine other statistics, use the drop-down menu in the upper right corner. Other statistics include accuracy and F1 score. This result is displayed only if the target is a class variable.
- **Fit Statistics** — Displays a table of the fit statistics for the model, broken down by data role.
- **Event Classification** — Displays the confusion matrix at various cutoff values for each partition. The confusion matrix contains true positives (events that are correctly classified as events), false positives (non-events that are classified as events), false negatives (events that are classified as non-events), and true negatives (non-events that are correctly classified as non-events). The classification cutoffs that are used are the default value of 0.5, the Kolmogorov-Smirnov cutoff value for each partition, and the **Node binary classification cutoff** that is specified for binary targets. Use the drop-down menu to view the information in the chart as percentages or counts. You can also view the information summarized in a table. This result is displayed only if the target is a class variable.
- **Nominal Classification** — Displays either the percentage of observations or the number of observations that predict each level of the target variable. The plot is segmented by target level and partition level. The target level with the greatest predicted probability is the predicted outcome. This result is displayed only if the target is a nominal variable.

Model Interpretability

- **Surrogate Model Variable Importance** — Displays a table of the relative importance of the input variables, including variable level and label. Relative importance is calculated using a one-level decision tree for each input variable to predict the predicted value as a global surrogate model.
- **PD Plot** — Displays the functional relationship between the input variable and the model prediction. For interval input variables, the 95% confidence interval for the average target prediction is indicated by the shaded band around the line. The X axis includes a heat map that shows the distribution of the input variable. The extreme values of the input variables are removed by truncating the lower and upper tails of its distribution. The amount of truncation can be controlled by specifying values for the **Truncate lower tail** and the **Truncate upper tail** properties.
- **PD and ICE Overlay Plot** — Displays the functional relationship between the input variable and the model prediction as well as the functional relationship between individual observations and the model prediction. For each individual observation, the corresponding ICE curve displays values of the input variable on the X axis and the corresponding predicted probability or prediction of the target variable on the Y axis, holding the other input variable values constant at that observation. The X axis includes a heat map that shows the distribution of the input variable. The extreme values of the input variables are removed by truncating the lower and upper tails of its distribution. The amount of truncation can be controlled by specifying values for the **Truncate lower tail** and the **Truncate upper tail** properties.
- **LIME Explanations** — The LIME algorithm calculates a more easily interpretable linear model around an individual observation. The chart displays the regression estimates for the input variables that are selected in the local

surrogate linear regression model. The input variables are ordered by significance such that the most important variable for the local regression model is at the bottom of the chart. The LASSO technique is used to select the most significant predictors from the set of input variables that are used to train the model. Each nominal input variable is binary-encoded based on whether it matches the level of the individual observation. A positive estimate indicates that the observed value of the input variable increases the predicted probability of the event.

- **Kernel SHAP Values** — For each individual observation, an input variable's Kernel SHAP value is the contribution of the observed value of the input variable to the predicted probability or prediction of the target variable. The Kernel SHAP values are the regression coefficients that are obtained by fitting a weighted least squares regression. The Kernel SHAP values of all input variables sum to one. The input variables are ordered by significance such that the most important variable according to the absolute Kernel SHAP values is at the top of the chart. Each nominal input is binary-encoded based on whether it matches the individual observation. Interval inputs are binary encoded based on their proximity to the individual observation.
- **LIME Information Table** — Displays information about the settings used to perform the LIME method.
- **Kernel SHAP Information Table** — Displays information about the settings used to perform the Kernel SHAP method.
- **LIME Fidelity Table** — Displays information about the accuracy of the local explanation predictions that are calculated using the LIME method, compared to the black-box machine learning model. The model prediction, the explainer prediction, and the explainer root mean square error (RMSE) are displayed.
- **Kernel SHAP Fidelity Table** — Displays information about the accuracy of the local explanation predictions that are calculated using the Kernel SHAP method, compared to the black-box machine learning model. The model prediction, the explainer prediction, and the explainer root mean square error (RMSE) are displayed.

Feature Extraction

<i>Overview of Feature Extraction</i>	63
<i>Feature Extraction Properties</i>	65
Input Variables Property	65
PCA Options	65
RPCA Options	66
Additional RPCA and SVD Options	66
Autoencoder Options	67
<i>Feature Extraction Results</i>	69

Overview of Feature Extraction

The **Feature Extraction** node is a Data Mining Preprocessing node. Use the **Feature Extraction** node to create new features from the initial set of data. These features encapsulate the central properties of a data set and represent it in a low dimensional space. The initial data set of raw features might be too large and unwieldy to be effectively managed, requiring an unreasonable amount of computing resources. Alternatively, the data set might be too robust, causing a classification algorithm to overfit, and providing poor extrapolation in the event of new observations. In either case, the **Feature Extraction** can be used to provide a more manageable, representative subset of input variables.

The **Feature Extraction** node offers four methods. You can either specify the method that you want to use or let the node automatically select the method based on the number of inputs in your data set. The available methods include the following:

- Principal Component Analysis (PCA)
- Singular Value Decomposition (SVD)
- Robust Principal Component Analysis (RPCA)
- Autoencoder

The automatic selection uses PCA when the number of interval inputs is less than or equal to 500. Otherwise, SVD with randomized optimization is used. Only the training partition is used for extracting the new features for the PCA and SVD methods. For the autoencoder method, early stopping is available. Early stopping can be performed based on the validation partition that already exists in your data or on a validation partition created from the training data.

PCA is a multivariate technique for examining linear relationships among several interval variables. It provides an optimal way to reduce dimensionality by projecting the data onto a lower-dimensional orthogonal subspace that explains as much variation as possible in those variables.

SVD is similar to PCA in that it projects the data onto a lower-dimensional orthogonal subspace and is a generalization of the eigenvalue decomposition. It can handle more input variables because it supports Eigen, Iterative, or Randomized optimizations. Randomized optimization provides a fast approximation to computing principal components and is recommended when the number of input variables is greater than 500. For this method, the number of principal components to keep is controlled by the **Maximum rank** property.

RPCA decomposes an input matrix into a sum of two matrices: a low-rank matrix and a sparse matrix. You can use the low-rank matrix to do feature extraction and use the sparse matrix to detect anomalies. Robustness in RPCA is due to the fact that the principal components are computed from observations after removing the outliers. That is, they are removed from the low-rank matrix. Use the **Penalty weight** property to control sparsity because it specifies the weight placed on the penalty that is applied to the sparse matrix. A smaller value places more emphasis on sparsity. Two optimization algorithms are supported for RPCA: the **Augmented Lagrange multiplier** (ALM) and the **Accelerated proximal gradient** (APG). APG is generally slower than ALM. Research suggests that APG is used when observations are significantly corrupted by noise. Because RPCA performs an SVD to decompose the low-rank matrix, the number of principal components to keep is controlled by the **Maximum rank** property.

Autoencoder builds a three-layer or five-layer fully connected, symmetric neural network where the output layer is the same as the input layer. The goal of the autoencoder network is to train and learn the low dimension representation of the data from the middle layer called the code layer. The code layer is the second layer in a three-layer network and third layer in a five-layer network. Therefore, the number of features to keep is controlled by the number of neurons in the middle layer. Choose the number of neurons in the middle layer and in hidden layers before and after the code layer with care. These layers affect the complexity and thus the computation time to train this neural network. The advantage of using an autoencoder compared to PCA, SVD, or RPCA is its ability to capture nonlinear mappings within the data. The autoencoder is trained using stochastic gradient descent (SGD) optimization. Early stopping can be performed based on the validation partition that already exists in your data or on a validation partition created from the training data. Use the **Stagnation for early stopping** property to control early stopping. Overfitting can be mitigated by using regularization with the **L1 weight decay** and **L2 weight decay** properties or by using dropout for denoising via the **Input layer dropout ratio** and **Hidden layer dropout ratio** properties. You can use the Iteration Plot in the results to see how the validation error, objective function, or loss varies across iterations during optimization.

Note: The PCA, SVD, and robust PCA methods are available only for interval inputs. The autoencoder is available for both class and interval inputs. Observations with missing values are omitted from the analysis for all methods.

Feature Extraction Properties

Input Variables Property

- **Feature extraction method** — Specifies the method used for feature extraction. Here are the possible values:
 - Autoencoder**
 - Automatic**
 - Principal component analysis (PCA)**
 - Robust PCA (RPCA)**
 - Singular value decomposition (SVD)**

The choice of feature extraction method determines the subsequent available options. If **Automatic** is selected, PCA is used if the number of input variables is less than or equal to 500. Otherwise, SVD with Randomized optimization is used. Note that PCA, SVD, and RPCA use interval inputs only.

- **Reject original input variables** — Specifies whether the role of the original input variables should be set to Rejected. Note that for PCA, SVD, and RPCA, only interval inputs are rejected.

PCA Options

- **Eigenvalue source** — Specifies the source matrix to calculate eigenvalues and eigenvectors. Possible values are **Correlation**, using the correlation matrix, and **Covariance**, using the covariance matrix.
- **Component prefix** — Specifies the principal component prefix, which is a valid SAS variable name. The default prefix string is PC. Length is limited to 25 characters.
- **Apply fixed number** — Specifies whether to use a fixed number of principal components to be passed to successor nodes. If this option is selected, the **Fixed number** of principal components can range from 1 to 50.
- **Apply maximum number** — Specifies whether to set a maximum limit for the number of principal components to be passed to successor nodes. If this option is selected, the **Maximum number** can range from 1 to 50.
- **Cumulative variance cutoff** — Specifies the cumulative proportional variance cutoff value. The last principal components with cumulative proportional variance greater than this cutoff value are not passed to successor nodes. The default value is 0.99.

- **Minimum variance increment** — Specifies the minimum increment for proportional variance. Principal components with a proportional increment less than this cutoff value are not passed to successor nodes. The default value is 0.001.

RPCA Options

- **Penalty weight** — Specifies the weight to place on the penalty that is applied to sparse matrices in Augmented LaGrange Multiplier (ALM) optimization. A lower value places emphasis on sparsity. The default value is 1.
- **Optimization method for RPCA** — Specifies the optimization method for RPCA. Possible values include the following:
 - **Accelerated proximal gradient** — specifies the APG method, which is a relaxation of the PCP with an unconstrained optimization formulation.
 - **Augmented lagrange multiplier** — Specifies the ALM method. This method is a relaxation of the principal component pursuit, or PCP, as a nonlinear unconstrained optimization problem.

The APG algorithm is generally slower than the ALM algorithm. The APG algorithm is suggested when the observations are significantly corrupted by noise. The default method is **Augmented lagrange multiplier**.

- **Maximum iterations for RPCA** — Specifies the maximum number of iterations to solve the RPCA. The default value is 1000.
- **Tolerance for RPCA** — Specifies the tolerance criteria to solve RPCA—that is, the desired accuracy of the recovered solution. The default value is 0.0000001.

Additional RPCA and SVD Options

- **Input standardization** — Specifies whether to center, scale, or standardize (center and scale) input variables using the traditional standardization method. The possible values are as follows:
 - **(none)**
 - **Center**
 - **Scale**
 - **Z score**

The default value is **(none)**.

- **Maximum rank** — Specifies the maximum number of latent variables that are passed to successor nodes. This number is set to the number of input variables if it exceeds that value. The default value is 100.
- **Component prefix** — Specifies the prefix for output variables, which should be a valid SAS variable name. Length is limited to 25 characters. The default prefix string is COMP.

- **Optimization method for SVD** — Specifies the optimization method used to solve the SVD problem. The possible values are as follows:
 - **Automatic**
 - **Eigen**
 - **Iterative**
 - **Randomized**

The default **Automatic** selection uses **Eigen** optimization if the number of input variables is less than or equal to 500. Otherwise, **Randomized** optimization is used. **Randomized** optimization is recommended for wide data because it is a faster approximation.

Autoencoder Options

- **Input standardization** — Specifies the method that is used to standardize the interval input variables. The possible values are as follows:
 - **(none)**
 - **Range**
 - **Z Score**

The default input standardization method is **Range** .
- **Use missing as level** — Specifies whether missing values should be treated as a level for class inputs.
- **Feature prefix** — Specifies the prefix for output variables, which should be a valid SAS variable name. Length is limited to 25 characters. The default prefix string is Hidden.
- **Hidden layer options** — Specifies the configurations for the hidden layer stage of the network.
 - **Number of hidden layers** — Specifies the number of hidden layers to include in the neural network. The possible values are 3 and 5. The default value is 3.
 - **Middle hidden layer: number of neurons (features)** — Specifies the number of hidden neurons in the middle hidden layer for an autoencoder. For a network with three hidden layers, this number is the second hidden layer. For a network with five hidden layers, this is the third hidden layer. This is the number of features to use to represent the inputs. The default value is 10.
 - **First and last hidden layer: number of neurons** — Specifies the number of hidden neurons in the first and last hidden layer for an autoencoder. The default value is 100.
 - **Second and fourth hidden layer: number of neurons** — Specifies the number of hidden neurons in the second and fourth hidden layer for an autoencoder when **Number of hidden layers** is 5. The default value is 50.
 - **Hidden layer activation function** — Specifies the activation function to use in the hidden layers. Here are the possible values:
 - **Exponential**

- **Identity**
- **Logistic**
- **ReLU**
- **Sine**
- **Softplus**
- **Tanh**
- **Varies**

If **Varies** is selected, you individually select the activation function for each hidden layer. The default value is **Tanh**.

- **Common Optimization Options**

- **Number of tries** — Specifies the number of times to train the network. This is done by using different initial estimates for the weights. Possible values range from 1 to 64. The default value is 1.
- **Maximum iterations** — Specifies the maximum number of iterations allowed within each try. The default value is 300.
- **Stagnation for early stopping** — Specifies the number of successive iterations without improvement in the validation error before stopping the optimization early. The value 0 indicates that early stopping is not performed. The default value is 5.
- **Random seed** — Specifies the random seed to use for generating random numbers to initialize the network weights. The default value is 12345.
- **L1 weight decay** — Specifies the weight decay for L1 regularization. The default value is 0.
- **L2 weight decay** — Specifies the weight decay for L2 regularization. The default value is 0.1.

- **SGD Optimization Options** — Specifies the configurations for the stochastic gradient descent.

- **Learning rate** — Specifies the learning rate parameter for SGD optimization. The default value is 0.001.
- **Annealing rate** — Specifies the annealing rate parameter for SGD optimization. The default value is 0.000001.
- **Input layer dropout ratio** — Specifies the dropout ratio for the input layer when SGD optimization is used. The default rate is 0.
- **Hidden layer dropout ratio** — Specifies the dropout ratio for the hidden layer when SGD optimization is used. The default rate is 0.
- **Minibatch size** — Specifies the size of the minibatches used for SGD optimization. The default value is 10.
- **Momentum** — Specifies the value for momentum for SGD optimization. The default value is 0.
- **Create deterministic results** — Specifies whether to create deterministic (reproducible) results using the specified **SGD seed**.

Note: Selecting this property can significantly increase run time.


If **Create deterministic results** is selected, the **SGD seed** can be set. The default value is 12345.

- **Create validation** — Specifies whether a validation sample should be created from the incoming training data. If **Create validation** is selected, the following configurations are available:
 - **Validation proportion** — Specifies the probability of a given record being selected for the validation partition. The default value is 0.3.
 - **Partition seed** — Specifies the partition seed to generate the sample that will be used for validation. The default value is 12345.

Feature Extraction Results

After running the node, open the Results window by right-clicking the node and selecting **Results**.

Note: The results available for this node vary based on the feature extraction method.

You can download the data that is used to build the charts and tables in the Results window by clicking . A CSV file with the chart or table data will immediately begin downloading.

In the **Output Data** tab, you can view, save, and explore the data that is generated by the node. For more information, see [Explore and Visualize](#).

Summary

- **Principal Components Coefficient** — Displays the coefficient values for each principal component. Switch the selected principal component using the drop-down menu in the upper right corner. This display is available only if the feature extraction method is **Principal component analysis**, **Robust PCA**, or **Singular value decomposition**.
- **Eigenvalue Plots** — Displays the eigenvalue that corresponds to each principal component via a line graph. The vertical line that corresponds to the number of Principal Components selected is in bold. Switch the display between Eigenvalue, Proportional Eigenvalue, Cumulative Proportional Eigenvalue, and Log Eigenvalue using the drop-down menu in the upper right corner. This display is available only if the feature extraction method is **Principal component analysis**, **Robust PCA**, or **Singular value decomposition**.
- **2D Representation of the Data Points Projections** — Displays a scatter plot relating two principal components, with the target variable that determines the color of the point. Switch the principal component pair being displayed using the drop-down menu in the upper right corner. This display is available only if the feature extraction method is **Principal component analysis**.
- **Iteration Plot** — displays a graph that measures the Valid Error after a given number of iterations. Switch the display between Valid Error, Objective, and Loss using the drop-down menu in the upper right corner. This graph is available only if the feature extraction method is **Autoencoder**.

- **2D Representation of Hidden Neurons** — Displays a scatter plot that relates two hidden neurons in the middle layer, with the target variable that determines the color of the point. Switch the hidden neuron pair being displayed using the drop-down menu in the upper right corner. This plot is available only if the feature extraction method is **Autoencoder** and **Middle hidden layer: number of neurons (features)** is less than 6.
- **Node Score Code** — Displays SAS score code that was created by the node. The SAS score code can be used outside the Model Studio environment to score new data.
- **Properties** — Specifies the various properties that you selected before running the node.
- **Output** — Displays the SAS output of the feature extraction run. The output given varies based on the selected feature extraction method.

Feature Machine

<i>Overview of Feature Machine</i>	71
<i>Feature Machine Properties</i>	72
Transformation Policy	72
Input Variable Screening	73
Feature Selection	73
Reject Original Variables Property	74
<i>Feature Machine Results</i>	74

Overview of Feature Machine

The **Feature Machine** node is a Data Mining Preprocessing node. It generates new features by performing variable transformations to improve data quality and improve model accuracy. New features are generated to fix the following data quality issues:

- High cardinality
- High kurtosis
- High skewness
- Low entropy
- Low indices of qualitative variation
- Missing values
- Outliers

After identifying data quality issues, the **Feature Machine** node performs one or more of the following transformations to generate new features:

- Box-Cox transformation — Applies to interval variables that are characterized as having significant skewness. The Box-Cox transformation is applied, followed by median imputation.
- Decision tree binning — Similar to regression tree binning, but applies to classification problems.

- MDLP binning — A binning algorithm is applied based on the minimum description length principle (MDLP).
- Median imputation — Applies to all interval variables, except those that are converted to binary missing indicator variables.
- Missing level and group rare — Applies to nominal variables whose missing rate makes them ineligible for mode imputation. A new missing level is created and the resulting variable is transformed using the group rare transformation.
- Missing level and label encoding — Applies to nominal variables whose missing rate makes them ineligible for mode imputation. A new missing level is created and the resulting variable is transformed using the label encoding transformation.
- Missing indicator — Binary missing indicator features are generated from the input variables.
- Mode imputation and group rare — Applies to nominal variables that have very low missing values. The missing values are imputed using the mode value, and the resulting variable is transformed using the group rare transformation.
- Mode imputation and label encoding — Applies to nominal variables that have very low missing values. The missing values are imputed using the mode value, and the resulting variable is transformed using the label encoding transformation.
- Quantile binning with missing bins — Applies to interval variables that have significant skewness, kurtosis, or outliers.
- Regression tree binning — Applies to both interval and nominal variables and is applicable only to regression problems. It is applied to interval variables when the variables are deemed to have significant skewness, kurtosis, or outliers. It is applied to nominal variables when the variables have significant distinct values.
- Target encoding — Applies to nominal variables that have significant distinct counts. For regression problems, mean, minimum, and maximum target encoding operators are available. For classification problems, frequency ratio, event probability, and weight of evidence target encoding operators are available. Label count and input count encodings (and their log transformations) are applicable to both regression and classification problems.
- Yeo-Johnson transformation and median imputation — Applies to interval variables that are characterized as having significant kurtosis. The Yeo-Johnson transformation is applied, followed by median imputation.

Feature Machine Properties

Transformation Policy

- **Cardinality** — Specifies whether to generate features that include transformations for the treatment of high cardinality.
- **Entropy** — Specifies whether to generate features that include transformations for the treatment of low entropy.

- **Kurtosis** — Specifies whether to generate features that include transformations for the treatment of high kurtosis.
- **Missingness** — Specifies whether to generate features that include transformations for the treatment of missing values.
- **Outliers** — Specifies whether to generate features that include transformations for the treatment of outliers.
- **Qualitative variation** — Specifies whether to generate features that include transformations for the treatment of low indices of qualitative variation.
- **Skewness** — Specifies whether to generate features that include transformations for the treatment of high skewness.

Input Variable Screening

- **Coefficient of variation** — Specifies whether to identify variables that have a low coefficient of variation (close to constant value). The identified variables are excluded from feature processing.
- **Group rare levels** — Specifies whether to identify nominal variables that have rare levels. The identified variables are flagged for Group Rare transformations
- **Leakage percent threshold** — Specifies the entropy reduction threshold for identifying variables that have a very high level of information about the target (leakage variables). Variables that exceed this threshold are excluded from feature processing. The default value is 90.
- **Mutual information threshold** — Specifies the mutual information threshold for identifying variables that have a low level of information about the target (not informative). Variables that are below this threshold are excluded from feature processing. The default value is 0.05.
- **Redundancy threshold** — Specifies the symmetric uncertainty (SU) threshold for identifying redundant variables. If the SU for two variables exceeds the threshold, the variable that has less information about the target variable is excluded from feature processing. The redundancy check is not performed when the threshold is set to 1. The default value is 1.

.....
Note: Performing a redundancy check can significantly increase processing time.
.....

Feature Selection

- **Number of features per input** — Specifies the number of top ranked features to select for each input variable. More features might be selected than the number specified if multiple features have a tied ranking. The number of available features depends on the number of selected transformation policies, where a maximum of about 10 to 12 features per input variable are generated and ranked for selection.


Reject Original Variables Property

- **Reject original variables** — Specifies whether the original input variables are assigned the role **Rejected**. Deselect this property to keep the original variables assigned the role **Input**.

.....
Note: If no features are created from an input variable, that input variable is not assigned the role **Rejected**.
.....

Feature Machine Results

After running the node, open the Results window by right-clicking the node and selecting **Results**.

You can download the data that is used to build the charts and tables in the Results window by clicking . A CSV file with the chart or table data will immediately begin downloading.

In the **Output Data** tab, you can view, save, and explore the data that is generated by the node. For more information, see [Explore and Visualize](#).

Summary

- **Generated Features** — Displays a table of the features that were generated by the node. The description column explains how the features were calculated.
- **Selected Features** — Displays a table of the features that were generated and selected by the node. The description column explains how the features were calculated. This table replaces the Generated Features table when **Feature Selection** is enabled.
- **Properties** — Specifies the various properties that you selected before running the node.
- **Output** — Displays the SAS output of the feature machine run. This displays the list of all features that were generated by the node.

Filtering

<i>Overview of Filtering</i>	75
<i>Filtering Properties</i>	76
Class Variable Filtering	76
Interval Variable Filtering	76
Filter Indicator Property	77
<i>Filtering Results</i>	77

Overview of Filtering

The **Filtering** node is a Data Mining Preprocessing node. Based on data values in the training data set, it is used to filter the training, validation, and test data sets. You can use filters to identify certain observations, such as extreme outliers, rare class values, missing values, and errant data. These observations are treated in one of two ways: they can be excluded when training subsequent data mining models, or the binary variable (M_FILTER) that is used to flag these observations can be used as input for these models. Filtering extreme values from the training data tends to produce better models because the parameter estimates are more stable and clustering results are more reliable. The **Filtering** node ignores target and rejected variables.

Filtering Properties

Class Variable Filtering

- **Class filtering method** — Specifies the method for filtering class variables. Levels that do not exceed the rare value cutoff are excluded. Here are the possible values:
 - (none)**
 - Rare values (count)**
 - Rare values (percentage)**

The default value is **Rare values (percentage)**.

- **Frequency cutoff** — Specifies the maximum frequency for excluding rare values. The default value is 1.
- **Percentage cutoff** — Specifies the maximum percentage for excluding rare values. The default value is 1.
- **Maximum number of levels cutoff** — Specifies the maximum number of class levels cutoff. Only Class variables that have fewer levels than the cutoff value are considered for filtering. The default value is 25.

Note: If **(none)** is selected for the class filtering method, there is no cutoff available.

- **Keep missing class values** — Specifies whether to keep missing levels for class variables.

Interval Variable Filtering

- **Interval filtering limits method** — Specifies the default method for calculating interval variable filtering limits. Here are the possible values:
 - (none)**
 - Extreme percentiles**
 - Median absolute deviation (MAD)**
 - Metadata limits**
 - Standard deviation from the mean**

The default value is **Standard deviation from the mean**.

- **Cutoff for standard deviations** — Specifies the number of standard deviations from the mean to be used as a cutoff value. That is, values that exceed the

specified number of standard deviations away from the mean are filtered out. The default value is 3.

- **Cutoff for median absolute deviations** — Specifies the number of deviations from the median to be used as a cutoff value. That is, values that exceed the specified number of absolute deviations away from the median are filtered out. The default value is 9.
- **Cutoff for extreme percentiles** — Specifies the cutoff percentile for extreme percentiles. The default value is 0.5.
- **Alternate interval filtering limits method** — Specifies the method for calculating interval variable filtering limits where metadata lacks stored limits. If the interval filtering limits method is **Metadata limits**, any limits stored in metadata are used for filtering. Here are the possible values:
 - (none)**
 - Extreme percentiles**
 - Median absolute deviation (MAD)**
 - Standard deviation from the mean**
- **Keep missing interval values** — Specifies whether to keep missing values for interval variables.

Filter Indicator Property

- **Filter indicator usage** — Specifies how the filter indicator is used. The filter indicator is a flag that indicates whether a row of data contains an outlier or rare value. The possible values are **Filter** and **Model input**. For **Model input**, the filter indicator is used as an input in the modeling process. For **Filter**, the filter indicator is applied to the training, validation, and test data to exclude the outlier and rare value rows.


.....

Note: Rows that contain missing values are also flagged in the filtering indicator, as required by the **Keep missing interval values** and **Keep missing class values** properties.

.....

Filtering Results

After running the node, open the Results window by right-clicking the node and selecting **Results**.

You can download the data that is used to build the charts and tables in the Results window by clicking . A CSV file with the chart or table data will immediately begin downloading.

In the **Output Data** tab, you can view, save, and explore the data that is generated by the node. For more information, see [Explore and Visualize](#).

Summary

- **Excluded Class Values** — Displays a table of all the class variable levels that were filtered from the training data set. The table provides data for variable label, variable role, level, train count, train percent, and filter method.
- **Limits for Interval Variables** — Displays a table that shows the upper and lower limits that were established by the filter method for interval variables. The table provides data for variable label, variable role, limits method, lower limit, upper limit, and keep missing values.
- **Number of Observations** — Displays the number of observations for the training data, noting the original amount, the number that were excluded, and the number that were included.
- **Node Score Code** — Displays the SAS score code that was created by the node. The SAS score code can be used outside the Model Studio environment in custom user applications.
- **Properties** — Specifies the various properties that you selected before running the node.
- **Output** — Displays the SAS output of the filtering run.

Forest

Overview of Forest	79
Forest Properties	80
General Properties	80
Tree-Splitting Options	80
Perform Autotuning	81
Lift Calculation Property	84
Binary Classification Cutoff	84
Post Training Properties	85
Forest Results	86

Overview of Forest

The **Forest** node is a Supervised Learning node that creates a predictive model called a forest. A forest consists of several decision trees that differ from each other in two ways. First, the training data for a tree is a sample with replacement from all available observations. Second, the input variables that are considered for splitting a node are randomly selected from all available inputs. Among these randomly selected variables, the input variable that is most often associated with the target is used for the splitting rule. In other respects, trees in a forest are trained like standard trees.

The training data for an individual tree excludes some of the available data. The data that is withheld from training is called the *out-of-bag sample*. The out-of-bag sample is used to assess the fit of the model.

The **Forest** node accepts interval and class target variables. For an interval target, the prediction in a leaf of an individual tree equals the average of the target values among the bagged training observations in that leaf. For a class target, the posterior probability of a target category equals the proportion of that category among the bagged training observations in that leaf. Predictions or posterior probabilities are then averaged across all the trees in the forest. Averaging over trees with different training samples reduces the dependence of the predictions on a particular training sample. Increasing the number of trees does not increase the risk of overfitting the

data and can decrease it. However, if the predictions from different trees are correlated, then increasing the number of trees makes little or no improvement.

Forest Properties

General Properties

- **Number of trees** — Specifies the number of trees in the forest. The default value is 100.
- **Class target voting method** — Specifies the method to use for calculating the predicted probability for a class target. Here are the possible values:
 - **Probability** — Uses the average target level probability across all trees.
 - **Majority** — Uses the proportion of trees in the forest that predicted the level as target.

The default value is **Probability**.

Tree-Splitting Options

- **Class target criterion** — Specifies the splitting criterion to use for determining the best splits on inputs given a class target. Here are the possible values:
 - **CHAID**
 - **Chi-square**
 - **Entropy**
 - **Gini**
 - **Information gain ratio**The default value is **Information gain ratio**.
- **Interval target criterion** — Specifies the splitting criterion to use for determining the best splits on inputs given an interval target. Here are the possible values:
 - **CHAID**
 - **F test**
 - **Variance**The default value is **Variance**.
- **Maximum number of branches** — Specifies the maximum number of branches to consider for a each node split in the tree. Possible values range from 2 to 5. The default value is 2.
- **Maximum depth** — Specifies the maximum depth for each generated tree within the forest. Possible values range from 1 to 50. The default value is 20.

- **Leaf-size specification** — Specifies the method for determining the minimum leaf size. Possible values are **Count** and **Proportion**. The default value is **Count**.
- **Minimum leaf count** — Specifies the smallest number of training observations that a new branch can have. That number is expressed as the count of the available observations. This option is available when **Leaf-size specification** is set to **Count**. Possible values range from 1 to 64. The default value is 5.
- **Minimum leaf proportion** — Specifies the smallest number of training observations that a new branch can have. That number is expressed as a fraction of the training observations with a nonmissing target value. This option is available when **Leaf-size specification** is set to **Proportion**. The default value is 0.00005.
- **Missing values** — Specifies the action to take when there are values missing. You can specify to **Ignore**, **Use as machine smallest**, or **Use in search**. The default value is **Use in search**.
- **Minimum missing use in search** — Specifies a threshold for using missing values in the split search. The default value is 1.
- **Number of interval bins** — Specifies the number of bins that are used for interval inputs. Bin size is (maximum value - minimum value)/(**Number of interval bins**). The default value is 50.
- **Interval bin method** — Specifies the method used to bin the interval input variables. Select **Bucket** to divide input variables into evenly spaced intervals based on the difference between maximum and minimum values. Select **Quantile** to divide input variables into approximately equal sized groups. The default value is **Quantile**.
- **In-bag sample proportion** — Specifies the proportion of training observations to train a tree with—that is, the “in-bag” proportion. The default value is 0.6.
- **Use default number of inputs to consider per split** — Specifies whether to use the default number of inputs to consider per split—that is, the square root of the number of inputs. By default, this option is selected.
- **Number of inputs to consider per split** — Specifies the number of input variables that are randomly sampled to use per split. This option is available only if you deselect **Use default number of inputs to consider per split**. The default value is 100.
- **Number of inputs to subset with Loh’s method** — Specifies the number of input variables to further sample using the Loh’s method. When set to 0, no further sampling of inputs is performed. The default value is 0.
- **Seed** — Specifies the seed for generating random numbers. This option is used to select training observations for each tree and to select candidate variables in each node to split on. The default value is 12345.

Perform Autotuning

This feature specifies whether to perform autotuning of any forest parameters. Warning: Performing autotuning can substantially increase run time. If **Perform Autotuning** is selected, the following options are available:

- **Maximum Depth** — Specifies whether to autotune the maximum depth parameter. The following option is available:
 - **Initial value** — Specifies the initial value for autotuning maximum depth. Possible values range from 1 to 50. The default value is 20. Use the **From** and **To** options to specify the range. The default **From** value is 1, and the default **To** value is 29.
- **Minimum Leaf Size** — Specifies whether to autotune the minimum leaf size parameter. The following option is available:
 - **Initial value** — Specifies the initial value for autotuning the minimum leaf size. The default value is 5. Use the **From** and **To** options to specify the range. The default **From** value is 1, and the default **To** value is 100.
- **Number of Interval Bins** — Specifies whether to autotune the number of interval bins parameter. The following option is available:
 - **Initial value** — Specifies the initial value for autotuning the number of interval bins. The default value is 50. Use the **From** and **To** options to specify the range. The default **From** value is 20, and the default **To** value is 100.
- **Number of Trees** — Specifies whether to autotune the tree number parameter. The following option is available:
 - **Initial value** — Specifies the initial value for autotuning the number of trees. The default value is 100. Use the **From** and **To** options to specify the range. The default **From** value is 20, and the default **To** value is 150.
- **In-bag Sample Proportion** — Specifies whether to autotune the in-bag sample proportion parameter. If this option is selected, the following option is available:
 - **Initial value** — Specifies the initial value for autotuning the in-bag sample proportions. The default value is 0.6. Use the **From** and **To** options to specify the range. The default **From** value is 0.1, and the default **To** value is 0.9.
- **Number of Inputs per Split** — Specifies whether to autotune the number of inputs per split parameter. If this option is selected, the following option is available:
 - **Initial value** — Specifies the initial value for autotuning the number of inputs per split. The default value is 100. Use the **From** and **To** options to specify the range. The default **From** value is 1, and the default **To** value is 100.
- **Search Options** — Specifies the autotune search method and properties. The following options are available:
 - **Search method** — Specifies the tuning search method. Here are the possible values:
 - **Bayesian** — Specifies the Bayesian method. This method uses priors to seed the iterative optimization.
 - **Genetic algorithm** — Specifies the genetic algorithm method. This method uses an initial Latin Hypercube sample that seeds a genetic algorithm to generate a new population of alternative configurations at each iteration.
 - **Grid** — Specifies the grid method. This method performs an exhaustive search of all configurations. Each hyperparameter of interest is discretized into a desired set of values to be studied. Models are trained and assessed for all combinations of values across all the hyperparameters (thus forming a multi-dimensional “grid”).

- **Latin hypercube sample** — Specifies the Latin Hypercube method. This method performs an optimized grid search that is uniform in each tuning parameter, but random in combinations.
- **Random** — Specifies the Random method. This method generates a single sample of purely random configurations.

The default value is **Genetic algorithm**.

- **Number of evaluations per iteration** — Specifies the number of tuning evaluations in one iteration. This option is available only if the **Search method** is **Genetic algorithm** or **Bayesian**. The default value is 10.
- **Maximum number of evaluations** — Specifies the maximum number of tuning evaluations. This option is available only if the **Search method** is **Genetic algorithm** or **Bayesian**. The default value is 50.
- **Maximum number of iterations** — Specifies the maximum number of tuning iterations. This option is available only if the **Search method** is **Genetic algorithm** or **Bayesian**. The default value is 5.
- **Maximum number of points in model** — Specifies the maximum number of points in the model. This option is available only if the **Search method** is **Bayesian**. The default value is 100.
- **Sample size** — Specifies the sample size. This option is available only if the **Search method** is **Random** or **Latin hypercube sample**. The default value is 50.
- **General Options** — Specifies the general properties for autotuning. The following options are available:

- **Validation method** — Specifies the validation method for finding the objective value. Note that if your data is partitioned, then that partition is used. **Validation method**, **Validation data proportion**, and **Cross validation number of folds** are all ignored. Here are the possible values:
 - **Partition** — Specifies using a single partition of a training set. With partition, you specify proportions to use for randomly assigning observations to each role.
 - **K-fold cross validation** — Specifies using the cross validation method. In cross validation, each model evaluation requires k training executions (on $k-1$ data folds) and k scoring executions (on one holdout fold). This increases the evaluation time by approximately a factor of k .

For small to medium data tables, cross validation provides, on average, a better representation of error across the whole data table. **Partition** is the default value.

- **Training data proportion** — Specifies the proportion of training data to be used for the **Partition** validation method. The default value is 0.7.
- **Validation data proportion** — Specifies the proportion of data to be used for the **Partition** validation method. The default value is 0.3.
- **Cross validation number of folds** — Specifies the number of partition folds in the cross validation process (the k defined above). Possible values range from 2 to 20. The default value is 5.
- **Class target objective function** — Specifies the objective function to optimize for tuning parameters for a class target. Here are the possible values:
 - **Area under the curve**

- **Average squared error**
- **F0.5 score**
- **F1 score**
- **Gamma**
- **Gini coefficient**
- **Kolmogorov-Smirnov statistic**
- **Misclassification rate**
- **Multi-class log loss**
- **Root average squared error**
- **Tau**

The default value is **Kolmogorov-Smirnov statistic**.

- **Interval target objective function** — Specifies the objective function to optimize for tuning parameters for an interval target. Here are the possible values:
 - **Average squared error**
 - **Mean absolute error**
 - **Mean squared logarithmic error**
 - **Root average squared error**
 - **Root mean absolute error**
 - **Root mean squared logarithmic error**

The default value is **Average squared error**.

- **Maximum time (minutes)** — Specifies the maximum time (in minutes) for the optimization tuner. The default value is 60.
- **Maximum training time for single model (minutes)** — Specifies the maximum training time (in minutes) for the single model.

Lift Calculation Property

- **Use the exact percentile method for lift calculations** — Specifies whether to use the exact percentile method for calculating lift and related assessment measures. When this property is deselected, the iterative method is used. The exact method should be used if there are convergence errors when using the iterative method.

Binary Classification Cutoff

- **Specify node binary classification cutoff** — Specifies whether to use the binary classification cutoff specified below for the node. If this property is deselected, the project binary classification cutoff is used for determining the

predicted value for a binary target based on the posterior probabilities. By default, this property is deselected.

- **Node binary classification cutoff** — Specifies the cutoff to use in the node for determining the predicted value for a binary target based on the posterior probabilities. This option is available only if **Specify node binary classification cutoff** is selected. The default value is 0.5.

.....

Note: The **Rules** settings in the Project Settings window enable you to specify a new cutoff value by selecting **Override the default classification cutoff**. If **Specify node binary classification cutoff** is deselected, then the value that you specified in the Project Settings window is used. If **Specify node binary classification cutoff** is selected, then the value that you specified in **Node binary classification cutoff** is used.

.....

Post Training Properties

Changing the following properties will not retrain the model.

Model Interpretability — Specifies the method of model interpretation in order to show how the model responds to changing inputs.

- **Global Interpretability**

- **Variable importance** — Specifies whether to display a table of the relative importance values for the input variables. The surrogate variable importance values are calculated using a one-level decision tree for each input to predict the predicted value as a global surrogate model. For tree-based supervised learning models, the model variable importance table is displayed as well.
- **PD plots** — Specifies whether to display partial dependency (PD) plots that show the relationship between model input variables and their predictors. PD plots are generated for only up to the top ten variables in the relative importance table. Here is how to determine the variables that are displayed in the PD plot: for tree-based models (forest, decision tree, and gradient boosting), the model variable importance table is used. For other supervised learning models, the surrogate variable importance table is used.


- **Local Interpretability**

- **ICE plots** — Specifies whether to display Individual Conditional Expectation (ICE) plots. ICE plots can reveal interesting subgroups and interactions between model variables. ICE plots are generated for only up to the top ten variables in the relative importance table. Here is how to determine the variables that are displayed in the ICE plot: for tree-based models (forest, decision tree, and gradient boosting), the model variable importance table is used. For other supervised learning models, the surrogate variable importance table is used.
- **LIME** — Specifies whether to use the Locally Interpretable Model-Agnostic Explanation (LIME) method to explain the model predictions.
- **Kernel SHAP** — Specifies whether to use the Kernel Shapley additive explanations (Kernel SHAP) method to explain the model predictions.

- **Maximum number of Kernel SHAP variables** — Specifies the maximum number of Kernel SHAP variables to plot. Possible values range from 1 to 100. The default value is 20.
- **Specify instances to explain** — Specifies the individual observations that you want to explain. The **Random** observations option provides explanations for five randomly selected observations. The **Specify up to 5** observations option enables you to provide IDs for the specific observations in the data. The IDs should be unique values of the variable that has the role **Key** in the metadata. If no variable is assigned the role **Key**, then you can use the variable `_dmIndex_` to specify the individual observations.
- **LIME/Kernel SHAP Tables**
 - **Explainer Information table** — Specifies whether the Explainer Information tables for the LIME and Kernel SHAP methods should be included in the node results. The Explainer Information table contains information about what settings were used to perform the LIME and Kernel SHAP methods.
 - **Explainer Fidelity table** — Specifies whether the Explainer Fidelity tables for the LIME and Kernel SHAP methods should be included in the node results. The Explainer Fidelity table contains information about the accuracy of the local explanation predictions, compared to the black-box machine learning model.
- **PD/ICE Options**
 - **Maximum number of variables** — Specifies the maximum number of input variables to use to generate the PD and ICE plots. The input variables are selected based on their relative importance. Possible values range from 1 to 10. The default value is 5.
 - **Number of observations** — Specifies the maximum number of observations to sample for the PD and ICE plots. The default value is 1000.
 - **Number of tick points** — Specifies the number of tick points for an interval analysis variable to generate for the PD and ICE plots. Possible values range from 3 to 100. The default value is 50.
 - **Truncate lower tail** — Specifies under what percentage to discard the values of the interval analysis variables for the PD and ICE plots. Possible values range from 0 to 99. The default value is 5%.
 - **Truncate upper tail** — Specifies above what percentage to discard the values of the interval analysis variables for the PD and ICE plots. Possible values range from 0 to 99. The default value is 95%.
- **Seed** — Specifies the seed for generating random numbers. The seed is used for selecting individual observations to explain, selecting observations to generate PD and ICE plots, and simulating data for LIME and Kernel SHAP.



Forest Results

After running the node, open the Results window by right-clicking the node and selecting **Results**.

You can download the data that is used to build the charts and tables in the Results window by clicking . A CSV file with the chart or table data will immediately begin downloading.

In the **Output Data** tab, you can view, save, and explore the data that is generated by the node. For more information, see [Explore and Visualize](#).

Several charts have detailed descriptions available that explain how the charts are generated. The descriptions also provide information about how to interpret the charts that is custom to your particular data and models. To access these detailed descriptions:

- Expand an individual chart by clicking . The detailed description is in the right panel.
- Select  on an individual chart. A window appears with the detailed description.

Note: The detailed descriptions are available only in the English language.

Node

- **Error Plot** — Displays the error statistics for the forest model, giving the average squared error as a function of the number of trees. The average squared error is given for each of the data roles, as well as for the out-of-bag data. To examine the misclassification rate as a function of the number of trees when you have a class target, use the drop-down menu in the upper right corner.
- **Variable Importance** — Displays a table of importance data for each variable. These statistics include train importance, importance standard deviation, and relative importance.
- **Autotune Best Configuration** — Displays a table of the resulting values of parameters that were autotuned. This table is not displayed if autotuning is not selected.
- **Autotune Results** — Displays the iterative results of the autotuning process, showing how each of the parameter values evolve as the model iterates. This table is not displayed if autotuning is not selected.
- **Path EP Score Code** — Displays the SAS code that was created by the node. The score code can be used outside of the Model Studio environment to score new data.
- **DS2 Package Code** — Displays the SAS DS2 package code that was created by the node. The score code can be used outside of the Model Studio environment to score new data. DS2 package code is used when publishing models to SAS Micro Analytic Service.
- **Score Inputs** — Displays a table of data on the input variables for scoring calculations. The table includes the variables' name, role, level, type, label, format, and length.
- **Score Outputs** — Displays a table of data on the predicted response variable for scoring calculations. The table includes the variables' name, role, type, format and length, as well as the creator, creator GUID, and function.
- **Training Code** — Displays the SAS code that Model Studio used to train the node.
- **Properties** — Specifies the various properties that you selected before running the node.

- **Output** — Displays the SAS output of the Forest run. For the **Forest** node, the output includes model information, number of observations read and used, variable importance, fit statistics, predicted probability variables, and predicted target variable.

Assessment

- **Predicted Reports** — Displays the predicted and target mean as a function of the depth for the model. The predicted and target mean are given for each of the data roles. To examine the predicted mean as a function of the target mean, use the drop-down menu in the upper right corner. This result is displayed only if the target is an interval variable.
- **Lift Reports** — Displays the cumulative lift as a function of depth for the model. The cumulative lift is given for each of the data roles. To examine other statistics as a function of depth, use the drop-down menu in the upper right corner. Other statistics include lift, gain, captured response percentage, cumulative captured response percentage, response percentage, and cumulative response percentage. This result is displayed only if the target is a class variable.
- **ROC Reports** — Displays the ROC (receiver operating characteristic) chart for a model, giving the sensitivity as a function of 1-specificity. The sensitivity is given for each of the data roles. To examine other statistics, use the drop-down menu in the upper right corner. Other statistics include accuracy and F1 score. This result is displayed only if the target is a class variable.
- **Fit Statistics** — Displays the fit statistics for the model, broken down by data role.
- **Event Classification** — Displays the confusion matrix at various cutoff values for each partition. The confusion matrix contains true positives (events that are correctly classified as events), false positives (non-events that are classified as events), false negatives (events that are classified as non-events), and true negatives (non-events that are correctly classified as non-events). The classification cutoffs that are used are the default value of 0.5, the Kolmogorov-Smirnov cutoff value for each partition, and the **Node binary classification cutoff** that is specified for binary targets. Use the drop-down menu to view the information in the chart as percentages or counts. You can also view the information summarized in a table. This result is displayed only if the target is a class variable.
- **Nominal Classification** — Displays either the percentage of observations or the number of observations that predict each level of the target variable. The plot is segmented by target level and partition level. The target level with the greatest predicted probability is the predicted outcome. This result is displayed only if the target is a nominal variable.

Model Interpretability

- **Surrogate Model Variable Importance** — Displays a table of the relative importance of the input variables, including variable level and label. Relative importance is calculated using a one-level decision tree for each input variable to predict the predicted value as a global surrogate model.
- **Model Variable Importance** — Displays a table of the relative importance of the input variables, including variable level and label. This is the variable importance table that is generated when training the original forest model.
- **PD Plot** — Displays the functional relationship between the input variable and the model prediction. For interval input variables, the 95% confidence interval for the average target prediction is indicated by the shaded band around the line. The X axis includes a heat map that shows the distribution of the input variable.

The extreme values of the input variables are removed by truncating the lower and upper tails of its distribution. The amount of truncation can be controlled by specifying values for the **Truncate lower tail** and the **Truncate upper tail** properties.

- **PD and ICE Overlay Plot** — Displays the functional relationship between the input variable and the model prediction as well as the functional relationship between individual observations and the model prediction. For each individual observation, the corresponding ICE curve displays values of the input variable on the X axis and the corresponding predicted probability or prediction of the target variable on the Y axis, holding the other input variable values constant at that observation. The X axis includes a heat map that shows the distribution of the input variable. The extreme values of the input variables are removed by truncating the lower and upper tails of its distribution. The amount of truncation can be controlled by specifying values for the **Truncate lower tail** and the **Truncate upper tail** properties.
- **LIME Explanations** — The LIME algorithm calculates a more easily interpretable linear model around an individual observation. The chart displays the regression estimates for the input variables that are selected in the local surrogate linear regression model. The input variables are ordered by significance such that the most important variable for the local regression model is at the bottom of the chart. The LASSO technique is used to select the most significant predictors from the set of input variables that are used to train the model. Each nominal input variable is binary-encoded based on whether it matches the level of the individual observation. A positive estimate indicates that the observed value of the input variable increases the predicted probability of the event.
- **Kernel SHAP Values** — For each individual observation, an input variable's Kernel SHAP value is the contribution of the observed value of the input variable to the predicted probability or prediction of the target variable. The Kernel SHAP values are the regression coefficients that are obtained by fitting a weighted least squares regression. The Kernel SHAP values of all input variables sum to one. The input variables are ordered by significance such that the most important variable according to the absolute Kernel SHAP values is at the top of the chart. Each nominal input is binary-encoded based on whether it matches the individual observation. Interval inputs are binary encoded based on their proximity to the individual observation.
- **LIME Information Table** — Displays information about the settings used to perform the LIME method.
- **Kernel SHAP Information Table** — Displays information about the settings used to perform the Kernel SHAP method.
- **LIME Fidelity Table** — Displays information about the accuracy of the local explanation predictions that are calculated using the LIME method, compared to the black-box machine learning model. The model prediction, the explainer prediction, and the explainer root mean square error (RMSE) are displayed.
- **Kernel SHAP Fidelity Table** — Displays information about the accuracy of the local explanation predictions that are calculated using the Kernel SHAP method, compared to the black-box machine learning model. The model prediction, the explainer prediction, and the explainer root mean square error (RMSE) are displayed.

GLM

Overview of GLM	91
GLM Properties	92
General Properties	92
Effects Options	93
Selection Method Property	94
Selection Options	94
Optimization Options	96
Convergence Options	98
Tweedie Model Options	98
Relativity Plot Properties	99
Lift Calculation Property	99
Post Training Properties	99
GLM Results	101

Overview of GLM

The **GLM** node is a Supervised Learning node that fits a generalized linear model (GLM) for an interval target with a specified target distribution and link function. A GLM is an extension of the traditional linear model. This extension allows the population mean to depend on a linear predictor through a nonlinear link function. For example, GLMs can be used to model traditional insurance measures such as claim frequency, severity, or pure premium. Claim frequency is typically modeled with a Poisson distribution and a logarithmic link function. Claim severity is typically modeled with a gamma distribution and a logarithmic link function. Pure premiums are modeled with the Tweedie distribution.

The **GLM** node can fit models for standard distributions in the exponential family, as well as the beta and generalized Poisson distributions.

The **GLM** node supports offset variables. You must meet the following conditions to use an offset variable:

- One variable must be assigned the role **Offset**.

- Your target variable must be interval.

GLM Properties

General Properties

- **Target probability distribution** — Specifies the probability distribution to use in the model for the target. Here are the possible values:

- Beta**
- Binary**
- Exponential**
- Gamma**
- Generalized Poisson**
- Geometric**
- Inverse Gaussian**
- Negative binomial**
- Normal**
- Poisson**
- Tweedie**

The default value is **Poisson**.

- **Link function** — Specifies the link function to link the response mean to the linear predictor. Here are the possible values:

- Complementary log-log**
- Identity**
- Inverse**
- Inverse squared**
- Log**
- Log-log**
- Logit**
- Probit**

The default value is **Log**.

Effects Options

- **Class input order** — Specifies the sort order for the class inputs. For the GLM coding that is used, the last level according to the sorted order is used as the reference level. Here are the possible values:
 - **Formatted**
 - **Formatted descending**
 - **Frequency**
 - **Frequency descending**
 - **Unformatted**
 - **Unformatted descending**The default value is **Frequency**.
- **Two-factor interactions** — Specifies whether to include all two-factor interactions of class variables in the model. By default, **Two-factor interactions** is deselected.
- **Factor split** — Specifies whether levels of a factor can enter or leave a model independently. This is done automatically for LASSO selection. By default, **Factor split** is deselected.
- **Spline** — Specifies whether to expand interval input variables into cubic B-spline bases with three equally spaced knots. (This action yields seven design matrix columns for each of the variables.) If **Spline** is selected, **Polynomial** is unavailable. By default, **Spline** is deselected.
- **Spline split** — Specifies whether each individual column in the design matrix that corresponds to the spline effect is treated as a separate effect that can enter or leave the model. This option is available only if **Spline** is selected. By default, **Spline split** is deselected.
- **Polynomial** — Specifies whether to use model polynomial effects up to the degree specified for all interval variables. If **Polynomial** is selected, **Spline** is unavailable. By default, **Polynomial** is deselected.
- **Polynomial degree** — Specifies the polynomial degree when polynomial terms are included in the model. Possible values are 2 and 3. The default value is 2.
- **Suppress intercept** — Specifies whether the intercept should be suppressed. By default, **Suppress intercept** is deselected.
- **Model missing values for inputs** — Specifies whether to model missing values for inputs by including indicator values in the model for missing values for interval inputs and imputing values in the original effect with the mean as well as treating missing values for class input variables as valid levels. By default, **Model missing values for inputs** is deselected.
- **Use missing as level** — Specifies whether missing values should be treated as a level for class inputs. By default, **Use missing as level** is deselected.
- **Standardize interval inputs** — Specifies whether interval inputs should be standardized (centered and scaled) when used as main effects. By default, **Standardize interval inputs** is deselected.

Selection Method Property

- **Selection method** — Specifies a model selection method. Here are the possible values:
 - **(none)** — Specifies that there is no model selection method.
 - **Backward** — Specifies that training is done by starting with all candidate effects in the model and then removing effects until the Stay significance level or the stop criterion is met.
 - **Fast backward** — Specifies that training is done by starting with all effects in the model and then deleting effects without refitting the model.
 - **Forward** — Specifies that training is done by starting with no candidate effects in the model and then adding effects until the Entry significance level or the stop criterion is met.
 - **LASSO** — Specifies that training is done using the group LASSO method, which adds and removes effects by a sequence of LASSO steps.
 - **Stepwise** — Specifies that training is done as in the forward model but that it might remove effects already in the model.

The default value is **Stepwise**.

Selection Options

- **Effect-selection criterion** — Specifies the criterion that the procedure uses to determine the order in which effects enter or leave at each step of the selection method. Here are the possible values:
 - **AIC** — Specifies Akaike's Information Criterion. Smaller values indicate better models, and AIC values can become negative. AIC is based on the Kullback-Leibler information measure of discrepancy between the true distribution of the response variable and the distribution specified by the model.
 - **AICC** — Specifies Corrected Akaike's Information Criterion. This version of AIC adjusts the value to account for sample size. The result is that extra effects penalize AICC more than AIC. As the sample size increases, AICC and AIC converge.
 - **SBC (BIC)** — Specifies Schwarz's Bayesian Criterion (SBC), also known as the Bayesian Information Criterion (BIC). SBC is an increasing function of the model's residual sum of squares and the number of effects. Unexplained variations in the response variable and the number of effects increase the value of the SBC. As a result, a lower SBC implies either fewer explanatory variables, better fit, or both. SBC penalizes free parameters more strongly than AIC.
 - **Significance level** — Specifies the standard statistical significance level criterion.

If the **Selection method** above is **LASSO**, this option is unavailable. The default value is **SBC (BIC)**.

- **Selection-process stopping criterion** — Specifies the criterion to stop the selection process. Here are the possible values:
 - **AIC** — Specifies Akaike's Information Criterion. Smaller values indicate better models, and AIC values can become negative. AIC is based on the Kullback-Leibler information measure of discrepancy between the true distribution of the response variable and the distribution specified by the model.
 - **AICC** — Specifies Corrected Akaike's Information Criterion. This version of AIC adjusts the value to account for sample size. The result is that extra effects penalize AICC more than AIC. As the sample size increases, AICC and AIC converge.
 - **SBC (BIC)** — Specifies Schwarz's Bayesian Criterion (SBC), also known as the Bayesian Information Criterion (BIC). SBC is an increasing function of the model's residual sum of squares and the number of effects. Unexplained variations in the response variable and the number of effects increase the value of the SBC. As a result, a lower SBC implies either fewer explanatory variables, better fit, or both. SBC penalizes free parameters more strongly than AIC.
 - **Significance level** — Specifies the standard statistical significance level criterion.
 - **Validation ASE** — Average square error (ASE) of the model is computed using the validation data, and selection stops at the step where the ASE starts to increase. This method requires partitioned data.

The default value is **SBC (BIC)**.

- **Model-selection criterion** — Specifies the criterion to choose the model from the list of models at each step of the selection process that yields the best value of the specified criterion. If the optimal value of the specified criterion occurs for models at more than one step, then the model that has the smallest number of parameters is chosen. Here are the possible values:
 - **AIC** — Specifies Akaike's Information Criterion. Smaller values indicate better models, and AIC values can become negative. AIC is based on the Kullback-Leibler information measure of discrepancy between the true distribution of the response variable and the distribution specified by the model. The model that has the minimal AIC value is chosen.
 - **AICC** — Specifies Corrected Akaike's Information Criterion. This version of AIC adjusts the value to account for sample size. The result is that extra effects penalize AICC more than AIC. As the sample size increases, AICC and AIC converge.
 - **SBC (BIC)** — Specifies Schwarz's Bayesian Criterion (SBC), also known as the Bayesian Information Criterion (BIC). SBC is an increasing function of the model's residual sum of squares and the number of effects. Unexplained variations in the response variable and the number of effects increase the value of the SBC. As a result, a lower SBC implies either fewer explanatory variables, better fit, or both. SBC penalizes free parameters more strongly than AIC. The model that has the minimal SBC value is chosen.
 - **Validation ASE** — Average square error (ASE) of the model is computed using the validation data, and the model that has the minimal ASE is chosen. This method requires partitioned data.

The default value is **SBC (BIC)**.

- **Entry significance level** — Specifies the significance level for adding variables in the forward and stepwise directions. The default value is 0.05.
- **Stay significance level** — Specifies the significance level for removing variables in backward and stepwise directions. The default value is 0.05.
- **Maximum number of effects** — Specifies the maximum number of effects to be considered in any model during the selection process. If a model at some step of the selection process contains the specified maximum number of effects, then no additional effects are considered. If **Maximum number of effects** is set to 0 (the default value), this option is ignored.
- **Minimum number of effects** — Specifies the minimum number of effects to be considered in any model during the backward selection process. If **Minimum number of effects** is set to 0 (the default value), this option is ignored.
- **Maximum number of steps** — Specifies the maximum number of selection steps that are performed. If **Maximum number of steps** is set to 0 (the default value), this option is ignored.
- **Hierarchy** — Specifies whether and how the model hierarchy requirement is applied. Model hierarchy refers to the requirement that, for any term to be in the model, all model effects that are contained in the term must be present in the model. For example, in order for the interaction A*B to enter the model, the main effects A and B must also be in the model. Likewise, neither effect A nor effect B can leave the model while the interaction A*B is in the model. Here are the possible values:
 - **(none)** — Specifies that the hierarchy requirement is never applied.
 - **All variables** — Specifies that both interval and class variables are subject to the model hierarchy requirement.
 - **Class variables** — Specifies that only class variables are subject to the model hierarchy requirement.

The default value is **(none)**.

Optimization Options

- **Optimization technique** — Specifies the optimization method used when fitting a model. Here are the possible values:
 - **(none)**
 - **Conjugate-gradient**
 - **Double-dogleg**
 - **Dual quasi-Newton**
 - **Nelder-Mead simplex**
 - **Newton-Raphson**
 - **Newton-Raphson with ridging**
 - **Trust-region**

The default value is **Newton-Raphson with ridging**. For more information, see the PROC NL MIXED documentation in *SAS/STAT 15.1 User's Guide*.

Note: Optimization options are not available for the **LASSO** selection method. LASSO uses the Nesterov algorithm as the default optimization technique.

- **Maximum number of iterations** — Specifies the maximum number of iterations of any optimization. The default value depends on the optimization technique used:

Optimization Technique	Default Value
Conjugate-gradient	400
Double-dogleg	200
Dual quasi-Newton	200
Nelder-Mead simplex	1000
Newton-Raphson	50
Newton-Raphson with ridging	50
Trust-region	50

To use the default value, leave **Maximum number of iterations** blank or use a dot.

- **Maximum number of function evaluations** — Specifies the maximum number of function calls of any optimization. The default value depends on the optimization technique used:

Optimization Technique	Default Value
Conjugate-gradient	1000
Double-dogleg	500
Dual quasi-Newton	500
Nelder-Mead simplex	3000
Newton-Raphson	125
Newton-Raphson with ridging	125
Trust-region	125

To use the default value, leave **Maximum number of function evaluations** blank or use a dot.

- **Maximum CPU time in second** — Specifies an upper limit of CPU time (in seconds) for the optimization process. The default value is the largest floating-point double representation of your computer. To use the default value, do not enter a value for **Maximum CPU time in second**, or enter a dot.
- **Minimum number of iterations** — Specifies the minimum number of iterations in any optimization. The default value is 1. To use the default value, do not enter a value for **Minimum number of iterations** or enter a dot.
- **Normalize objective function** — Specifies whether the objective function should be normalized during optimization by the reciprocal of the used frequency count. By default, this option is selected.

Convergence Options

- **Absolute function convergence** — Specifies the threshold for absolute function convergence. The default value is the negative square root of the largest double-precision value. To use the default value, do not enter a value for **Absolute function convergence** or enter a dot.
- **Absolute function difference convergence** — Specifies the threshold for absolute function difference convergence. The default value is 0. To use the default value, do not enter a value for **Absolute function difference convergence** or enter a dot.
- **Absolute gradient convergence** — Specifies the threshold for absolute gradient convergence. The default value is 1E-5. To use the default value, do not enter a value for **Absolute gradient convergence** or enter a dot.
- **Relative function difference convergence** — Specifies the relative function difference convergence. The default value is the machine precision times 2. To use the default value, do not enter a value for **Relative function difference convergence** or enter a dot.
- **Relative gradient convergence** — Specifies the threshold for relative gradient convergence. The default value is 1E-8. To use the default value, leave **Relative gradient convergence** blank or use a dot.

Tweedie Model Options

These configurations are available only when the **Target probability distribution** is **Tweedie**.

- **Optimization method** — Specifies the optimization method for iterative estimation of Tweedie model parameters. Here are the possible values:
 - **EQL/Likelihood** — Specifies using EQL for a sample of the data, followed by Tweedie log likelihood for the full data.
 - **Extended quasi-likelihood** — Specifies using extended quasi-likelihood (EQL) for a sample of the data, followed by EQL for the full data.

- **Final likelihood** — Specifies using a four-stage approach of EQL and likelihood.
- **Likelihood** — Specifies using Tweedie log likelihood for a sample of the data, and then specifies again for the full data.

The default value is **EQL/Likelihood**.

- **Use fixed power parameter** — Specifies whether to use a fixed power parameter. Otherwise, iterative estimation is used for the Tweedie power parameter with the specified starting value. By default, **Use fixed power parameter** is selected.
- **Fixed power parameter value** — Specifies the value of the Tweedie power parameter when it is fixed. The default value is 1.5.
- **Starting power parameter value** — Specifies the starting value for the iterative estimation of the Tweedie power parameter. This option is available only if **Use fixed power parameter** is deselected. The default value is 1.5.
- **Sample fraction for Tweedie starting values** — Specifies the fraction of training data to use to compute the starting values for the Tweedie distribution. The default value is 1.

Relativity Plot Properties

- **Create relativity plots** — Specifies whether relativity plots are created for each class main effect that is included in the final model. By default, this option is deselected.
- **Confidence level** — Specifies the confidence level as a percentage of confidence intervals used in relativity plots. This option is available only if **Create relativity plots** is selected. The default value is 95.

Lift Calculation Property

- **Use the exact percentile method for lift calculations** — Specifies whether to use the exact percentile method for calculating lift and related assessment measures. When this property is deselected, the iterative method is used. The exact method should be used if there are convergence errors when using the iterative method.

Post Training Properties

Changing the following properties will not retrain the model.

Model Interpretability — Specifies the method of model interpretation in order to show how the model responds to changing inputs.

- **Global Interpretability**


- **Variable importance** — Specifies whether to display a table of the relative importance values for the input variables. The surrogate variable importance values are calculated using a one-level decision tree for each input to predict the predicted value as a global surrogate model. For tree-based supervised learning models, the model variable importance table is displayed as well.
 - **PD plots** — Specifies whether to display partial dependency (PD) plots that show the relationship between model input variables and their predictors. PD plots are generated for only up to the top ten variables in the relative importance table. Here is how to determine the variables that are displayed in the PD plot: for tree-based models (forest, decision tree, and gradient boosting), the model variable importance table is used. For other supervised learning models, the surrogate variable importance table is used.
- **Local Interpretability**
 - **ICE plots** — Specifies whether to display Individual Conditional Expectation (ICE) plots. ICE plots can reveal interesting subgroups and interactions between model variables. ICE plots are generated for only up to the top ten variables in the relative importance table. Here is how to determine the variables that are displayed in the ICE plot: for tree-based models (forest, decision tree, and gradient boosting), the model variable importance table is used. For other supervised learning models, the surrogate variable importance table is used.
 - **LIME** — Specifies whether to use the Locally Interpretable Model-Agnostic Explanation (LIME) method to explain the model predictions.
 - **Kernel SHAP** — Specifies whether to use the Kernel Shapley additive explanations (Kernel SHAP) method to explain the model predictions.
 - **Maximum number of Kernel SHAP variables** — Specifies the maximum number of Kernel SHAP variables to plot. Possible values range from 1 to 100. The default value is 20.
 - **Specify instances to explain** — Specifies the individual observations that you want to explain. The **Random** observations option provides explanations for five randomly selected observations. The **Specify up to 5** observations option enables you to provide IDs for the specific observations in the data. The IDs should be unique values of the variable that has the role **Key** in the metadata. If no variable is assigned the role **Key**, then you can use the variable `_dmIndex_` to specify the individual observations.
 - **LIME/Kernel SHAP Tables**
 - **Explainer Information table** — Specifies whether the Explainer Information tables for the LIME and Kernel SHAP methods should be included in the node results. The Explainer Information table contains information about what settings were used to perform the LIME and Kernel SHAP methods.
 - **Explainer Fidelity table** — Specifies whether the Explainer Fidelity tables for the LIME and Kernel SHAP methods should be included in the node results. The Explainer Fidelity table contains information about the accuracy of the local explanation predictions, compared to the black-box machine learning model.
 - **PD/ICE Options**
 - **Maximum number of variables** — Specifies the maximum number of input variables to use to generate the PD and ICE plots. The input variables are

selected based on their relative importance. Possible values range from 1 to 10. The default value is 5.

- **Number of observations** — Specifies the maximum number of observations to sample for the PD and ICE plots. The default value is 1000.
- **Number of tick points** — Specifies the number of tick points for an interval analysis variable to generate for the PD and ICE plots. Possible values range from 3 to 100. The default value is 50.
- **Truncate lower tail** — Specifies under what percentage to discard the values of the interval analysis variables for the PD and ICE plots. Possible values range from 0 to 99. The default value is 5%.
- **Truncate upper tail** — Specifies above what percentage to discard the values of the interval analysis variables for the PD and ICE plots. Possible values range from 0 to 99. The default value is 95%.
- **Seed** — Specifies the seed for generating random numbers. The seed is used for selecting individual observations to explain, selecting observations to generate PD and ICE plots, and simulating data for LIME and Kernel SHAP.



GLM Results

After running the node, open the Results window by right-clicking the node and selecting **Results**.

You can download the data that is used to build the charts and tables in the Results window by clicking . A CSV file with the chart or table data will immediately begin downloading.

In the **Output Data** tab, you can view, save, and explore the data that is generated by the node. For more information, see [Explore and Visualize](#).

Several charts have detailed descriptions available that explain how the charts are generated. The descriptions also provide information about how to interpret the charts that is custom to your particular data and models. To access these detailed descriptions:

- Expand an individual chart by clicking . The detailed description is in the right panel.
- Select  on an individual chart. A window appears with the detailed description.

Note: The detailed descriptions are available only in the English language.

Node

- **Relativity Plots** — Displays a relativity plot for each class main effect included in the final model. This plot shows the relativity for each level with regard to the reference level—that is, the exponential of the parameter estimate. The bands on the plots indicate the lower and upper confidence limits, which are based on the specified **Confidence level**. You can select which variable to display in the plot using the drop-down menu in the upper right corner. This result is displayed only if **Create relativity plots** is selected.

- **t Values by Parameter** — Displays a bar chart of the t values associated with each of the input parameters, as well as the intercept. Positive t values are denoted with blue bars, and negative t values are denoted with orange bars.
- **Parameter Estimates** — Displays a table of various statistics related to estimates for the parameters. These statistics include the t value, sign, estimate, absolute estimate, the p-value, chi-square value, and standard error.
- **Selection Summary** — Displays a table of the iterations, adding in and removing effects. This table also includes the SBC and optimal SBC.
- **GLM Fit Statistics** — Displays a table of data on the GLM fit statistics. The table includes the statistic, a corresponding qualitative description, and the training, validation, and testing values.
- **Node Score Code** — Displays the SAS score code that was created by the node. The SAS score code can be used outside the Model Studio environment to score new data.
- **Path Score Code** — Displays the SAS score code that was created by the node, as well as all preceding nodes. The SAS model score code can be used outside the Model Studio environment to score new data.
- **DS2 Package Code** — Displays the SAS DS2 package code that was created by the node. The SAS score code can be used outside the Model Studio environment in custom user applications. DS2 package code is used when publishing models to SAS Micro Analytic Service.
- **Score Inputs** — Displays a table of data on the input variables for scoring calculations. The table includes the variables' name, role, level, type, label, format, and length.
- **Score Outputs** — Displays a table of data on the predicted response variable for scoring calculations. The table includes the variables' name, role, type, format and length, as well as the creator, creator GUID, and function.
- **Training Code** — Displays the SAS code that Model Studio used to train the node.
- **Properties** — Specifies the various properties that you selected before running the node.
- **Output** — Displays the SAS output of the GLM run.

Assessment

- **Predicted Reports** — Displays the predicted and target mean as a function of the depth for the model. The predicted and target mean are given for each of the data roles. To examine the predicted mean as a function of the target mean, use the drop-down menu in the upper right corner.
- **Fit Statistics** — Displays the fit statistics for the model, broken down by data role.

Model Interpretability

- **Surrogate Model Variable Importance** — Displays a table of the relative importance of the input variables, including variable level and label. Relative importance is calculated using a one-level decision tree for each input variable to predict the predicted value as a global surrogate model.
- **PD Plot** — Displays the functional relationship between the input variable and the model prediction. For interval input variables, the 95% confidence interval for the average target prediction is indicated by the shaded band around the line. The X axis includes a heat map that shows the distribution of the input variable.

The extreme values of the input variables are removed by truncating the lower and upper tails of its distribution. The amount of truncation can be controlled by specifying values for the **Truncate lower tail** and the **Truncate upper tail** properties.

- **PD and ICE Overlay Plot** — Displays the functional relationship between the input variable and the model prediction as well as the functional relationship between individual observations and the model prediction. For each individual observation, the corresponding ICE curve displays values of the input variable on the X axis and the corresponding predicted probability or prediction of the target variable on the Y axis, holding the other input variable values constant at that observation. The X axis includes a heat map that shows the distribution of the input variable. The extreme values of the input variables are removed by truncating the lower and upper tails of its distribution. The amount of truncation can be controlled by specifying values for the **Truncate lower tail** and the **Truncate upper tail** properties.
- **LIME Explanations** — The LIME algorithm calculates a more easily interpretable linear model around an individual observation. The chart displays the regression estimates for the input variables that are selected in the local surrogate linear regression model. The input variables are ordered by significance such that the most important variable for the local regression model is at the bottom of the chart. The LASSO technique is used to select the most significant predictors from the set of input variables that are used to train the model. Each nominal input variable is binary-encoded based on whether it matches the level of the individual observation. A positive estimate indicates that the observed value of the input variable increases the predicted probability of the event.
- **Kernel SHAP Values** — For each individual observation, an input variable's Kernel SHAP value is the contribution of the observed value of the input variable to the predicted probability or prediction of the target variable. The Kernel SHAP values are the regression coefficients that are obtained by fitting a weighted least squares regression. The Kernel SHAP values of all input variables sum to one. The input variables are ordered by significance such that the most important variable according to the absolute Kernel SHAP values is at the top of the chart. Each nominal input is binary-encoded based on whether it matches the individual observation. Interval inputs are binary encoded based on their proximity to the individual observation.
- **LIME Information Table** — Displays information about the settings used to perform the LIME method.
- **Kernel SHAP Information Table** — Displays information about the settings used to perform the Kernel SHAP method.
- **LIME Fidelity Table** — Displays information about the accuracy of the local explanation predictions that are calculated using the LIME method, compared to the black-box machine learning model. The model prediction, the explainer prediction, and the explainer root mean square error (RMSE) are displayed.
- **Kernel SHAP Fidelity Table** — Displays information about the accuracy of the local explanation predictions that are calculated using the Kernel SHAP method, compared to the black-box machine learning model. The model prediction, the explainer prediction, and the explainer root mean square error (RMSE) are displayed.

Gradient Boosting

<i>Overview of Gradient Boosting</i>	105
<i>Gradient Boosting Properties</i>	106
Basic Options	106
Tree-Splitting Options	106
Perform Early Stopping	107
Perform Autotuning	108
Lift Calculation Property	111
Binary Classification Cutoff	111
Post Training Properties	112
<i>Gradient Boosting Results</i>	113

Overview of Gradient Boosting

The **Gradient Boosting** node is a Supervised Learning node. Gradient boosting is an iterative approach that creates multiple trees where, typically, each tree is based on an independent sample without replacement of the data. The gradient boosting model hones its predictions by minimizing a specified loss function, such as average square error. The first step creates a baseline tree. Each subsequent tree is fit to the residuals of the previous tree, and the loss function is minimized. This process is repeated a specific number of times. The final model is a single function, which is an aggregation of the series of trees that can be used to predict the target value of a new observation.

The term “stochastic gradient boosting” refers to training each new tree based on a subsample of the data. This typically results in a better model. For gradient boosting models, each new observation is fed through a sequence of trees that are created to predict the target value of each new observation.

The **Gradient Boosting** node supports early stopping to avoid overtraining. It works by monitoring the performance of the model that is being trained on a validation partition of the data. When the performance on the validation data no longer improves after a fixed number of training iterations (that is, performance has stagnated), the training procedure stops.

The **Gradient Boosting** node supports offset variables. You must meet the following conditions to use an offset variable:

- One variable must be assigned the role **Offset**.
- Your target variable must be interval.
- The **Interval target distribution** property must be Tweedie or Poisson.

Gradient Boosting Properties

Basic Options

- **Number of trees** — Specifies the number of iterations in the boosting series. For interval and binary targets, the number of iterations equals the number of trees. For a nominal target, a separate tree is created for each target category in each iteration. The default value is 100.
- **Learning rate** — Specifies the step size for gradient descent optimization. The default value is 0.1.
- **Subsample rate** — Specifies the proportion of training observations to train a tree with. A different training sample is taken in each iteration. Trees trained in the same iteration have the same training data. The default value is 0.5.
- **L1 regularization** — Specifies the L1 regularization parameter. The default value is 0.
- **L2 regularization** — Specifies the L2 regularization parameter. The default value is 1.
- **Interval target distribution** — Specifies the distribution of the objective function for an interval target. Possible values are **Normal**, **Poisson**, and **Tweedie**. For a binary or nominal target, the distribution is binary or multinomial, respectively.
- **Tweedie power parameter** — Specifies the power parameter to use when **Tweedie** is selected as the **Interval target distribution**.
- **Seed** — Specifies the seed for generating random numbers. The subsample rate property uses this value to select a training sample at each iteration. The default value is 12345.

Tree-Splitting Options

- **Maximum number of branches** — Specifies the maximum number of branches to consider for a tree node split in the tree. Possible values range from 2 to 5. The default value is 2.
- **Maximum depth** — Specifies the maximum number of generations of nodes. The original node, generation 0, is called the root node. The children of the root

node are the first generation. Possible values range from 1 to 50. The default value is 4.

- **Minimum leaf size** — Specifies the smallest number of training observations that a leaf can have. The default value is 5.
- **Missing values** — Specifies the action to take when there are values missing. You can specify to **Ignore**, **Use as machine smallest**, or **Use in search**. The default value is **Use in search**.
- **Minimum missing use in search** — Specifies a threshold for using missing values in the split search. The default value is 1.
- **Number of interval bins** — Specifies the number of bins in which to bin the interval input variables. Bin size is $(\text{maximum value} - \text{minimum value}) / (\text{Number of interval bins})$. The default value is 50.
- **Interval bin method** — Specifies the method used to bin the interval input variables. Select **Bucket** to divide input variables into evenly spaced intervals based on the difference between maximum and minimum values. Select **Quantile** to divide input variables into approximately equal sized groups. The default value is **Quantile**.
- **Use default number of inputs to consider per split** — Specifies whether to use the default number of inputs to consider per split. By default, this option is selected.
- **Number of inputs to consider per split** — Specifies the number of input variables randomly sampled to use per split. This option is not available if **Use default number of inputs to consider per split** is selected. The default value is 100.

Perform Early Stopping

This feature specifies whether to stop training when the model begins to overfit. Early stopping cannot be used if there is no validation partition. By default, **Perform Early Stopping** is selected. The following options are available:

- **Class target metric** — Specifies the error metric to use for a class target. Possible values are **Log loss** and **Misclassification rate**. Average squared error is used for an interval target.
- **Early stopping method** — Specifies the method to use for early stopping. The **Stagnation** method stops when the relative change in the validation error in consecutive iterations satisfies criteria based on the **Stagnation**, **Tolerance**, and **Start from minimum error** properties. The **Threshold** method stops when the validation error exceeds a threshold based on the **Threshold** and **Threshold iterations** properties.
- **Stagnation** — Specifies the number of consecutive iterations (N) to consider when using the **Stagnation** early stopping method. The default value is 5.
- **Tolerance** — Specifies the threshold for the relative change in validation error when using the **Stagnation** early stopping method. The default value is 0.
- **Start from minimum error** — Specifies whether to count iterations starting from the iteration that has the smallest validation error when using the **Stagnation** early stopping method.

- **Threshold** — Specifies the threshold value when using the **Threshold** early stopping method. Training is stopped when the validation error equals or exceeds this value.
- **Threshold iterations** — Specifies the minimum number of training iterations to run before the validation error is compared to the specified **Threshold**.

Note: When early stopping is enabled and a validation partition exists, autotuning **Number of Trees** is not performed if selected.

Perform Autotuning

This feature specifies whether to perform autotuning of any gradient boosting parameters. Warning: Performing autotuning can substantially increase run time. If **Perform Autotuning** is selected, the following options are available:

- **L1 Regularization** — Specifies whether to autotune L1 Regularization. If this option is selected, the following option is available:
 - **Initial value** — Specifies the initial value for autotuning the L1 Regularization. The default value is 0. Use the **From** and **To** options to specify the range. The default **From** value is 0, and the default **To** value is 10.
- **L2 Regularization** — Specifies whether to autotune L2 Regularization. If this option is selected, the following option is available:
 - **Initial value** — Specifies the initial value for autotuning the L2 Regularization. The default value is 1. Use the **From** and **To** options to specify the range. The default **From** value is 0, and the default **To** value is 10.
- **Learning Rate** — Specifies whether to autotune the learn rate. If this option is selected, the following option is available:
 - **Initial value** — Specifies the initial value for autotuning the learn rate. The default value is 0.1. Use the **From** and **To** options to specify the range. The default **From** value is 0.01, and the default **To** value is 1.
- **Maximum Depth** — Specifies whether to autotune the maximum number of generations of nodes. If this option is selected, the following option is available:
 - **Initial value** — Specifies the initial value for autotuning the maximum number of generations of nodes. The default value is 4. Use the **From** and **To** options to specify the range. The default **From** value is 1, and the default **To** value is 6.
- **Minimum Leaf Size** — Specifies whether to autotune the minimum leaf size. The following option is available:
 - **Initial value** — Specifies the initial value for autotuning the minimum leaf size. The default value is 5. Use the **From** and **To** options to specify the range. The default **From** value is 1, and the default **To** value is 100.
- **Number of Interval Bins** — Specifies whether to autotune the number of interval bins. The following option is available:

- **Initial value** — Specifies the initial value for autotuning the number of interval bins. The default value is 50. Use the **From** and **To** options to specify the range. The default **From** value is 20, and the default **To** value is 100.
- **Number of Inputs per Split** — Specifies whether to autotune the number of inputs per split. If selected, the following option is available:
 - **Initial value** — Specifies the initial value for the number of inputs per split. The default value is 100. Use the **From** and **To** options to specify the range. The default **From** value is 1, and the default **To** value is 100.
- **Number of Trees** — Specifies whether to autotune the number of trees in a boosting series. If selected, the following option is available:
 - **Initial value** — Specifies the initial value for the number of trees. The default value is 100. Use the **From** and **To** options to specify the range. The default **From** value is 20, and the default **To** value is 150.

Note: When early stopping is enabled and a validation partition exists, autotuning **Number of Trees** is not performed if selected.

- **Subsample Rate** — Specifies whether to autotune the subsample rate. If selected, the following option is available:
 - **Initial value** — Specifies the initial value for autotuning the subsample rate. The default value is 0.5. Use the **From** and **To** options to specify the range. The default **From** value is 0.1, and the default **To** value is 1.
- **Search Options** — Specifies the options for autotuning searching. The following options are available:
 - **Search method** — Specifies the tuning search method. Here are the possible values:
 - **Bayesian** — Specifies the Bayesian method. This method uses priors to seed the iterative optimization.
 - **Genetic algorithm** — Specifies the genetic algorithm method. This method uses an initial Latin Hypercube sample that seeds a genetic algorithm to generate a new population of alternative configurations at each iteration.
 - **Grid** — Specifies the grid method. This method performs an exhaustive search of all configurations. Each hyperparameter of interest is discretized into a desired set of values to be studied. Models are trained and assessed for all combinations of values across all the hyperparameters (thus forming a multi-dimensional “grid”).
 - **Latin hypercube sample** — Specifies the Latin Hypercube method. This method performs an optimized grid search that is uniform in each tuning parameter, but random in combinations.
 - **Random** — Specifies the Random method. This method generates a single sample of purely random configurations.

The default value is **Genetic algorithm**.

- **Number of evaluations per iteration** — Specifies the number of tuning evaluations in one iteration. This option is available only if the **Search method** is **Genetic algorithm** or **Bayesian**. The default value is 10.

- **Maximum number of evaluations** — Specifies the maximum number of tuning evaluations. This option is available only if the **Search method** is **Genetic algorithm** or **Bayesian**. The default value is 50.
- **Maximum number of iterations** — Specifies the maximum number of tuning iterations. This option is available only if the **Search method** is **Genetic algorithm** or **Bayesian**. The default value is 5.
- **Maximum number of points in model** — Specifies the maximum number of points in the model. This option is available only if the **Search method** is **Bayesian**. The default value is 100.
- **Sample size** — Specifies the sample size. This option is available only if the **Search method** is **Random** or **Latin hypercube sample**. The default value is 50.
- **General Options** — Specifies the general properties for autotuning. The following options are available:
 - **Validation method** — Specifies how to partition the data for assessing the models. Note that if your data is partitioned, then that partition is used and **Validation method**, **Validation data proportion**, and **Cross validation number of folds** are all ignored. Here are the possible values:
 - **Partition** — Specifies using a single partition of a training set. With partition, you specify proportions to use for randomly assigning observations to each role.
 - **K-fold cross validation** — Specifies using the k -fold cross validation method. In k -fold cross validation, each model evaluation requires k training executions (on $k-1$ data folds) and k scoring executions (on one holdout fold). This increases the evaluation time by approximately a factor of k .

For small to medium data tables, cross validation provides, on average, a better representation of error across the whole data table. **Partition** is the default value.
 - **Training data proportion** — Specifies the proportion of training data to be used for the **Partition** validation method. The default value is 0.7.
 - **Validation data proportion** — Specifies the proportion of data to be used for the **Partition** validation method. The default value is 0.3.
 - **Cross validation number of folds** — Specifies the number of partition folds in the cross validation process (the k defined above). Possible values range from 2 to 20. The default value is 5.
 - **Class target objective function** — Specifies the objective function to optimize for tuning parameters for a class target. Here are the possible values:
 - **Area under the curve**
 - **Average squared error**
 - **F0.5 score**
 - **F1 score**
 - **Gamma**
 - **Gini coefficient**
 - **Kolmogorov-Smirnov statistic**

- **Misclassification rate**
- **Multi-class log loss**
- **Root average squared error**
- **Tau**

The default value is **Kolmogorov-Smirnov statistic**.

- **Interval target objective function** — Specifies the objective function to optimize for tuning parameters for an interval target. Here are the possible values:
 - **Average squared error**
 - **Mean absolute error**
 - **Mean squared logarithmic error**
 - **Root average squared error**
 - **Root mean absolute error**
 - **Root mean squared logarithmic error**

The default value is **Average squared error**.

- **Maximum time (minutes)** — Specifies the maximum time (in minutes) for the optimization tuner. The default value is 60.
- **Maximum training time for single model (minutes)** — Specifies the maximum training time (in minutes) for the single model.

Lift Calculation Property

- **Use the exact percentile method for lift calculations** — Specifies whether to use the exact percentile method for calculating lift and related assessment measures. When this property is deselected, the iterative method is used. The exact method should be used if there are convergence errors when using the iterative method.

Binary Classification Cutoff

- **Specify node binary classification cutoff** — Specifies whether to use the binary classification cutoff specified below for the node. If this property is deselected, the project binary classification cutoff is used for determining the predicted value for a binary target based on the posterior probabilities. By default, this property is deselected.
- **Node binary classification cutoff** — Specifies the cutoff to use in the node for determining the predicted value for a binary target based on the posterior probabilities. This option is available only if **Specify node binary classification cutoff** is selected. The default value is 0.5.

Note: The **Rules** settings in the Project Settings window enable you to specify a new cutoff value by selecting **Override the default classification cutoff**. If **Specify node binary classification cutoff** is deselected, then the value that you specified in the Project Settings window is used. If **Specify node binary classification cutoff** is selected, then the value that you specified in **Node binary classification cutoff** is used.

Post Training Properties

Changing the following properties will not retrain the model.

Model Interpretability — Specifies the method of model interpretation in order to show how the model responds to changing inputs.

- **Global Interpretability**

- **Variable importance** — Specifies whether to display a table of the relative importance values for the input variables. The surrogate variable importance values are calculated using a one-level decision tree for each input to predict the predicted value as a global surrogate model. For tree-based supervised learning models, the model variable importance table is displayed as well.
- **PD plots** — Specifies whether to display partial dependency (PD) plots that show the relationship between model input variables and their predictors. PD plots are generated for only up to the top ten variables in the relative importance table. Here is how to determine the variables that are displayed in the PD plot: for tree-based models (forest, decision tree, and gradient boosting), the model variable importance table is used. For other supervised learning models, the surrogate variable importance table is used.

- **Local Interpretability**


- **ICE plots** — Specifies whether to display Individual Conditional Expectation (ICE) plots. ICE plots can reveal interesting subgroups and interactions between model variables. ICE plots are generated for only up to the top ten variables in the relative importance table. Here is how to determine the variables that are displayed in the ICE plot: for tree-based models (forest, decision tree, and gradient boosting), the model variable importance table is used. For other supervised learning models, the surrogate variable importance table is used.
- **LIME** — Specifies whether to use the Locally Interpretable Model-Agnostic Explanation (LIME) method to explain the model predictions.
- **Kernel SHAP** — Specifies whether to use the Kernel Shapley additive explanations (Kernel SHAP) method to explain the model predictions.
- **Maximum number of Kernel SHAP variables** — Specifies the maximum number of Kernel SHAP variables to plot. Possible values range from 1 to 100. The default value is 20.
- **Specify instances to explain** — Specifies the individual observations that you want to explain. The **Random** observations option provides explanations for five randomly selected observations. The **Specify up to 5 observations** option enables you to provide IDs for the specific observations in the data. The IDs should be unique values of the variable that has the role **Key** in the

metadata. If no variable is assigned the role **Key**, then you can use the variable `_dmIndex_` to specify the individual observations.

- **LIME/Kernel SHAP Tables**
 - **Explainer Information table** — Specifies whether the Explainer Information tables for the LIME and Kernel SHAP methods should be included in the node results. The Explainer Information table contains information about what settings were used to perform the LIME and Kernel SHAP methods.
 - **Explainer Fidelity table** — Specifies whether the Explainer Fidelity tables for the LIME and Kernel SHAP methods should be included in the node results. The Explainer Fidelity table contains information about the accuracy of the local explanation predictions, compared to the black-box machine learning model.
- **PD/ICE Options**
 - **Maximum number of variables** — Specifies the maximum number of input variables to use to generate the PD and ICE plots. The input variables are selected based on their relative importance. Possible values range from 1 to 10. The default value is 5.
 - **Number of observations** — Specifies the maximum number of observations to sample for the PD and ICE plots. The default value is 1000.
 - **Number of tick points** — Specifies the number of tick points for an interval analysis variable to generate for the PD and ICE plots. Possible values range from 3 to 100. The default value is 50.
 - **Truncate lower tail** — Specifies under what percentage to discard the values of the interval analysis variables for the PD and ICE plots. Possible values range from 0 to 99. The default value is 5%.
 - **Truncate upper tail** — Specifies above what percentage to discard the values of the interval analysis variables for the PD and ICE plots. Possible values range from 0 to 99. The default value is 95%.
- **Seed** — Specifies the seed for generating random numbers. The seed is used for selecting individual observations to explain, selecting observations to generate PD and ICE plots, and simulating data for LIME and Kernel SHAP.

Gradient Boosting Results

After running the node, open the Results window by right-clicking the node and selecting **Results**.

You can download the data that is used to build the charts and tables in the Results window by clicking . A CSV file with the chart or table data will immediately begin downloading.

In the **Output Data** tab, you can view, save, and explore the data that is generated by the node. For more information, see [Explore and Visualize](#).

Several charts have detailed descriptions available that explain how the charts are generated. The descriptions also provide information about how to interpret the

charts that is custom to your particular data and models. To access these detailed descriptions:

- Expand an individual chart by clicking ↕. The detailed description is in the right panel.
- Select ⓘ on an individual chart. A window appears with the detailed description.

Note: The detailed descriptions are available only in the English language.

Node

- **Error Plot** — Displays a graph of the average squared error as a function of the number of trees. The average squared error is given for each of the data roles. To examine the misclassification rate as a function of the number of trees when you have a class target, use the drop-down menu in the upper right corner.
- **Variable Importance** — Displays a table of importance data for each variable. These statistics include train importance, importance standard deviation, and relative importance.
- **Autotune Best Configuration** — Displays a table of the resulting values of parameters that were autotuned. This table is not displayed if autotuning is not selected.
- **Autotune Results** — Displays the iterative results of the autotuning process, showing how each of the parameter values evolve as the model iterates. This table is not displayed if autotuning is not selected.
- **Path EP Score Code** — Displays the SAS code that was created by the node. The score code can be used outside the Model Studio environment to score new data.
- **DS2 Package Code** — Displays the SAS DS2 package code that was created by the node. The score code can be used outside the Model Studio environment to score new data. DS2 package code is used when publishing models to SAS Micro Analytic Service.
- **Score Inputs** — Displays a table of data on the input variables for scoring calculations. The table includes the variables' name, role, level, type, label, format, and length.
- **Score Outputs** — Displays a table of data on the predicted response variable for scoring calculations. The table includes the variables' name, role, type, format and length, as well as the creator, creator GUID, and function.
- **Training Code** — Displays the SAS code that Model Studio used to train the node.
- **Properties** — Specifies the various properties that you selected before running the node.
- **Output** — Displays the SAS output of the Gradient Boosting run. For the **Gradient Boosting** node, the output includes the model information, number of observations read and used, variable importance, fit statistics, and predicted target variable.

Assessment

- **Predicted Reports** — Displays the predicted and target mean as a function of the depth for the model. The predicted and target mean are given for each of the data roles. To examine the predicted mean as a function of the target mean, use

the drop-down menu in the upper right corner. This result is displayed only if the target is an interval variable.

- **Lift Reports** — Displays the cumulative lift as a function of the depth for the model. The cumulative lift is given for each of the data roles. To examine other statistics as a function of depth, use the drop-down menu in the upper right corner. Other statistics include lift, gain, captured response percentage, cumulative captured response percentage, response percentage, and cumulative response percentage. This result is displayed only if the target is a class variable.
- **ROC Reports** — Displays the ROC (receiver operating characteristic) chart for a model, giving the sensitivity as a function of 1-specificity. The sensitivity is given for each of the data roles. To examine other statistics, use the drop-down menu in the upper right corner. Other statistics include accuracy and F1 score. This result is displayed only if the target is a class variable.
- **Fit Statistics** — Displays a table of the fit statistics for the model, broken down by data role.
- **Event Classification** — Displays the confusion matrix at various cutoff values for each partition. The confusion matrix contains true positives (events that are correctly classified as events), false positives (non-events that are classified as events), false negatives (events that are classified as non-events), and true negatives (non-events that are correctly classified as non-events). The classification cutoffs that are used are the default value of 0.5, the Kolmogorov-Smirnov cutoff value for each partition, and the **Node binary classification cutoff** that is specified for binary targets. Use the drop-down menu to view the information in the chart as percentages or counts. You can also view the information summarized in a table. This result is displayed only if the target is a class variable.
- **Nominal Classification** — Displays either the percentage of observations or the number of observations that predict each level of the target variable. The plot is segmented by target level and partition level. The target level with the greatest predicted probability is the predicted outcome. This result is displayed only if the target is a nominal variable.

Model Interpretability

- **Surrogate Model Variable Importance** — Displays a table of the relative importance of the input variables, including variable level and label. Relative importance is calculated using a one-level decision tree for each input variable to predict the predicted value as a global surrogate model.
- **Model Variable Importance** — Displays a table of the relative importance of the input variables, including variable level and label. This is the variable importance table that is generated when training the original gradient boosting model.
- **PD Plot** — Displays the functional relationship between the input variable and the model prediction. For interval input variables, the 95% confidence interval for the average target prediction is indicated by the shaded band around the line. The X axis includes a heat map that shows the distribution of the input variable. The extreme values of the input variables are removed by truncating the lower and upper tails of its distribution. The amount of truncation can be controlled by specifying values for the **Truncate lower tail** and the **Truncate upper tail** properties.
- **PD and ICE Overlay Plot** — Displays the functional relationship between the input variable and the model prediction as well as the functional relationship between individual observations and the model prediction. For each individual

observation, the corresponding ICE curve displays values of the input variable on the X axis and the corresponding predicted probability or prediction of the target variable on the Y axis, holding the other input variable values constant at that observation. The X axis includes a heat map that shows the distribution of the input variable. The extreme values of the input variables are removed by truncating the lower and upper tails of its distribution. The amount of truncation can be controlled by specifying values for the **Truncate lower tail** and the **Truncate upper tail** properties.

- **LIME Explanations** — The LIME algorithm calculates a more easily interpretable linear model around an individual observation. The chart displays the regression estimates for the input variables that are selected in the local surrogate linear regression model. The input variables are ordered by significance such that the most important variable for the local regression model is at the bottom of the chart. The LASSO technique is used to select the most significant predictors from the set of input variables that are used to train the model. Each nominal input variable is binary-encoded based on whether it matches the level of the individual observation. A positive estimate indicates that the observed value of the input variable increases the predicted probability of the event.
- **Kernel SHAP Values** — For each individual observation, an input variable's Kernel SHAP value is the contribution of the observed value of the input variable to the predicted probability or prediction of the target variable. The Kernel SHAP values are the regression coefficients that are obtained by fitting a weighted least squares regression. The Kernel SHAP values of all input variables sum to one. The input variables are ordered by significance such that the most important variable according to the absolute Kernel SHAP values is at the top of the chart. Each nominal input is binary-encoded based on whether it matches the individual observation. Interval inputs are binary encoded based on their proximity to the individual observation.
- **LIME Information Table** — Displays information about the settings used to perform the LIME method.
- **Kernel SHAP Information Table** — Displays information about the settings used to perform the Kernel SHAP method.
- **LIME Fidelity Table** — Displays information about the accuracy of the local explanation predictions that are calculated using the LIME method, compared to the black-box machine learning model. The model prediction, the explainer prediction, and the explainer root mean square error (RMSE) are displayed.
- **Kernel SHAP Fidelity Table** — Displays information about the accuracy of the local explanation predictions that are calculated using the Kernel SHAP method, compared to the black-box machine learning model. The model prediction, the explainer prediction, and the explainer root mean square error (RMSE) are displayed.

Imputation

<i>Overview of Imputation</i>	117
<i>Imputation Properties</i>	118
General Properties	118
Class Inputs	119
Interval Inputs	119
Constant Values	120
Indicators	120
Ignore Methods in Metadata Property	121
<i>Imputation Results</i>	121

Overview of Imputation

The **Imputation** node is a Data Mining Preprocessing node. Use the **Imputation** node to replace missing values in data sets to improve data quality and improve model accuracy.

Data mining databases often contain observations that have missing values for one or more variables. Missing values can result from data collection errors, incomplete customer responses, and actual system and measurement failures. They might also result from a revision of the data collection scope over time, such as tracking new variables that were not included in the previous data collection schema.

If an observation contains a missing value, then by default that observation is not used for modeling by nodes such as **Neural Network** or **Regression**. However, rejecting all incomplete observations might ignore useful or important information that is still contained in the nonmissing variables. Rejecting all incomplete observations might also bias the sample, since observations that have missing values might have other things in common as well.

How should missing data values be treated? There is no single correct answer. Choosing the "best" missing value replacement technique inherently requires the researcher to make assumptions about the true (missing) data. For example, researchers often replace a missing value with the mean of the variable. This approach assumes that the variable's data distribution follows a normal population

response. Replacing missing values with the mean, median, or another measure of central tendency is simple. But it can greatly affect a variable's sample distribution. Always try to choose an imputation strategy that minimizes the impact on the relationship between the predictor and the response.

In the data that is exported from the **Imputation** node, a new variable is created for each variable for which missing values are imputed. The original variable is not overwritten. Instead, the new variable has the same name as the original variable but is prefaced with **IMP_**. The original version of each variable also exists in the exported data and by default has the role of **Rejected**. Some variables behave differently with respect to the outcome when they have a missing value. The **Imputation** node supports the creation of both imputation and missing value indicators to help handle such issues.

Input variables that have a **DATE**, **DATETIME**, or **TIME** format are ignored by the **Imputation** node.

You can set the imputation method for individual inputs by editing an input variable's value for **Impute** on the **Data** tab or by using the **Manage Variables** node. The method that is specified in the metadata (on the **Data** tab) takes precedence over the default method that was set in the **Manage Variables** node.

Note: When you specify an imputation method for a variable on the **Data** tab, that method is always used, regardless of the property settings for this node.

Imputation Properties

General Properties

- **Impute non-missing variables** — Specifies whether to generate an imputation score code regardless of the existence of missing values in the training data. By default, this option is deselected.
- **Missing percentage cutoff** — Specifies the maximum percent of missing values that are allowed for a variable to be imputed. Variables whose missing percentage exceeds this cutoff are ignored. The default value is 50.
- **Reject original variables** — Specifies whether to mark original variables that are imputed as rejected. Deselecting this option keeps original variables as input. By default, this option is selected.
- **Summary statistics** — Specifies whether to generate summary statistics for the imputed variables. By default, this option is deselected.

Class Inputs

- **Default method** — Specifies the default imputation method for all class input variables. Note that any imputation specified in the metadata takes precedence over the method specified here. Possible values are as follows:
 - **(none)** — Specifies that only the imputations specified on the **Data** pane are performed.
 - **Cluster count** — Specifies that missing values be replaced with the variable's most frequent nonmissing value in the observation's cluster. In order to use this method of imputation, you must have a **Clustering** node in the pipeline immediately preceding the **Imputation** node.
 - **Constant value** — Specifies that missing values are replaced with a character specified in the **Constant character value** field.
 - **Count** — Specifies that missing values are replaced with the variable's most frequent value. There are no other configurations to set.
 - **Distribution** — Specifies that missing values be replaced with randomly assigned values from an empirical distribution of the nonmissing values of the variable. As a result, the Distribution imputation typically does not significantly change the distribution of the data. The initial seed value for randomization is specified in the **Distribution method random seed** field. The default value is 12345.

The default value is **Count**.

.....

Note: If an input variable is unary and **Constant value** is specified, the imputed variable will be binary and assigned the role Input. Otherwise, the imputed variable will be unary and assigned the role Rejected.

.....

Interval Inputs

- **Default method** — Specifies the imputation statistic that you want to use to replace missing interval variables. Here are the possible values:
 - **(none)** — Specifies that only the imputations specified on the **Data** pane are performed.
 - **Cluster mean** — Specifies that missing values be replaced with the arithmetic average of the observation's cluster. In order to use this method of imputation, you must have a **Clustering** node in the pipeline immediately preceding the **Imputation** node.
 - **Constant value** — Specifies that missing values be replaced with the value that is specified in the **Constant number value** field. The default value is 0.
 - **Distribution** — Specifies that missing values be replaced with randomly assigned values from an empirical distribution of the nonmissing values of the variable. As a result, the Distribution imputation typically does not significantly change the distribution of the data. The initial seed value for randomization is

specified in the **Distribution method random seed** field. The default value is 12345.

- Maximum** — Specifies that missing values be replaced with the maximum value for the variable found in training.
- Mean** — Specifies that missing values be replaced with the arithmetic average.
- Median** — Specifies that missing values be replaced with the midpoint of a frequency distribution of the observed values.
- Midrange** — Specifies that missing values be replaced with the maximum value plus the minimum value divided by 2.
- Minimum** — Specifies that missing values be replaced with the minimum value for the variable found in training.

The default value is **Mean**.

- **Data limits for calculating values** — Specifies how the data is used to calculate values used for default methods. Here are the possible values:
 - All data** — Specifies that the values used are based on the entire set of data.
 - Trimmed** — Specifies that trimmed data is used.
 - Winsorized** — Specifies that Winsorized data is used.

The default value is **All data**.

Note: The **Data limits for calculating values** field is available only when the **Default Method** property is set to **Mean**, **Midrange**, **Minimum**, or **Maximum**.

- **Data limit percentage** — Specifies the percentage of data that is removed from both tails of the distribution when using trimmed or Winsorized data for calculating the mean, maximum, minimum, or midrange. For example, setting the data limit percentage to 2.5 creates a 95% trimmed or Winsorized data set. The default value is 5.

Constant Values

- **Constant character value** — Specifies the replacement value that is used when the default class imputation method is **Constant value**.
- **Constant number value** — Specifies the replacement value that is used when the default interval imputation method is **Constant value**.

Indicators

- **Single indicator** — Specifies the creation of a single variable that indicates a count of any inputs that were missing in training.
- **Unique indicator** — Specifies the creation of a separate variable for each input with missing values in the training data.

- If either the **Single indicator** or **Unique indicator** options are selected, the **Indicator subject** option is available. This property specifies the subject of the indicator variables to be created. Here are the possible values:
 - **Imputed variables** — Specifies to set the indicator for variables that have been imputed.
 - **Missing variables** — Specifies to set the indicator for missing values.

The default value is **Imputed variables**.

- If either the **Single indicator** or **Unique indicator** options are selected, the **Indicator role** option is available. This property specifies the role to be assigned to the created indicator variables. Here are the possible values:
 - **Input**
 - **Rejected**


The default value is **Rejected**.

Ignore Methods in Metadata Property

- **Ignore methods in metadata** — Specifies whether to ignore imputation methods that are specified on the **Data** tab, in global metadata, or in a preceding **Manage Variables** node. By default, this option is deselected.

Imputation Results

After running the node, open the Results window by right-clicking the node and selecting **Results**.

You can download the data that is used to build the charts and tables in the Results window by clicking . A CSV file with the chart or table data will immediately begin downloading.

In the **Output Data** tab, you can view, save, and explore the data that is generated by the node. For more information, see [Explore and Visualize](#).

Summary

- **Input Variable Statistics** — Displays relevant imputation statistics on input variables. This table gives the variable name, variable level, number of missing values, percent of missing values, and whether the variable is imputable (1 for yes, 0 for no). It also lists the minimum, maximum, mean, midrange, and standard deviation for each input.
- **Imputed Variables Summary** — Displays a summary of the imputation that was run for each imputed variable. This table gives the following properties:
 - Imputed Variable
 - Method
 - Input Variable

- Value
- Percent Missing
- Variable Level
- Type
- Variable Label
- **Imputation Summary for Interval Variables** — Displays a summary of the imputation run for interval variables. This table gives the following properties:
 - Imputed Variable
 - Method
 - Input Variable
 - Indicator Variable
 - Value
 - Number of Imputed Values
 - Percentage of Imputed Values
 - Minimum
 - Maximum
 - Mean
 - Midrange
 - Importance Standard Deviation
 - Skewness
 - Kurtosis
 - Variable Label

This result is displayed only if you select the **Summary statistics** property. This table replaces the **Imputed Variables Summary** table.

- **Imputation Summary for Class Variables** — Displays a summary of the imputation run for class variables. This table gives the following properties:
 - Imputed Variable
 - Method
 - Input Variable
 - Indicator Variable
 - Value
 - Variable Level
 - Number of Imputed Values
 - Percentage of Imputed Values
 - Type
 - Variable Length
 - Variable Label

This result is displayed only if you select the **Summary statistics** property. This table replaces the **Imputed Variables Summary** table.

- **Node Score Code** — Displays the SAS score code that was created by the node. The SAS score code can be used outside the Model Studio environment in custom user applications.
- **Properties** — Specifies the various properties that you selected before running the node.
- **Output** — Displays the SAS output of the imputation run.

Linear Regression

<i>Overview of Linear Regression</i>	125
<i>Linear Regression Properties</i>	126
Effects Options	126
Selection Method Property	126
Selection Options	127
Lift Calculation Property	130
Post Training Properties	130
<i>Linear Regression Results</i>	131

Overview of Linear Regression

The **Linear Regression** node is a Supervised Learning node. A linear regression attempts to predict the value of an interval target variable as a linear function of one or more features. The linear regression uses the least squares method to perform variable selection and to determine the model. The least squares method creates a line of best fit by minimizing the residual sum of squares for every instance in the input data set. The residual sum of squares is the vertical distance between an instance and the line of best fit. The least squares method assumes the distribution of the target variable is normal with constant variance.

The linear regression requires an interval target variable and at least one feature.

Linear Regression Properties

Effects Options

- **Two-factor interactions** — Specifies whether to include all two-factor interactions of nominal variables in the model. By default, this option is deselected.
- **Factor split** — Specifies whether levels of a nominal variable can enter or leave a model independently. This is done automatically for LASSO selection. For other selection methods, this option is deselected.
- **Spline** — Specifies whether to expand interval input variables into cubic B-spline bases with three equally spaced knots. (Expansion yields seven design matrix columns for each of the variables.) Spline and polynomial effects cannot be specified together. By default, this option is deselected.
- **Spline split** — Specifies whether each individual column in the design matrix that corresponds to the spline effect is treated as a separate effect that can enter or leave the model. By default, this option is deselected. This option is available only if **Spline** is selected.
- **Polynomial** — Specifies whether to use polynomial effects up to the specified **Polynomial degree** for all interval variables. Spline and polynomial effects cannot be specified together. By default, this option is deselected.
- **Polynomial degree** — Specifies the polynomial degree when polynomial terms are included in the model. This option is available only if **Polynomial** has been selected. Possible values are 2 and 3. The default value is 2.
- **Suppress intercept** — Specifies whether to suppress the intercept. By default, this is deselected.
- **Model missing values for inputs** — Specifies whether to model missing values for input variables. This is done by including indicator values in the model for missing values for interval input variables and by imputing missing values in the original effect with the mean. In addition, missing values for class input variables are treated as valid levels. By default, this is deselected.
- **Use missing as level** — Specifies whether missing values should be treated as a separate level for all nominal inputs. By default, this is deselected.

Selection Method Property

- **Selection method** — Specifies a model selection method. Here are the possible values:
 - **(none)** — Specifies not to perform variable selection.

- **Adaptive LASSO** — Specifies to use adaptive weights when applying L1 regularization of the LASSO method. By default, ordinary least squares estimates of the regression coefficients are used as adaptive weights.
- **Backward** — Specifies that training is done by least squares regression that starts with all candidate features in the model. Features are removed one at a time until the stop criterion is met.
- **Forward** — Specifies that training is done by least squares regression that starts with no candidate features in the model. Features are added one at a time until the stop criterion is met.
- **LASSO** — Specifies that training is done using the group LASSO method, which adds and removes features by a sequence of LASSO steps.
- **Stepwise** — Specifies that training is done by least squares regression that starts as in the forward model but might remove features already in the model.

The default value is **Stepwise**.

Selection Options

- **Effect-selection criterion** — Specifies the criterion that the procedure uses to determine the order in which features enter or leave at each step of the selection method. Here are the possible values:
 - **Adjusted R-square** — Specifies the Adjusted R-square value. The Adjusted R-Square adjusts the R-square value by accounting for the addition of more features. Values can range from 0 to 1. Values closer to 1 are preferred.
 - **AIC** — Specifies Akaike's Information Criterion. AIC is based on the Kullback-Leibler information measure of discrepancy between the true distribution of the response variable and the distribution specified by the model. Smaller values indicate better models, and AIC values can become negative.
 - **AICC** — Specifies Corrected Akaike's Information Criterion. AICC adjusts the AIC value by accounting for sample size. Extra features penalize AICC more than AIC. As the sample size increases, AICC and AIC converge.
 - **Mallows' Cp** — Specifies Mallows's Cp value. Smaller values indicate better models, and Cp values can become negative.
 - **R-square** — Specifies the R-square value. R-square value is an indicator of how well the model fits the data. R-square values can range from 0 to 1. Values closer to 1 are preferred.
 - **SBC (BIC)** — Specifies Schwarz's Bayesian Criterion (SBC), also known as the Bayesian Information Criterion (BIC). Unexplained variations in the response variable and the number of features increase the value of the SBC. Smaller values indicate better models. SBC penalizes total number of model parameters more strongly than AIC, and hence tends to choose more sparse models.
 - **Significance level** — Specifies the standard statistical significance level criterion.

If the **Selection method** above is **LASSO**, this option is unavailable. The default value is **SBC (BIC)**.

- **Selection-process stopping criterion** — Specifies the criterion to stop the selection process. Here are the possible values:
 - **(none)** — Specifies that the selection process continues until all the features are in the model or if a size-based limit (see **Maximum number of effects**) is achieved.
 - **Adjusted R-square** — Specifies the Adjusted R-square value. The Adjusted R-Square value adjusts the R-square value by accounting for the number of features in the model. Values can range from 0 to 1. Values closer to 1 are preferred. Selection stops at the step where adjusted R-square starts to decrease.
 - **AIC** — Specifies Akaike's Information Criterion. AIC is based on the Kullback-Leibler information measure of discrepancy between the true distribution of the response variable and the distribution specified by the model. Smaller values indicate better models, and AIC values can become negative. Selection stops at the step where AIC starts to increase.
 - **AICC** — Specifies Corrected Akaike's Information Criterion. AICC adjusts the AIC value by accounting for sample size. Extra features penalize AICC more than AIC. Smaller values indicate better models. As the sample size increases, AICC and AIC converge. Selection stops at the step where AICC starts to increase.
 - **Mallows' Cp** — Specifies Mallows's Cp value. Smaller values indicate better models, and Cp values can become negative.
 - **Predicted RSS** — Specifies that the selection process continues until the predicted residual sum of squares (RSS) starts to increase. This option is not valid with **LASSO** selection.
 - **SBC (BIC)** — Specifies Schwarz's Bayesian Criterion (SBC), also known as the Bayesian Information Criterion (BIC). Unexplained variations in the response variable and the number of features increase the value of the SBC. SBC penalizes total number of parameters more strongly than AIC, and hence tends to choose more sparse models. Selection stops at the step where SBC starts to increase.
 - **Significance level** — Specifies the standard statistical significance level criterion. Selection stops at the step where the significance level exceeds the specified level. The default value is 0.05.
 - **Validation ASE** — Average square error (ASE) of the model that is computed by using the validation data. Selection stops at the step where the ASE starts to increase. This method requires partitioned data.

The default value is **SBC (BIC)**.

- **Model-selection criterion** — Specifies the criterion to choose the model (from the list of models at each step of the selection process) that yields the best value of the specified criterion. If the optimal value of the specified criterion occurs for models at more than one step, then the model that has the smallest number of parameters is chosen. Here are the possible values:
 - **Adjusted R-square** — Specifies the Adjusted R-square value. The Adjusted R-Square value adjusts the R-square value by accounting for the number of features in the model. Values can range from 0 to 1. The model with the largest R-square value is chosen.

- **AIC** — Specifies Akaike's Information Criterion. AIC is based on the Kullback-Leibler information measure of discrepancy between the true distribution of the response variable and the distribution specified by the model. Smaller values indicate better models, and AIC values can become negative. The model that has the minimal AIC value is chosen.
- **AICC** — Specifies Corrected Akaike's Information Criterion. AICC adjusts the AIC value by accounting for sample size. Extra features penalize AICC more than AIC. Smaller values indicate better models. As the sample size increases, AICC and AIC converge.
- **Mallows' Cp** — Specifies Mallows's Cp value. Smaller values indicate better models, and the model with the minimal Cp is chosen.
- **Predicted RSS** — Specifies that the model that has the minimal predicted residual sum of squares (RSS) is chosen. This option is not valid with **LASSO** selection.
- **SBC (BIC)** — Specifies Schwarz Bayesian Criterion (SBC), also known as the Bayesian Information Criterion (BIC). Unexplained variations in the response variable and the number of features increase the value of the SBC. SBC penalizes the total number of parameters more strongly than AIC and therefore tends to choose more sparse models. The model that has the minimal SBC value is chosen.
- **Validation ASE** — Average square error (ASE) of the model is computed by using the validation data. The model that has the minimal validation ASE is chosen. This method requires partitioned data.

The default value is **SBC (BIC)**.

- **Entry significance level** — Specifies the significance level for adding variables in the forward and stepwise directions. The default value is 0.05.
- **Stay significance level** — Specifies the significance level for removing variables in backward and stepwise directions. The default value is 0.05.
- **Maximum number of effects** — Specifies the maximum number of effects in any model that is considered during the selection process. If a model at some step of the selection process contains the specified maximum number of effects, then no additional effects are considered. If **Maximum number of effects** is set to 0 (the default value), this option is ignored.
- **Minimum number of effects** — Specifies the minimum number of effects in any model that is considered during the backward selection process. If **Minimum number of effects** is set to 0 (the default value), this option is ignored.
- **Maximum number of steps** — Specifies the maximum number of selection steps that are performed. If **Maximum number of steps** is set to 0 (the default value), this option is ignored.
- **Hierarchy** — Specifies whether and how the model hierarchy requirement is applied. Model hierarchy refers to the requirement that, for any term to be in the model, all model effects that are contained in the term must be present in the model. For example, in order for the interaction A*B to enter the model, the main effects A and B must also be in the model. Likewise, neither effect A nor effect B can leave the model while the interaction A*B is in the model. Here are the possible values:
 - **(none)** — Specifies that the hierarchy requirement is never applied.
 - **All variables** — Specifies that both interval and nominal variables are subject to the model hierarchy requirement.

- **Class variables** — Specifies that only nominal variables are subject to the model hierarchy requirement.

The default value is **(none)**.

Lift Calculation Property

- **Use the exact percentile method for lift calculations** — Specifies whether to use the exact percentile method for calculating lift and related assessment measures. When this property is deselected, the iterative method is used. The exact method should be used if there are convergence errors when using the iterative method.

Post Training Properties

Changing the following properties will not retrain the model.

Model Interpretability — Specifies the method of model interpretation in order to show how the model responds to changing inputs.

- **Global Interpretability**

- **Variable importance** — Specifies whether to display a table of the relative importance values for the input variables. The surrogate variable importance values are calculated using a one-level decision tree for each input to predict the predicted value as a global surrogate model. For tree-based supervised learning models, the model variable importance table is displayed as well.
- **PD plots** — Specifies whether to display partial dependency (PD) plots that show the relationship between model input variables and their predictors. PD plots are generated for only up to the top ten variables in the relative importance table. Here is how to determine the variables that are displayed in the PD plot: for tree-based models (forest, decision tree, and gradient boosting), the model variable importance table is used. For other supervised learning models, the surrogate variable importance table is used.


- **Local Interpretability**

- **ICE plots** — Specifies whether to display Individual Conditional Expectation (ICE) plots. ICE plots can reveal interesting subgroups and interactions between model variables. ICE plots are generated for only up to the top ten variables in the relative importance table. Here is how to determine the variables that are displayed in the ICE plot: for tree-based models (forest, decision tree, and gradient boosting), the model variable importance table is used. For other supervised learning models, the surrogate variable importance table is used.
- **LIME** — Specifies whether to use the Locally Interpretable Model-Agnostic Explanation (LIME) method to explain the model predictions.
- **Kernel SHAP** — Specifies whether to use the Kernel Shapley additive explanations (Kernel SHAP) method to explain the model predictions.

- **Maximum number of Kernel SHAP variables** — Specifies the maximum number of Kernel SHAP variables to plot. Possible values range from 1 to 100. The default value is 20.
- **Specify instances to explain** — Specifies the individual observations that you want to explain. The **Random** observations option provides explanations for five randomly selected observations. The **Specify up to 5** observations option enables you to provide IDs for the specific observations in the data. The IDs should be unique values of the variable that has the role **Key** in the metadata. If no variable is assigned the role **Key**, then you can use the variable `_dmIndex_` to specify the individual observations.
- **LIME/Kernel SHAP Tables**
 - **Explainer Information table** — Specifies whether the Explainer Information tables for the LIME and Kernel SHAP methods should be included in the node results. The Explainer Information table contains information about what settings were used to perform the LIME and Kernel SHAP methods.
 - **Explainer Fidelity table** — Specifies whether the Explainer Fidelity tables for the LIME and Kernel SHAP methods should be included in the node results. The Explainer Fidelity table contains information about the accuracy of the local explanation predictions, compared to the black-box machine learning model.
- **PD/ICE Options**
 - **Maximum number of variables** — Specifies the maximum number of input variables to use to generate the PD and ICE plots. The input variables are selected based on their relative importance. Possible values range from 1 to 10. The default value is 5.
 - **Number of observations** — Specifies the maximum number of observations to sample for the PD and ICE plots. The default value is 1000.
 - **Number of tick points** — Specifies the number of tick points for an interval analysis variable to generate for the PD and ICE plots. Possible values range from 3 to 100. The default value is 50.
 - **Truncate lower tail** — Specifies under what percentage to discard the values of the interval analysis variables for the PD and ICE plots. Possible values range from 0 to 99. The default value is 5%.
 - **Truncate upper tail** — Specifies above what percentage to discard the values of the interval analysis variables for the PD and ICE plots. Possible values range from 0 to 99. The default value is 95%.
- **Seed** — Specifies the seed for generating random numbers. The seed is used for selecting individual observations to explain, selecting observations to generate PD and ICE plots, and simulating data for LIME and Kernel SHAP.



Linear Regression Results

After running the node, open the Results window by right-clicking the node and selecting **Results**.

You can download the data that is used to build the charts and tables in the Results window by clicking . A CSV file with the chart or table data will immediately begin downloading.

In the **Output Data** tab, you can view, save, and explore the data that is generated by the node. For more information, see [Explore and Visualize](#).

Several charts have detailed descriptions available that explain how the charts are generated. The descriptions also provide information about how to interpret the charts that is custom to your particular data and models. To access these detailed descriptions:

- Expand an individual chart by clicking . The detailed description is in the right panel.
- Select  on an individual chart. A window appears with the detailed description.

Note: The detailed descriptions are available only in the English language.

Node

- **t Values by Parameter** — Displays bar charts for the t values in the final model. The bars are color-coded to indicate the algebraic signs of the coefficients.
- **Parameter Estimates** — Displays a table of the various statistics related to estimates for the parameters. These statistics include the t value, sign, estimate, absolute estimate, the p-value, chi-square value, and standard error.
- **Selection Summary** — Displays a table of the iterations, adding in and removing effects. This table also includes the SBC and optimal SBC values.
- **Regression Fit Statistics** — Displays a table of data on the linear regression fit statistics. The table includes the statistics, a corresponding qualitative description, and the values.
- **Node Score Code** — Displays the SAS score code that was created by the node. The SAS score code can be used outside the Model Studio environment to score new data.
- **Path Score Code** — Displays the SAS score code that was created by the node, as well as all preceding nodes. The SAS model score code can be used outside the Model Studio environment to score new data.
- **DS2 Package Code** — Displays the SAS DS2 package code that was created by the node. The SAS score code can be used outside the Model Studio environment in custom user applications. DS2 package code is used when publishing models to SAS Micro Analytic Service.
- **Score Inputs** — Displays a table of data on the input variables for scoring calculations. The table includes the variables' name, role, level, type, label, format, and length.
- **Score Outputs** — Displays a table of data on the predicted response variable for scoring calculations. The table includes the variables' name, role, type, format and length, as well as the creator, creator GUID, and function.
- **Training Code** — Displays the SAS code that Model Studio used to train the node.
- **Properties** — Specifies the various properties that you selected before running the node.
- **Output** — Displays the SAS output of the linear regression run.

Assessment

- **Predicted Reports** — Displays the predicted and target mean as a function of the depth for the model. The predicted and target mean are given for each of the data roles. To examine the predicted mean as a function of the target mean, use the drop-down menu in the upper right corner. This result is displayed only if the target is an interval variable.
- **Fit Statistics** — Displays a table of the fit statistics for the model, broken down by data role.

Model Interpretability

- **Surrogate Model Variable Importance** — Displays a table of the relative importance of the input variables, including variable level and label. Relative importance is calculated using a one-level decision tree for each input variable to predict the predicted value as a global surrogate model.
- **PD Plot** — Displays the functional relationship between the input variable and the model prediction. For interval input variables, the 95% confidence interval for the average target prediction is indicated by the shaded band around the line. The X axis includes a heat map that shows the distribution of the input variable. The extreme values of the input variables are removed by truncating the lower and upper tails of its distribution. The amount of truncation can be controlled by specifying values for the **Truncate lower tail** and the **Truncate upper tail** properties.
- **PD and ICE Overlay Plot** — Displays the functional relationship between the input variable and the model prediction as well as the functional relationship between individual observations and the model prediction. For each individual observation, the corresponding ICE curve displays values of the input variable on the X axis and the corresponding predicted probability or prediction of the target variable on the Y axis, holding the other input variable values constant at that observation. The X axis includes a heat map that shows the distribution of the input variable. The extreme values of the input variables are removed by truncating the lower and upper tails of its distribution. The amount of truncation can be controlled by specifying values for the **Truncate lower tail** and the **Truncate upper tail** properties.
- **LIME Explanations** — The LIME algorithm calculates a more easily interpretable linear model around an individual observation. The chart displays the regression estimates for the input variables that are selected in the local surrogate linear regression model. The input variables are ordered by significance such that the most important variable for the local regression model is at the bottom of the chart. The LASSO technique is used to select the most significant predictors from the set of input variables that are used to train the model. Each nominal input variable is binary-encoded based on whether it matches the level of the individual observation. A positive estimate indicates that the observed value of the input variable increases the predicted probability of the event.
- **Kernel SHAP Values** — For each individual observation, an input variable's Kernel SHAP value is the contribution of the observed value of the input variable to the predicted probability or prediction of the target variable. The Kernel SHAP values are the regression coefficients that are obtained by fitting a weighted least squares regression. The Kernel SHAP values of all input variables sum to one. The input variables are ordered by significance such that the most important variable according to the absolute Kernel SHAP values is at the top of the chart. Each nominal input is binary-encoded based on whether it matches the

individual observation. Interval inputs are binary encoded based on their proximity to the individual observation.

- **LIME Information Table** — Displays information about the settings used to perform the LIME method.
- **Kernel SHAP Information Table** — Displays information about the settings used to perform the Kernel SHAP method.
- **LIME Fidelity Table** — Displays information about the accuracy of the local explanation predictions that are calculated using the LIME method, compared to the black-box machine learning model. The model prediction, the explainer prediction, and the explainer root mean square error (RMSE) are displayed.
- **Kernel SHAP Fidelity Table** — Displays information about the accuracy of the local explanation predictions that are calculated using the Kernel SHAP method, compared to the black-box machine learning model. The model prediction, the explainer prediction, and the explainer root mean square error (RMSE) are displayed.

Logistic Regression

<i>Overview of Logistic Regression</i>	135
<i>Logistic Regression Properties</i>	136
General Properties	136
Effects Options	136
Selection Method Property	137
Selection Options	138
Optimization Options	140
Convergence Options	141
Lift Calculation Property	142
Binary Classification Cutoff	142
Post Training Properties	143
<i>Logistic Regression Results</i>	144

Overview of Logistic Regression

The **Logistic Regression** node is a Supervised Learning node. A logistic regression attempts to predict the value of a binary or nominal response variable. A logistic regression analysis models the natural logarithm of the odds ratio as a linear combination of the explanatory variables. This approach enables the logistic regression model to approximate the probability that an individual observation belongs to the level of interest.

The **Logistic Regression** node supports offset variables. When you are on the **Data** tab, you can select a variable and modify its role to **Offset**.

Logistic Regression Properties

General Properties

- **Binary target link function** — Specifies the link that links the binary target to a linear function of input variables. Here are the possible values:
 - **Complementary log-log**
 - **Log-log**
 - **Logit**
 - **Probit**

The default value is **Logit**.

- **Nominal target link function** — Specifies the link function that links the nominal target to a linear function of input variables. Cumulative link functions treat the target as an ordinal variable. Here are the possible values:
 - **Cumulative complementary log-log**
 - **Cumulative log-log**
 - **Cumulative logit**
 - **Cumulative probit**
 - **Generalized logit**

The default value is **Generalized logit**.

Effects Options

- **Class input order** — Specifies the sort order for the class inputs. Here are the possible values:
 - **Formatted**
 - **Formatted descending**
 - **Frequency**
 - **Frequency descending**
 - **Unformatted**
 - **Unformatted descending**

The default value is **Formatted**.

- **Two-factor interactions** — Specifies whether to include all two-factor interactions of nominal variables in the model. By default, this option is deselected.
- **Factor split** — Specifies whether levels of a nominal variable enter or leave a model independently. This is done automatically for LASSO selection. For other selection methods, this option is deselected.
- **Spline** — Specifies whether to expand interval input variables into cubic B-spline bases with three equally spaced knots. (This action yields seven design matrix columns for each of the variables.) Spline and polynomial effects cannot be specified together. By default, this option is deselected.
- **Spline split** — Specifies whether each individual column in the design matrix that corresponds to the spline effect is treated as a separate effect that can enter or leave the model. By default, this option is deselected. This option is available only if **Spline** is selected.
- **Polynomial** — Specifies whether to use model polynomial effects up to the specified polynomial degree for all interval variables. Spline and polynomial effects cannot be specified together. By default, this option is deselected.
- **Polynomial degree** — Specifies the polynomial degree when polynomial terms are included in the model. This option is available only if **Polynomial** has been selected. Possible values are 2 and 3. The default value is 2.
- **Suppress intercept** — Specifies whether to suppress the intercept. By default, this is deselected.
- **Model missing values for inputs** — Specifies whether to model missing values for input variables. This is done by including indicator values in the model for missing values for interval input variables and by imputing missing values in the original effect with the mean. In addition, missing values for class input variables are treated as valid levels. By default, this is deselected.
- **Use missing as level** — Specifies whether missing values should be treated as a separate level for all nominal inputs. By default, this is deselected.

Selection Method Property

- **Selection method** — Specifies a model selection method. Here are the possible values:
 - **(none)** — Specifies not to perform variable selection.
 - **Backward** — Specifies that training is done by least squares regression with all candidate effects in the model. Features are removed until the stop criterion is met.
 - **Fast backward** — Specifies that training is done by least squares regression that starts with all effects in the model. Features are deleted without refitting the model.
 - **Forward** — Specifies that training is done by least squares regression that starts with no candidate effects in the model. Effects are added until the Entry significance level or the stop criterion is met.
 - **LASSO** — Specifies that training is done using the group LASSO method, adding and removing effects by a sequence of LASSO steps.

- **Stepwise** — Specifies that training is done by least squares regression. The training starts as in the forward model, but it might remove effects already in the model.

The default value is **Stepwise**.

Selection Options

- **Effect-selection criterion** — Specifies the criterion that the procedure uses to determine the order in which effects enter or leave at each step of the selection method. Here are the possible values:
 - **AIC** — Specifies Akaike's Information Criterion. AIC is based on the Kullback-Leibler information measure of discrepancy between the true distribution of the response variable and the distribution specified by the model. Smaller values indicate better models, and AIC values can become negative.
 - **AICC** — Specifies Corrected Akaike's Information Criterion. AICC adjusts the AIC value by accounting for sample size. Extra features penalize AICC more than AIC. As the sample size increases, AICC and AIC converge.
 - **SBC (BIC)** — Specifies Schwarz Bayesian Criterion (SBC), also known as the Bayesian Information Criterion (BIC). Unexplained variations in the response variable and the number of features increase the value of the SBC. Smaller values indicate better models. SBC penalizes total number of model parameters more strongly than AIC and therefore tends to choose more sparse models.
 - **Significance level** — Specifies the standard statistical significance level criterion.

If the **Selection method** above is **LASSO**, this option is unavailable. The default value is **SBC (BIC)**.

- **Selection-process stopping criterion** — Specifies the criterion to stop the selection process. Here are the possible values:
 - **AIC** — Specifies Akaike's Information Criterion. AIC is based on the Kullback-Leibler information measure of discrepancy between the true distribution of the response variable and the distribution specified by the model. Smaller values indicate better models, and AIC values can become negative. Selection stops at the step where AIC starts to increase.
 - **AICC** — Specifies Corrected Akaike's Information Criterion. AICC adjusts the AIC value by accounting for sample size. Extra features penalize AICC more than AIC. Smaller values indicate better models. As the sample size increases, AICC and AIC converge. Selection stops at the step where AICC starts to increase.
 - **SBC (BIC)** — Specifies Schwarz Bayesian Criterion (SBC), also known as the Bayesian Information Criterion (BIC). Unexplained variations in the response variable and the number of features increase the value of the SBC. SBC penalizes the total number of model parameters more strongly than AIC and therefore tends to choose more sparse models. Selection stops at the step where SBC starts to increase.

- **Significance level** — Specifies the standard statistical significance level criterion. Selection stops at the step where significance level exceeds the specified level. The default value is 0.05.
- **Validation ASE** — Average square error (ASE) of the model is computed by using the validation data, and selection stops at the step where the validation ASE starts to increase. This method requires partitioned data.

The default value is **SBC (BIC)**.

- **Model-selection criterion** — Specifies the criterion to choose the model (from the list of models at each step of the selection process) that yields the best value of the specified criterion. If the optimal value of the specified criterion occurs for models at more than one step, then the model that has the smallest number of parameters is chosen. Here are the possible values:
 - **AIC** — Specifies Akaike's Information Criterion. AIC is based on the Kullback-Leibler information measure of discrepancy between the true distribution of the response variable and the distribution specified by the model. Smaller values indicate better models, and AIC values can become negative. The model that has the minimal AIC value is chosen.
 - **AICC** — Specifies Corrected Akaike's Information Criterion. AICC adjusts the AIC value by accounting for sample size. Extra features penalize AICC more than AIC. As the sample size increases, AICC and AIC converge.
 - **SBC (BIC)** — Specifies Schwarz Bayesian Criterion (SBC), also known as the Bayesian Information Criterion (BIC). Unexplained variations in the response variable and the number of features increase the value of the SBC. SBC penalizes total number of model parameters more strongly than AIC and therefore tends to choose more sparse models. The model that has the minimal SBC value is chosen.
 - **Validation ASE** — Average square error (ASE) of the model is computed by using the validation data, and the model that has the minimal validation ASE is chosen. This method requires partitioned data.

The default value is **SBC (BIC)**.

- **Entry significance level** — Specifies the significance level for adding variables in the forward and stepwise directions. The default value is 0.05.
- **Stay significance level** — Specifies the significance level for removing variables in backward and stepwise directions. The default value is 0.05.
- **Maximum number of effects** — Specifies the maximum number of effects in any model that is considered during the selection process. If a model at some step of the selection process contains the specified maximum number of effects, then no additional effects are considered. If **Maximum number of effects** is set to 0 (the default value), this option is ignored.
- **Minimum number of effects** — Specifies the minimum number of effects in any model that is considered during the backward selection process. If **Minimum number of effects** is set to 0 (the default value), this option is ignored.
- **Maximum number of steps** — Specifies the maximum number of selection steps that are performed. If **Maximum number of steps** is set to 0 (the default value), this option is ignored.
- **Hierarchy** — Specifies whether and how the model hierarchy requirement is applied. Model hierarchy refers to the requirement that, for any term to be in the model, all model effects that are contained in the term must be present in the model. For example, in order for the interaction A*B to enter the model, the main

effects A and B must also be in the model. Likewise, neither effect A nor effect B can leave the model while the interaction A*B is in the model. Here are the possible values:

- (none)** — Specifies that the hierarchy requirement is never applied.
- All variables** — Specifies that both interval and nominal variables are subject to the model hierarchy requirement.
- Class variables** — Specifies that only nominal variables are subject to the model hierarchy requirement.

The default value is **(none)**.

Optimization Options

- **Optimization technique** — Specifies the optimization method used when fitting a model. Here are the possible values:
 - (none)**
 - Conjugate-gradient**
 - Double-dogleg**
 - Dual quasi-Newton**
 - Nelder-Mead simplex**
 - Newton-Raphson**
 - Newton-Raphson with ridging**
 - Trust-region**

The default value is **Newton-Raphson with ridging**. For more information, see the PROC NL MIXED documentation in *SAS/STAT 15.1 User's Guide*.

.....

Note: Optimization options are not available for the **LASSO** selection method. LASSO uses the Nesterov algorithm as the default optimization technique.

.....

- **Maximum number of iterations** — Specifies the maximum number of iterations of any optimization. The default value depends on the optimization technique used:

Optimization Technique	Default Value
Conjugate-gradient	400
Double-dogleg	200
Dual quasi-Newton	200
Nelder-Mead simplex	1000
Newton-Raphson	50

Newton-Raphson with ridging	50
Trust-region	50

To use the default value, leave **Maximum number of iterations** blank or use a dot.

- **Maximum number of function evaluations** — Specifies the maximum number of function calls of any optimization. The default value depends on the optimization technique used:

Optimization Technique	Default Value
Conjugate-gradient	1000
Double-dogleg	500
Dual quasi-Newton	500
Nelder-Mead simplex	3000
Newton-Raphson	125
Newton-Raphson with ridging	125
Trust-region	125

To use the default value, leave **Maximum number of function evaluations** blank or use a dot.

- **Maximum CPU time in second** — Specifies an upper limit of CPU time (in seconds) for the optimization process. The default value is the largest floating-point double representation of your computer. To use the default value, leave **Maximum CPU time in second** blank or use a dot.
- **Minimum number of iterations** — Specifies the minimum number of iterations in any optimization. The default value is 1. To use the default value, leave **Minimum number of iterations** blank or use a dot.
- **Normalize objective function** — Specifies whether the objective function should be normalized during optimization by the reciprocal of the used frequency count. By default, this option is selected.

Convergence Options

- **Absolute function convergence** — Specifies the threshold for absolute function convergence. The default value is the negative square root of the largest double-precision value. To use the default value, leave **Absolute function convergence** blank or use a dot.

- **Absolute function difference convergence** — Specifies the threshold for absolute function difference convergence. The default value is 0. To use the default value, leave **Absolute function difference convergence** blank or use a dot.
- **Absolute gradient convergence** — Specifies the threshold for absolute gradient convergence. The default value is 1E-5. To use the default value, leave **Absolute gradient convergence** blank or use a dot.
- **Relative function difference convergence** — Specifies the relative function difference convergence. The default value is the machine precision times 2. To use the default value, leave **Relative function difference convergence** blank or use a dot.
- **Relative gradient convergence** — Specifies the threshold for relative gradient convergence. The default value is 1E-8. To use the default value, leave **Relative gradient convergence** blank or use a dot.

Lift Calculation Property

- **Use the exact percentile method for lift calculations** — Specifies whether to use the exact percentile method for calculating lift and related assessment measures. When this property is deselected, the iterative method is used. The exact method should be used if there are convergence errors when using the iterative method.

Binary Classification Cutoff

- **Specify node binary classification cutoff** — Specifies whether to use the binary classification cutoff specified below for the node. If this property is deselected, the project binary classification cutoff is used for determining the predicted value for a binary target based on the posterior probabilities. By default, this property is deselected.
- **Node binary classification cutoff** — Specifies the cutoff to use in the node for determining the predicted value for a binary target based on the posterior probabilities. This option is available only if **Specify node binary classification cutoff** is selected. The default value is 0.5.

Note: The **Rules** settings in the Project Settings window enable you to specify a new cutoff value by selecting **Override the default classification cutoff**. If **Specify node binary classification cutoff** is deselected, then the value that you specified in the Project Settings window is used. If **Specify node binary classification cutoff** is selected, then the value that you specified in **Node binary classification cutoff** is used.

Post Training Properties

Changing the following properties will not retrain the model.

Model Interpretability — Specifies the method of model interpretation in order to show how the model responds to changing inputs.

■ Global Interpretability

- **Variable importance** — Specifies whether to display a table of the relative importance values for the input variables. The surrogate variable importance values are calculated using a one-level decision tree for each input to predict the predicted value as a global surrogate model. For tree-based supervised learning models, the model variable importance table is displayed as well.
- **PD plots** — Specifies whether to display partial dependency (PD) plots that show the relationship between model input variables and their predictors. PD plots are generated for only up to the top ten variables in the relative importance table. Here is how to determine the variables that are displayed in the PD plot: for tree-based models (forest, decision tree, and gradient boosting), the model variable importance table is used. For other supervised learning models, the surrogate variable importance table is used.

■ Local Interpretability


- **ICE plots** — Specifies whether to display Individual Conditional Expectation (ICE) plots. ICE plots can reveal interesting subgroups and interactions between model variables. ICE plots are generated for only up to the top ten variables in the relative importance table. Here is how to determine the variables that are displayed in the ICE plot: for tree-based models (forest, decision tree, and gradient boosting), the model variable importance table is used. For other supervised learning models, the surrogate variable importance table is used.
- **LIME** — Specifies whether to use the Locally Interpretable Model-Agnostic Explanation (LIME) method to explain the model predictions.
- **Kernel SHAP** — Specifies whether to use the Kernel Shapley additive explanations (Kernel SHAP) method to explain the model predictions.
- **Maximum number of Kernel SHAP variables** — Specifies the maximum number of Kernel SHAP variables to plot. Possible values range from 1 to 100. The default value is 20.
- **Specify instances to explain** — Specifies the individual observations that you want to explain. The **Random** observations option provides explanations for five randomly selected observations. The **Specify up to 5** observations option enables you to provide IDs for the specific observations in the data. The IDs should be unique values of the variable that has the role **Key** in the metadata. If no variable is assigned the role **Key**, then you can use the variable `_dmIndex_` to specify the individual observations.
- **LIME/Kernel SHAP Tables**
 - **Explainer Information table** — Specifies whether the Explainer Information tables for the LIME and Kernel SHAP methods should be included in the node results. The Explainer Information table contains

information about what settings were used to perform the LIME and Kernel SHAP methods.

- **Explainer Fidelity table** — Specifies whether the Explainer Fidelity tables for the LIME and Kernel SHAP methods should be included in the node results. The Explainer Fidelity table contains information about the accuracy of the local explanation predictions, compared to the black-box machine learning model.
- **PD/ICE Options**
 - **Maximum number of variables** — Specifies the maximum number of input variables to use to generate the PD and ICE plots. The input variables are selected based on their relative importance. Possible values range from 1 to 10. The default value is 5.
 - **Number of observations** — Specifies the maximum number of observations to sample for the PD and ICE plots. The default value is 1000.
 - **Number of tick points** — Specifies the number of tick points for an interval analysis variable to generate for the PD and ICE plots. Possible values range from 3 to 100. The default value is 50.
 - **Truncate lower tail** — Specifies under what percentage to discard the values of the interval analysis variables for the PD and ICE plots. Possible values range from 0 to 99. The default value is 5%.
 - **Truncate upper tail** — Specifies above what percentage to discard the values of the interval analysis variables for the PD and ICE plots. Possible values range from 0 to 99. The default value is 95%.
- **Seed** — Specifies the seed for generating random numbers. The seed is used for selecting individual observations to explain, selecting observations to generate PD and ICE plots, and simulating data for LIME and Kernel SHAP.



Logistic Regression Results

After running the node, open the Results window by right-clicking the node and selecting **Results**.

You can download the data that is used to build the charts and tables in the Results window by clicking . A CSV file with the chart or table data will immediately begin downloading.

In the **Output Data** tab, you can view, save, and explore the data that is generated by the node. For more information, see [Explore and Visualize](#).

Several charts have detailed descriptions available that explain how the charts are generated. The descriptions also provide information about how to interpret the charts that is custom to your particular data and models. To access these detailed descriptions:

- Expand an individual chart by clicking . The detailed description is in the right panel.
- Select  on an individual chart. A window appears with the detailed description.

Note: The detailed descriptions are available only in the English language.

Node

- **t Values by Parameter** — Displays bar charts for the t values in the final model. The bars are color-coded to indicate the algebraic signs of the coefficients.
- **Parameter Estimates** — Displays a table of the various statistics related to estimates for the parameters. These statistics include the t value, sign, estimate, absolute estimate, the p-value, chi-square value, and standard error.
- **Selection Summary** — Displays a table of the iterations, adding in and removing effects. This table also includes the SBC and optimal SBC values.
- **Regression Fit Statistics** — Displays a table of data on the logistic regression fit statistics. The table includes the statistic, a corresponding qualitative description, and the training, validation, and testing values.
- **Node Score Code** — Displays the SAS score code that was created by the node. The SAS score code can be used outside the Model Studio environment to score new data.
- **Path Score Code** — Displays the SAS score code that was created by the node, as well as all preceding nodes. The SAS model score code can be used outside the Model Studio environment to score new data.
- **DS2 Package Code** — Displays the SAS DS2 package code that was created by the node. The SAS score code can be used outside the Model Studio environment in custom user applications DS2 package code is used when publishing models to SAS Micro Analytic Service..
- **Score Inputs** — Displays a table of data on the input variables for scoring calculations. The table includes the variables' name, role, level, type, label, format, and length.
- **Score Outputs** — Displays a table of data on the predicted response variable for scoring calculations. The table includes the variables' name, role, type, format and length, as well as the creator, creator GUID, and function.
- **Training Code** — Displays the SAS code that Model Studio used to train the node.
- **Properties** — Specifies the various properties that you selected before running the node.
- **Output** — Displays the SAS output of the logistic regression run.

Assessment

- **Lift Reports** — Displays the cumulative lift as a function of the depth for the model. The cumulative lift is given for each of the data roles. To examine other statistics as a function of depth, use the drop-down menu in the upper right corner. Other statistics include lift, gain, captured response percentage, cumulative captured response percentage, response percentage, and cumulative response percentage. This result is displayed only if the target is a class variable.
- **ROC Reports** — Displays the ROC (receiver operating characteristic) chart for a model, giving the sensitivity as a function of 1-specificity. The sensitivity is given for each of the data roles. To examine other statistics, use the drop-down menu in the upper right corner. Other statistics include accuracy and F1 score. This result is displayed only if the target is a class variable.

- **Fit Statistics** — Displays a table of the fit statistics for the model, broken down by data role.
- **Event Classification** — Displays the confusion matrix at various cutoff values for each partition. The confusion matrix contains true positives (events that are correctly classified as events), false positives (non-events that are classified as events), false negatives (events that are classified as non-events), and true negatives (non-events that are correctly classified as non-events). The classification cutoffs that are used are the default value of 0.5, the Kolmogorov-Smirnov cutoff value for each partition, and the **Node binary classification cutoff** that is specified for binary targets. Use the drop-down menu to view the information in the chart as percentages or counts. You can also view the information summarized in a table. This result is displayed only if the target is a class variable.
- **Nominal Classification** — Displays either the percentage of observations or the number of observations that predict each level of the target variable. The plot is segmented by target level and partition level. The target level with the greatest predicted probability is the predicted outcome. This result is displayed only if the target is a nominal variable.

Model Interpretability

- **Surrogate Model Variable Importance** — Displays a table of the relative importance of the input variables, including variable level and label. Relative importance is calculated using a one-level decision tree for each input variable to predict the predicted value as a global surrogate model.
- **PD Plot** — Displays the functional relationship between the input variable and the model prediction. For interval input variables, the 95% confidence interval for the average target prediction is indicated by the shaded band around the line. The X axis includes a heat map that shows the distribution of the input variable. The extreme values of the input variables are removed by truncating the lower and upper tails of its distribution. The amount of truncation can be controlled by specifying values for the **Truncate lower tail** and the **Truncate upper tail** properties.
- **PD and ICE Overlay Plot** — Displays the functional relationship between the input variable and the model prediction as well as the functional relationship between individual observations and the model prediction. For each individual observation, the corresponding ICE curve displays values of the input variable on the X axis and the corresponding predicted probability or prediction of the target variable on the Y axis, holding the other input variable values constant at that observation. The X axis includes a heat map that shows the distribution of the input variable. The extreme values of the input variables are removed by truncating the lower and upper tails of its distribution. The amount of truncation can be controlled by specifying values for the **Truncate lower tail** and the **Truncate upper tail** properties.
- **LIME Explanations** — The LIME algorithm calculates a more easily interpretable linear model around an individual observation. The chart displays the regression estimates for the input variables that are selected in the local surrogate linear regression model. The input variables are ordered by significance such that the most important variable for the local regression model is at the bottom of the chart. The LASSO technique is used to select the most significant predictors from the set of input variables that are used to train the model. Each nominal input variable is binary-encoded based on whether it matches the level of the individual observation. A positive estimate indicates that the observed value of the input variable increases the predicted probability of the event.

- **Kernel SHAP Values** — For each individual observation, an input variable's Kernel SHAP value is the contribution of the observed value of the input variable to the predicted probability or prediction of the target variable. The Kernel SHAP values are the regression coefficients that are obtained by fitting a weighted least squares regression. The Kernel SHAP values of all input variables sum to one. The input variables are ordered by significance such that the most important variable according to the absolute Kernel SHAP values is at the top of the chart. Each nominal input is binary-encoded based on whether it matches the individual observation. Interval inputs are binary encoded based on their proximity to the individual observation.
- **LIME Information Table** — Displays information about the settings used to perform the LIME method.
- **Kernel SHAP Information Table** — Displays information about the settings used to perform the Kernel SHAP method.
- **LIME Fidelity Table** — Displays information about the accuracy of the local explanation predictions that are calculated using the LIME method, compared to the black-box machine learning model. The model prediction, the explainer prediction, and the explainer root mean square error (RMSE) are displayed.
- **Kernel SHAP Fidelity Table** — Displays information about the accuracy of the local explanation predictions that are calculated using the Kernel SHAP method, compared to the black-box machine learning model. The model prediction, the explainer prediction, and the explainer root mean square error (RMSE) are displayed.

Manage Variables

<i>Overview of Manage Variables</i>	149
<i>Manage Variables Properties</i>	149
<i>Manage Variables Results</i>	150

Overview of Manage Variables

The **Manage Variables** node is a preprocessing node that enables you to make modifications to the metadata while it is within a Model Studio pipeline. Available options include a subset of the options that are under the **Data** tab. For example, you cannot modify the target variable attributes, change the target variable, modify the partition indicator variable, or modify the key variable. This is to ensure that models in the **Model Comparison** node and Pipeline Comparison tab remain comparable. That is, this ensures that all models are based on the same partitioned data and same target variable. For more information about the **Data** tab, see [Data Management Overview](#) in the *Model Studio User's Guide*.

To modify the metadata, you must first run the **Manage Variables** node, which creates a copy of the incoming metadata. The metadata in a pipeline is dynamically generated.

The results consist of tables that identify the incoming variable metadata as well as the changes that were made interactively.


Manage Variables Properties

To manage variables, place the **Manage Variables** node on the diagram, right-click the node, and select **Run**. Let the node run, right-click the node again, and then select the **Manage Variables** option. This opens a window that lists the set of

variables that you can edit (change the role, level, order, and so on). After editing a variable, save the edit in the upper right corner, and click **Close**.

Manage Variables Results

After running the node or editing the metadata, open the Results window by right-clicking the node and selecting **Results**.

You can download the data that is used to build the charts and tables in the Results window by clicking . A CSV file with the chart or table data will immediately begin downloading.

In the **Output Data** tab, you can view, save, and explore the data that is generated by the node. For more information, see [Explore and Visualize](#).

Summary

- **Incoming Variables** — Displays a list of the incoming variables and their attributes from the data set, including input and target variables. This list also includes summary statistics.
- **Metadata Changes** — Displays a list of the attribute alterations made to the variables. Only variables that have been changed appear here.
- **Properties** — Specifies a list of the properties used by the node. Note that these are not set by you. They are static properties.
- **Output** — Displays the SAS output of the manage variables run.

Model Comparison

<i>Overview of Model Comparison</i>	151
<i>Model Comparison Properties</i>	152
General Properties	152
<i>Model Comparison Results</i>	154

Overview of Model Comparison

The **Model Comparison** node is a Model Studio node that is automatically added to a pipeline when a Supervised Learning node is also added. The **Model Comparison** node enables you to compare the performance of competing models that are within the same pipeline using various benchmarking criteria.

There are many assessment criteria that can be used to compare models. For class targets, these include measures of error, lift-based measures, and measures derived from the Receiver Operating Characteristic (ROC) curve. You can select the measure, and specify the depth to use when applying a lift-based measure or the cutoff to use when applying an ROC-based measure. For interval targets, there are various measures of error available for choosing the champion model. All measures of assessment are computed for each of the data partitions that are available (train, validate, and test). You can also select which data partition to use for selecting the champion. By default, the node uses the criterion specified in the **Project Settings** menu.

Note: If multiple supervised learning nodes are connected to the **Model Comparison** node, then only successfully completed models are compared. Models that have failed or been stopped are not considered. The log of the Model Comparison node lists the models that were evaluated, as well as the selected model. The selected model from the **Model Comparison** node of each pipeline is added to the **Pipeline Comparison** tab. This enables you to compare models from the different pipelines in your project and to select a champion model. To add models that were not selected by the **Model Comparison** node to the **Pipeline Comparison** tab, right-click on the given model and select **Add challenger model**.

Model Comparison Properties

General Properties

- **Class selection statistic** — Specifies the fit statistic to use for selecting the champion model when there is a class target. Here are the possible values:
 - Accuracy**
 - Area under curve (C statistic)**
 - Average squared error**
 - Captured response**
 - Cumulative captured response**
 - Cumulative lift**
 - F1 score**
 - False discovery rate**
 - False positive rate**
 - Gain**
 - Gini**
 - Kolmogorov-Smirnov statistic (KS)**
 - Lift**
 - Misclassification (Event)**
 - Misclassification (MCE)**
 - Misclassification at cutoff**
 - Multiclass log loss**
 - ROC separation**
 - Root average squared error**
 - Use rule from project settings**

If **Use rule from project settings** is selected (it is selected by default), the class selection statistic chosen in the **Project Settings** menu in the upper right corner is used. In **Project Settings**, the default is **Kolmogorov-Smirnov statistic (KS)**.

Note: **Misclassification (MCE)** is the true misclassification rate. That is, every observation where the observed target level is predicted to be a different level counts in the misclassification rate. **Misclassification (Event)** considers only the classification of the event level versus all other levels. Thus, a non-event level that is classified as another non-event level does not count in the misclassification. For binary targets, these two measures are the same. The

Misclassification (Event) statistic is computed in the context of the ROC report. That is, at each cutoff value, this measure is calculated. The **Misclassification at cutoff** statistic is not supported for nominal targets. If you specify **Misclassification at cutoff** with a nominal target, then the **Misclassification (Event)** statistic is automatically applied.

For more information, see the [ASSESS procedure](#) documentation in *SAS Visual Statistics: Procedures*.

- **Interval selection statistic** — Specifies the fit statistic to use for selecting the champion model when there is an interval target. Here are the possible values:

- Average squared error**
- Root average squared error**
- Root mean absolute error**
- Root mean squared logarithmic error**
- Use rule from project settings**

If **Use rule from project settings** is selected (it is selected by default), the interval selection statistic chosen in the **Project Settings** menu in the upper right corner is used. The default in **Project Settings** is **Average squared error**.

For more information, see the [ASSESS procedure](#) documentation in *SAS Visual Statistics: Procedures*.

- **Selection partition** — Specifies from which of the data roles the fit statistic is used for selecting the champion model. Here are the possible values:

- Test**
- Train**
- Use rule from project settings**
- Validate**

If **Use rule from project settings** is selected (it is selected by default), the selection partition chosen in the **Project Settings** menu in the upper right corner is used. In the **Project Settings** menu, the default partition used is **Test**, then **Validate**, then **Train**, based on availability.

- **Selection Depth** — Specifies the depth to use when there is a class target. This option is available for the following class selection statistics:

- Lift**
- Cumulative Lift**
- Gain**
- Gini**
- Captured response**
- Cumulative captured response**

Here are the possible values:

- 5**
- 10**
- 15**
- 20**

- **Use rule from project settings**

If **Use rule from project settings** is selected (it is by default), the selection depth chosen in the **Project Settings** menu in the upper right corner is used. The default value in **Project Settings** is 10.


- **ROC-based cutoff** — Specifies the posterior probability cutoff to use for classifying a binary target when there is a class target. This option is available for the following class selection statistics:

- **F1 score**
- **False discovery rate**
- **False positive rate**
- **Accuracy**
- **ROC separation**
- **Misclassification (Event)**

Possible values are all multiples of 0.05 (from 0.05 to 0.95, inclusive), and **Use rule from project settings**. If **Use rule from project settings** is selected (it is selected by default), the cutoff chosen in the **Project Settings** menu in the upper right corner is used. The default in **Project Settings** is 0.50.


Model Comparison Results

After running the node, open the Results window by right-clicking the node and selecting **Results**.

You can download the data that is used to build the charts and tables in the Results window by clicking . A CSV file with the chart or table data will immediately begin downloading.

In the **Output Data** tab, you can view, save, and explore the data that is generated by the node. For more information, see [Explore and Visualize](#).

Node

- **Model Comparison** — Displays a table of the results of the model comparison. The table lists the selected model, as well as the various comparison statistics. Click  to view a more extensive list of comparison statistics.

The Model Comparison table is a combination of the fit statistics table, the lift statistics at the specified depth, and the ROC-based statistics at the specified cutoff value for all models being compared.

- **Properties** — Displays a table of the properties that are used to select the best model. This includes the property names and values.

Assessment

- **Predicted Reports** — Displays the predicted mean of the target as a function of the actual target mean for each model and each of the data roles. To examine the predicted mean and the actual target mean both as a function of depth, use the drop-down menu in the upper right corner. This result is displayed only if the target is an interval variable.

- **Lift Reports (for class target)** — Displays the response percentage as a function of the depth for the models. To examine other statistics as a function of depth, use the drop-down menu in the upper right corner. Other statistics include cumulative lift, lift, gain, captured response percentage, cumulative captured response percentage, and cumulative response percentage.
- **ROC Reports (for class target)** — Displays accuracy as a function of the cutoff for the models and each of the data roles. You can also examine the F1 score as a function of the cutoff or the ROC chart that displays the sensitivity as a function of 1-specificity. To examine these other statistics, use the drop-down menu in the upper right corner.
- **Fit Statistics** — Displays the fit statistics for the models, broken down by data role.

Model Composer

<i>Overview of Model Composer</i>	157
<i>Model Composer Properties</i>	158
Models to Autotune	158
General Properties	158
Autotuning Options	158
Lift Calculation Property	160
Binary Classification Cutoff	161
Post Training Properties	161
<i>Model Composer Results</i>	163

Overview of Model Composer

The **Model Composer** node is a Supervised Learning node. It automatically tunes hyperparameters for multiple model types concurrently with optimal allocations of evaluations performed across multiple rounds of hyperparameter tuning. The number of evaluations that are allocated to each model type in each round is determined by Bayesian optimization. The top model is then selected from the best configuration of hyperparameters for each model type. The **Model Composer** node supports both class and interval targets. The following models can be autotuned:

- decision tree
- forest
- gradient boosting
- neural network
- Bayesian network (class target only)
- support vector machine (binary target only)

Model Composer Properties

Models to Autotune

- **Decision tree** — Specifies whether to include a decision tree model in the autotuning process.
- **Forest** — Specifies whether to include a forest model in the autotuning process.
- **Gradient boosting** — Specifies whether to include a gradient boosting model in the autotuning process.
- **Neural network** — Specifies whether to include a neural network model in the autotuning process.
- **Bayesian network (class target only)** — Specifies whether to include a Bayesian network model in the autotuning process. This option is available only with a class target.
- **Support vector machine (binary target only)** — Specifies whether to include an SVM model in the autotuning process. This option is available only with a binary target.

General Properties

- **Number of autotuning rounds** — Specifies the number of rounds to tune multiple model types concurrently with optimal allocations.
- **Number of evaluations per round** — Specifies the total number of evaluations to allocate between model types in each round of autotuning. There must be at least two evaluations per round per model type.

Autotuning Options

- **Seed** — Specifies the seed for generating random numbers that are used for autotuning.
- **Search Options** — Specifies the autotune search method and properties. The following options are available:
 - **Search method** — Specifies the tuning search method. Here are the possible values:
 - **Bayesian** — Specifies the Bayesian method. This method uses priors to seed the iterative optimization.

- **Genetic algorithm** — Specifies the genetic algorithm method. This method uses an initial Latin Hypercube sample that seeds a genetic algorithm to generate a new population of alternative configurations at each iteration.
- **Grid** — Specifies the grid method. This method performs an exhaustive search of all configurations. Each hyperparameter of interest is discretized into a desired set of values to be studied. Models are trained and assessed for all combinations of values across all the hyperparameters (thus forming a multi-dimensional “grid”).
- **Latin hypercube sample** — Specifies the Latin Hypercube method. This method performs an optimized grid search that is uniform in each tuning parameter, but random in combinations.
- **Random** — Specifies the Random method. This method generates a single sample of purely random configurations.

The default value is **Genetic algorithm**.

- **Number of evaluations per iteration** — Specifies the number of tuning evaluations in one iteration. This option is available only if the **Search method** is **Genetic algorithm** or **Bayesian**. The default value is 10.
- **Maximum number of evaluations** — Specifies the maximum number of tuning evaluations. This option is available only if the **Search method** is **Genetic algorithm** or **Bayesian**. The default value is 50.
- **Maximum number of iterations** — Specifies the maximum number of tuning iterations. This option is available only if the **Search method** is **Genetic algorithm** or **Bayesian**. The default value is 5.
- **Maximum number of points in model** — Specifies the maximum number of points in the model. This option is available only if the **Search method** is **Bayesian**. The default value is 100.
- **Sample size** — Specifies the sample size. This option is available only if the **Search method** is **Random** or **Latin hypercube sample**. The default value is 50.
- **General Options** — Specifies the general properties for autotuning. The following options are available:
 - **Validation method** — Specifies the validation method for finding the objective value. Note that if your data is partitioned, then that partition is used. **Validation method**, **Validation data proportion**, and **Cross validation number of folds** are all ignored. Here are the possible values:
 - **Partition** — Specifies using a single partition of a training set. With partition, you specify proportions to use for randomly assigning observations to each role.
 - **K-fold cross validation** — Specifies using the cross validation method. In cross validation, each model evaluation requires k training executions (on $k-1$ data folds) and k scoring executions (on one holdout fold). This increases the evaluation time by approximately a factor of k .

For small to medium data tables, cross validation provides, on average, a better representation of error across the whole data table. **Partition** is the default value.

- **Training data proportion** — Specifies the proportion of training data to be used for the **Partition** validation method. The default value is 0.7.

- **Validation data proportion** — Specifies the proportion of data to be used for the **Partition** validation method. The default value is 0.3.
- **Cross validation number of folds** — Specifies the number of partition folds in the cross validation process (the k defined above). Possible values range from 2 to 20. The default value is 5.
- **Class target objective function** — Specifies the objective function to optimize for tuning parameters for a class target. Here are the possible values:
 - **Area under the curve**
 - **Average squared error**
 - **F0.5 score**
 - **F1 score**
 - **Gamma**
 - **Gini coefficient**
 - **Kolmogorov-Smirnov statistic**
 - **Misclassification rate**
 - **Multi-class log loss**
 - **Root average squared error**
 - **Tau**

The default value is **Kolmogorov-Smirnov statistic**.

- **Interval target objective function** — Specifies the objective function to optimize for tuning parameters for an interval target. Here are the possible values:
 - **Average squared error**
 - **Mean absolute error**
 - **Mean squared logarithmic error**
 - **Root average squared error**
 - **Root mean absolute error**
 - **Root mean squared logarithmic error**

The default value is **Average squared error**.

- **Maximum time (minutes)** — Specifies the maximum time (in minutes) for the optimization tuner. The default value is 60.
- **Maximum training time for single model (minutes)** — Specifies the maximum training time (in minutes) for the single model.

Lift Calculation Property

- **Use the exact percentile method for lift calculations** — Specifies whether to use the exact percentile method for calculating lift and related assessment measures. When this property is deselected, the iterative method is used. The

exact method should be used if there are convergence errors when using the iterative method.

Binary Classification Cutoff

- **Specify node binary classification cutoff** — Specifies whether to use the binary classification cutoff specified below for the node. If this property is deselected, the project binary classification cutoff is used for determining the predicted value for a binary target based on the posterior probabilities. By default, this property is deselected.
- **Node binary classification cutoff** — Specifies the cutoff to use in the node for determining the predicted value for a binary target based on the posterior probabilities. This option is available only if **Specify node binary classification cutoff** is selected. The default value is 0.5.

.....

Note: The **Rules** settings in the Project Settings window enable you to specify a new cutoff value by selecting **Override the default classification cutoff**. If **Specify node binary classification cutoff** is deselected, then the value that you specified in the Project Settings window is used. If **Specify node binary classification cutoff** is selected, then the value that you specified in **Node binary classification cutoff** is used.

.....

Post Training Properties

Changing the following properties will not retrain the model.

Model Interpretability — Specifies the method of model interpretation in order to show how the model responds to changing inputs.


- **Global Interpretability**
 - **Variable importance** — Specifies whether to display a table of the relative importance values for the input variables. The surrogate variable importance values are calculated using a one-level decision tree for each input to predict the predicted value as a global surrogate model. For tree-based supervised learning models, the model variable importance table is displayed as well.
 - **PD plots** — Specifies whether to display partial dependency (PD) plots that show the relationship between model input variables and their predictors. PD plots are generated for only up to the top ten variables in the relative importance table. Here is how to determine the variables that are displayed in the PD plot: for tree-based models (forest, decision tree, and gradient boosting), the model variable importance table is used. For other supervised learning models, the surrogate variable importance table is used.
- **Local Interpretability**
 - **ICE plots** — Specifies whether to display Individual Conditional Expectation (ICE) plots. ICE plots can reveal interesting subgroups and interactions between model variables. ICE plots are generated for only up to the top ten variables in the relative importance table. Here is how to determine the

variables that are displayed in the ICE plot: for tree-based models (forest, decision tree, and gradient boosting), the model variable importance table is used. For other supervised learning models, the surrogate variable importance table is used.

- **LIME** — Specifies whether to use the Locally Interpretable Model-Agnostic Explanation (LIME) method to explain the model predictions.
- **Kernel SHAP** — Specifies whether to use the Kernel Shapley additive explanations (Kernel SHAP) method to explain the model predictions.
- **Maximum number of Kernel SHAP variables** — Specifies the maximum number of Kernel SHAP variables to plot. Possible values range from 1 to 100. The default value is 20.
- **Specify instances to explain** — Specifies the individual observations that you want to explain. The **Random** observations option provides explanations for five randomly selected observations. The **Specify up to 5** observations option enables you to provide IDs for the specific observations in the data. The IDs should be unique values of the variable that has the role **Key** in the metadata. If no variable is assigned the role **Key**, then you can use the variable `_dmIndex_` to specify the individual observations.
- **LIME/Kernel SHAP Tables**
 - **Explainer Information table** — Specifies whether the Explainer Information tables for the LIME and Kernel SHAP methods should be included in the node results. The Explainer Information table contains information about what settings were used to perform the LIME and Kernel SHAP methods.
 - **Explainer Fidelity table** — Specifies whether the Explainer Fidelity tables for the LIME and Kernel SHAP methods should be included in the node results. The Explainer Fidelity table contains information about the accuracy of the local explanation predictions, compared to the black-box machine learning model.
- **PD/ICE Options**
 - **Maximum number of variables** — Specifies the maximum number of input variables to use to generate the PD and ICE plots. The input variables are selected based on their relative importance. Possible values range from 1 to 10. The default value is 5.
 - **Number of observations** — Specifies the maximum number of observations to sample for the PD and ICE plots. The default value is 1000.
 - **Number of tick points** — Specifies the number of tick points for an interval analysis variable to generate for the PD and ICE plots. Possible values range from 3 to 100. The default value is 50.
 - **Truncate lower tail** — Specifies under what percentage to discard the values of the interval analysis variables for the PD and ICE plots. Possible values range from 0 to 99. The default value is 5%.
 - **Truncate upper tail** — Specifies above what percentage to discard the values of the interval analysis variables for the PD and ICE plots. Possible values range from 0 to 99. The default value is 95%.
- **Seed** — Specifies the seed for generating random numbers. The seed is used for selecting individual observations to explain, selecting observations to generate PD and ICE plots, and simulating data for LIME and Kernel SHAP.



Model Composer Results

After running the node, open the Results window by right-clicking the node and selecting **Results**.

You can download the data that is used to build the charts and tables in the Results window by clicking . A CSV file with the chart or table data will immediately begin downloading.

In the **Output Data** tab, you can view, save, and explore the data that is generated by the node. For more information, see [Explore and Visualize](#).

Several charts have detailed descriptions available that explain how the charts are generated. The descriptions also provide information about how to interpret the charts that is custom to your particular data and models. To access these detailed descriptions:

- Expand an individual chart by clicking . The detailed description is in the right panel.
- Select  on an individual chart. A window appears with the detailed description.

Note: The detailed descriptions are available only in the English language.

Node

- **Top Models** — Displays information about the top model for each model type that is specified. The model that is chosen is the model that has the best value of the specified objective function.
- **Configuration for Best Model** — Displays a table of the resulting values of parameters that were autotuned and the value of the specified objective function for the overall best model.
- **Best Configuration for Each Model Type** — Displays a table of the resulting values of parameters that were autotuned and the value of the specified objective function for each model type.
- **Evaluations Allocated per Round** — Displays a table with the number of evaluations that were run for each model in each round.
- **Node Score Code** — Displays the SAS code that was created by the node when the overall best model creates DATA step score code. The score code can be used outside the Model Studio environment to score new data.
- **Path Score Code** — Displays the SAS code that was created by the node, as well as all preceding nodes, when the overall best model creates DATA step score code. The score code can be used outside the Model Studio environment to score new data.
- **Path EP Score Code** — Displays the SAS code that was created by the node when the overall best model creates an analytic store for scoring. The score code can be used outside the Model Studio environment to score new data.
- **DS2 Package Code** — Displays the SAS DS2 package code that was created by the node. The score code can be used outside the Model Studio environment

to score new data. DS2 package code is used when publishing models to SAS Micro Analytic Service.

- **Score Inputs** — Displays a table of data on the input variables for scoring calculations. The table includes the variables' name, role, level, type, label, format, and length.
- **Score Outputs** — Displays a table of data on the predicted response variable for scoring calculations. The table includes the variables' name, role, type, format and length, as well as the creator, creator GUID, and function.
- **Training Code** — Displays the SAS code that Model Studio used to train the node.
- **Properties** — Specifies the various properties that you selected before running the node.
- **Output** — Displays the SAS output of the model composer run.

Assessment

- **Predicted Reports** — Displays the predicted and target mean as a function of the depth for the model. The predicted and target mean are given for each of the data roles. To examine the predicted mean as a function of the target mean, use the drop-down menu in the upper right corner. This result is displayed only if the target is an interval variable.
- **Lift Reports** — Displays the cumulative lift as a function of depth for the model. The cumulative lift is given for each of the data roles. To examine other statistics as a function of depth, use the drop-down menu in the upper right corner. Other statistics include lift, gain, captured response percentage, cumulative captured response percentage, response percentage, and cumulative response percentage. This result is displayed only if the target is a class variable.
- **ROC Reports** — Displays the ROC (receiver operating characteristic) chart for a model, giving the sensitivity as a function of 1-specificity. The sensitivity is given for each of the data roles. To examine other statistics, use the drop-down menu in the upper right corner. Other statistics include accuracy and F1 score. This result is displayed only if the target is a class variable.
- **Fit Statistics** — Displays the fit statistics for the model, broken down by data role.
- **Event Classification** — Displays the confusion matrix at various cutoff values for each partition. The confusion matrix contains true positives (events that are correctly classified as events), false positives (non-events that are classified as events), false negatives (events that are classified as non-events), and true negatives (non-events that are correctly classified as non-events). The classification cutoffs that are used are the default value of 0.5, the Kolmogorov-Smirnov cutoff value for each partition, and the **Node binary classification cutoff** that is specified for binary targets. Use the drop-down menu to view the information in the chart as percentages or counts. You can also view the information summarized in a table. This result is displayed only if the target is a class variable.
- **Nominal Classification** — Displays either the percentage of observations or the number of observations that predict each level of the target variable. The plot is segmented by target level and partition level. The target level with the greatest predicted probability is the predicted outcome. This result is displayed only if the target is a nominal variable.

Model Interpretability

- **Surrogate Model Variable Importance** — Displays a table of the relative importance of the input variables, including variable level and label. Relative importance is calculated using a one-level decision tree for each input variable to predict the predicted value as a global surrogate model.
- **Model Variable Importance** — Displays a table of the relative importance of the input variables, including variable level and label. This is the variable importance of the final model.
- **PD Plot** — Displays the functional relationship between the input variable and the model prediction. For interval input variables, the 95% confidence interval for the average target prediction is indicated by the shaded band around the line. The X axis includes a heat map that shows the distribution of the input variable. The extreme values of the input variables are removed by truncating the lower and upper tails of its distribution. The amount of truncation can be controlled by specifying values for the **Truncate lower tail** and the **Truncate upper tail** properties.
- **PD and ICE Overlay Plot** — Displays the functional relationship between the input variable and the model prediction as well as the functional relationship between individual observations and the model prediction. For each individual observation, the corresponding ICE curve displays values of the input variable on the X axis and the corresponding predicted probability or prediction of the target variable on the Y axis, holding the other input variable values constant at that observation. The X axis includes a heat map that shows the distribution of the input variable. The extreme values of the input variables are removed by truncating the lower and upper tails of its distribution. The amount of truncation can be controlled by specifying values for the **Truncate lower tail** and the **Truncate upper tail** properties.
- **LIME Explanations** — The LIME algorithm calculates a more easily interpretable linear model around an individual observation. The chart displays the regression estimates for the input variables that are selected in the local surrogate linear regression model. The input variables are ordered by significance such that the most important variable for the local regression model is at the bottom of the chart. The LASSO technique is used to select the most significant predictors from the set of input variables that are used to train the model. Each nominal input variable is binary-encoded based on whether it matches the level of the individual observation. A positive estimate indicates that the observed value of the input variable increases the predicted probability of the event.
- **Kernel SHAP Values** — For each individual observation, an input variable's Kernel SHAP value is the contribution of the observed value of the input variable to the predicted probability or prediction of the target variable. The Kernel SHAP values are the regression coefficients that are obtained by fitting a weighted least squares regression. The Kernel SHAP values of all input variables sum to one. The input variables are ordered by significance such that the most important variable according to the absolute Kernel SHAP values is at the top of the chart. Each nominal input is binary-encoded based on whether it matches the individual observation. Interval inputs are binary encoded based on their proximity to the individual observation.
- **LIME Information Table** — Displays information about the settings used to perform the LIME method.
- **Kernel SHAP Information Table** — Displays information about the settings used to perform the Kernel SHAP method.

- **LIME Fidelity Table** — Displays information about the accuracy of the local explanation predictions that are calculated using the LIME method, compared to the black-box machine learning model. The model prediction, the explainer prediction, and the explainer root mean square error (RMSE) are displayed.
- **Kernel SHAP Fidelity Table** — Displays information about the accuracy of the local explanation predictions that are calculated using the Kernel SHAP method, compared to the black-box machine learning model. The model prediction, the explainer prediction, and the explainer root mean square error (RMSE) are displayed.

Neural Network

<i>Overview of Neural Network</i>	167
<i>Neural Network Properties</i>	168
General Properties	168
Hidden Layer Options	169
Target Layer Options	169
Common Optimization Options	170
SGD Options (≥ 3 Hidden Layers)	171
Perform Early Stopping	172
Perform Autotuning	173
Lift Calculation Property	176
Binary Classification Cutoff	176
Post Training Properties	176
<i>Neural Network Results</i>	178

Overview of Neural Network

The **Neural Network** node is a Supervised Learning node. A neural network is a statistical model that is designed to mimic the biological structures of the human brain. A neural network consists of units (neurons) and connections between those units. The units are organized into layers:

- an input layer containing units that represent the input variables. The interval input variables can be standardized. (Optional)
- any number of hidden layers with any number of units in each that perform internal computations, providing the nonlinearity that makes neural networks powerful
- an output layer with one or more units (representing the target) that compute predicted values and compare those predicted values with the values of the target variables

Predictors can also be connected directly to the output layer. You can specify an activation function for each hidden layer and the target layer.

Units pass information to units in the subsequent layer through connections (links). Each unit produces a single computed value. For input units, this computed value is passed along the connections to the hidden units in the first hidden layer. The connections in a network have an associated numeric value called a *weight* or *parameter estimate*. The value for units in the hidden layers and output layers is calculated as a linear combination of the incoming weights. You can apply an activation function to the values of the units in the previous layer to transform the value. Hidden units pass along their computed value to the connections to hidden units in subsequent hidden layers. For the last hidden layer, the units are passed to the connections to output units. For output units, the computed value is the predicted value. The predicted value is compared with the target value to compute the error function, which the training attempts to minimize. The training methods minimize the error function by iteratively adjusting the values of the connection weights. The **Neural Network** node supports fully connected networks with up to 10 hidden layers.

Predictors can be connected directly to the output layer. You can specify an activation function for each hidden layer and the target layer.

You can view a visual representation of your network in the node results when you are using five or fewer hidden layers for the units. Connections that correspond to the top 200 weights are displayed. In the hidden layers and output layers, the color of a given unit represents the average weight for the incoming connections. The size of a given unit represents the average absolute weight for the incoming connections. The connections are colored by their weight and sized by their absolute weight. The Iteration Plot window shows how measures like objective, loss, and error for the validation partition (when it exists) change over the iterations.

Neural Network Properties

General Properties

- **Include missing inputs** — Specifies whether to impute missing values with the mean for interval inputs and to treat missing as a level for class inputs. By default, this is deselected.
- **Input standardization** — Specifies the method that is used to standardize the interval input variables. Here are the possible values:
 - (none)**
 - Midrange**
 - Z score**

The default value is **Midrange**.

- **Number of hidden layers** — Specifies the number of hidden layers to include in the neural network. Possible values range from 0 to 10. The default value is 1.

Hidden Layer Options

- **Use same number of neurons in hidden layers** — Specifies to use the same number of neurons in each hidden layer. By default, this is selected. If deselected, the **Custom Hidden Layer Options** menu enables you to vary the number of neurons in each hidden layer.
- **Number of neurons per hidden layer** — Specifies the number of neurons in each hidden layer. The default value is 50.
- **Hidden layer activation function** — Specifies the activation function to use in the hidden layers. Here are the possible values:
 - Exponential**
 - Identity**
 - Logistic**
 - ReLU**
 - Sine**
 - Softplus**
 - Tanh**
 - Varies**

Choosing **Varies** enables you to vary the activation function in each hidden layer, using the **Custom Hidden Layer Options** menu. The default value is **Tanh**.

Target Layer Options

- **Direct connections** — Specifies whether direct connections from nodes in the input layer to nodes in the output layer should be included in the network. By default, this is deselected.
- **Interval target standardization** — Specifies the method that is used to standardize the interval target variable. Here are the possible values:
 - (none)**
 - Midrange**
 - Z score**The default value is **Midrange**.
- **Interval target error function** — Specifies the target layer error function for interval targets. When there are no hidden layers, the normal distribution is used. Here are the possible values:
 - Gamma**
 - Normal**
 - Poisson**

The default value is **Normal**.

- **Interval target activation function** — Specifies the target layer activation function for interval targets. When there are no hidden layers, the identity function is used. Here are the possible values:
 - **Identity**
 - **Sine**
 - **Tanh**

If the **Interval target error function** is **Gamma** or **Poisson**, this option is unavailable, and the Exponential activation function is used. The default value is **Identity**.

Common Optimization Options

- **Optimization method** — Specifies the optimization method to use to train the network. This option is in effect only for networks with fewer than 6 hidden layers.
 - **Automatic** — Selects the optimization method based on the number of hidden layers. LBFGS is used when there are 2 or fewer hidden layers. SGD is used otherwise.
 - **LBFGS**
 - **SGD**

LBFGS cannot be used in networks with more than 2 hidden layers. The default value is **Automatic**.
- **Number of tries** — Specifies the number of times to train the network. This is done by using different initial estimates for the weights. Possible values range from 1 to 64. The default value is 1.
- **Maximum iterations** — Specifies the maximum number of iterations allowed within each try. The default value is 300.
- **Maximum time** — Specifies the maximum time (in minutes) allowed for optimizations. When this value is reached, the optimization terminates the search and returns the results. The value of 0 indicates that no maximum time is set. The default value is 0.
- **Random seed** — Specifies the random seed to use for generating random numbers to initialize the network weights. The default value is 12345.
- **L1 weight decay** — Specifies the weight decay for L1 regularization. The default value is 0.
- **L2 weight decay** — Specifies the weight decay for L2 regularization. The default value is 0.1.
- **Input layer dropout ratio** — Specifies the dropout ratio for the input layer when SGD optimization is used. The default value is 0.
- **Hidden layer dropout ratio** — Specifies the dropout ratio for the hidden layers when SGD optimization is used. The default value is 0.

SGD Options (≥ 3 Hidden Layers)

SGD Options — Specifies the configurations for stochastic gradient descent (SGD). This menu is available only if there are 3 or more hidden layers.

- **Learning rate** — Specifies the learning rate parameter for SGD optimization. The default value is 0.001.
- **Annealing rate** — Specifies the annealing rate parameter for SGD optimization. The default value is 0.000001.
- **Minibatch size** — Specifies the size of the minibatches used for SGD optimization. The default value is 50.
- **Momentum** — Specifies the value for momentum for SGD optimization. The default value is 0.
- **Create deterministic results** — Specifies whether to create deterministic (reproducible) results using the specified SGD seed. Note that selecting this option can significantly increase run time. By default, this option is deselected.
- **SGD seed** — Specifies the seed to use for SGD optimization to create reproducible results. This property is available only when the **Create deterministic results** property is selected. The default value is 12345.

For neural networks with more than 5 hidden layers, the SGD options are different. The following options are available:

- **Learning rate** — Specifies the learning rate parameter for SGD optimization. The default value is 0.001.
- **Maximum epochs** — Specifies the maximum number of epochs. One epoch is reached after a single pass through the data. The default value is 10.
- **Minibatch size** — Specifies the size of the minibatches used for SGD optimization. The default value is 50.
- **Advanced Options** — Specifies the advanced SGD options available for neural networks with more than 5 hidden layers. The following options are available:
 - **SGD algorithm** — Specifies the SGD algorithm to use when training more than 5 hidden layers. Possible values are as follows:
 - **Adam SGD**
 - **Momentum SGD**
 - **SGD**
 The default value is **Adam SGD**.
 - **First moment exponential decay rate** — Specifies the exponential decay rate for the first moment in the Adam learning algorithm. This option is available only if **SGD algorithm** is **Adam SGD**. The default value is 0.9.
 - **Second moment exponential decay rate** — Specifies the exponential decay rate for the second moment in the Adam learning algorithm. This option is available only if **SGD algorithm** is **Adam SGD**. The default value is 0.999.

- **Momentum** — Specifies the momentum value for SGD optimization. This option is available only if **SGD algorithm** is **Momentum SGD**. The default value is 0.9.
- **Minimum gradient value** — Specifies the minimum gradient value. All gradients that are less than the specified value are set to the specified value. The default value is 0.
- **Maximum gradient value** — Specifies the maximum gradient value. All gradients that are greater than the specified value are set to the specified value. The default value is 0.
- **Learning rate policy** — Specifies the learning rate policy. Here are the possible values:
 - **Fixed decay**
 - **Inverse decay**
 - **Polynomial decay**
 - **Step down**
 The default value is **Fixed decay**
- **Gamma** — Specifies the gamma value for the learning rate policy. This option is available only if **Learning rate policy** is **Inverse decay** or **Step down**. The default value is 0.1.
- **Power** — Specifies the power for the learning rate policy. This option is available only if **Learning rate policy** is **Inverse decay** or **Polynomial decay**. The default value is 0.75.
- **Step size** — Specifies the step size for the learning rate policy. This option is available only if **Learning rate policy** is **Step down**. The default value is 10.

Perform Early Stopping

Perform Early Stopping — Specifies whether to stop training when the model begins to overfit. The training stops after N consecutive iterations (**Stagnation**) without improvement in the validation partition. Early stopping cannot be used if there is no validation partition. By default, this option is selected. The following options are available:

- **Stagnation** — Specifies the number of consecutive iterations (N) for early stopping. The default value is 5.
- **Validation error goal** — Specifies a goal for early stopping based on the validation error rate. When the error gets below this value, the optimization stops. This option is in effect only for networks with fewer than 6 hidden layers. The value of 0 indicates that no validation error goal is set. The default value is 0.

Note: When early stopping is performed, either the value of the **Stagnation** property or the value of the **Validation error goal** property must be nonzero.

Perform Autotuning

This feature specifies whether to perform autotuning of any neural network parameters.

Note: Autotuning is available only when **Number of hidden layers** is less than 6.

Warning: Performing autotuning can substantially increase run time. If **Perform Autotuning** is selected, the following options are available:

- **Number of Hidden Layers** — Specifies whether to autotune the number of hidden layers. If selected, the following options are available:
 - **Hidden layers initial value** — Specifies the initial value for autotuning the number of hidden layers. Possible values range from 0 to 5. The default value is 1. Use the **From** and **To** options to specify the range. The default **From** value is 0, and the default **To** value is 2.
 - **Neurons initial value** — Specifies the initial value for autotuning the number of neurons. The default value is 1. Use the **From** and **To** options to specify the range. The default **From** value is 1, and the default **To** value is 100.
- **L1 Weight Decay** — Specifies whether to autotune the L1 weight decay parameter. If selected, the following option is available:
 - **Initial value** — Specifies the initial value for autotuning the L1 weight decay. The default value is 0. Use the **From** and **To** options to specify the range. The default **From** value is 0, and the default **To** value is 10.
- **L2 Weight Decay** — Specifies whether to autotune the L2 weight decay parameter. If selected, the following option is available:
 - **Initial value** — Specifies the initial value for autotuning the L2 weight decay. The default value is 0. Use the **From** and **To** options to specify the range. The default **From** value is 0, and the default **To** value is 10.
- **Learning Rate** — Specifies whether to autotune the learning rate for the hidden layers. If selected, the following option is available:
 - **Initial value** — Specifies the initial value for autotuning the hidden layer learning rate. The default value is 0.001. Use the **From** and **To** options to specify the range. The default **From** value is 0, and the default **To** value is 0.1.
- **Annealing Rate** — Specifies whether to autotune the annealing rate for the hidden layers. If selected, the following option is available:
 - **Initial value** — Specifies the initial value for autotuning the hidden layer annealing rate. The default value is 0.001. Use the **From** and **To** options to specify the range. The default **From** value is 0.000001, and the default **To** value is 0.1.
- **Search Options** — Specifies the options for autotuning searching. The following options are available:
 - **Search method** — Specifies the tuning search method. Here are the possible values:

- **Bayesian** — Specifies the Bayesian method. This method uses priors to seed the iterative optimization.
- **Genetic algorithm** — Specifies the genetic algorithm method. This method uses an initial Latin Hypercube sample that seeds a genetic algorithm to generate a new population of alternative configurations at each iteration.
- **Grid** — Specifies the grid method. This method performs an exhaustive search of all configurations. Each hyperparameter of interest is discretized into a desired set of values to be studied. Models are trained and assessed for all combinations of values across all the hyperparameters (thus forming a multi-dimensional “grid”).
- **Latin hypercube sample** — Specifies the Latin Hypercube method. This method performs an optimized grid search that is uniform in each tuning parameter, but random in combinations.
- **Random** — Specifies the Random method. This method generates a single sample of purely random configurations.

The default value is **Genetic algorithm**.

- **Number of evaluations per iteration** — Specifies the number of tuning evaluations in one iteration. This option is available only if the **Search method** is **Genetic algorithm** or **Bayesian**. The default value is 10.
- **Maximum number of evaluations** — Specifies the maximum number of tuning evaluations. This option is available only if the **Search method** is **Genetic algorithm** or **Bayesian**. The default value is 50.
- **Maximum number of iterations** — Specifies the maximum number of tuning iterations. This option is available only if the **Search method** is **Genetic algorithm** or **Bayesian**. The default value is 5.
- **Maximum number of points in model** — Specifies the maximum number of points in the model for a Bayesian search. This option is available only if the **Search method** is **Bayesian**. The default value is 100.
- **Sample size** — Specifies the sample size. This option is available only if the **Search method** is **Random** or **Latin hypercube sample**. The default value is 50.
- **General Options** — Specifies the general properties for autotuning. The following options are available:
 - **Validation method** — Specifies the validation method for finding the objective value. Note that if your data is partitioned, then that partition is used and **Validation method**, **Validation data proportion**, and **Cross validation number of folds** are all ignored. Here are the possible values:
 - **Partition** — Specifies using a single partition of a training set. With partition, you specify proportions to use for randomly assigning observations to each role.
 - **K-fold cross validation** — specifies using the cross validation method. In cross validation, each model evaluation requires k training executions (on $k-1$ data folds) and k scoring executions (on one holdout fold). This increases the evaluation time by approximately a factor of k .

For small to medium data tables, cross validation provides, on average, a better representation of error across the whole data table. **Partition** is the default value.

- **Training data proportion** — Specifies the proportion of data to be used for training the **Partition** validation method. The default value is 0.7.
- **Validation data proportion** — Specifies the proportion of data to be used for the **Partition** validation method. The default value is 0.3.
- **Cross validation number of folds** — Specifies the number of partition folds in the cross validation process (the k defined above). Possible values range from 2 to 20. The default value is 5.
- **Class target objective function** — Specifies the objective function to optimize for tuning parameters for a class target. Here are the possible values:
 - **Area under the curve**
 - **Average squared error**
 - **F0.5 score**
 - **F1 score**
 - **Gamma**
 - **Gini coefficient**
 - **Kolmogorov-Smirnov statistic**
 - **Misclassification rate**
 - **Multi-class log loss**
 - **Root average squared error**
 - **Tau**

The default value is **Kolmogorov-Smirnov statistic**.

- **Interval target objective function** — Specifies the objective function to optimize for tuning parameters for an interval target. Here are the possible values:
 - **Average squared error**
 - **Mean absolute error**
 - **Mean squared logarithmic error**
 - **Root average squared error**
 - **Root mean absolute error**
 - **Root mean squared logarithmic error**

The default value is **Average squared error**.

- **Maximum time (minutes)** — Specifies the maximum time (in minutes) for the optimization tuner. The default value is 60.
- **Maximum training time for single model (minutes)** — Specifies the maximum time (in minutes) for a single model train. To indicate no maximum, enter no value in the field.

Lift Calculation Property

- **Use the exact percentile method for lift calculations** — Specifies whether to use the exact percentile method for calculating lift and related assessment measures. When this property is deselected, the iterative method is used. The exact method should be used if there are convergence errors when using the iterative method.

Binary Classification Cutoff

- **Specify node binary classification cutoff** — Specifies whether to use the binary classification cutoff specified below for the node. If this property is deselected, the project binary classification cutoff is used for determining the predicted value for a binary target based on the posterior probabilities. By default, this property is deselected.
- **Node binary classification cutoff** — Specifies the cutoff to use in the node for determining the predicted value for a binary target based on the posterior probabilities. This option is available only if **Specify node binary classification cutoff** is selected. The default value is 0.5.

Note: The **Rules** settings in the Project Settings window enable you to specify a new cutoff value by selecting **Override the default classification cutoff**. If **Specify node binary classification cutoff** is deselected, then the value that you specified in the Project Settings window is used. If **Specify node binary classification cutoff** is selected, then the value that you specified in **Node binary classification cutoff** is used.

Post Training Properties

Changing the following properties will not retrain the model.

Model Interpretability — Specifies the method of model interpretation in order to show how the model responds to changing inputs.

- **Global Interpretability**
 - **Variable importance** — Specifies whether to display a table of the relative importance values for the input variables. The surrogate variable importance values are calculated using a one-level decision tree for each input to predict the predicted value as a global surrogate model. For tree-based supervised learning models, the model variable importance table is displayed as well.
 - **PD plots** — Specifies whether to display partial dependency (PD) plots that show the relationship between model input variables and their predictors. PD plots are generated for only up to the top ten variables in the relative

importance table. Here is how to determine the variables that are displayed in the PD plot: for tree-based models (forest, decision tree, and gradient boosting), the model variable importance table is used. For other supervised learning models, the surrogate variable importance table is used.

■ Local Interpretability

- **ICE plots** — Specifies whether to display Individual Conditional Expectation (ICE) plots. ICE plots can reveal interesting subgroups and interactions between model variables. ICE plots are generated for only up to the top ten variables in the relative importance table. Here is how to determine the variables that are displayed in the ICE plot: for tree-based models (forest, decision tree, and gradient boosting), the model variable importance table is used. For other supervised learning models, the surrogate variable importance table is used.
- **LIME** — Specifies whether to use the Locally Interpretable Model-Agnostic Explanation (LIME) method to explain the model predictions.
- **Kernel SHAP** — Specifies whether to use the Kernel Shapley additive explanations (Kernel SHAP) method to explain the model predictions.
- **Maximum number of Kernel SHAP variables** — Specifies the maximum number of Kernel SHAP variables to plot. Possible values range from 1 to 100. The default value is 20.
- **Specify instances to explain** — Specifies the individual observations that you want to explain. The **Random** observations option provides explanations for five randomly selected observations. The **Specify up to 5** observations option enables you to provide IDs for the specific observations in the data. The IDs should be unique values of the variable that has the role **Key** in the metadata. If no variable is assigned the role **Key**, then you can use the variable `_dmIndex_` to specify the individual observations.
- **LIME/Kernel SHAP Tables**
 - **Explainer Information table** — Specifies whether the Explainer Information tables for the LIME and Kernel SHAP methods should be included in the node results. The Explainer Information table contains information about what settings were used to perform the LIME and Kernel SHAP methods.
 - **Explainer Fidelity table** — Specifies whether the Explainer Fidelity tables for the LIME and Kernel SHAP methods should be included in the node results. The Explainer Fidelity table contains information about the accuracy of the local explanation predictions, compared to the black-box machine learning model.


■ PD/ICE Options

- **Maximum number of variables** — Specifies the maximum number of input variables to use to generate the PD and ICE plots. The input variables are selected based on their relative importance. Possible values range from 1 to 10. The default value is 5.
- **Number of observations** — Specifies the maximum number of observations to sample for the PD and ICE plots. The default value is 1000.
- **Number of tick points** — Specifies the number of tick points for an interval analysis variable to generate for the PD and ICE plots. Possible values range from 3 to 100. The default value is 50.

- **Truncate lower tail** — Specifies under what percentage to discard the values of the interval analysis variables for the PD and ICE plots. Possible values range from 0 to 99. The default value is 5%.
- **Truncate upper tail** — Specifies above what percentage to discard the values of the interval analysis variables for the PD and ICE plots. Possible values range from 0 to 99. The default value is 95%.
- **Seed** — Specifies the seed for generating random numbers. The seed is used for selecting individual observations to explain, selecting observations to generate PD and ICE plots, and simulating data for LIME and Kernel SHAP.



Neural Network Results

After running the node, open the Results window by right-clicking the node and selecting **Results**.

You can download the data that is used to build the charts and tables in the Results window by clicking . A CSV file with the chart or table data will immediately begin downloading.

In the **Output Data** tab, you can view, save, and explore the data that is generated by the node. For more information, see [Explore and Visualize](#).

Several charts have detailed descriptions available that explain how the charts are generated. The descriptions also provide information about how to interpret the charts that is custom to your particular data and models. To access these detailed descriptions:

- Expand an individual chart by clicking . The detailed description is in the right panel.
- Select  on an individual chart. A window appears with the detailed description.

Note: The detailed descriptions are available only in the English language.

Node

- **Network Diagram: Top 200 Weights** — Displays a diagram of the neural network when there are fewer than 6 hidden layers in the network. Only the links that correspond to the top 200 weights are displayed. You can interactively control the range of these weights to display.
- **Iteration Plot** — Displays a line graph of the validation error, if reported, as a function of the epoch. To examine the loss or objective as a function of the epoch, use the drop-down menu in the upper right corner.
- **Autotune Best Configuration** — Displays a table of the resulting values of parameters that were autotuned. This table is not displayed if autotuning is not selected.
- **Autotune Results** — Displays the iterative results of the autotuning process, showing how each of the parameter values evolve as the model iterates. This table is not displayed if autotuning is not selected.

- **Node Score Code** — Displays the SAS score code that was created by the node. The SAS score code can be used outside the Model Studio environment to score new data.

Note: If you train a network with more than 5 hidden layers, you get EP score code instead of DATA step code since an analytic store is used.

- **Path EP Score Code** — Displays the SAS score code that was created by the node when training a network with more than 5 hidden layers. The score code can be used outside the Model Studio environment to score new data.
- **Path Score Code** — Displays the SAS score code that was created by the node, as well as all preceding nodes. The SAS model score code can be used outside the Model Studio environment to score new data.

Note: If you trained a network with more than 5 hidden layers, this code is not included in the results.

- **DS2 Package Code** — Displays the SAS DS2 package code that was created by the node. The score code can be used outside the Model Studio environment to score new data. DS2 package code is used when publishing models to SAS Micro Analytic Service.
- **Score Inputs** — Displays a table of data on the input variables for scoring calculations. The table includes the variables' name, role, level, type, label, format, and length.
- **Score Outputs** — Displays a table of data on the predicted response variable for scoring calculations. The table includes the variables' name, role, type, format and length, as well as the creator, creator GUID, and function.
- **Training Code** — Displays the SAS code that Model Studio used to train the node.
- **Properties** — Specifies the various properties that you selected before running the node.
- **Output** — Displays the SAS output of the neural network run.

Assessment

- **Predicted Reports** — Displays the predicted and target mean as a function of the depth for the model. The predicted and target mean are given for each of the data roles. To examine the predicted mean as a function of the target mean, use the drop-down menu in the upper right corner. This result is displayed only if the target is an interval variable.
- **Lift Reports** — Displays the cumulative lift as a function of the depth for the model. The cumulative lift is given for each of the data roles. To examine other statistics as a function of depth, use the drop-down menu in the upper right corner. Other statistics include lift, gain, captured response percentage, cumulative captured response percentage, response percentage, and cumulative response percentage. This result is displayed only if the target is a class variable.
- **ROC Reports** — Displays the ROC (receiver operating characteristic) chart for a model, giving the sensitivity as a function of 1-specificity. The sensitivity is given for each of the data roles. To examine other statistics, use the drop-down menu in the upper right corner. Other statistics include accuracy and F1 score. This result is displayed only if the target is a class variable.

- **Fit Statistics** — Displays a table of the fit statistics for the model, broken down by data role.
- **Event Classification** — Displays the confusion matrix at various cutoff values for each partition. The confusion matrix contains true positives (events that are correctly classified as events), false positives (non-events that are classified as events), false negatives (events that are classified as non-events), and true negatives (non-events that are correctly classified as non-events). The classification cutoffs that are used are the default value of 0.5, the Kolmogorov-Smirnov cutoff value for each partition, and the **Node binary classification cutoff** that is specified for binary targets. Use the drop-down menu to view the information in the chart as percentages or counts. You can also view the information summarized in a table. This result is displayed only if the target is a class variable.
- **Nominal Classification** — Displays either the percentage of observations or the number of observations that predict each level of the target variable. The plot is segmented by target level and partition level. The target level with the greatest predicted probability is the predicted outcome. This result is displayed only if the target is a nominal variable.

Model Interpretability

- **Surrogate Model Variable Importance** — Displays a table of the relative importance of the input variables, including variable level and label. Relative importance is calculated using a one-level decision tree for each input variable to predict the predicted value as a global surrogate model.
- **PD Plot** — Displays the functional relationship between the input variable and the model prediction. For interval input variables, the 95% confidence interval for the average target prediction is indicated by the shaded band around the line. The X axis includes a heat map that shows the distribution of the input variable. The extreme values of the input variables are removed by truncating the lower and upper tails of its distribution. The amount of truncation can be controlled by specifying values for the **Truncate lower tail** and the **Truncate upper tail** properties.
- **PD and ICE Overlay Plot** — Displays the functional relationship between the input variable and the model prediction as well as the functional relationship between individual observations and the model prediction. For each individual observation, the corresponding ICE curve displays values of the input variable on the X axis and the corresponding predicted probability or prediction of the target variable on the Y axis, holding the other input variable values constant at that observation. The X axis includes a heat map that shows the distribution of the input variable. The extreme values of the input variables are removed by truncating the lower and upper tails of its distribution. The amount of truncation can be controlled by specifying values for the **Truncate lower tail** and the **Truncate upper tail** properties.
- **LIME Explanations** — The LIME algorithm calculates a more easily interpretable linear model around an individual observation. The chart displays the regression estimates for the input variables that are selected in the local surrogate linear regression model. The input variables are ordered by significance such that the most important variable for the local regression model is at the bottom of the chart. The LASSO technique is used to select the most significant predictors from the set of input variables that are used to train the model. Each nominal input variable is binary-encoded based on whether it matches the level of the individual observation. A positive estimate indicates that the observed value of the input variable increases the predicted probability of the event.

- **Kernel SHAP Values** — For each individual observation, an input variable's Kernel SHAP value is the contribution of the observed value of the input variable to the predicted probability or prediction of the target variable. The Kernel SHAP values are the regression coefficients that are obtained by fitting a weighted least squares regression. The Kernel SHAP values of all input variables sum to one. The input variables are ordered by significance such that the most important variable according to the absolute Kernel SHAP values is at the top of the chart. Each nominal input is binary-encoded based on whether it matches the individual observation. Interval inputs are binary encoded based on their proximity to the individual observation.
- **LIME Information Table** — Displays information about the settings used to perform the LIME method.
- **Kernel SHAP Information Table** — Displays information about the settings used to perform the Kernel SHAP method.
- **LIME Fidelity Table** — Displays information about the accuracy of the local explanation predictions that are calculated using the LIME method, compared to the black-box machine learning model. The model prediction, the explainer prediction, and the explainer root mean square error (RMSE) are displayed.
- **Kernel SHAP Fidelity Table** — Displays information about the accuracy of the local explanation predictions that are calculated using the Kernel SHAP method, compared to the black-box machine learning model. The model prediction, the explainer prediction, and the explainer root mean square error (RMSE) are displayed.

Open Source Code

<i>Overview of Open Source Code</i>	183
<i>Open Source Code Properties</i>	184
<i>Open Source Code Editor User Interface</i>	185
<i>Open Source Code Results</i>	187
<i>Using the Open Source Code Node</i>	188
<i>Installation of Python or R Software</i>	189
<i>Common Questions</i>	189

Overview of Open Source Code

The **Open Source Code** node is a Miscellaneous node that can run Python or R code. This node can subsequently be moved to the **Supervised Learning** group if a Python or R model needs to be assessed and included in the **Model Comparison** node.

Because this code is not executed in CAS, a sample of the data is created and downloaded to avoid the movement of large data. The default sample size is 10,000 observations. Use the **Data Sample** properties to control the sample size and method. Apply caution and do not specify the entire data table or a relatively large sample when the input data is large. When you use the **Model Comparison** node to compare the results of your Python or R models, note that this node might not be using the entire input data set.

Input data is accessed by the Python or R code via a comma-separated value (CSV) file or as a data frame. When **Generate data frame** is selected, a data frame is generated for the CSV. In this case, the input data is available in `dm_inputdf`, which is a pandas data frame for Python code or an R data frame for R code. When the data is partitioned, an additional data frame, `dm_traindf`, is available in the editor. This data frame contains the training data. If a Python or R model is built and must be assessed, the corresponding predictions or posterior probabilities are made available in the `dm_scoredf` data frame.

The **Open Source Code** node can be moved from the Miscellaneous group to the Supervised Learning group. To move the node, right-click the **Open Source Code** node and select **Move** ⇨ **Supervised Learning**. Do this to indicate that your **Open Source Code** node contains a model that should be merged with the input data and that model assessment should be performed. To successfully merge your Python or R model, ensure that the number of observations in `dm_inputdf` and `dm_scoreddf` are equal.

To override the default source encoding in Python 2 or to add global Python code, select the **Prepend Python configuration code** check box on the **Node Configuration** tab of the project or user settings, and then enter code. This code is automatically prepended to every **Open Source Code** node when the **Language** property is set to Python.

For an example using the **Open Source Code** node, see [Open Source Code Node Example](#).

Note: This node does not support the operations **Download score code**, **Register models**, **Publish models**, **Score holdoutdata**, and other similar operations in the **Pipeline Comparison** tab. This node cannot be a member of the **Ensemble** node when used in a Supervised Learning group. These operations are not supported because the **Open Source Code** node does not generate SAS score code.

Note: The **Open Source Code** node cannot be run if the LOCKDOWN feature is enabled or if you are working in a multi-tenant environment.

Open Source Code Properties

- **Open Code Editor** — Invokes the Open Source Code Editor.
- **Language** — Specifies the language of your code. The options are **Python** and **R**.
- **Input to Open Source**
 - **Data Sample**
 - **Sampling method** — Specifies the method used to create the data sample. Options include **(none)**, **Simple random**, and **Stratify**. By default, **Simple random** is selected.
 - **Sample using** — Specifies whether sampling is performed on a fixed number of observations or a fixed percentage of observations.
 - **Number of observations** — Specifies the fixed number of observations used in the sample. The default value is 10000.
 - **Percentage of observations** — Specifies the fixed percentage of observations used in the sample. The default value is 10%.
 - **Include SAS formats** — Specifies whether SAS formats are used in the data sample. By default, this option is selected.

- **Drop rejected variables** — Specifies whether variables with the role **Rejected** should be dropped in the output data set.
- **Generate data frame** — Specifies whether to generate an R data frame or a Pandas DataFrame in Python. Categorical inputs are encoded as factors in R. If this option is disabled, the input data should be accessed as a CSV file. By default, this option is selected.
- **Use output data in child nodes** — Specifies whether to save a copy of the output data that is used by successive nodes. This output data is used in the successive nodes instead of the project data and the metadata associated with this data overwrites the metadata of the project. When this property is selected, the **Open Source Code** node expects that output data is in the `dm_scoreddf` data frame. The run fails if that is not the case.
- **Use the exact percentile method for lift calculations** — Specifies whether to use the exact percentile method for calculating lift and related assessment measures. When this option is deselected, the iterative method is used. The exact method should be used if there are convergence errors using the iterative method.

Open Source Code Editor User Interface


The following data items are defined and available for certain expected cases in your Python or R code. For R code, the `setwd(dm_nodedir)` function call is also added so that you can specify `dm_nodedir` as the current working directory.

Data Item	Description
<code>dm_class_input</code>	A list of names that identify the categorical (nominal and ordinal) variables. The data type is a list in Python and a vector in R.
<code>dm_classtarget_intovar</code>	A variable that identifies the classification variable for a class target. The data type is a string.
<code>dm_classtarget_level</code>	A list of values that identify the various levels in the target variable. The data type is a list in Python and a vector in R.
<code>dm_dec_target</code>	A variable that identifies the name of the target. The data type is string.
<code>dm_input</code>	A list of names that identify the input (interval and categorical) variables. The data type is a list in Python and a vector in R.
<code>dm_inputdf</code>	A data frame that contains the sampled input data. This is available only when the Generate data frame property is selected. The data type is a pandas data frame in Python or

Data Item	Description
	a data frame where categorical variables are encoded as factors in R.
dm_interval_input	A list of names that identify the interval variables. The data type is a list in Python and a vector in R.
dm_model	An object that identifies the model. The usage of this object is optional but recommended for future use.
dm_model_formula	An object that identifies the model formula in R of the form $\sim input_var1 + input_var2$. The data type is formula object. This object is available only in R.
dm_nodedir	A variable that identifies the node's working directory. That is, where the local files are created. The data type is string.
dm_partition_train_val	A variable that identifies the value that identifies the training partition. The data type is string or integer.
dm_partitionvar	A variable that identifies the name of the data partition. The data type is string.
dm_predictionvar	A list of names that identify the prediction variables. The data type is a list in Python and a vector in R.
dm_scoreddf	A data frame that you must create with model predictions when you execute the node in the Supervised Learning group. This item must be created when the Generate data frame property is selected and contain the same number of rows as dm_inputdf. The data type is a pandas data frame in Python or a data frame in R.
dm_traindf	A data frame that contains the sampled training data. This is available only when the Generate data frame property is selected. The data type is a pandas data frame in Python or a data frame where categorical variables are encoded as factors in R.
node_data.csv	A file in comma-separated value format that identifies the sampled input data in the dm_nodedir directory.
node_scored.csv	A file in comma-separated value format that you must create with model predictions when you execute the node in the Supervised Learning group and model assessment is performed. This file must be created when Generate data frame is not selected and must contain the same number of rows as node_data.csv.



Open Source Code Results

After running the node, open the Results window by right-clicking the node and selecting **Results**.

You can download the data that is used to build the charts and tables in the Results window by clicking . A CSV file with the chart or table data will immediately begin downloading.

In the **Output Data** tab, you can view, save, and explore the data that is generated by the node. For more information, see [Explore and Visualize](#).

Several charts have detailed descriptions available that explain how the charts are generated. The descriptions also provide information about how to interpret the charts that is custom to your particular data and models. To access these detailed descriptions:

- Expand an individual chart by clicking . The detailed description is in the right panel.
- Select  on an individual chart. A window appears with the detailed description.

Note: The detailed descriptions are available only in the English language.

The following results are available:

- Plots, tables, or an editor that display the output from the open-source program. These are displayed when the open-source program creates output files with the `rpt_` prefix and a `csv`, `txt`, `jpg`, `jpeg`, `png`, or `GIF` file extension. For example, `rpt_ForestVariableImportance.csv` displays a variable importance table. For a CSV file, the results are always a table and the first row must contain the column names.
- The Python code or R code that was executed is displayed.
- The standard Python output or R output is displayed.
- The node properties are displayed.
- When in **Supervised Learning** mode, the list of Predicted Variables expected by the project is displayed. In addition, if available, the top 10 observations from the input data and the scored data are displayed.
- The model assessment reports, such as Lift, ROC, Fit Statistics, Event Classification, and Nominal Classification, are displayed.

Using the Open Source Code Node

When running the **Open Source Code** node, additional Python or R code is generated and added before and after your code. The precursor code creates target, input, and partition variables; input handles; and other necessary data items. The posterior code, when applicable, converts the scored data frame to a CSV file.

You can use the **Open Source Code** node to produce a custom model that is incorporated into your pipeline. To indicate that your **Code** node will produce a model that should be assessed and processed by the **Model Comparison** node:

- 1 Right-click the node, and select **Move**.
- 2 Select **Supervised Learning**.
- 3 The **Open Source Code** node is now connected to a **Model Comparison** node.

In order for the model produced by the **Open Source Code** node to be assessed, it must produce a scored data set with the appropriate prediction variables names. Otherwise, no assessment reports are generated. To ensure that reports are generated, ensure that the following criteria are met:

- If the **Generate data frame** property is selected, the scored data set must be saved to `dm_scoreddf`. Otherwise, it must be saved to `node_scored.csv`.
- The prediction variables in the scored data set must be named according to the following convention:
 - Interval targets must be named `P_<targetVariableName>`. For example, if MSRP is the target variable then the prediction variable must be `P_MS RP`.
 - Categorical targets must be named `P_<targetVariableName><targetLevel>`. All target level properties must be computed. For example, if BAD is the target variable with levels 0 and 1 then the posterior probabilities should be `P_BAD0` and `P_BAD1`.
- The scored data set must contain the same number of observations as the input data set.

When an open-source model is the project champion, a classification variable in the form `I_<targetVariableName>` must be created. The classification variable is necessary to generate the variable importance plot in the Insights tab. The classification variable can be identified with the variable `dm_classtarget_intovar`. The various levels of the target variable are defined in `dm_classtarget_level`. The `I_<targetVariableName>` is also required to produce the **Event Classification** chart and the **Nominal Classification** chart on the Assessment tab of the Results window.

Installation of Python or R Software

The Python or R software must be installed on the same machine as the compute server. The executable python or Rscript file must be available in the system path. On Windows, the environment variables PYTHONHOME or RHOME specify the home directories of Python or R, where the executable file is expected to be located in %PYTHONHOME%/python or %RHOME%/bin/Rscript. On Linux, the path for the compute server microservice can be modified by editing the sas-compsrv file under the /opt/sas/viya/config/etc/sysconfig/compsrv/default directory. The following line can be added to the sas-compsrv file:

```
export PATH=path_to_your_python_or_R_bin_directory:${PATH}
```

Consult your SAS Administrator to ensure that the above configurations are satisfied. You also need to install any necessary packages with sudo privileges so that they are accessible to all users. The **Open Source Code** node functions regardless of the version of Python or R that is installed. It acts to transfer the code to the package that is installed.

To run the **Open Source Code** node with the **Language** set to **Python** and **Generate data frame** selected, pandas must be installed. Pandas is unnecessary if you deselect **Generate data frame**.

Common Questions

What happens if you run the **Open Source Code** node when Python or R is not installed on the compute server?

The **Open Source Code** node fails if Python or R is not installed or if it is not configured correctly. The executable python or Rscript file must be available in the system path on Linux or their install directories can be specified with PYTHONHOME or RHOME on Windows.

When should you deselect **Include SAS formats** property?

The **Include SAS formats** property controls whether the downloaded data sent to the Python or R software should keep SAS formats. It is strongly recommended that you keep SAS formats and this should work in most cases. However, some numeric formats like DOLLARw.d add a dollar sign and change the data type of the variable when exporting to CSV. In such cases, these formats must be removed.

This property either keeps or removes SAS formats for all variables in the data set. If input variables or target variables have SAS formats or user-defined formats that significantly modify the data, it is not recommended to deselect this option. This is because the model built might not be comparable to other models.

How are missing values in input data handled in open-source software?

Both Python and R packages sometimes do not support missing values in data. It is your responsibility to prepare the data as necessary for these packages. It is

highly recommended that you add an **Imputation** node before the **Open Source Code** node to handle missing values. If the training data does not contain missing values but if either the validation or test data does contain missing values, you should consider enabling the **Impute non-missing variables** property in the **Imputation** node.

The node failure displays the message `Encountered error code XX` when executing `{PYTHON/R}` program in the log. Where do I see detailed messages to debug the problem?

When an error occurs in the **Open Source Code** node, the generic message `Encountered error code 1 when executing R program` is highlighted in the log. The detailed error messages that help pinpoint the problem are displayed above this generic message and you can scroll up to view them. You can also search for the `Executing python` or `Executing Rscript` strings in the log to see the start of these detailed error messages.

Why do I get `dm_traindf` not found error when using the training data frame in code?

The data frame variable `dm_traindf` is created only when input data is partitioned. Ensure that there is a training partition in the project.

Can SWAT packages be installed and CAS actions invoked from Python or R installations on the compute server?

This scenario has not been tested and is not supported. Instead, if CAS actions need to be invoked, it is recommended that you use the **SAS Code** node and PROC CAS. There is no benefit in making a call to CAS via the Python or R SWAT packages when you can directly do that with PROC CAS in the **SAS Code** node.

Why does the **Open Source Code** node not have **Assessment** results even though it was successfully executed under the **Supervised Learning** group?

Assessment results might not be generated for one or more of the following reasons:

- When a scored data set (`dm_scoreddf` or `nodescored.csv`) is not generated. In this case, the node fails with an error in the Python or R code.
- The scored data set does not contain the expected predicted variable names. The node does not fail but does not produce assessment plots or statistics in the **Results** tab.
- The scored data set does not contain the same number of observations as the input data set. The node does not fail but does not produce assessment plots or statistics in the **Results** tab. The following warning message is generated in the log: Data merge was skipped as observations in input data (<number of observations>) and scored data (<number of observations>) did not match. Therefore, model assessment is not performed.

If any of the above conditions are met, the **Open Source Code** node is not included in the **Model Comparison** node that follows.

Do variable names have any length requirements?

When using the R language, variable names are limited to 31 characters. When using Python, variable names are limited to 32 characters. Variable names longer than this are truncated, which causes the node to fail.

Why does the **Open Source Code** node require pandas package when executing Python code?

The node requires pandas package because the **Generate data frame** property creates a pandas data frame for input data. There is no pandas requirement when the **Generate data frame** property is deselected.

Can the **Open Source Code** node be executed when there is no code entered in the editor?

Yes, it is good practice to execute the node in an empty state to validate whether Python or R is correctly installed and configured. In addition, you can view the precursor code that is added as part of the executed code. The code that is added depends on combination of properties selected. The precursor code is part of the node results.

Any other debugging tips?

If the **Open Source Code** node fails with errors related to working with data frames, consider deselecting the **Generate data frame** property and working with CSV files.

Quantile Regression

<i>Overview of Quantile Regression</i>	193
<i>Quantile Regression Properties</i>	194
General Options	194
Effects Options	194
Selection Method Property	195
Selection Options	195
Lift Calculation Property	197
Post Training Properties	198
<i>Quantile Regression Results</i>	199

Overview of Quantile Regression

The **Quantile Regression** node is a Supervised Learning node. A quantile regression attempts to fit the quantiles of an interval target as a linear function of one or more features. The **Quantile Regression** node requires an interval target variable and one or more effect variables. Effect variables can be interval variables, classification variables, interaction effects, or splines. Rows that have at least one missing value are removed from the model.

Fitting a quantile regression model is especially useful if you want to understand which features are important in predicting the target at its certain quantiles (also known as percentiles). For example, for a target variable that is highly skewed or has outliers, you might want to know what predicts median values (50th percentile) as opposed to an ordinary least squares model that is based on predicting the mean of the target variable. Similarly, you might want to model lower or higher quantiles of the target variable to understand the features that are important at those specific quantiles. For example, if the target is income from many countries, the features that predict a lower quantile of the target distribution can be very different from those features that predict higher quantiles.

Quantile Regression Properties

General Options

Quantile — Specifies the quantile level for the regression. You can specify any value between 0 and 1, exclusive. For example, specify 0.5 to perform a median regression.

Effects Options

- **Two-factor interactions** — Specifies whether to include all two-factor interactions of nominal variables in the model. By default, this option is deselected.
- **Factor split** — Specifies whether levels of a nominal variable can enter or leave a model independently. This is done automatically for LASSO selection. For other selection methods, this option is deselected.
- **Spline** — Specifies whether to expand interval input variables into cubic B-spline bases with three equally spaced knots. (Expansion yields seven design matrix columns for each of the variables.) Spline and polynomial effects cannot be specified together. By default, this option is deselected.
- **Spline split** — Specifies whether each individual column in the design matrix that corresponds to the spline effect is treated as a separate effect that can enter or leave the model. By default, this option is deselected. This option is available only if **Spline** is selected.
- **Polynomial** — Specifies whether to use polynomial effects up to the specified **Polynomial degree** for all interval variables. Spline and polynomial effects cannot be specified together. By default, this option is deselected.
- **Polynomial degree** — Specifies the polynomial degree when polynomial terms are included in the model. This option is available only if **Polynomial** has been selected. Possible values are 2 and 3. The default value is 2.
- **Suppress intercept** — Specifies whether to suppress the intercept. By default, this is deselected.
- **Use missing as level** — Specifies whether missing values should be treated as a separate level for all nominal inputs. By default, this is deselected.
- **Model missing values for inputs** — Specifies whether to include observations with missing values. For interval variables, an indicator variable is created to indicate whether an observation contains a missing value is created and the original value is imputed with the mean. For categorical variables, missing values are treated as their own measurement level.

Selection Method Property

- **Selection method** — Specifies a model selection method. Here are the possible values:
 - **(none)** — Specifies not to perform variable selection.
 - **Backward** — Specifies that training is done by least squares regression that starts with all candidate features in the model. Features are removed one at a time until the stop criterion is met.
 - **Forward** — Specifies that training is done by least squares regression that starts with no candidate features in the model. Features are added one at a time until the stop criterion is met.
 - **Stepwise** — Specifies that training is done by least squares regression that starts as in the forward model but might remove features already in the model.

The default value is **Stepwise**.

Selection Options

- **Effect-selection criterion** — Specifies the criterion used to determine how effects are added to the model. Here are the possible values:
 - **Adjusted R1 statistic** — Specifies the Adjusted R1 statistic. The Adjusted R1 statistic is an adjustment of the tau statistic that adjusts for ties and can vary between -1 and 1, inclusive. Values closer to 1 or -1 indicate a stronger relationship.
 - **AIC** — Specifies Akaike's Information Criterion. AIC is based on the Kullback-Leibler information measure of discrepancy between the true distribution of the response variable and the distribution specified by the model. Smaller values indicate better models, and AIC values can become negative.
 - **AICC** — Specifies Corrected Akaike's Information Criterion. AICC adjusts the AIC value by accounting for sample size. Extra features penalize AICC more than AIC. As the sample size increases, AICC and AIC converge.
 - **Average check loss validation** — Specifies average check loss for the validation data as the selection method.
 - **R1 statistic** — Specifies the R1 statistic. The R1 statistic is also known as the tau statistic and can vary between -1 and 1, inclusive.
 - **SBC (BIC)** — Specifies Schwarz's Bayesian Criterion (SBC), also known as the Bayesian Information Criterion (BIC). Unexplained variations in the response variable and the number of features increase the value of the SBC. Smaller values indicate better models. SBC penalizes total number of model parameters more strongly than AIC, and hence tends to choose more sparse models.

- **Significance level** — Specifies the standard statistical significance level criterion.

The default value is **SBC (BIC)**.

- **Selection-process stopping criterion** — Specifies the criterion to stop the selection process. Here are the possible values:
 - **(none)** — Specifies no selection—process stopping criterion.
 - **Adjusted R1 statistic** — Specifies the Adjusted R1 statistic. The Adjusted R1 statistic is an adjustment of the tau statistic that adjusts for ties and can vary between -1 and 1, inclusive. Values closer to 1 or -1 indicate a stronger relationship.
 - **AIC** — Specifies Akaike's Information Criterion. AIC is based on the Kullback-Leibler information measure of discrepancy between the true distribution of the response variable and the distribution specified by the model. Smaller values indicate better models, and AIC values can become negative.
 - **AICC** — Specifies Corrected Akaike's Information Criterion. AICC adjusts the AIC value by accounting for sample size. Extra features penalize AICC more than AIC. As the sample size increases, AICC and AIC converge.
 - **Average check loss validation** — Specifies average check loss for the validation data as the selection method.
 - **SBC (BIC)** — Specifies Schwarz's Bayesian Criterion (SBC), also known as the Bayesian Information Criterion (BIC). Unexplained variations in the response variable and the number of features increase the value of the SBC. Smaller values indicate better models. SBC penalizes total number of model parameters more strongly than AIC, and hence tends to choose more sparse models.
 - **Significance level** — Specifies the standard statistical significance level criterion.

The default value is **SBC (BIC)**.

- **Model-selection criterion** — Specifies the criterion to choose the model (from the list of models at each step of the selection process) that yields the best value of the specified criterion. If the optimal value of the specified criterion occurs for models at more than one step, then the model that has the smallest number of parameters is chosen. Here are the possible values:
 - **Adjusted R1 statistic** — Specifies the Adjusted R1 statistic. The Adjusted R1 statistic is an adjustment of the tau statistic that adjusts for ties and can vary between -1 and 1, inclusive. Values closer to 1 or -1 indicate a stronger relationship.
 - **AIC** — Specifies Akaike's Information Criterion. AIC is based on the Kullback-Leibler information measure of discrepancy between the true distribution of the response variable and the distribution specified by the model. Smaller values indicate better models, and AIC values can become negative.
 - **AICC** — Specifies Corrected Akaike's Information Criterion. AICC adjusts the AIC value by accounting for sample size. Extra features penalize AICC more than AIC. As the sample size increases, AICC and AIC converge.
 - **Average check loss validation** — Specifies average check loss for the validation data as the selection method.

- **SBC (BIC)** — Specifies Schwarz’s Bayesian Criterion (SBC), also known as the Bayesian Information Criterion (BIC). Unexplained variations in the response variable and the number of features increase the value of the SBC. Smaller values indicate better models. SBC penalizes total number of model parameters more strongly than AIC, and hence tends to choose more sparse models.

The default value is **SBC (BIC)**.

- **Entry significance level** — Specifies the significance level for adding variables in the forward and stepwise directions. The default value is 0.05.
- **Stay significance level** — Specifies the significance level for removing variables in backward and stepwise directions. The default value is 0.05.
- **Maximum number of effects** — Specifies the maximum number of effects in any model that is considered during the selection process. If a model at some step of the selection process contains the specified maximum number of effects, then no additional effects are considered. If **Maximum number of effects** is set to 0 (the default value), this option is ignored.
- **Minimum number of effects** — Specifies the minimum number of effects in any model that is considered during the backward selection process. If **Minimum number of effects** is set to 0 (the default value), this option is ignored.
- **Maximum number of steps** — Specifies the maximum number of selection steps that are performed. If **Maximum number of steps** is set to 0 (the default value), this option is ignored.
- **Hierarchy** — Specifies whether and how the model hierarchy requirement is applied. Model hierarchy refers to the requirement that, for any term to be in the model, all model effects that are contained in the term must be present in the model. For example, in order for the interaction A*B to enter the model, the main effects A and B must also be in the model. Likewise, neither effect A nor effect B can leave the model while the interaction A*B is in the model. Here are the possible values:
 - **(none)** — Specifies that the hierarchy requirement is never applied.
 - **All variables** — Specifies that both interval and nominal variables are subject to the model hierarchy requirement.
 - **Class variables** — Specifies that only nominal variables are subject to the model hierarchy requirement.

The default value is **(none)**.

Lift Calculation Property

- **Use the exact percentile method for lift calculations** — Specifies whether to use the exact percentile method for calculating lift and related assessment measures. When this property is deselected, the iterative method is used. The exact method should be used if there are convergence errors when using the iterative method.

Post Training Properties

Changing the following properties will not retrain the model.

Model Interpretability — Specifies the method of model interpretation in order to show how the model responds to changing inputs.

- **Global Interpretability**

- **Variable importance** — Specifies whether to display a table of the relative importance values for the input variables. The surrogate variable importance values are calculated using a one-level decision tree for each input to predict the predicted value as a global surrogate model. For tree-based supervised learning models, the model variable importance table is displayed as well.
- **PD plots** — Specifies whether to display partial dependency (PD) plots that show the relationship between model input variables and their predictors. PD plots are generated for only up to the top ten variables in the relative importance table. Here is how to determine the variables that are displayed in the PD plot: for tree-based models (forest, decision tree, and gradient boosting), the model variable importance table is used. For other supervised learning models, the surrogate variable importance table is used.

- **Local Interpretability**


- **ICE plots** — Specifies whether to display Individual Conditional Expectation (ICE) plots. ICE plots can reveal interesting subgroups and interactions between model variables. ICE plots are generated for only up to the top ten variables in the relative importance table. Here is how to determine the variables that are displayed in the ICE plot: for tree-based models (forest, decision tree, and gradient boosting), the model variable importance table is used. For other supervised learning models, the surrogate variable importance table is used.
- **LIME** — Specifies whether to use the Locally Interpretable Model-Agnostic Explanation (LIME) method to explain the model predictions.
- **Kernel SHAP** — Specifies whether to use the Kernel Shapley additive explanations (Kernel SHAP) method to explain the model predictions.
- **Maximum number of Kernel SHAP variables** — Specifies the maximum number of Kernel SHAP variables to plot. Possible values range from 1 to 100. The default value is 20.
- **Specify instances to explain** — Specifies the individual observations that you want to explain. The **Random** observations option provides explanations for five randomly selected observations. The **Specify up to 5** observations option enables you to provide IDs for the specific observations in the data. The IDs should be unique values of the variable that has the role **Key** in the metadata. If no variable is assigned the role **Key**, then you can use the variable `_dmIndex_` to specify the individual observations.
- **LIME/Kernel SHAP Tables**
 - **Explainer Information table** — Specifies whether the Explainer Information tables for the LIME and Kernel SHAP methods should be included in the node results. The Explainer Information table contains

information about what settings were used to perform the LIME and Kernel SHAP methods.

- **Explainer Fidelity table** — Specifies whether the Explainer Fidelity tables for the LIME and Kernel SHAP methods should be included in the node results. The Explainer Fidelity table contains information about the accuracy of the local explanation predictions, compared to the black-box machine learning model.
- **PD/ICE Options**
 - **Maximum number of variables** — Specifies the maximum number of input variables to use to generate the PD and ICE plots. The input variables are selected based on their relative importance. Possible values range from 1 to 10. The default value is 5.
 - **Number of observations** — Specifies the maximum number of observations to sample for the PD and ICE plots. The default value is 1000.
 - **Number of tick points** — Specifies the number of tick points for an interval analysis variable to generate for the PD and ICE plots. Possible values range from 3 to 100. The default value is 50.
 - **Truncate lower tail** — Specifies under what percentage to discard the values of the interval analysis variables for the PD and ICE plots. Possible values range from 0 to 99. The default value is 5%.
 - **Truncate upper tail** — Specifies above what percentage to discard the values of the interval analysis variables for the PD and ICE plots. Possible values range from 0 to 99. The default value is 95%.
- **Seed** — Specifies the seed for generating random numbers. The seed is used for selecting individual observations to explain, selecting observations to generate PD and ICE plots, and simulating data for LIME and Kernel SHAP.



Quantile Regression Results

After running the node, open the Results window by right-clicking the node and selecting **Results**.

You can download the data that is used to build the charts and tables in the Results window by clicking . A CSV file with the chart or table data will immediately begin downloading.

In the **Output Data** tab, you can view, save, and explore the data that is generated by the node. For more information, see [Explore and Visualize](#).

Several charts have detailed descriptions available that explain how the charts are generated. The descriptions also provide information about how to interpret the charts that is custom to your particular data and models. To access these detailed descriptions:

- Expand an individual chart by clicking . The detailed description is in the right panel.
- Select  on an individual chart. A window appears with the detailed description.

Note: The detailed descriptions are available only in the English language.

Node

- **t Values by Parameter** — Displays bar charts for the t values in the final model. The bars are color-coded to indicate the algebraic signs of the coefficients.
- **Parameter Estimates** — Displays a table of the various statistics related to estimates for the parameters. These statistics include the t value, sign, estimate, absolute estimate, the p-value, chi-square value, and standard error.
- **Selection Summary** — Displays a table of the iterations, adding in and removing effects. This table also includes the SBC and optimal SBC values.
- **Regression Fit Statistics** — Displays a table of data on the quantile regression fit statistics. The table includes the quantile level, statistics, a corresponding qualitative description, and the values.
- **Node Score Code** — Displays the SAS score code that was created by the node. The SAS score code can be used outside the Model Studio environment to score new data.
- **Path Score Code** — Displays the SAS score code that was created by the node, as well as all preceding nodes. The SAS model score code can be used outside the Model Studio environment to score new data.
- **DS2 Package Code** — Displays the SAS DS2 package code that was created by the node. The SAS score code can be used outside the Model Studio environment in custom user applications. DS2 package code is used when publishing models to SAS Micro Analytic Service.
- **Score Inputs** — Displays a table of data on the input variables for scoring calculations. The table includes the variables' name, role, level, type, label, format, and length.
- **Score Outputs** — Displays a table of data on the predicted response variable for scoring calculations. The table includes the variables' name, role, type, format and length, as well as the creator, creator GUID, and function.
- **Training Code** — Displays the SAS code that Model Studio used to train the node.
- **Properties** — Specifies the various properties that you selected before running the node.
- **Output** — Displays the SAS output of the quantile regression run.

Assessment

- **Predicted Reports** — Displays the predicted and target mean as a function of the depth for the model. The predicted and target mean are given for each of the data roles. To examine the predicted mean as a function of the target mean, use the drop-down menu in the upper right corner. This result is displayed only if the target is an interval variable.
- **Fit Statistics** — Displays a table of the fit statistics for the model, broken down by data role.

Model Interpretability

- **Surrogate Model Variable Importance** — Displays a table of the relative importance of the input variables, including variable level and label. Relative importance is calculated using a one-level decision tree for each input variable to predict the predicted value as a global surrogate model.

- **PD Plot** — Displays the functional relationship between the input variable and the model prediction. For interval input variables, the 95% confidence interval for the average target prediction is indicated by the shaded band around the line. The X axis includes a heat map that shows the distribution of the input variable. The extreme values of the input variables are removed by truncating the lower and upper tails of its distribution. The amount of truncation can be controlled by specifying values for the **Truncate lower tail** and the **Truncate upper tail** properties.
- **PD and ICE Overlay Plot** — Displays the functional relationship between the input variable and the model prediction as well as the functional relationship between individual observations and the model prediction. For each individual observation, the corresponding ICE curve displays values of the input variable on the X axis and the corresponding predicted probability or prediction of the target variable on the Y axis, holding the other input variable values constant at that observation. The X axis includes a heat map that shows the distribution of the input variable. The extreme values of the input variables are removed by truncating the lower and upper tails of its distribution. The amount of truncation can be controlled by specifying values for the **Truncate lower tail** and the **Truncate upper tail** properties.
- **LIME Explanations** — The LIME algorithm calculates a more easily interpretable linear model around an individual observation. The chart displays the regression estimates for the input variables that are selected in the local surrogate linear regression model. The input variables are ordered by significance such that the most important variable for the local regression model is at the bottom of the chart. The LASSO technique is used to select the most significant predictors from the set of input variables that are used to train the model. Each nominal input variable is binary-encoded based on whether it matches the level of the individual observation. A positive estimate indicates that the observed value of the input variable increases the predicted probability of the event.
- **Kernel SHAP Values** — For each individual observation, an input variable's Kernel SHAP value is the contribution of the observed value of the input variable to the predicted probability or prediction of the target variable. The Kernel SHAP values are the regression coefficients that are obtained by fitting a weighted least squares regression. The Kernel SHAP values of all input variables sum to one. The input variables are ordered by significance such that the most important variable according to the absolute Kernel SHAP values is at the top of the chart. Each nominal input is binary-encoded based on whether it matches the individual observation. Interval inputs are binary encoded based on their proximity to the individual observation.
- **LIME Information Table** — Displays information about the settings used to perform the LIME method.
- **Kernel SHAP Information Table** — Displays information about the settings used to perform the Kernel SHAP method.
- **LIME Fidelity Table** — Displays information about the accuracy of the local explanation predictions that are calculated using the LIME method, compared to the black-box machine learning model. The model prediction, the explainer prediction, and the explainer root mean square error (RMSE) are displayed.
- **Kernel SHAP Fidelity Table** — Displays information about the accuracy of the local explanation predictions that are calculated using the Kernel SHAP method, compared to the black-box machine learning model. The model prediction, the explainer prediction, and the explainer root mean square error (RMSE) are displayed.

Replacement

<i>Overview of Replacement</i>	203
<i>Replacement Properties</i>	203
Replacement Value Property	203
Interval Inputs	204
<i>Replacement Results</i>	205

Overview of Replacement

The **Replacement** node is a Data Mining Preprocessing node. It is used to generate score code to replace outliers and unknown class levels with specified values. In some cases, you might want to reassign specified nonmissing values (trim your variable's distribution) before performing imputation calculations for the missing values. This is a typical task for the **Replacement** node.

Input variables that have a DATE, DATETIME, or TIME format are ignored by the **Replacement** node.

Replacement Properties

Replacement Value Property

- **Replacement value for unknown class levels** — Specifies the replacement method for unknown class levels encountered in the scoring process. Here are the possible values:

- Ignore** — Specifies that variables that contain unknown levels are left unmodified.
- Missing value** — Specifies that variables that contain unknown levels are replaced with SAS missing value notations.
- Mode (most frequent level)** — Specifies that variables that contain unknown levels are replaced with the mode.

The default value is **Ignore**.

Interval Inputs

- **Default limits method** — Specifies the default method by which the lower and upper limits are derived for interval variables. Here are the possible values:
 - (none)**
 - Extreme percentiles**
 - Median absolute deviation (MAD)**
 - Metadata limits** — Specifies that any limits stored in the metadata are used for replacement.
 - Standard deviation from the mean**

The default value is **Standard deviation from the mean**.


- **Extreme percentile** — Specifies the maximum and minimum upper and lower percentile to be used in deriving the upper and lower limits. The default value is 0.5.
- **MAD deviations** — Specifies the number of MAD deviations from the median to be used in deriving the lower and upper limits. The default value is 9.
- **Alternate limits method** — Specifies the method for calculating interval variable limits when metadata lacks stored limits. Here are the possible values:
 - (none)**
 - Standard deviation from the mean**
 - Median absolute deviation (MAD)**
 - Extreme percentiles**

The default value is **Standard deviation from the mean**.

- **Standard deviations** — Specifies the number of Standard deviations from the mean to be used in deriving the lower and upper limits. The default value is 3.
- **Replacement value** — Specifies the replacement value for lower and upper outlier values. Possible values are **Computed limits** and **Missing value**. The default setting is **Computed limits**.

Replacement Results

After running the node, open the Results window by right-clicking the node and selecting **Results**.

You can download the data that is used to build the charts and tables in the Results window by clicking . A CSV file with the chart or table data will immediately begin downloading.

In the **Output Data** tab, you can view, save, and explore the data that is generated by the node. For more information, see [Explore and Visualize](#).

Summary

- **Class Variables** — Displays a list table of the replacement class variables and their replacement values for unknown levels.
- **Interval Variables** — Displays a list table that summarizes the replaced variables, the limits method, the lower and upper limits, the replacement method, and the lower and upper replacement values.
- **Replacement Counts** — Displays a list table that summarizes the variable replacement counts, their role, and level.
- **Node Score Code** — Displays SAS score code that was created by the node. The SAS score code can be used outside the Model Studio environment in custom user applications.
- **Properties** — Specifies the various properties that you selected before running the node.
- **Output** — Displays the SAS output of the replacement run.

SAS Code

<i>Overview of SAS Code</i>	207
<i>SAS Code Properties</i>	208
Post Training Properties	208
<i>Code Editor User Interface</i>	210
Macros Table	210
Macro Variables Table	213
Code Editor Menu	219
<i>SAS Code Results</i>	220
<i>Using the SAS Code Node</i>	223

Overview of SAS Code

The **SAS Code** node is a Miscellaneous node that enables you to incorporate new or existing SAS code into Model Studio pipelines. The **SAS Code** node enables you to write separate training code and DS1 scoring code. The **SAS Code** node extends the functionality of Model Studio by making other SAS procedures available for use in your data mining analysis. You can also write SAS DATA steps to create customized scoring code, conditionally process data, or manipulate existing data sets. The **SAS Code** node is also useful for building predictive models, formatting SAS output, defining table and plot views in the user interface, and for modifying variable metadata. The **SAS Code** node can be placed at any location within a pipeline. By default, the **SAS Code** node does not require data. However, a training table is always available to use. The exported data that is produced by a successful **SAS Code** node run can be used by successive nodes in a pipeline. To use the exported data in successive nodes, you must do one of the following:

- Produce score code that is applied to the incoming data to create a new table for the successive node. This table is not dynamically generated and is not persisted.
- Use the `dmcas_register` macro to indicate that the table should be persisted in CAS and used by successive nodes. Use the `dm_output_data` or

```
dm_output_memname macro variables to specify the output table: %dmcas
register(dataset = &dm_output_memname, type = cas);
```

To indicate that the **SAS Code** node produces a model that should be assessed, right-click the **SAS Code** node, select **Move**, and then select **Supervised Learning**. If a **SAS Code** node that is marked as a Supervised Learning node does not generate any score code, either as DS1 or as an analytic store, then no assessment reports or model interpretability reports are generated. If the node produces score code that does not create the expected predicted or posterior probability variables, then the node run fails.

For Model Studio examples using the **SAS Code** node, see [SAS Code Node Examples](#) in the Model Studio user guide documentation.

SAS Code Properties

- **Open Code Editor** — Invokes the SAS Code Editor. There are editors for writing both **Training code** and **Scoring code**.
- **Training data only** — If the data is partitioned, this option specifies whether the node should receive only the training observations. By default, this option is deselected.
- **Use the exact percentile method for lift calculations** — Specifies whether to use the exact percentile method for calculating lift and related assessment measures. When this property is deselected, the iterative method is used. The exact method should be used if there are convergence errors when using the iterative method.

Post Training Properties

Changing the following properties will not retrain the model.

Model Interpretability — Specifies the method of model interpretation in order to show how the model responds to changing inputs.

- **Global Interpretability**
 - **Variable importance** — Specifies whether to display a table of the relative importance values for the input variables. The surrogate variable importance values are calculated using a one-level decision tree for each input to predict the predicted value as a global surrogate model. For tree-based supervised learning models, the model variable importance table is displayed as well.
 - **PD plots** — Specifies whether to display partial dependency (PD) plots that show the relationship between model input variables and their predictors. PD plots are generated for only up to the top ten variables in the relative importance table. Here is how to determine the variables that are displayed in the PD plot: for tree-based models (forest, decision tree, and gradient boosting), the model variable importance table is used. For other supervised learning models, the surrogate variable importance table is used.
- **Local Interpretability**

- **ICE plots** — Specifies whether to display Individual Conditional Expectation (ICE) plots. ICE plots can reveal interesting subgroups and interactions between model variables. ICE plots are generated for only up to the top ten variables in the relative importance table. Here is how to determine the variables that are displayed in the ICE plot: for tree-based models (forest, decision tree, and gradient boosting), the model variable importance table is used. For other supervised learning models, the surrogate variable importance table is used.
- **LIME** — Specifies whether to use the Locally Interpretable Model-Agnostic Explanation (LIME) method to explain the model predictions.
- **Kernel SHAP** — Specifies whether to use the Kernel Shapley additive explanations (Kernel SHAP) method to explain the model predictions.
- **Maximum number of Kernel SHAP variables** — Specifies the maximum number of Kernel SHAP variables to plot. Possible values range from 1 to 100. The default value is 20.
- **Specify instances to explain** — Specifies the individual observations that you want to explain. The **Random** observations option provides explanations for five randomly selected observations. The **Specify up to 5** observations option enables you to provide IDs for the specific observations in the data. The IDs should be unique values of the variable that has the role **Key** in the metadata. If no variable is assigned the role **Key**, then you can use the variable `_dmIndex_` to specify the individual observations.
- **LIME/Kernel SHAP Tables**
 - **Explainer Information table** — Specifies whether the Explainer Information tables for the LIME and Kernel SHAP methods should be included in the node results. The Explainer Information table contains information about what settings were used to perform the LIME and Kernel SHAP methods.
 - **Explainer Fidelity table** — Specifies whether the Explainer Fidelity tables for the LIME and Kernel SHAP methods should be included in the node results. The Explainer Fidelity table contains information about the accuracy of the local explanation predictions, compared to the black-box machine learning model.
- **PD/ICE Options**
 - **Maximum number of variables** — Specifies the maximum number of input variables to use to generate the PD and ICE plots. The input variables are selected based on their relative importance. Possible values range from 1 to 10. The default value is 5.
 - **Number of observations** — Specifies the maximum number of observations to sample for the PD and ICE plots. The default value is 1000.
 - **Number of tick points** — Specifies the number of tick points for an interval analysis variable to generate for the PD and ICE plots. Possible values range from 3 to 100. The default value is 50.
 - **Truncate lower tail** — Specifies under what percentage to discard the values of the interval analysis variables for the PD and ICE plots. Possible values range from 0 to 99. The default value is 5%.
 - **Truncate upper tail** — Specifies above what percentage to discard the values of the interval analysis variables for the PD and ICE plots. Possible values range from 0 to 99. The default value is 95%.

- **Seed** — Specifies the seed for generating random numbers. The seed is used for selecting individual observations to explain, selecting observations to generate PD and ICE plots, and simulating data for LIME and Kernel SHAP.

Code Editor User Interface

Macros Table

The following macros are defined and available for use in the SAS Code Editor.

Note: The macros are not available for use in the **Scoring code** window.

Macro Group	Macro Name	Macro Description
DATA: VARIABLES	dm_binary_input	A macro that identifies the nliteral names of the binary input variables.
	dm_dec_target	A macro that identifies the nliteral name of the target variable.
	dm_id	A macro that identifies the nliteral names of the ID variables.
	dm_interval_input	A macro that identifies the nliteral names of the interval input variables.
	dm_into_var	A macro that identifies the nliteral name of the classification variable for a class target variable. This macro is defined, but empty for an interval target variable.
	dm_key	A macro that identifies the nliteral name of the key variable.
	dm_nominal_input	A macro that identifies the nliteral names of the nominal input variables.
	dm_offset	A macro that identifies the nliteral name of the offset variable.
	dm_ordinal_input	A macro that identifies the nliteral names of the ordinal input variables.

Macro Group	Macro Name	Macro Description
	dm_predicted_var	A macro that identifies the nliteral names of the posterior probability variables for a class target variable or the predicted variable for an interval target variable.
	dm_segment	A macro that identifies the nliteral names of the segment variables.
	dm_text	A macro that identifies the nliteral names of the text variables.
	dm_unary_input	A macro that identifies the nliteral names of the unary input variables with missing values.
UTILITIES	dmcas_checkMacroVar	A macro that checks the existence of a macro variable. A default value is assigned if the macro variable does not exist.
UTILITIES	dmcas_copybinaryfile	A macro that copies a binary file.
UTILITIES	dmcas_copyfile	A macro that copies a text file.
UTILITIES	dmcas_metachange	<p>A macro that specifies changes to the metadata. You must specify the actual variable name (instead of the nliteral name). The following attributes can be modified:</p> <ul style="list-style-type: none"> ■ comment ■ label ■ level ■ lower limit ■ impute ■ order ■ role. The supported roles are ID, Input, Rejected, and Segment. ■ transform ■ upper limit
UTILITIES	dmcas_register	<p>A macro that registers files, data sets, and CAS tables. The macro arguments are:</p> <ul style="list-style-type: none"> ■ DATASET — The data set name. ■ FILE — The file name.

Macro Group	Macro Name	Macro Description
		<ul style="list-style-type: none"> ■ TYPE — The type of file. Possible values are text, folder, and cas. When a data set has the type set to cas, the table is persisted to the project CASLIB. ■ PROPERTY — Specifies (Y or N) property file. The default value is N. ■ MODELMANAGER — Specifies (Y or N) whether the file or data set is part of the package when registering the associated model to SAS Model Manager. The default value is N.
UTILITIES	dmcas_report	<p>A macro that is used to register and define reports. The macro arguments are:</p> <ul style="list-style-type: none"> ■ DATASET — The data set name. ■ FILE — The file name. ■ REPORTTYPE — The report type. Possible values are BandPlot, BarChart, CodeEditor, DecisionTree, Html, IciclePlot, NetworkDiagram, PieChart, ScatterPlot, SeriesPlot, and Table. ■ DESCRIPTION — The report description. ■ X — The category variable. ■ Y — The response variable. ■ GROUP — The group variables. ■ SUBGROUP — The subgroup variables. ■ FREQ — The frequency variable. ■ BY — The BY or lattice variables. ■ WHERE — The WHERE clause. ■ AUTODISPLAY — Specifies (Y or N) whether to auto display the report when the result viewer is open. ■ BLOCK — Defines, if applicable, the group in which the report belongs when opening the result viewer.
UTILITIES	dmcas_varmacro	<p>A macro that is used to generate a macro that contains a list of</p>

Macro Group	Macro Name	Macro Description
		<p>variables that satisfy a specified WHERE clause. The macro also generates a macro variable that identifies the number of variables in the list. The macro arguments are:</p> <ul style="list-style-type: none"> ■ NAME — The macro name. ■ METADATA — The metadata data set that contains a list of variables and their attributes. ■ WHERE — The WHERE clause. This argument is optional. ■ KEY — The variable that identifies the variable names. The default value is NAME. ■ NUMMACRO — The name of the macro variable to contain the number of variables retrieved. ■ UPCASE — Specifies (Y or N) whether to upcase the variable names. The default value is N. ■ SINGLEQUOTE — Specifies (Y or N) whether to enclose the variable names in single quotation marks. The default value is N. ■ QUOTE — Specifies (Y or N) whether to enclose the variable names in double quotation marks. The default value is N. ■ COMMA — Specifies (Y or N) whether to separate the variable names with a comma. The default value is N.

Macro Variables Table

The following macro variables are defined and available for use in the SAS Code Editor.

Note: The macro variables are not available for use in the **Scoring code** window.

Macro Variable Group	Macro Variable	Macro Variable Definition
CAS: FILES	dm_data	A macro variable that identifies the CAS training table.
	dm_data_caslib	A macro variable that identifies the CASLIB that is associated with the training table.
	dm_data_outmodel	A macro variable that identifies the remote CAS outmodel table.
	dm_data_rstore	A macro variable that identifies the remote analytical store.
	dm_datalib	A macro variable that identifies the libref that is associated with the training table.
	dm_ds_caslib	A macro variable that identifies the CAS library of the project source table.
	dm_memname	A macro variable that identifies the name of the training table (that is, a one-level name).
	dm_memnamenlit	A macro variable that identifies the nliteral name of the training table (that is, a one-level name).
	dm_outmodeltable	A macro variable that identifies the name of the remote CAS outmodel table (a one-level name).
	dm_output_data	A macro variable that identifies the CAS output table. This table is used to produce assessment reports for supervised learning nodes.
	dm_output_memname	A macro variable that identifies the name of the output table (that is, a one-level name).
	dm_projectData	A macro variable that identifies the project CAS source table.
	dm_rstoretable	A macro variable that identifies the name of the remote analytic store associated with the node (a one-level name).

Macro Variable Group	Macro Variable	Macro Variable Definition
CAS: GENERAL	dm_cashost	A macro variable that identifies the host name for the server that is associated with the CAS session.
	dm_caslib	A macro variable that identifies the CASLIB that is associated with the training table.
	dm_casport	A macro variable that identifies the port number for the server associated with the CAS session.
	dm_casessionid	A macro variable that identifies the session ID of the CAS session.
	dm_casessref	A macro variable that identifies the name of the CAS session to which you want to connect.
DATA: PARTITION	dm_partition_statement	A macro variable that identifies the partition statement.
	dm_partition_test_val	A macro variable that identifies the value of the partition variable that corresponds to the test observations.
	dm_partition_train_val	A macro variable that identifies the value of the partition variable that corresponds to the training observations.
	dm_partition_valid_val	A macro variable that identifies the value of the partition variable that corresponds to the validation observations.
	dm_partitiontestwhereclausenlit	A macro variable that identifies the WHERE clause based on the nliteral name of the partition variable that can be applied to retrieve the test observations (for example, '_PartInd_'n = 2).
	dm_partitiontrainwhereclausenlit	A macro variable that identifies the WHERE clause based on the nliteral name of the partition variable that can be applied to retrieve the training observations (for example, '_PartInd_'n = 1).


Macro Variable Group	Macro Variable	Macro Variable Definition
	dm_partitionvalidwhereclauselit	A macro variable that identifies the WHERE clause based on the literal name of the partition variable that can be applied to retrieve the validation observations (for example, '_PartInd_'n = 0).
	dm_partitionvar	A macro variable that identifies the name of the partition variable. This macro variable is blank if the data is not partitioned. This macro variable is _partInd_ if the partitioning is done by Model Studio. If a partition variable is specified in the Data tab, then this macro variable is the name of the specified partition variable.
DATA: TARGET	dm_data_targetInfo	A macro variable that identifies the SAS data set that contains information about the target variable. For a class target, this data set contains the level values and frequencies, the associated posterior probability values, and the name of the classification variable. For an interval target, the data set contains the name of the predicted variable.
	dm_dec_event	A macro variable that identifies the event level of a class target variable. By default, the event is the first level as defined by the order of the target variable. You can also select the event level of a class target variable in the Data tab.
	dm_dec_format	A macro variable that identifies the format of the target variable.
	dm_dec_label	A macro variable that identifies the label of the target variable.
	dm_dec_level	A macro variable that identifies the measurement level of the target variable. Possible measurement levels are BINARY, INTERVAL, ORDINAL, and NOMINAL.
	dm_dec_nonevent	A macro variable that identifies the nonevent value for a binary target.

Macro Variable Group	Macro Variable	Macro Variable Definition
	dm_dec_order	A macro variable that identifies the order of the target variable. Possible values are ASCFMT, ASC, DESC, and DESFMT. The default value is DESC.
	dm_dec_type	A macro variable that identifies the type (C or N) of the target variable.
	dm_dec_vvntartget	A macro variable that identifies the name of the target variable.
	dm_into_vvnvar	A macro variable that identifies the classification variable for a class target.
	dm_predicted_vvnvar	A macro variable that identifies the prediction variable for an interval target or the posterior variable associated with the target event for a class target.
DATA: VARIABLE	dm_key	A macro variable that identifies the name of the key variable.
	dm_num_binary_input	A macro variable that identifies the number of binary input variables.
	dm_num_id	A macro variable that identifies the number of ID variables.
	dm_num_interval_input	A macro variable that identifies the number of interval input variables.
	dm_num_nominal_input	A macro variable that identifies the number of nominal input variables.
	dm_num_ordinal_input	A macro variable that identifies the number of ordinal input variables.
	dm_num_segment	A macro variable that identifies the number of segment variables.
	dm_num_text	A macro variable that identifies the number of text variables.
	dm_num_unary_input	A macro variable that identifies the number of unary input variables with missing values. Variables that are

Macro Variable Group	Macro Variable	Macro Variable Definition
		constant without missing values are assigned the role rejected.
	dm_projectmetadata	A macro variable that identifies the SAS data set that contains the project metadata.
NODE: GENERAL	dm_codebar	A macro variable that identifies a code separator.
	dm_dsep	A macro variable that identifies the operating system file delimiter (for example, “\” for Windows).
	dm_labelid	A macro variable that identifies a number that is used to construct array names and statement labels in the generated DS1 code. This is used to prevent naming issues when combining models.
	dm_maxnamelen	A macro variable that identifies the maximum length for the name of a new variable.
	dm_nodeguid	A macro variable that identifies the global identifier of the node. This identifier is used to register tables and results.
	dm_nodeid	A macro variable that identifies the nodeID.
NODE: LOCAL FILES	dm_data_outfit	A macro variable that identifies the fit-statistics data set.
	dm_data_scorevar	A macro variable that identifies the data set to be used to register individual variable transformations.
	dm_file_astore	A macro variable that identifies the local analytical store.
	dm_file_deltacode	A macro variable that identifies the file that contains the DATA step code that modifies the columnmeta metadata that is exported by the node.

Macro Variable Group	Macro Variable	Macro Variable Definition
	dm_file_scorecode	A macro variable that identifies the file that contains the score code.
	dm_file_traincode	A macro variable that identifies the file that contains the training code.
	dm_folder_scorevar	A macro variable that identifies the local folder that contains the score code files. The score code files contain individual variable transformations.
	dm_lib	A macro variable that identifies the SAS library where components should save and create local tables.
	dm_metadata	A macro variable that identifies the local data set that contains the imported metadata for the node.
	dm_nodedir	A macro variable that identifies the folder where local files are created.

Code Editor Menu

- **Code Editor Settings** — Specifies options for code editor settings. Click  to open the **Code Editor Settings** window. The following options are available:
 - **General** — Specifies general property configurations:
 - **Enable code folding** — Specifies whether to enable wraparound of code text. By default, this option is selected.
 - **Show line numbers** — Specifies whether to display line numbers to the left of the code editor pane. By default, this option is selected.
 - **Enable line highlighting** — Specifies whether to highlight the line the cursor is currently on. By default, this option is selected.
 - **Font size** — Specifies the font size of the text inside the editor pane. Possible font sizes range from 8 to 72, 16 being the default value.
 To set all values back to default, click **Reset**.
 - **Editing** — Specifies properties related to the editor. The following properties are available:
 - **Enable autocomplete** — Specifies whether to enable autocomplete for SAS keywords, properties, and functions. By default, this option is selected.

- **Enable color coding** — Specifies whether to enable the color coding system that uses different colors to differentiate between different types of SAS keywords. By default, this option is selected.
- **Tab width** — Specifies the width of a Tab keystroke in spaces. The default value is 4.
- **Replace tabs with spaces** — Specifies whether to substitute a Tab keystroke with the equivalent number of spaces. By default, this option is deselected.
- **Enable auto indentation** — Specifies whether to automatically indent at the beginning of new code blocks. By default, this option is selected.

To set all values back to default, click **Reset**.


- **Undo** — Reverses the most recent change to the code pane.
- **Redo** — Restores the most recent undone change to the code pane.
- **Cut** — Deletes the selected item and copies it to the clipboard.
- **Copy** — Copies the selected item to the clipboard.
- **Paste** — Pastes a copied item from the clipboard.
- **Find and replace** — Opens the Find and Replace dialog box, which enables you to search for and replace text in the code, output, and log.
- **Clear all code** — Clears all of the text from the current code pane.

The menu to the upper right corner of the code editor pane has the following items:

- **Save** — Saves the contents in the current view of the code pane.
- **Close** — Closes the code pane.



SAS Code Results

After running the node, open the Results window by right-clicking the node and selecting **Results**.

You can download the data that is used to build the charts and tables in the Results window by clicking . A CSV file with the chart or table data will immediately begin downloading.

In the **Output Data** tab, you can view, save, and explore the data that is generated by the node. For more information, see [Explore and Visualize](#).

Several charts have detailed descriptions available that explain how the charts are generated. The descriptions also provide information about how to interpret the charts that is custom to your particular data and models. To access these detailed descriptions:

- Expand an individual chart by clicking . The detailed description is in the right panel.
- Select  on an individual chart. A window appears with the detailed description.

Note: The detailed descriptions are available only in the English language.

Node

- **SAS Code** — Displays the SAS code from the Training code editor.
- **Node Score Code** — Displays the SAS scoring code.
- **Path EP Score Code** — Displays the SAS code that was created by the node. The score code can be used outside the Model Studio environment to score new data.
- **DS2 Package Code** — Displays the SAS DS2 package code that was created by the node. The score code can be used outside the Model Studio environment to score new data. DS2 package code is used when publishing models to SAS Micro Analytic Service.
- **Score Inputs** — Displays a table of data on the input variables for scoring calculations. The table includes the variables' name, role, level, type, label, format, and length.
- **Score Outputs** — Displays a table of data on the predicted response variable for scoring calculations. The table includes the variables' name, role, type, format and length, as well as the creator, creator GUID, and function.
- **Properties** — Specifies the various properties that you selected before running the node.
- **Output** — Displays the SAS output of the code node run—that is, the SAS output of the code in the Code Editor.

Assessment

- **Predicted Reports** — Displays the predicted and target mean as a function of the depth for the model. The predicted and target mean are given for each of the data roles. To examine the predicted mean as a function of the target mean, use the drop-down menu in the upper right corner. This result is displayed only if the target is an interval variable.
- **Lift Reports** — Displays the cumulative lift as a function of the depth for the model. The cumulative lift is given for each of the data roles. To examine other statistics as a function of depth, use the drop-down menu in the upper right corner. Other statistics include lift, gain, captured response percentage, cumulative captured response percentage, response percentage, and cumulative response percentage. This result is displayed only if the target is a class variable.
- **ROC Reports** — Displays the ROC (receiver operating characteristic) chart for a model, giving the sensitivity as a function of 1-specificity. The sensitivity is given for each of the data roles. To examine other statistics, use the drop-down menu in the upper right corner. Other statistics include accuracy and F1 score. This result is displayed only if the target is a class variable.
- **Fit Statistics** — Displays a table of the fit statistics for the model, broken down by data role.
- **Event Classification** — Displays the confusion matrix at various cutoff values for each partition. The confusion matrix contains true positives (events that are correctly classified as events), false positives (non-events that are classified as events), false negatives (events that are classified as non-events), and true negatives (non-events that are correctly classified as non-events). The classification cutoffs that are used are the default value of 0.5, the Kolmogorov-Smirnov cutoff value for each partition, and the **Node binary classification**

cutoff that is specified for binary targets. Use the drop-down menu to view the information in the chart as percentages or counts. You can also view the information summarized in a table. This result is displayed only if the target is a class variable.

- **Nominal Classification** — Displays either the percentage of observations or the number of observations that predict each level of the target variable. The plot is segmented by target level and partition level. The target level with the greatest predicted probability is the predicted outcome. This result is displayed only if the target is a nominal variable.

Model Interpretability

- **Surrogate Model Variable Importance** — Displays a table of the relative importance of the input variables, including variable level and label. Relative importance is calculated using a one-level decision tree for each input variable to predict the predicted value as a global surrogate model.
- **PD Plot** — Displays the functional relationship between the input variable and the model prediction. For interval input variables, the 95% confidence interval for the average target prediction is indicated by the shaded band around the line. The X axis includes a heat map that shows the distribution of the input variable. The extreme values of the input variables are removed by truncating the lower and upper tails of its distribution. The amount of truncation can be controlled by specifying values for the **Truncate lower tail** and the **Truncate upper tail** properties.
- **PD and ICE Overlay Plot** — Displays the functional relationship between the input variable and the model prediction as well as the functional relationship between individual observations and the model prediction. For each individual observation, the corresponding ICE curve displays values of the input variable on the X axis and the corresponding predicted probability or prediction of the target variable on the Y axis, holding the other input variable values constant at that observation. The X axis includes a heat map that shows the distribution of the input variable. The extreme values of the input variables are removed by truncating the lower and upper tails of its distribution. The amount of truncation can be controlled by specifying values for the **Truncate lower tail** and the **Truncate upper tail** properties.
- **LIME Explanations** — The LIME algorithm calculates a more easily interpretable linear model around an individual observation. The chart displays the regression estimates for the input variables that are selected in the local surrogate linear regression model. The input variables are ordered by significance such that the most important variable for the local regression model is at the bottom of the chart. The LASSO technique is used to select the most significant predictors from the set of input variables that are used to train the model. Each nominal input variable is binary-encoded based on whether it matches the level of the individual observation. A positive estimate indicates that the observed value of the input variable increases the predicted probability of the event.
- **Kernel SHAP Values** — For each individual observation, an input variable's Kernel SHAP value is the contribution of the observed value of the input variable to the predicted probability or prediction of the target variable. The Kernel SHAP values are the regression coefficients that are obtained by fitting a weighted least squares regression. The Kernel SHAP values of all input variables sum to one. The input variables are ordered by significance such that the most important variable according to the absolute Kernel SHAP values is at the top of the chart. Each nominal input is binary-encoded based on whether it matches the

individual observation. Interval inputs are binary encoded based on their proximity to the individual observation.

- **LIME Information Table** — Displays information about the settings used to perform the LIME method.
- **Kernel SHAP Information Table** — Displays information about the settings used to perform the Kernel SHAP method.
- **LIME Fidelity Table** — Displays information about the accuracy of the local explanation predictions that are calculated using the LIME method, compared to the black-box machine learning model. The model prediction, the explainer prediction, and the explainer root mean square error (RMSE) are displayed.
- **Kernel SHAP Fidelity Table** — Displays information about the accuracy of the local explanation predictions that are calculated using the Kernel SHAP method, compared to the black-box machine learning model. The model prediction, the explainer prediction, and the explainer root mean square error (RMSE) are displayed.

Using the SAS Code Node

You can use your **SAS Code** node to produce a custom model that is incorporated into your pipeline. To indicate that your **SAS Code** node will produce a model that should be assessed and processed by the **Model Comparison** node:

- 1 Right-click the node, and select **Move**.
- 2 Select **Supervised Learning**.
- 3 The **SAS Code** node is now connected to a **Model Comparison** node.

In order for the model produced by the **SAS Code** node to be assessed, it must either produce DATA step score code or an analytic store that will produce the expected predictor or posterior variables. The `&dm_lib..dmcas_targetLevel` data set provides the expected names as a macro variable.

In order to produce the **Surrogate Model Variable Importance** chart, the **Event Classification** chart, and the **Nominal Classification** chart, you must create a classification variable in the form `I_<targetVariableName>`. The classification variable can be identified with the variable `dm_classtarget_intovar`. The various levels of the target variable are defined in `dm_classtarget_level`.

To move your **SAS Code** node out of the Supervised Learning group:

- 1 Right-click the **SAS Code** node, and select **Move**.
- 2 Select **Preprocessing**.

DS1 code must be written in the **Scoring code** editor. Otherwise, the node run fails.

Save Data

<i>Overview of Save Data</i>	225
<i>Save Data Properties</i>	226
General Properties	226
<i>Save Data Results</i>	226

Overview of Save Data

The **Save Data** node is a Miscellaneous node that enables you to save the training table that is produced by a predecessor node to a CAS library. This table could be partitioned into training, validation, or test sets based on the project settings. In that case, the table contains the `_partind_` variable that identifies the partitions.

By default, the training table that is produced by a pipeline is temporary, exists only for the duration of the run of a node, and has local session scope. The **Save Data** node enables you to save that table to disk and promote that table to the location that is associated with the specified output library. This table can then be used later by other applications for further analysis or reporting.

The default output CAS library where tables are saved can be specified in **Output Library Project Settings**. You can overwrite this location using the **Output library** property. In addition, you can load the table in memory and promote this table to have global scope in the specified CAS library. This enables multiple CAS sessions to access the table.

Save Data Properties

General Properties

- **Output library** — Specifies the output CAS library where the table is saved on disk. Use **Browse** to navigate to the proper library. If the user has specified an **Output Library** under **Project Settings**, then this library is used by default.

Note: The **Output library** property is not saved or set if you save the node or pipeline to the exchange, duplicate or modify the node in the exchange, or import a project with a **Save Data** node. This avoids potential issues that can arise when running the node outside the initial environment. For example, the CAS library might not exist in the new environment or a different user who runs the node might not have access to the CAS library or table. The exception to this is a user who has specified **Output Library** in the Project Settings window. The Project Settings library specified is always used by default.


- **Table name** — Specifies the name for the CAS table being saved. The default value is **tmpSaveData**.
- **Replace existing table** — Specifies whether to override an existing CAS table with the same name when saving. By default, this option is deselected.

Note: The node run fails if these two conditions are true: the **Replace existing table** option is deselected, and if there is a table that already exists with the same name in the specified output library.

- **Promote table** — Specifies whether to load the table in memory and promote the table to global space. By default, this option is deselected.

Save Data Results

After running the node, open the Results window by right-clicking the node and selecting **Results**.

You can download the data that is used to build the charts and tables in the Results window by clicking . A CSV file with the chart or table data will immediately begin downloading.

In the **Output Data** tab, you can view, save, and explore the data that is generated by the node. For more information, see [Explore and Visualize](#).

Summary

- **Properties** — Specifies the various properties selected prior to running the node. For the **Save Data** node, these include the output library, the table name, whether to replace or promote the table, and the CAS session ID.
- **Output** — Displays the SAS output of the save data run.

Score Code Import

<i>Overview of Score Code Import</i>	229
<i>Using the Score Code Import Node</i>	230
<i>Score Code Import Properties</i>	230
Post Training Properties	231
<i>Score Code Import Results</i>	232

Overview of Score Code Import

The **Score Code Import** node is a Supervised Learning node that enables you to import the score code from an existing model that might have been created outside Model Studio. The existing model must have the same target variable that is specified in the Model Studio project data. You can import models with two different code representations:

- A single DATA step code file that must be accessible from the Model Studio client.
- A SAS analytic store, which is a binary representation of the state of an analytic procedure after training. This analytic store must also be accompanied by analytic store score code, also referred to as *EP score code*. These files represent your model and can be used by the **Score Code Import** node.

The EP score code file must be accessible from the client. In order to import an analytic store, it must be loaded into a caslib available in Model Studio. When the node runs, the score code is used to score the project data and assess this model. The input variables that are used in the model must also exist in the Model Studio project data.

The score code or analytic store that is imported remains static each time you run your pipeline.

Note: Any references to CAS libraries, URIs, or analytic stores are not saved or set if you save the node or pipeline to the exchange, duplicate or modify the node in the exchange, or import a project with a **Score Code Import** node. This avoids potential

issues that can arise when running the node outside the initial environment. For example, the CAS library might not exist in the new environment or a different user who runs the node might not have access to the CAS library or table.


Using the Score Code Import Node

To use the **Score Code Import** node:

- 1 Build a process flow diagram in SAS Enterprise Miner. Your process flow diagram must end in a **Score** node.
- 2 Right-click the **Score** node in your diagram and select **Results**. In the Results window, select **View** ⇒ **Scoring** ⇒ **SAS Code**. In the **SAS Code** pane, select **File** ⇒ **Save As**, and specify a location to save the score code. Once you select a location to save the score code, click **OK**.

To use analytic store code, save the EP score code file in a folder that the Model Studio client can access and save the score data to a CAS library that the mid-tier can access, such as a public CAS library. The Model Studio mid-tier cannot access an analytic store that is located in a user's personal CAS library.

The analytic store that is produced by SAS Enterprise Miner is a binary file that cannot be accessed directly by Model Studio. The file must be uploaded to a CAS library that Model Studio can access. You can do this by using the **UPLOAD** statement in the **ASTORE** procedure.

- 3 In a Model Studio pipeline, add a **Score Code Import** node to your pipeline.
- 4 In the **Score Code Import** node properties pane, click **Open Code Editor** and select either **DATA step code** or **Analytic store code**.
- 5 Navigate to the file that you saved earlier. If you are using analytic store code, you must specify the EP score code file and the analytic store table. The score code that you created in SAS Enterprise Miner is opened in the code editor.
- 6 Click  to save this code. Then click **Close**.
- 7 Continue building your pipeline. When you run your pipeline, the **Score Code Import** node runs alongside all other nodes.

Score Code Import Properties

- **Open Code editor** — Invokes the Score Code Editor. For more information, see [Code Editor User Interface on page 210](#).
- **Use the exact percentile method for lift calculations** — Specifies whether to use the exact percentile method for calculating lift and related assessment measures. When this property is deselected, the iterative method is used. The

exact method should be used if there are convergence errors when using the iterative method.

Post Training Properties

Changing the following properties will not retrain the model.

Model Interpretability — Specifies the method of model interpretation in order to show how the model responds to changing inputs.

Note: If you are importing a model from SAS Enterprise Miner that uses Date variables as input variables, you must change the variable role to **ID** in order to generate model interpretability reports.

■ Global Interpretability

- **Variable importance** — Specifies whether to display a table of the relative importance values for the input variables. The surrogate variable importance values are calculated using a one-level decision tree for each input to predict the predicted value as a global surrogate model. For tree-based supervised learning models, the model variable importance table is displayed as well.
- **PD plots** — Specifies whether to display partial dependency (PD) plots that show the relationship between model input variables and their predictors. PD plots are generated for only up to the top ten variables in the relative importance table. Here is how to determine the variables that are displayed in the PD plot: for tree-based models (forest, decision tree, and gradient boosting), the model variable importance table is used. For other supervised learning models, the surrogate variable importance table is used.

■ Local Interpretability


- **ICE plots** — Specifies whether to display Individual Conditional Expectation (ICE) plots. ICE plots can reveal interesting subgroups and interactions between model variables. ICE plots are generated for only up to the top ten variables in the relative importance table. Here is how to determine the variables that are displayed in the ICE plot: for tree-based models (forest, decision tree, and gradient boosting), the model variable importance table is used. For other supervised learning models, the surrogate variable importance table is used.
- **LIME** — Specifies whether to use the Locally Interpretable Model-Agnostic Explanation (LIME) method to explain the model predictions.
- **Kernel SHAP** — Specifies whether to use the Kernel Shapley additive explanations (Kernel SHAP) method to explain the model predictions.
- **Maximum number of Kernel SHAP variables** — Specifies the maximum number of Kernel SHAP variables to plot. Possible values range from 1 to 100. The default value is 20.
- **Specify instances to explain** — Specifies the individual observations that you want to explain. The **Random** observations option provides explanations for five randomly selected observations. The **Specify up to 5** observations option enables you to provide IDs for the specific observations in the data. The IDs should be unique values of the variable that has the role **Key** in the

metadata. If no variable is assigned the role **Key**, then you can use the variable `_dmIndex_` to specify the individual observations.

- **LIME/Kernel SHAP Tables**
 - **Explainer Information table** — Specifies whether the Explainer Information tables for the LIME and Kernel SHAP methods should be included in the node results. The Explainer Information table contains information about what settings were used to perform the LIME and Kernel SHAP methods.
 - **Explainer Fidelity table** — Specifies whether the Explainer Fidelity tables for the LIME and Kernel SHAP methods should be included in the node results. The Explainer Fidelity table contains information about the accuracy of the local explanation predictions, compared to the black-box machine learning model.
- **PD/ICE Options**
 - **Maximum number of variables** — Specifies the maximum number of input variables to use to generate the PD and ICE plots. The input variables are selected based on their relative importance. Possible values range from 1 to 10. The default value is 5.
 - **Number of observations** — Specifies the maximum number of observations to sample for the PD and ICE plots. The default value is 1000.
 - **Number of tick points** — Specifies the number of tick points for an interval analysis variable to generate for the PD and ICE plots. Possible values range from 3 to 100. The default value is 50.
 - **Truncate lower tail** — Specifies under what percentage to discard the values of the interval analysis variables for the PD and ICE plots. Possible values range from 0 to 99. The default value is 5%.
 - **Truncate upper tail** — Specifies above what percentage to discard the values of the interval analysis variables for the PD and ICE plots. Possible values range from 0 to 99. The default value is 95%.
- **Seed** — Specifies the seed for generating random numbers. The seed is used for selecting individual observations to explain, selecting observations to generate PD and ICE plots, and simulating data for LIME and Kernel SHAP.

Score Code Import Results

After running the node, open the Results window by right-clicking the node and selecting **Results**.

You can download the data that is used to build the charts and tables in the Results window by clicking . A CSV file with the chart or table data will immediately begin downloading.

In the **Output Data** tab, you can view, save, and explore the data that is generated by the node. For more information, see [Explore and Visualize](#).

Several charts have detailed descriptions available that explain how the charts are generated. The descriptions also provide information about how to interpret the

charts that is custom to your particular data and models. To access these detailed descriptions:

- Expand an individual chart by clicking ↕. The detailed description is in the right panel.
- Select ⓘ on an individual chart. A window appears with the detailed description.

Note: The detailed descriptions are available only in the English language.

Node

- **Path Score Code** — Provides the optimized SAS code from SAS Enterprise Miner.
- **Path EP Score Code** — Provides the optimized SAS code from SAS Enterprise Miner when you import an analytic store model.
- **Properties** — Specifies the various properties that you selected before running the node.
- **Score Inputs** — Displays a table of data on the input variables for scoring calculations. The table includes the variables' name, role, level, type, label, format, and length.
- **Score Outputs** — Displays a table of data on the predicted response variable for scoring calculations. The table includes the variables' name, role, type, format and length, as well as the creator, creator GUID, and function.

Assessment

- **Predicted Reports** — Displays the predicted and target mean as a function of the depth for the model. The predicted and target mean are given for each of the data roles. To examine the predicted mean as a function of the target mean, use the drop-down menu in the upper right corner. This result is displayed only if the target is an interval variable.
- **Lift Reports** — Displays the cumulative lift as a function of the depth for the model. The cumulative lift is given for each of the data roles. To examine other statistics as a function of depth, use the drop-down menu in the upper right corner. Other statistics include lift, gain, captured response percentage, cumulative captured response percentage, response percentage, and cumulative response percentage. This result is displayed only if the target is a class variable.
- **ROC Reports** — Displays the ROC (receiver operating characteristic) chart for a model, giving the sensitivity as a function of 1-specificity. The sensitivity is given for each of the data roles. To examine other statistics, use the drop-down menu in the upper right corner. Other statistics include accuracy and F1 score. This result is displayed only if the target is a class variable.
- **Fit Statistics** — Displays a table of the fit statistics for the model, broken down by data role.
- **Event Classification** — Displays the confusion matrix at various cutoff values for each partition. The confusion matrix contains true positives (events that are correctly classified as events), false positives (non-events that are classified as events), false negatives (events that are classified as non-events), and true negatives (non-events that are correctly classified as non-events). The classification cutoffs that are used are the default value of 0.5, the Kolmogorov-Smirnov cutoff value for each partition, and the **Node binary classification cutoff** that is specified for binary targets. Use the drop-down menu to view the

information in the chart as percentages or counts. You can also view the information summarized in a table. This result is displayed only if the target is a class variable.

- **Nominal Classification** — Displays either the percentage of observations or the number of observations that predict each level of the target variable. The plot is segmented by target level and partition level. The target level with the greatest predicted probability is the predicted outcome. This result is displayed only if the target is a nominal variable.

Model Interpretability

- **Surrogate Model Variable Importance** — Displays a table of the relative importance of the input variables, including variable level and label. Relative importance is calculated using a one-level decision tree for each input variable to predict the predicted value as a global surrogate model.
- **PD Plot** — Displays the functional relationship between the input variable and the model prediction. For interval input variables, the 95% confidence interval for the average target prediction is indicated by the shaded band around the line. The X axis includes a heat map that shows the distribution of the input variable. The extreme values of the input variables are removed by truncating the lower and upper tails of its distribution. The amount of truncation can be controlled by specifying values for the **Truncate lower tail** and the **Truncate upper tail** properties.
- **PD and ICE Overlay Plot** — Displays the functional relationship between the input variable and the model prediction as well as the functional relationship between individual observations and the model prediction. For each individual observation, the corresponding ICE curve displays values of the input variable on the X axis and the corresponding predicted probability or prediction of the target variable on the Y axis, holding the other input variable values constant at that observation. The X axis includes a heat map that shows the distribution of the input variable. The extreme values of the input variables are removed by truncating the lower and upper tails of its distribution. The amount of truncation can be controlled by specifying values for the **Truncate lower tail** and the **Truncate upper tail** properties.
- **LIME Explanations** — The LIME algorithm calculates a more easily interpretable linear model around an individual observation. The chart displays the regression estimates for the input variables that are selected in the local surrogate linear regression model. The input variables are ordered by significance such that the most important variable for the local regression model is at the bottom of the chart. The LASSO technique is used to select the most significant predictors from the set of input variables that are used to train the model. Each nominal input variable is binary-encoded based on whether it matches the level of the individual observation. A positive estimate indicates that the observed value of the input variable increases the predicted probability of the event.
- **Kernel SHAP Values** — For each individual observation, an input variable's Kernel SHAP value is the contribution of the observed value of the input variable to the predicted probability or prediction of the target variable. The Kernel SHAP values are the regression coefficients that are obtained by fitting a weighted least squares regression. The Kernel SHAP values of all input variables sum to one. The input variables are ordered by significance such that the most important variable according to the absolute Kernel SHAP values is at the top of the chart. Each nominal input is binary-encoded based on whether it matches the individual observation. Interval inputs are binary encoded based on their proximity to the individual observation.

- **LIME Information Table** — Displays information about the settings used to perform the LIME method.
- **Kernel SHAP Information Table** — Displays information about the settings used to perform the Kernel SHAP method.
- **LIME Fidelity Table** — Displays information about the accuracy of the local explanation predictions that are calculated using the LIME method, compared to the black-box machine learning model. The model prediction, the explainer prediction, and the explainer root mean square error (RMSE) are displayed.
- **Kernel SHAP Fidelity Table** — Displays information about the accuracy of the local explanation predictions that are calculated using the Kernel SHAP method, compared to the black-box machine learning model. The model prediction, the explainer prediction, and the explainer root mean square error (RMSE) are displayed.

Score Data

<i>Overview of Score Data</i>	237
<i>Score Data Properties</i>	238
Score Data	238
Output Data	238
<i>Score Data Results</i>	239

Overview of Score Data

The **Score Data** node is a Miscellaneous node that enables you to score a data table with the score code that was generated by the predecessor nodes in the pipeline. The scored table can be saved or promoted to a CAS library.

By default, the scored table is temporary, exists only for the duration of the run of a node, and has local session scope. The **Score Data** node enables you to save the scored table to disk in the location that is associated with the specified output library. After it is saved to disk, this table can be used by other applications for further analysis or reporting.

Use the **Output Library** field in the Project Settings window to specify the default output CAS library. That is, the location where the scored table is saved. You can override the default location by specifying a different value in the **Output library** property of the **Score Data** node. In addition, you can load the table in memory and promote this table to have global scope in the specified CAS library. This enables multiple CAS sessions to access the table.

If the output table contains the project target variable and the score generates the expected predicted variables, then assessment results are generated.

Nodes cannot be added after the **Score Data** node because it does not prepare data for successor nodes.

Score Data Properties

Score Data

- **Table name** — Specifies the input table to be scored. Use **Browse** to navigate to the proper data source.

Note: The **Table name** property is not saved or set if you save the node or pipeline to the exchange, duplicate or modify the node in the exchange, or import a project with a **Score Data** node. This avoids potential issues that can arise when running the node outside the initial environment. For example, the CAS library might not exist in the new environment or a different user who runs the node might not have access to the CAS library or table.

Output Data

- **Output library** — Specifies the output library where the score table is created. Use **Browse** to navigate to the proper data library. If the user has specified an **Output Library** under **Project Settings**, then this library is used by default.

Note: The **Output library** property is not saved or set if you save the node or pipeline to the exchange, duplicate or modify the node in the exchange, or import a project with a **Score Data** node. This avoids potential issues that can arise when running the node outside the initial environment. For example, the CAS library might not exist in the new environment or a different user who runs the node might not have access to the CAS library or table.

- **Table name** — Specifies the name for the CAS table being saved. The default value is **tmpScoreData**.
- **Save table** — Specifies whether to save the table to the output library. By default, this option is selected.
- **Replace existing table** — Specifies whether to overwrite the existing CAS table with the same name when saving. By default, this option is deselected.

Note: The node run fails if these two conditions are true: the **Replace existing table** option is deselected, and if there is a table that already exists with the same name in the specified output library.


- **Promote table** — Specifies whether to promote the table to a global space. By default, this option is deselected.

- **Drop rejected variables** — Specifies whether variables with a role of **Rejected** should be dropped in the output data set. By default, this option is deselected.

Score Data Results

After running the node, open the Results window by right-clicking the node and selecting **Results**.

The scored input data that you specified in the **Table name** property (located under the **Scored Data** properties) is available on the **Output Data** tab.

You can download the data that is used to build the charts and tables in the Results window by clicking . A CSV file with the chart or table data will immediately begin downloading.

In the **Output Data** tab, you can view, save, and explore the data that is generated by the node. For more information, see [Explore and Visualize](#).

Summary

- **Path Score Code** — Displays the SAS code that was created by the node, as well as all preceding nodes, if there are no analytic stores that are generated in the pipeline. The score code can be used outside the Model Studio environment to score new data.
- **Path EP Score Code** — Displays the SAS code that was created by the node if there are analytic stores that are generated in the pipeline. The score code can be used outside the Model Studio environment to score new data.
- **DS2 Package Code** — Displays the SAS DS2 package code that was created by the node. The score code can be used outside the Model Studio environment to score new data. DS2 package code is used when publishing models to SAS Micro Analytic Service.
- **Score Inputs** — Displays a table of data on the input variables for scoring calculations. The table includes the variables' name, role, level, type, label, format, and length.
- **Score Outputs** — Displays a table of all variables that are created by the score code. The table includes the variables' name, role, type, format and length, as well as the creator, creator GUID, and function.
- **Properties** — Specifies the various properties that you selected before running the node.

Segment Profile

<i>Overview of Segment Profile</i>	241
<i>Segment Profile Properties</i>	242
General Properties	242
Input Variables	242
<i>Segment Profile Results</i>	243

Overview of Segment Profile

The **Segment Profile** node is a Miscellaneous node that enables you to analyze segmented data in order to identify factors that differentiate each segment from the population. Segments are defined by the unique levels of the segment variable and, if specified, the unique levels of the target variable. The target can also be treated as an input for the purposes of this analysis. Two frequency profile reports and a parallel coordinates report are the primary means for analyzing the segmented data.

To prepare the data for these reports, the input variables for each segment are analyzed using a classification tree to extract the important inputs. The important inputs are used to profile each segment. Then, the frequency distribution of each input in its associated segment is derived, in addition to the overall input frequency distribution. You can Winsorize the distributions at the 1st and 99th percentiles in order to remove the effect of extreme outliers.

Segments can be derived from a variable in the input data set with the role **Segment**. Segments can also be derived from the output of the **Clustering** node. A **Clustering** node must precede the **Segment Profile** node in the pipeline, and the derived cluster variable must be identified with the **Segment** role in the **Clustering** node properties.

Segment Profile Properties

General Properties

- **Input data partition** — Specifies which data partition to analyze. Here are the possible values:
 - All data**
 - Test**
 - Train**
 - Validate**The default value is **All data**.
- **Profile all segments** — Specifies whether to profile all segments or to group small segments into the `_OTHER_` category. By default, this option is deselected.
- **Cutoff percentage** — Specifies the cutoff percentage value at which segment categories that occur infrequently are combined into the `_OTHER_` category. This option is available when **Profile all segments** is deselected. The default value is 95.
- **Exclude missing segment values** — Specifies whether to exclude observations with missing segment values from the profile analysis. By default, this option is deselected.
- **Winsorize frequency profiles** — Specifies whether to address extreme outliers by winsorizing the frequency distribution profiles. Winsorization is performed at the 1st and 99th percentiles. By default, this option is selected.
- **Target role** — Specifies the analysis role for the target variable. Here are the possible values:
 - (none)** — Specifies to exclude the target from the analysis.
 - Input** — Specifies to include the target as an input variable
 - Segment** — Specifies to treat the target variable as another segment variable

The default value is **(none)**.


Input Variables

- **Number of bins for interval inputs** — Specifies the number of histogram bars that are used in the frequency distribution profiles of interval input variables. Possible values range from 2 to 32. The default value is 8.

- **Maximum number of inputs to display** — Specifies the maximum number of input variables that are selected for each segment to be displayed in the Profile by Segment and Overall results chart. This property also identifies the maximum number of input variables that are displayed in the Profile results chart. A tree-based algorithm is used to filter the input variables. Possible values range from 1 to 25. The default value is 10.
- **Maximum tree depth** — Specifies the maximum tree depth for input selection. If the maximum depth is set to 1, then the input variables are evaluated one at a time and selected based on their worth. If the maximum depth is greater than 1, then the input variables are selected based on their importance generated from different splits in the tree. Possible values range from 1 to 4. The default value is 1.
- **Minimum logworth** — Specifies the minimum logworth that is used to filter the input variables for a 1-level tree. Input variables with logworth values less than this value are excluded from the analysis. The default value is 30.

Segment Profile Results

After running the node, open the Results window by right-clicking the node and selecting **Results**.

You can download the data that is used to build the charts and tables in the Results window by clicking . A CSV file with the chart or table data will immediately begin downloading.

In the **Output Data** tab, you can view, save, and explore the data that is generated by the node. For more information, see [Explore and Visualize](#).

Summary

- **Segment Plot** — The segment frequency pie chart.
- **Parallel Coordinates** — A visual display of how the observations in each segment are spread among the top ranked inputs. This report is limited to a maximum of 8 inputs, excluding the segment variable. Click a segment to show only the observations in that particular segment.

The **Parallel Coordinates** chart might not be displayed for performance reasons. The cardinality of the input variables, the maximum number of input variables, and the number of interval input variable bins determine whether this chart is displayed. You can enable the display of this chart by re-running the node after decreasing the values specified in the **Number of bins for interval inputs** property or the **Maximum number of inputs to display** property.

The **Parallel Coordinates** chart does not include observations where input variables contain missing values. Therefore, this chart might not contain all expected polylines, input variable values, input variable levels, or segment variable levels. The number of input variables and the number of missing values in those input variables determine what data is displayed in the chart. To ensure that all data is included in this chart, you can run the **Imputation** node prior to the **Segment Profile** node to impute missing values.

Note: If the **Parallel Coordinates** chart is not displayed in the Results window, the following CAS errors appear in the SAS log. Other than the fact that the chart does not appear, these errors have no impact.

ERROR: There are more levels than the specified limit.

ERROR: The action stopped due to errors.

- **Profile** — A bar chart that contains the frequency distribution of each of the top ranked inputs, grouped by segment. You can specify an input variable with the menu at the top of the pane.
- **Profile by Segment and Overall** — A segment lattice of bar charts by input. For each input selected in the menu at the top of the pane, the frequency distribution bar chart is displayed for each segment where the input was selected. In addition, an overlay of the overall input distribution is included, which is plotted as unfilled outlines.
- **Variable Worth and Importance by Segment** — When **Maximum tree depth** is 1, this displays a variable-worth bar chart of the selected inputs in the specified segment. When **Maximum tree depth** is greater than 1, this displays a variable importance (relative importance) bar chart of the selected inputs in the specified segment. The segment is specified via the menu at the top of the pane.
- **Properties** — Specifies the various properties that you selected before running the node.
- **Output** — Displays the SAS output of the segment profile run.

SVM

<i>Overview of SVM</i>	245
<i>SVM Properties</i>	246
General Properties	246
Perform Autotuning	246
Lift Calculation Property	249
Binary Classification Cutoff	249
Post Training Properties	249
<i>SVM Results</i>	251

Overview of SVM

The **SVM** node is a Supervised Learning node. An SVM, or support vector machine, is a supervised machine-learning method that performs support vector classification for a binary target and support vector regression for an interval target.

Most problems in a finite dimensional space are not linearly separable. In this case, the original space must be mapped into a much higher dimensional space, which makes the separation easier. The node currently supports linear and low-degree polynomial kernels and uses interior-point method for training optimization. With a polynomial kernel, each observation is explicitly projected. Thus, setting **Polynomial degree** to 2 is not recommended when the number of columns in the design matrix is much greater than 100. Similarly, setting **Polynomial degree** to 3 is not recommended when the number of columns in the design matrix is greater than 32.

SVM Properties

General Properties

- **Scale inputs** — Specifies whether to scale input variables to be between 0 and 1 (inclusive). By default, this option is selected. This option is available only for binary targets.
- **Use missing** — Specifies whether to use missing values. Missing values are treated as a level for class input variables, and missing values for interval input variables are imputed to the mean before training. By default, this option is deselected.
- **Include iterations report** — Specifies whether to include an iterations report. The iterations report displays model accuracy after each iteration. Generating this report requires accuracy to be computed after every iteration, which might result in longer computation times. By default, this option is deselected. This option is available only for binary targets.
- **Penalty** — Specifies the penalty value. The default value is 1.
- **Kernel** — Specifies the kernel type that the support vector machine uses. Possible values are **Linear** and **Polynomial**. The default value is **Linear**.

Note: Specifying a polynomial kernel with a degree of 2 is not recommended when the number of columns in the design matrix is much greater than 100. Specifying a polynomial kernel with a degree of 3 is not recommended when the number of columns in the design matrix is greater than 32.

- **Polynomial Degree** — Specifies the degree of the polynomial that is used. This option is available only if the **Kernel** type is **Polynomial**. Possible values are 2 and 3. The default value is 2.
- **Tolerance** — Specifies the minimum number at which the iteration stops. The default value is 0.000001.
- **Maximum iterations** — Specifies the maximum number of iterations allowed with each try. The default value is 25.

Perform Autotuning

Specifies whether to perform autotuning of any SVM parameters. Warning: Performing autotuning can substantially increase run time.

Note: Autotuning is available only for binary targets.

If **Perform Autotuning** is selected, the following options are available:

- **Penalty** — Specifies whether to autotune the penalty value. If selected, the following option is available:
 - **Initial value** — Specifies the initial value for autotuning the penalty parameter. The default value is 1. Use the **From** and **To** options to specify the range. The default **From** value is 0.000001, and the default **To** value is 100.
- **Polynomial degree** — Specifies whether to autotune the polynomial degree for the SVM model. If selected, the following option is available:
 - **Initial value** — Specifies the initial value for autotuning the polynomial degree. The default value is 1. Use the **From** and **To** options to specify the range. The default **From** value is 1, and the default **To** value is 3.

Note: Specifying a polynomial kernel with a degree of 2 is not recommended when the number of columns in the design matrix is much greater than 100. Specifying a polynomial kernel with a degree of 3 is not recommended when the number of columns in the design matrix is greater than 32.

- **Search Options** — Specifies the options for autotuning searching. The following options are available:
 - **Search method** — Specifies the tuning search method. Here are the possible values:
 - **Bayesian** — Specifies the Bayesian method. This method uses priors to seed the iterative optimization.
 - **Genetic algorithm** — Specifies the genetic algorithm method. This method uses an initial Latin Hypercube sample that seeds a genetic algorithm to generate a new population of alternative configurations at each iteration.
 - **Grid** — Specifies the grid search method. This method performs an exhaustive search of all configurations. Each hyperparameter of interest is discretized into a desired set of values to be studied. Models are trained and assessed for all combinations of values across all the hyperparameters (thus forming a multi-dimensional “grid”).
 - **Latin hypercube sample** — Specifies the Latin Hypercube method. This method performs an optimized grid search that is uniform in each tuning parameter, but random in combinations.
 - **Random** — Specifies the Random method. This method generates a single sample of purely random configurations.

The default value is **Genetic algorithm**.

- **Number of evaluations per iteration** — Specifies the number of tuning evaluations in one iteration. This option is available only if the **Search method** is **Genetic algorithm** or **Bayesian**. The default value is 10.
- **Maximum number of evaluations** — Specifies the maximum number of tuning evaluations. This option is available only if the **Search method** is **Genetic algorithm** or **Bayesian**. The default value is 50.
- **Maximum number of iterations** — Specifies the maximum number of tuning iterations. This option is available only if the **Search method** is **Genetic algorithm** or **Bayesian**. The default value is 5.

- **Maximum number of points in model** — Specifies the maximum number of points in the model. This option is available only if the **Search method** is **Bayesian**. The default value is 100.
- **Sample size** — Specifies the sample size. This option is available only if the **Search method** is **Random** or **Latin hypercube sample**. The default value is 50.
- **General Options** — Specifies the general properties for autotuning. The following options are available:
 - **Validation method** — Specifies the validation method for finding the objective value. Note that if your data is partitioned, then that partition is used and **Validation method**, **Validation data proportion**, and **Cross validation number of folds** are all ignored. Here are the possible values:
 - **Partition** — Specifies using a single partition of a training set. With partition, you specify proportions to use for randomly assigning observations to each role.
 - **K-fold cross validation** — Specifies using the cross validation method. In cross validation, each model evaluation requires k training executions (on $k-1$ data folds) and k scoring executions (on one holdout fold). This increases the evaluation time by approximately a factor of k .

For small to medium data tables, cross validation provides, on average, a better representation of error across the whole data table. **Partition** is the default value.
 - **Training data proportion** — Specifies the proportion of data to be used for training in the **Partition** validation method. The default value is 0.7.
 - **Validation data proportion** — Specifies the proportion of data to be used for the **Partition** validation method. The default value is 0.3.
 - **Cross validation number of folds** — Specifies the number of partition folds in the cross validation process (the k defined above). Possible values range from 2 to 20. The default value is 5.
 - **Class target objective function** — Specifies the objective function to optimize for tuning parameters for a class target. Here are the possible values:
 - **Area under the curve**
 - **Average squared error**
 - **F0.5 score**
 - **F1 score**
 - **Gamma**
 - **Gini coefficient**
 - **Kolmogorov-Smirnov statistic**
 - **Misclassification rate**
 - **Multi-class log loss**
 - **Root average squared error**
 - **Tau**

The default value is **Kolmogorov-Smirnov statistic**.

- **Maximum time (minutes)** — Specifies the maximum time (in minutes) for the optimization tuner. The default value is 60.
- **Maximum training time for single model (minutes)** — Specifies the maximum time (in minutes) for a single model train. To specify no maximum, do not enter a value for **Maximum training time for single model (minutes)**.

Lift Calculation Property

- **Use the exact percentile method for lift calculations** — Specifies whether to use the exact percentile method for calculating lift and related assessment measures. When this property is deselected, the iterative method is used. The exact method should be used if there are convergence errors when using the iterative method.

Binary Classification Cutoff

- **Specify node binary classification cutoff** — Specifies whether to use the binary classification cutoff specified below for the node. If this property is deselected, the project binary classification cutoff is used for determining the predicted value for a binary target based on the posterior probabilities. By default, this property is deselected.
- **Node binary classification cutoff** — Specifies the cutoff to use in the node for determining the predicted value for a binary target based on the posterior probabilities. This option is available only if **Specify node binary classification cutoff** is selected. The default value is 0.5.

.....

Note: The **Rules** settings in the Project Settings window enable you to specify a new cutoff value by selecting **Override the default classification cutoff**. If **Specify node binary classification cutoff** is deselected, then the value that you specified in the Project Settings window is used. If **Specify node binary classification cutoff** is selected, then the value that you specified in **Node binary classification cutoff** is used.

.....

Post Training Properties

Changing the following properties will not retrain the model.

Model Interpretability — Specifies the method of model interpretation in order to show how the model responds to changing inputs.

- **Global Interpretability**
 - **Variable importance** — Specifies whether to display a table of the relative importance values for the input variables. The surrogate variable importance


values are calculated using a one-level decision tree for each input to predict the predicted value as a global surrogate model. For tree-based supervised learning models, the model variable importance table is displayed as well.

- **PD plots** — Specifies whether to display partial dependency (PD) plots that show the relationship between model input variables and their predictors. PD plots are generated for only up to the top ten variables in the relative importance table. Here is how to determine the variables that are displayed in the PD plot: for tree-based models (forest, decision tree, and gradient boosting), the model variable importance table is used. For other supervised learning models, the surrogate variable importance table is used.
- **Local Interpretability**
 - **ICE plots** — Specifies whether to display Individual Conditional Expectation (ICE) plots. ICE plots can reveal interesting subgroups and interactions between model variables. ICE plots are generated for only up to the top ten variables in the relative importance table. Here is how to determine the variables that are displayed in the ICE plot: for tree-based models (forest, decision tree, and gradient boosting), the model variable importance table is used. For other supervised learning models, the surrogate variable importance table is used.
 - **LIME** — Specifies whether to use the Locally Interpretable Model-Agnostic Explanation (LIME) method to explain the model predictions.
 - **Kernel SHAP** — Specifies whether to use the Kernel Shapley additive explanations (Kernel SHAP) method to explain the model predictions.
 - **Maximum number of Kernel SHAP variables** — Specifies the maximum number of Kernel SHAP variables to plot. Possible values range from 1 to 100. The default value is 20.
 - **Specify instances to explain** — Specifies the individual observations that you want to explain. The **Random** observations option provides explanations for five randomly selected observations. The **Specify up to 5** observations option enables you to provide IDs for the specific observations in the data. The IDs should be unique values of the variable that has the role **Key** in the metadata. If no variable is assigned the role **Key**, then you can use the variable `_dmIndex_` to specify the individual observations.
 - **LIME/Kernel SHAP Tables**
 - **Explainer Information table** — Specifies whether the Explainer Information tables for the LIME and Kernel SHAP methods should be included in the node results. The Explainer Information table contains information about what settings were used to perform the LIME and Kernel SHAP methods.
 - **Explainer Fidelity table** — Specifies whether the Explainer Fidelity tables for the LIME and Kernel SHAP methods should be included in the node results. The Explainer Fidelity table contains information about the accuracy of the local explanation predictions, compared to the black-box machine learning model.
- **PD/ICE Options**
 - **Maximum number of variables** — Specifies the maximum number of input variables to use to generate the PD and ICE plots. The input variables are selected based on their relative importance. Possible values range from 1 to 10. The default value is 5.

- **Number of observations** — Specifies the maximum number of observations to sample for the PD and ICE plots. The default value is 1000.
- **Number of tick points** — Specifies the number of tick points for an interval analysis variable to generate for the PD and ICE plots. Possible values range from 3 to 100. The default value is 50.
- **Truncate lower tail** — Specifies under what percentage to discard the values of the interval analysis variables for the PD and ICE plots. Possible values range from 0 to 99. The default value is 5%.
- **Truncate upper tail** — Specifies above what percentage to discard the values of the interval analysis variables for the PD and ICE plots. Possible values range from 0 to 99. The default value is 95%.
- **Seed** — Specifies the seed for generating random numbers. The seed is used for selecting individual observations to explain, selecting observations to generate PD and ICE plots, and simulating data for LIME and Kernel SHAP.



SVM Results

After running the node, open the Results window by right-clicking the node and selecting **Results**.

You can download the data that is used to build the charts and tables in the Results window by clicking . A CSV file with the chart or table data will immediately begin downloading.

In the **Output Data** tab, you can view, save, and explore the data that is generated by the node. For more information, see [Explore and Visualize](#).

Several charts have detailed descriptions available that explain how the charts are generated. The descriptions also provide information about how to interpret the charts that is custom to your particular data and models. To access these detailed descriptions:

- Expand an individual chart by clicking . The detailed description is in the right panel.
- Select  on an individual chart. A window appears with the detailed description.

Note: The detailed descriptions are available only in the English language.

Node

- **Fit Statistics** — Displays the SVM fit statistics for the model. These include accuracy, error, sensitivity, and specificity, as well as their values for each data role (Train, Validate, or Test). The **Fit Statistics** table is available only for binary targets.
- **Training Results** — Displays a table of data on the SVM training statistics. The table includes the statistic, a corresponding qualitative description, and the value for the statistic.

- **Iteration Report** — Displays a table of data on model accuracy after every iteration of the training process for each data role. This chart is not displayed if **Include iterations report** is not selected.
- **Autotune Best Configuration** — Displays a table of the resulting values of parameters that were autotuned. This table is not displayed if autotuning is not selected.
- **Autotune Results** — Displays the iterative results of the autotuning process, showing how each of the parameter values evolve as the model iterates. This table is not displayed if autotuning is not selected.
- **Path EP Score Code** — Displays the SAS score code that was created by the node. The score code can be used outside the Model Studio environment to score new data.
- **DS2 Package Code** — Displays the SAS DS2 package code that was created by the node. The score code can be used outside the Model Studio environment to score new data. DS2 package code is used when publishing models to SAS Micro Analytic Service.
- **Score Inputs** — Displays a table of data on the input variables for scoring calculations. The table includes the variables' name, role, level, type, label, format, and length.
- **Score Outputs** — Displays a table of predicted response variables generated during scoring calculations. The table includes the variables' name, role, type, format and length, as well as the creator, creator GUID, and function.
- **Training Code** — Displays the SAS code that Model Studio used to train the node.
- **Properties** — Specifies the various properties that you selected before running the node.
- **Output** — Displays the SAS output of the SVM run.

Assessment

- **Lift Reports** — Displays the cumulative lift as a function of the depth for the model. The cumulative lift is given for each of the data roles. To examine other statistics as a function of depth, use the drop-down menu in the upper right corner. Other statistics include lift, gain, captured response percentage, cumulative captured response percentage, response percentage, and cumulative response percentage. This result is displayed only if the target is a class variable.
- **ROC Reports** — Displays the ROC (receiver operating characteristic) chart for a model, giving the sensitivity as a function of 1-specificity. The sensitivity is given for each of the data roles. To examine other statistics, use the drop-down menu in the upper right corner. Other statistics include accuracy and F1 score. This result is displayed only if the target is a class variable.
- **Fit Statistics** — Displays a table of the fit statistics for the model, broken down by data role.
- **Event Classification** — Displays the confusion matrix at various cutoff values for each partition. The confusion matrix contains true positives (events that are correctly classified as events), false positives (non-events that are classified as events), false negatives (events that are classified as non-events), and true negatives (non-events that are correctly classified as non-events). The classification cutoffs that are used are the default value of 0.5, the Kolmogorov-Smirnov cutoff value for each partition, and the **Node binary classification**

cutoff that is specified for binary targets. Use the drop-down menu to view the information in the chart as percentages or counts. You can also view the information summarized in a table. This result is displayed only if the target is a class variable.

Model Interpretability

- **Surrogate Model Variable Importance** — Displays a table of the relative importance of the input variables, including variable level and label. Relative importance is calculated using a one-level decision tree for each input variable to predict the predicted value as a global surrogate model.
- **PD Plot** — Displays the functional relationship between the input variable and the model prediction. For interval input variables, the 95% confidence interval for the average target prediction is indicated by the shaded band around the line. The X axis includes a heat map that shows the distribution of the input variable. The extreme values of the input variables are removed by truncating the lower and upper tails of its distribution. The amount of truncation can be controlled by specifying values for the **Truncate lower tail** and the **Truncate upper tail** properties.
- **PD and ICE Overlay Plot** — Displays the functional relationship between the input variable and the model prediction as well as the functional relationship between individual observations and the model prediction. For each individual observation, the corresponding ICE curve displays values of the input variable on the X axis and the corresponding predicted probability or prediction of the target variable on the Y axis, holding the other input variable values constant at that observation. The X axis includes a heat map that shows the distribution of the input variable. The extreme values of the input variables are removed by truncating the lower and upper tails of its distribution. The amount of truncation can be controlled by specifying values for the **Truncate lower tail** and the **Truncate upper tail** properties.
- **LIME Explanations** — The LIME algorithm calculates a more easily interpretable linear model around an individual observation. The chart displays the regression estimates for the input variables that are selected in the local surrogate linear regression model. The input variables are ordered by significance such that the most important variable for the local regression model is at the bottom of the chart. The LASSO technique is used to select the most significant predictors from the set of input variables that are used to train the model. Each nominal input variable is binary-encoded based on whether it matches the level of the individual observation. A positive estimate indicates that the observed value of the input variable increases the predicted probability of the event.
- **Kernel SHAP Values** — For each individual observation, an input variable's Kernel SHAP value is the contribution of the observed value of the input variable to the predicted probability or prediction of the target variable. The Kernel SHAP values are the regression coefficients that are obtained by fitting a weighted least squares regression. The Kernel SHAP values of all input variables sum to one. The input variables are ordered by significance such that the most important variable according to the absolute Kernel SHAP values is at the top of the chart. Each nominal input is binary-encoded based on whether it matches the individual observation. Interval inputs are binary encoded based on their proximity to the individual observation.
- **LIME Information Table** — Displays information about the settings used to perform the LIME method.

- **Kernel SHAP Information Table** — Displays information about the settings used to perform the Kernel SHAP method.
- **LIME Fidelity Table** — Displays information about the accuracy of the local explanation predictions that are calculated using the LIME method, compared to the black-box machine learning model. The model prediction, the explainer prediction, and the explainer root mean square error (RMSE) are displayed.
- **Kernel SHAP Fidelity Table** — Displays information about the accuracy of the local explanation predictions that are calculated using the Kernel SHAP method, compared to the black-box machine learning model. The model prediction, the explainer prediction, and the explainer root mean square error (RMSE) are displayed.

Text Mining

<i>Overview of Text Mining</i>	255
<i>Text Mining Properties</i>	256
Language Property	256
Parsing Options	256
Topic Discovery	257
<i>Text Mining Results</i>	257

Overview of Text Mining

The **Text Mining** node is a Data Mining Preprocessing node that enables you to process text data in a document collection. This data can be used to build predictive models in a distributed computing environment. Data is processed in two phases: text parsing and transformation. Text parsing processes textual data into a term-by-document frequency matrix. Transformations such as singular value decomposition (SVD) alter this matrix into a data set that is suitable for data mining purposes. Thus, a document collection with thousands of documents and terms can be represented in a compact and efficient form.

The Text Mining node uses Natural Language Processing (NLP) techniques. These techniques reduce the need for tedious manual analysis by supporting automated parsing, tokenization, part-of-speech tagging, stemming, and entity extraction.

Stop lists are automatically included and applied for all languages. A stop list is a data set that contains a list of low-information words or extraneous words to exclude from the parsing results.

The **Text Mining** node supports these languages:

Arabic	Chinese	Croatian	Czech
Danish	Dutch	English	Farsi

Finnish	French	German	Greek
Hebrew	Hindi	Hungarian	Indonesian
Italian	Japanese	Kazakh	Korean
Norwegian	Polish	Portuguese	Romanian
Russian	Slovak	Slovene	Spanish
Swedish	Tagalog	Thai	Turkish
Vietnamese			

Text Mining Properties

Language Property

- **Language** — Specifies the language to use for text parsing. Although all languages are displayed, the node fails if you attempt to use a language that you do not currently license. The default value is **English**.

Parsing Options


- **Include parts of speech** — Specifies whether to use part-of-speech tagging in parsing. By default, this option is selected.
- **Extract noun groups** — Specifies whether to extract noun groups in parsing. By default, this option is selected.
- **Extract entities** — Specifies whether to extract entities in parsing. By default, this option is deselected.
- **Stem terms** — Specifies whether to treat different terms with the same root as equivalent. By default, this option is selected.
- **Minimum number of documents** — Specifies the threshold for the number of documents in which a term must occur to appear in the text analysis. Possible values range from 1 to 100. The default value is 4.

Topic Discovery

- **Automatically determine number of topics** — Specifies whether to automatically determine the number of topics. When selected, topic generation is calculated based on a log function of the number of documents. By default, this option is selected.
- **Maximum topics** — Specifies the maximum number of topics to discover. This option is available only if **Automatically determine number of topics** is deselected. The default value is 25.

Text Mining Results

After running the node, open the Results window by right-clicking the node and selecting **Results**.

You can download the data that is used to build the charts and tables in the Results window by clicking . A CSV file with the chart or table data will immediately begin downloading.

In the **Output Data** tab, you can view, save, and explore the data that is generated by the node. For more information, see [Explore and Visualize](#).

Summary

- **Kept Terms** — Displays 2,000 rows from a list of the terms that are kept in the document collection. This list table includes information about the terms, such as their role, attribute, and frequency.
- **Dropped Terms** — Displays 2,000 rows from a list of the terms that have been dropped in the document collection. This list table includes information about the terms, such as their role, attribute, and frequency.
- **Terms: Role by Frequency** — Displays a bar chart giving the total frequency of each occurrence of parsed terms in the document collection, broken down by term role.
- **Topics** — Displays the topics created by the node and the unique ID for each topic. The **Text Mining** node creates topics based on groups of terms that occur together in several documents. Each term-document pair is assigned a score for every topic in which it appears. Thresholds are then used to determine whether the association is strong enough to consider if that document or term belongs in the topic. As a result of this, terms and documents can belong to multiple topics.
- **Properties** — Specifies the various properties that you selected before running the node.
- **Output** — Displays the SAS output for the text mining run.

Transformations

<i>Overview of Transformations</i>	259
<i>Transformations Properties</i>	260
Interval Inputs	260
Ranking Criterion for Best Transformation	261
Binning	262
Class Inputs	262
General Properties	263
<i>Transformations Results</i>	263

Overview of Transformations

The **Transformations** node is a Data Mining Preprocessing node. It enables making transformations by replacing an input variable with a function of that variable. Transformations are usually applied so that the data seems to more closely meet the assumptions of the modeling algorithm that is to be applied.

Such deterministic mathematical functions can be used to stabilize variances, remove nonlinearity, and correct non-normality in variables to improve the fit of your model.

The Transformations node allows the transformation of all variables. For interval variables, the following transformations are available:

- Simple transformations
- Binning transformations
- Best transformations

For class variables, the binning of rare levels and various encoding methods is supported. In general, with the **Transformations** node, you can make both computed transformations and customized actions and then choose the best transformation based on your data and your modeling objective.

To help evaluate your transformations, the **Transformations** node can create summary statistics for each variable after the transformation is applied. Summary statistics require two additional passes through the data.

In the data that is exported from the node, a new variable is created for each variable that is transformed. The original variable is not overwritten. Instead, the new variable has the same name as the original variable but is prefaced with an identifier of the transformation. For example, variables that use the log transformation are prefaced with LOG_, and variables that use the square transformation are prefaced with SQR_. Binning transformations use the prefix BIN_ and the original variable name. The original version of each variable also exists in the exported data and by default has the role Rejected.

Input variables that have a DATE, DATETIME, or TIME format are ignored by the **Transformations** node.

You can set the transformation method for individual inputs by editing the **Transform** value for an input variable on the **Data** tab or by using the **Manage Variables** node. The method specified in the metadata (on the **Data** tab) takes precedence over the default method set in the **Manage Variables** node.

Note: When you specify a transformation method for a variable on the **Data** pane, that method is always used, regardless of the property settings for this node.

Transformations Properties

Interval Inputs

- **Default interval inputs method** — Specifies a default transformation method for all interval input variables that have no method assigned in the metadata table in the project data tab. Here are the possible values:
 - (none)
 - Best
 - Bucket binning
 - Centering
 - Exponential
 - Inverse
 - Inverse square
 - Inverse square root
 - Log (the natural logarithm)
 - Log 10
 - Quantile binning
 - Range standardization

- Square**
- Square root**
- Standardization**
- Tree-based binning**

The **Best** transform option performs several transformations and uses the transformation that has the best Chi-Square test for the target variable. The default value is **(none)**.

Note: **Log**, **Log 10**, **Square root**, and **Inverse square root** add an offset to variables to ensure positive values. **Inverse** and **Inverse square** add an offset to variables to ensure nonzero values. This prevents creating missing values during the transformation when input variable values are zero.

Ranking Criterion for Best Transformation

- **Criterion for interval target** — Specifies the selection criterion that is used when you select **Best** in the **Default interval inputs method** property. Here are the possible values:

- Average quantile kurtosis**
- Average quantile skewness**
- Moment kurtosis**
- Moment skewness**
- Pearson correlation with target**

The default value is **Moment skewness**.

- **Criterion for binary target** — Specifies the selection criterion that is used when you select **Best** in the **Default interval inputs method** property. Here are the possible values:

- Anderson-Darling statistic with target**
- Average quantile kurtosis**
- Average quantile skewness**
- Cramer-Von Mises statistic with target**
- Kolmogorov-Smirnov statistic with target**
- Moment kurtosis**
- Moment skewness**

The default value is **Moment skewness**.

- **Criterion for nominal target** — Specifies the selection criterion that is used when you select **Best** in the **Default interval inputs method** property. Here are the possible values:

- Average quantile kurtosis**
- Average quantile skewness**

- Moment kurtosis**
- Moment skewness**

The default value is **Moment skewness**.

Binning

If **Bucket binning**, **Quantile binning**, or **Tree-based binning** are selected as the **Default interval inputs method**, the following **Binning** configurations are available:

- **Number of bins** — Specifies the number of nonmissing bins to create for interval input variables when applying a binning method. Possible values range from 2 to 50. The default value is 4.
- **Create extra bins for values outside training range** — Specifies whether two extra bins, for values below the training minimum and above the training maximum, should be created when bucket or quantile binning. If not selected, these values are placed in the first or last bin, respectively. By default, this is deselected.
- **Minimum tree bin size** — Specifies the minimum number of observations in a tree bin. Bins that do not meet or exceed the minimum number of observations are not split out. This option is available only when the **Interval inputs method** is **Tree-based binning**. The default value is 5.

Class Inputs

- **Default class inputs method** — Specifies the default transformation method to use for class inputs. Here are the possible values:
 - (none)**
 - Bin rare nominal levels**
 - Level encoding**
 - Level frequency encoding**
 - Level proportion encoding**
 - Target encoding**
 - WOE encoding**

Target encoding and **WOE encoding** are calculated with respect to the target. **WOE encoding** is available only for binary and nominal targets. The default value is **(none)**.


- **Rare cutoff value percentage** — Specifies the cutoff percentage to determine which levels are considered rare. This option is available only when **Bin rare nominal levels** is selected. The default value is 0.5.
- **WOE adjustment value** — Specifies the adjustment factor for the weight of evidence (WOE) calculation. This option is available only when **WOE encoding** is selected. The default value is 0.5.

General Properties

- **Missing values treatment** — Specifies how missing values should be treated when binning or encoding. Possible values are **Ignore** and **Separate**. When **Ignore** is specified, the bin value is set to missing. When **Separate** is specified, missing values are assigned to their own bin. The default value is **Separate**.
- **Reject original variables** — Specifies whether the original, non-transformed variables should be rejected, and thus unavailable for usage in proceeding nodes.
- **Summary statistics** — Specifies whether summary statistics are generated for the transformed variables.
- **Ignore methods in metadata** — Specifies whether to ignore transformation methods that are specified on the **Data** tab, in global metadata, or in a preceding **Manage Variables** node. By default, this option is deselected.

Transformations Results

After running the node, open the Results window by right-clicking the node and selecting **Results**.

You can download the data that is used to build the charts and tables in the Results window by clicking . A CSV file with the chart or table data will immediately begin downloading.

In the **Output Data** tab, you can view, save, and explore the data that is generated by the node. For more information, see [Explore and Visualize](#).

Summary

- **Input Variable Statistics** — Displays information about input variables, including variable level, number of missing values, percent missing, minimum and maximum value, mean, midrange, importance standard deviation, skewness, and kurtosis.
- **Binning Summary** — Displays information about the binned variables.
- **Transformed Variables Summary** — Displays information about the transformed variables, including how they were transformed, the corresponding input variable, the formula applied, the variable level, variable type, and variable label.
- **Transformation Summary** — Displays a table containing additional statistics on the results of the transformations applied to the training data. This table gives the following properties:
 - Transformed Variable
 - Method
 - Ranking Criteria for Best Transformation

- Input Variable
- Formula
- Number of Missing Values
- Minimum
- Maximum
- Mean
- Standard Deviation
- Skewness
- Kurtosis
- Variable Label

This result is displayed only if you select **Summary statistics**. This result replaces **Transformed Variables Summary**.

- **Node Score Code** — Displays SAS score code that was created by the node. The SAS score code can be used outside the Model Studio environment in custom user applications.
- **Properties** — Specifies the various properties that you selected before running the node.
- **Output** — Displays the SAS output of the transformation run. For the **Transformations** node, this includes the input variable statistics.

Variable Clustering

<i>Overview of Variable Clustering</i>	265
<i>Variable Clustering Properties</i>	266
General Properties	266
Advanced Options	267
<i>Variable Clustering Results</i>	267

Overview of Variable Clustering

The **Variable Clustering** node is a Data Mining Preprocessing node. This node is a useful tool for data reduction because it finds the best variables for analysis. Variable clustering removes collinearity, decreases variable redundancy, and helps reveal the underlying structure of the input variables in a data set.

Variable clustering divides numeric variables into disjoint or hierarchical clusters. Variables in different clusters are conditionally independent given their own clusters. In general, the clustering process starts with one variable per cluster, and the number of variables per cluster increases with additional clustering steps. Correspondingly, the total number of clusters decreases with additional clustering steps, because clusters in previous steps are merged together. For each cluster that contains more than one variable, the variable (or variables) that contribute the most to the variation in that cluster is chosen as the representative variable. All other variables are rejected.

There are three threshold properties that stop the clustering process when one of the thresholds is met or exceeded:

- Number of clustering steps (upper threshold)
- Number of clusters (lower threshold)
- Number of variables per cluster (upper threshold)

Class variables are handled differently in Model Studio than they are in SAS Enterprise Miner. In SAS Enterprise Miner, the original Class variables are replaced by individual binary Class level variables, with one variable per Class level. And

those variables are handled separately in the selection process. In Model Studio, the individual binary Class level variables are used in the clustering process, but the original Class variables might or might not be used in the selection process. This depends on whether the **Export class level indicators** property is selected. If deselected, the original class variables are kept and are used in the variable selection process.

You can export either the best representative variable (or variables) of each cluster, or the cluster component:

- The most representative variable is the variable that contributes the most to the variation in the cluster. You can use the **Number of representative variables** property to specify the maximum number of representative variables per cluster to export. When class variables are included, you can export either the original class variables or the class level indicator variables.
- The cluster component is derived by taking the first principal component when running a principal component analysis on the variables in the cluster.

Variable Clustering Properties

General Properties

- **Cluster representation** — Specifies the identifier that is used to represent each cluster in proceeding nodes. Select **Best variable** to refer to the cluster by the name of the variable that best describes the cluster. Select **Cluster component** to refer to the cluster by a feature of the cluster.
- **Number of representative variables** — Specifies the maximum number of representative variables that are selected per cluster. In general, a highly correlated cluster surfaces no more than one or two representative variables. A less correlated cluster surfaces more representative variables, up to the number specified here.
- **Cluster identifier prefix** — Specifies the prefix used to name the variable cluster identifiers and to name the cluster component variables when **Cluster representation** is set to **Cluster component**. Long prefixes are truncated to 29 bytes or fewer when the node runs.
- **Include class inputs** — Specifies whether to include class variables in the clustering process.
- **Export class level indicators** — Specifies that the class level indicators are available to proceeding nodes, which replace the original class variables. The original class variables is assigned the role Rejected.
- **Optimal cluster selection method** — Specifies the method used to select the optimal cluster configuration. Select **Penalized log-likelihood** to calculate the Penalized value at each clustering step. The optimal cluster configuration determined is the clustering step with the lowest Penalized value. The default value is **(none)**.
- **Cluster Stopping Criteria**


- **Number of clustering steps** — Specifies the maximum number of clustering steps that are performed. Possible values range between 1 and 50. The default value is 6.
- **Number of clusters lower threshold** — Specifies the number of clusters below which the clustering process is stopped. There is one variable per cluster at the start of the variable clustering process. The number of clusters decreases with additional clustering steps. The default value is 1.
- **Use default maximum number of variables per cluster** — Specifies to use all available variables as the number of variables per cluster upper threshold. By default, this is selected.
- **Number of variables per cluster upper threshold** — Specifies the number of variables in a cluster above which the clustering process is stopped. There is one variable per cluster at the start of the variable clustering process. The number of variables per cluster increases with additional clustering steps. This option is available only if the **Use default maximum number of variables per cluster** option is deselected. The default value is 20.
- **Clustering RHO value** — Specifies the value of RHO for determining the sequence of regularization parameters used in sequential clustering steps. Larger RHO values require a stronger correlation between two variables in order for them to be clustered together. The default value is 0.8.

Advanced Options

- **Maximum number of iterations** — Specifies the maximum number of coordinate descent iterations for estimating the sparse precision covariance matrix. The default value is 1000.
- **Iteration convergence criterion** — Specifies the minimal absolute tolerance at which an iteration stops. The default value is 0.0001.

Variable Clustering Results

After running the node, open the Results window by right-clicking the node and selecting **Results**.

You can download the data that is used to build the charts and tables in the Results window by clicking . A CSV file with the chart or table data will immediately begin downloading.

In the **Output Data** tab, you can view, save, and explore the data that is generated by the node. For more information, see [Explore and Visualize](#).

Summary

- **Clustered Variables Network** — Displays a spatial map that gives the orientation and relative distance of clusters and cluster members. Cluster members with a stronger link are connected by a thicker line.

- **Clustered Variables Table** — Displays a table that contains all of the clustered variables, listing their cluster IDs, variable labels, principal components, and whether they were selected. The value chosen for the **Number of representative variables** property determines the number of principal components that are displayed.

Note: For class variables, the principal component might be blank. This is valid and expected.

- **Selected Variable Summary** — Displays a table that contains information about the principal components for each variable that is selected.
- **Class Variable Mapping** — Displays a table that describes how individual class level variables are named in the clustering process. This table is displayed only when class input variables are included in the clustering.
- **Output Variable Metadata** — Displays the metadata table for the output variables. This table contains the role of each output variable and if rejected, the reason for the rejection.
- **Node Score Code** — Displays the SAS score code that was created by the node. The SAS score code can be used outside the Model Studio environment in custom user applications.
- **Properties** — Specifies the various properties that you selected before running the node.
- **Output** — Displays the SAS output of a variable clustering run.

Variable Selection

<i>Overview of Variable Selection</i>	269
<i>Variable Selection Properties</i>	270
General Properties	270
Unsupervised Selection	271
Fast Supervised Selection	272
Linear Regression Selection	273
Decision Tree Selection	276
Forest Selection	278
Gradient Boosting Selection	279
Create Validation from Training	281
<i>Variable Selection Results</i>	281

Overview of Variable Selection

Many data mining databases have hundreds of potential model inputs (independent or explanatory variables) that can be used to predict the target (dependent or response variable). The **Variable Selection** node can help you reduce the number of inputs by rejecting input variables based on the selection results. Although rejected variables are passed to subsequent nodes in the pipeline, these variables are not used as model inputs by a successor modeling node.

The **Variable Selection** node is a Data Mining Preprocessing node that quickly identifies input variables, which are useful for predicting the target variable. The information-rich inputs can then be evaluated in more detail by one of the modeling nodes.

Here are the available selection methods in the variable selection process:

- **Unsupervised Selection** — Identifies the set of input variables that jointly explain the maximum amount of data variance. The target variable is not considered with this method.
- **Supervised Selection Methods**

- **Fast Supervised Selection** — Identifies the set of input variables that jointly explain the maximum amount of variance contained in the target variable.
- **Linear Regression Selection** — Fits and performs variable selection on an ordinary least squares regression predictive model. This is valid for an interval target or a binary target. In the case of a character binary target (or a binary target with a user-defined format), a temporary numeric variable with values of 0 or 1 is created. This is then substituted for the target variable.
- **Decision Tree Selection** — Trains a decision tree predictive model. The residual sum of squares variable importance is calculated for each predictor variable. The relative variable importance threshold that you specify is used to select the most useful predictor variables.
- **Forest Selection** — Trains a forest predictive model by fitting multiple decision trees. The residual sum of squares variable importance is calculated for each predictor variable, averaged across all the trees. The relative variable importance threshold that you specify is used to select the most useful predictor variables.
- **Gradient Boosting Selection** — Trains a gradient boosting predictive model by fitting a set of additive decision trees. The residual sum of squares variable importance is calculated for each predictor variable, averaged across all the trees. The relative variable importance threshold that you specify is used to select the most useful predictor variables.

When you are performing multiple methods, the results from the individual methods are combined to generate the final selection result. Each selection method gets a vote on whether a variable is selected. The **Combination criterion** property lets you choose the voting level that is used to select a variable. The least restrictive option selects a variable if at least one method votes for that variable. The most restrictive option requires that all methods vote for that variable. Any variable that is not selected in the final outcome is rejected, and subsequent nodes in the pipeline do not use that variable.

You can pre-screen the input variables prior to running the chosen variable selection methods. When pre-screening, if a variable exceeds the maximum number of class levels threshold or a variable exceeds the maximum missing percentage threshold, that variable is rejected and not processed by the subsequent variable selection methods. The **Data** advisor also accomplishes variable pre-screening when the project is created, so this option can be used to increase the level of pre-screening beyond what is done at the project level.

Variable Selection Properties

General Properties

- **Pre-screen Input Variables** — Specifies whether to pre-screen input variables. If selected, this option checks the percentage of missing values for all variables and the number of levels for the class variables. It rejects a variable if it exceeds the **Maximum level** or **Maximum missing percent** properties that are set below. By default, **Pre-screen Input Variables** is deselected.

- **Maximum level** — Specifies the maximum number of levels for class variables. Possible values range from 1 to 100. The default value is 50.
- **Maximum missing percent** — Specifies the maximum percent for missing values. Possible values range from 1 to 100. The default value is 50.
- **Combination criterion** — Specifies the criterion used in keeping an input when you are using multiple variable selection methods. Here are the possible values:
 - **Selected by a majority**
 - **Selected by a tie or majority**
 - **Selected by all**
 - **Selected by at least 1**

The default value is **Selected by at least 1**.

Unsupervised Selection

Unsupervised Selection — Specifies whether to perform unsupervised variable selection. By default, **Unsupervised Selection** is deselected. If selected, **Unsupervised selection** has the following options:

- **Maximum steps** — Specifies the maximum number of selection steps that are performed. The default value is 200.
- **Maximum variables** — Specifies the maximum number of selected variables. The default value is 200.
- **Correlation statistics** — Specifies Pearson correlation statistics. Possible values are **Correlation**, **Covariance**, and **Sum of squares and crossproducts**. The default value is **Correlation**.
- **Cumulative variance cutoff** — Specifies the cutoff value for the cumulative variance, which is explained by selected variables. The selection process stops when this fraction of the total variance can be explained by the selected variables. The default value is 1.
- **Incremental variance cutoff** — Specifies the cutoff value for incremental variance. The selection process stops when the increment of the explained variance is less than this fraction of the total variance. The default value is 0.001.
- **Selection process** — Specifies the process used for selecting variables when you are performing both unsupervised and supervised methods. Here are the possible values:
 - **Combine with supervised methods** — Specifies to combine unsupervised selection with supervised methods via the Combination criterion.
 - **Perform sequential selection** — Specifies to use unsupervised selection before one supervised method or combination of methods.

This property is ignored if no supervised methods are selected. The default value is **Perform sequential selection**.

Fast Supervised Selection

Fast Supervised Selection — Specifies whether to perform fast supervised selection. By default, **Fast Supervised Selection** is selected. If selected, **Fast Supervised Selection** has the following options:

- **Stop criterion** — Specifies the criterion for stopping the selection process. Here are the possible values:
 - **(none)** — Specifies that no stop criterion is used and that the selection process continues until all effects are in the model, or that a limit, described by one of the options below, is achieved.
 - **AIC** — Specifies Akaike's Information Criterion. Smaller values indicate better models, and AIC values can become negative. AIC is based on the Kullback-Leibler information measure of discrepancy between the true distribution of the response variable and the distribution specified by the model.
 - **AICC** — Specifies Corrected Akaike's Information Criterion. This version of AIC adjusts the value to account for sample size. The result is that extra effects penalize AICC more than AIC. As the sample size increases, AICC and AIC converge.
 - **BIC** — Specifies the Bayesian Information Criterion (BIC), also known as Schwarz's Bayesian Criterion (SBC). BIC is an increasing function of the model's residual sum of squares and the number of effects. Unexplained variations in the response variable and the number of effects increase the value of the BIC. As a result, a lower BIC implies either fewer explanatory variables, better fit, or both. BIC penalizes free parameters more strongly than AIC.

The default value is **BIC**.

- **Maximum steps** — Specifies the maximum number of selection steps that are performed. The default value is 200.
- **Maximum variables** — Specifies the maximum number of selected variables. The default value is 200.
- **Correlation statistics** — Specifies Pearson correlation statistics. Possible values are **Correlation**, **Covariance**, and **Sum of squares and crossproducts**. The default value is **Correlation**.
- **Cumulative variance cutoff** — Specifies the cutoff value for the cumulative variance, which is explained by selected variables. The selection process stops when this fraction of the total variance can be explained by the selected variables. The default value is 1.
- **Incremental variance cutoff** — Specifies the cutoff value for the incremental variance. The selection process stops when the minimum increment of the explained variance is less than the fraction of the total variance. The default value is 0.001.

Linear Regression Selection

Linear Regression Selection — Specifies whether to perform linear regression selection. By default, **Linear Regression Selection** is deselected. If selected, **Linear Regression Selection** has the following options:

- **Selection method** — Specifies a model selection method. Here are the possible values:
 - **Adaptive LASSO** — Specifies that training uses adaptive weights when applying L1 regularization to the regression methods coefficients. By default, ordinary least squares estimates of the model parameters are used as adaptive weights.
 - **Backward** — Specifies that training is done by starting with all candidate effects in the model and removing effects until the Stay significance level or the stop criterion is met.
 - **Forward** — Specifies that training is done by starting with no candidate effects in the model and adding effects until the Entry significance level or stop criterion is met.
 - **Forward-swap** — Specifies that training is done by determining whether removing one effect and replacing it with the other improves the selection criterion.
 - **LASSO** — Specifies that training is done using the group LASSO method— that is, adding and removing effects by a sequence of LASSO steps.
 - **Stepwise** — Specifies that training is done as in the forward model but might remove effects already in the model.

The default value is **Stepwise**.

- **Effect-selection criterion** — Specifies the criterion that the procedure uses to determine the order in which effects enter or leave at each step of the selection method. Here are the possible values:
 - **Adjusted R-square** — Specifies the Adjusted R-square value. The Adjusted R-Square value attempts to account for the addition of more effect variables. Values can range from 0 to 1. Values closer to 1 are preferred.
 - **AIC** — Specifies Akaike's Information Criterion. Smaller values indicate better models, and AIC values can become negative. AIC is based on the Kullback-Leibler information measure of discrepancy between the true distribution of the response variable and the distribution that is specified by the model.
 - **AICC** — Specifies Corrected Akaike's Information Criterion. This version of AIC adjusts the value to account for sample size. The result is that extra effects penalize AICC more than AIC. As the sample size increases, AICC and AIC converge.
 - **Mallows' Cp** — Specifies Mallows' Cp value. Smaller values indicate better models, and Cp values can become negative.
 - **R-square** — Specifies the R-squared value. R-squared value is an indicator of how well the model fits the data. R-squared values can range from 0 to 1. Values closer to 1 are preferred.

- **SBC (BIC)** — Specifies Schwarz’s Bayesian Criterion (SBC), also known as the Bayesian Information Criterion (BIC). SBC is an increasing function of the model’s residual sum of squares and the number of effects. Unexplained variations in the response variable and the number of effects increase the value of the SBC. As a result, a lower SBC implies either fewer explanatory variables, better fit, or both. SBC penalizes free parameters more strongly than AIC.
- **Significance level** — Specifies the standard statistical significance level criterion.

If the **Selection method** above is **LASSO** or **Adaptive LASSO**, this option is unavailable. The default value is **SBC (BIC)**.

- **Selection-process stopping criterion** — Specifies the criterion to stop the selection process. Here are the possible values:
 - **(none)** — Specifies that the selection process continues until all the effects are in the model, or if a limit based on one of the options below is achieved.
 - **Adjusted R-square** — Specifies the Adjusted R-square value. The Adjusted R-Square value attempts to account for the addition of more effect variables. Values can range from 0 to 1. Values closer to 1 are preferred.
 - **AIC** — Specifies Akaike’s Information Criterion. Smaller values indicate better models, and AIC values can become negative. AIC is based on the Kullback-Leibler information measure of discrepancy between the true distribution of the response variable and the distribution specified by the model.
 - **AICC** — Specifies Corrected Akaike’s Information Criterion. This version of AIC adjusts the value to account for sample size. The result is that extra effects penalize AICC more than AIC. As the sample size increases, AICC and AIC converge.
 - **Mallows’ Cp** — Specifies Mallows’ Cp value. Smaller values indicate better models, and Cp values can become negative.
 - **Predicted RSS** — Specifies that the selection process continues until the predicted residual sum of squares (RSS) starts to increase. This option is not valid with **LASSO** selection.
 - **SBC (BIC)** — Specifies Schwarz’s Bayesian Criterion (SBC), also known as the Bayesian Information Criterion (BIC). SBC is an increasing function of the model’s residual sum of squares and the number of effects. Unexplained variations in the response variable and the number of effects increase the value of the SBC. As a result, a lower SBC implies either fewer explanatory variables, better fit, or both. SBC penalizes free parameters more strongly than AIC.
 - **Significance level** — Specifies the standard statistical significance level criterion.
 - **Validation ASE** — Average square error (ASE) of the model is computed using the validation data, and selection stops at the step where the ASE starts to increase. This method requires partitioned data.

The default value is **SBC (BIC)**.

- **Model-selection criterion** — Specifies the criterion to choose the model from the list of models (at each step of the selection process) that yields the best value of the specified criterion. If the optimal value of the specified criterion

occurs for models at more than one step, then the model that has the smallest number of parameters is chosen. Here are the possible values:

- **Adjusted R-square** — Specifies the Adjusted R-square value. The Adjusted R-Square value attempts to account for the addition of more effect variables. Values can range from 0 to 1. The model that has the maximal value is chosen.
- **AIC** — Specifies Akaike's Information Criterion. Smaller values indicate better models, and AIC values can become negative. AIC is based on the Kullback-Leibler information measure of discrepancy between the true distribution of the response variable and the distribution specified by the model. The model that has the minimal AIC value is chosen.
- **AICC** — Specifies Corrected Akaike's Information Criterion. This version of AIC adjusts the value to account for sample size. The result is that extra effects penalize AICC more than AIC. As the sample size increases, AICC and AIC converge.
- **Mallows' Cp** — Specifies Mallows' Cp value. Smaller values indicate better models, and the model with the minimal Cp is chosen.
- **Predicted RSS** — Specifies that the model that has the minimal predicted residual sum of squares (RSS) is chosen. This option is not valid with **LASSO** selection.
- **SBC (BIC)** — Specifies Schwarz's Bayesian Criterion (SBC), also known as the Bayesian Information Criterion (BIC). SBC is an increasing function of the model's residual sum of squares and the number of effects. Unexplained variations in the response variable and the number of effects increase the value of the SBC. As a result, a lower SBC implies either fewer explanatory variables, better fit, or both. SBC penalizes free parameters more strongly than AIC. The model that has the minimal SBC value is chosen.
- **Validation ASE** — Average square error (ASE) of the model is computed using the validation data, and the model that has the minimal ASE is chosen. This method requires partitioned data.

If the **Selection method** is **Forward-swap**, this option is not available. The default value is **SBC (BIC)**.

- **Entry significance level** — Specifies the significance level for adding variables in forward and stepwise directions. The default value is 0.05.
- **Stay significance level** — Specifies the significance level for removing variables in backward and stepwise selections. The default value is 0.05.
- **Maximum number of effects** — Specifies the maximum number of effects in any model considered during the selection process. This option is ignored with the backward regression. If a model at some step of the selection process contains the specified maximum number of effects, then no candidate effects for addition are considered. Use the default value (0) to ignore this option.
- **Minimum number of effects** — Specifies the minimum number of effects in any model considered during the backward selection process. The selection process terminates if a model at some step of the selection process contains the specified minimum number of effects. Use the default value (0) to ignore this option.
- **Maximum number of steps** — Specifies the maximum number of selection steps that are performed. Use the default value (0) to ignore this option.

- **Suppress intercept** — Specifies whether to suppress the intercept. By default, this is deselected.

Decision Tree Selection

Decision Tree Selection — Specifies whether to perform decision tree selection. By default, **Decision Tree Selection** is deselected. If selected, **Decision Tree Selection** has the following options:

- **Splitting Options** — Specifies the splitting criterion configurations:
 - **Class target criterion** — Specifies the splitting criterion to use for determining best splits on inputs given a class target. Here are the possible values:
 - **CHAID**
 - **Chi-square**
 - **Entropy**
 - **Gini**
 - **Information gain ratio**

The default value is **Entropy**.
 - **Interval target criterion** — Specifies the splitting criterion to use for determining the best splits on inputs given an interval target. Here are the possible values:
 - **CHAID**
 - **F test**
 - **Variance**

The default value is **Variance**.
 - **Significance level** — Specifies the significance level for splitting criteria CHAID, Chi-Square, and F Test. The default value is 0.2.
 - **Bonferroni** — Specifies whether a Bonferroni adjustment is applied to the top p-values for splitting criteria **CHAID**, **Chi-Square**, and **F Test**. By default, this is deselected.
 - **Maximum number of branches** — Specifies the maximum number of branches that a splitting rule produces. The default value of 2 results in binary splits. Possible values range from 2 to 6.
 - **Maximum depth** — Specifies the maximum number of generations in nodes. The original node, generation 0, is called the root node. The children of the root node are the first generation. Possible values range from 1 to 50. The default value is 10.
 - **Minimum leaf size** — Specifies the smallest number of training observations that a leaf can have. The default value is 5.
 - **Missing values** — Specifies how a splitting rule handles an observation with missing values. Here are the possible values:
 - **Largest branch**

- **Most correlated branch**
- **Separate branch**
- **Use in search**

Missing values are either used as a value during the split search or assigned to a node based on this selection. The default value is **Use in search**.

- **Minimum missing use in search** — Specifies the minimum number of missing values for a splitting variable to have for missing values to be treated as a separate level. This option is enabled when **Use in search** is specified for the **Missing values** property. The default value is 1.
- **Number of interval bins** — Specifies the number of bins used for interval inputs. Bin size is $(\text{maximum value} - \text{minimum value}) / (\text{Number of interval bins})$. The default value is 50.
- **Interval bin method** — Specifies the method used to bin the interval input variables. The default value is **Quantile**.
- **Surrogate rules** — Specifies the number of surrogate rules. The default value is 0.
- **Use input once** — If selected, specifies that no splitting rule will be based on an input variable used in a splitting rule of an ancestor node. This option is enabled when surrogate rules are not in effect. By default, this is deselected.
- **Pruning options** — Specifies the pruning configurations:
 - **Subtree method** — Specifies how to construct the subtree in terms of subtree pruning methods. Possible values include the following:
 - **C4.5** — The pruning is done with a C4.5 algorithm.
 - **Cost complexity** — The subtree with a minimum leaf-penalized ASE is chosen.
 - **Reduced error** — The smallest subtree with the best assessment value is chosen.

C4.5 is available only for class targets. For **Reduced error**, the assessment measure for class targets is Misclassification Rate. For interval targets, the assessment measure is ASE. The default value is **Cost complexity**.
 - **Selection method** — Specifies how to construct the subtree in terms of selection methods. Possible values include the following:
 - **Automatic** — Specifies the appropriate subtree for the specified subtree pruning method.
 - **Largest** — Specifies the full tree.
 - **N** — Specifies the largest subtree with at most *N* leaves.

The default value is **Automatic**.
 - **Confidence** — Specifies the binomial distribution confidence level to use to determine the error rates of merged and split nodes. The default value is 0.25. This option is available only when **C4.5** is the pruning method.
 - **Number of leaves** — Specifies the number of leaves that are used in creating the subtree when the subtree selection method is set to **N**. The default value is 1.
 - **Cross validation folds** — Specifies the number of cross validation folds to use for cost-complexity pruning when there is no validation data. This option

is enabled when the pruning selection method is **Automatic**, and the **Create Validation from Training** option is deselected. Possible values range from 2 to 20. The default value is 10.

- **1–SE rule** — Specifies whether to perform the 1–SE rule when performing cross validated cost-complexity pruning. This option is enabled when the pruning selection method is **Automatic**, and the **Create Validation from Training** option is deselected. By default, this is deselected.
- **Seed** — Specifies the random number seed used for cross validation cost-complexity. This option is enabled when the pruning selection method is **Automatic**, and the **Create Validation from Training** option is deselected. The default value is 12345.
- **Relative importance cutoff** — Specifies the relative importance cutoff used to determine whether an input is rejected. Any input with a relative importance below this cutoff is rejected. The default value is 0.25.

Forest Selection

Forest Selection — Specifies whether to perform forest selection. By default, **Forest Selection** is deselected. If selected, **Forest Selection** has the following options:

- **Tree-splitting Options** — Specifies the tree-splitting configurations:
 - **Class target criterion** — Specifies the splitting criterion to use for determining the best splits on inputs given a class target. Here are the possible values:
 - **CHAID**
 - **Chi-square**
 - **Entropy**
 - **Gini**
 - **Information gain ratio**

The default value is **Information gain ratio**.
 - **Interval target criterion** — Specifies the splitting criterion to use for determining the best splits on inputs given an interval target. Here are the possible values:
 - **CHAID**
 - **F test**
 - **Variance**

The default value is **Variance**.
 - **Maximum number of branches** — Specifies the maximum number of branches that a splitting rule produces. The default value of 2 results in binary splits. Possible values range from 2 to 5.
 - **Maximum depth** — Specifies the maximum depth for each generated tree within the forest. Possible values range from 1 to 50. The default value is 20.

- **Minimum leaf size** — Specifies the smallest number of training observations that a new branch can have, expressed as a count of the available observations. Possible values range from 1 to 64. The default value is 5.
- **Minimum missing use in search** — Specifies a threshold for using missing values in the split search. The default value is 1.
- **Number of interval bins** — Specifies the number of bins used for interval inputs. Bin size is defined as $(\text{maximum value} - \text{minimum value}) / (\text{Number of interval bins})$. The default value is 50.
- **Interval bin method** — Specifies the method used to bin the interval input variables. The default value is **Quantile**.
- **In-bag sample proportion** — Specifies the proportion of training observations with which to train a tree (the “in-bag” proportion). The default value is 0.6.
- **Use default number of inputs per split** — Specifies whether to use the default number of inputs per split, which is the square root of the number of inputs. By default, this is selected.
- **Number of inputs per split** — Specifies the number of input variables randomly sampled to use per split. This option is enabled when option **Use default number of inputs per split** is deselected. The default value is 100.
- **Number of inputs to subset with Loh’s method** — Specifies the number of input variables to further sample using LOH (where LOH is the method developed by Loh). When set to 0, no further sampling of inputs is performed. The default value is 0.
- **Seed** — Specifies the seed for generating random numbers. This is used to select training observations for each tree and to select candidate variables in each node to split on. The default value is 12345.
- **Number of trees** — Specifies the number of trees in the forest. The default value is 100.
- **Relative importance cutoff** — Specifies the relative importance cutoff used to determine whether an input is rejected. Any input with a relative importance below this cutoff is rejected. The default value is 0.25.

Gradient Boosting Selection

Gradient Boosting Selection — Specifies whether to perform gradient boosting. By default, **Gradient Boosting Selection** is deselected. If selected, **Gradient Boosting Selection** has the following options:

- **Basic Options**
 - **Number of trees** — Specifies the number of terms in the boosting series. For interval and binary targets, the number of iterations equals the trees. For a nominal target, a separate tree is created for each target category in each iteration. The default value is 100.
 - **Learning rate** — Specifies how much to reduce the prediction of each tree. The default value is 0.1.
 - **Subsample rate** — Specifies the proportion of training observations with which to train a tree. A different training sample is taken in each iteration.

Trees trained in the same iteration have the same training data. The default value is 0.5.

- **L1 regularization** — Specifies the L1 regularization parameter. The default value is 0.
- **L2 regularization** — Specifies the L2 regularization parameter. The default value is 1.
- **Interval target distribution** — Specifies the distribution of the objective function for an interval target. Possible values are **Normal**, **Poisson**, and **Tweedie**. For a binary or nominal target, the distribution is binary or multinomial, respectively.
- **Tweedie power parameter** — Specifies the power parameter to use when **Tweedie** is selected as the **Interval target distribution**.
- **Tree-splitting Options** — Specifies the tree-splitting configurations:
 - **Maximum number of branches** — Specifies the maximum number of branches to consider at each node split in the tree. Possible values range from 2 to 5. The default value is 2.
 - **Maximum depth** — Specifies the maximum number of generations of nodes. The original node, generation 0, is called the root node. The children of the root node are the first generation. Possible values range from 1 to 50. The default value is 4.
 - **Minimum leaf size** — Specifies the smallest number of training observations that a leaf can have. The default value is 5.
 - **Minimum missing use in search** — Specifies a threshold for using missing values in the split search. The default value is 1.
 - **Number of interval bins** — Specifies the number of bins in which to bin the interval input variables. Bin size is defined as $(\text{maximum value} - \text{minimum value}) / (\text{Number of interval bins})$. The default value is 50.
 - **Interval bin method** — Specifies the method used to bin the interval input variables. The default value is **Quantile**.
 - **Use the default number of inputs per split** — Specifies whether to use the default number of inputs per split, which is all of the inputs. By default, this option is selected.
 - **Number of inputs per tree** — Specifies the number of input variables randomly sampled to use per split. This option is used only if **Use the default number of inputs per split** is deselected. The default value is 100.
 - **Seed** — Specifies the seed for generating random numbers. The subsample rate property uses this value to select a training sample at each iteration. The default value is 12345.
- **Perform Early Stopping** — Specifies whether to perform early stopping—that is, stopping training when the model starts to overfit. Early stopping cannot be used if there is no validation partition. By default, this option is selected. The following options are available:
 - **Class target metric** — Specifies the error metric to use for a class target. Possible values are **Log loss** and **Misclassification rate**. Average squared error is used for an interval target.
 - **Early stopping method** — Specifies the method to use for early stopping. The **Stagnation** method stops when the relative change in the validation error in consecutive iterations satisfies criteria based on the **Stagnation**,

Tolerance, and **Start from minimum error** properties. The **Threshold** method stops when the validation error exceeds a threshold based on the **Threshold** and **Threshold iterations** properties.

- **Stagnation** — Specifies the number of consecutive iterations (N) to consider when using the **Stagnation** early stopping method. The default value is 5.
- **Tolerance** — Specifies the threshold for the relative change in validation error when using the **Stagnation** early stopping method. The default value is 0.
- **Start from minimum error** — Specifies whether to count iterations starting from the iteration that has the smallest validation error when using the **Stagnation** early stopping method.
- **Threshold** — Specifies the threshold value when using the **Threshold** early stopping method. Training is stopped when the validation error equals or exceeds this value.
- **Threshold iterations** — Specifies the minimum number of training iterations to run before the validation error is compared to the specified **Threshold**.
- **Relative importance cutoff** — Specifies the relative importance cutoff used to determine whether an input is rejected. Any input with a relative importance below this cutoff is rejected. The default value is 0.25.

Create Validation from Training

Create Validation Sample from Training Data — Specifies whether a validation sample should be created from the incoming training data. This is recommended even if the data has already been partitioned so that only the training partition is used for variable selection, and the validation partition can be used for modeling. By default, this is selected. When selected, it has the following options:

- **Validation proportion** — Specifies the probability of a given record being selected for the validation set. The default value is 0.3.
- **Partition seed** — Specifies the partition seed to generate the sample that is used for validation. The default value is 12345.


Variable Selection Results

After running the node, open the Results window by right-clicking the node and selecting **Results**.

.....

Note: The results that are available for this node vary based on the variable selection method.

.....

You can download the data that is used to build the charts and tables in the Results window by clicking . A CSV file with the chart or table data will immediately begin downloading.

In the **Output Data** tab, you can view, save, and explore the data that is generated by the node. For more information, see [Explore and Visualize](#).

Summary

- **Variable Selection** — Displays a table of variable metadata information about the data set submitted to the node. The table includes columns for variable name, label, level, role, and reason for rejection.
- **Variable Selection Combination Summary** — Displays a table of input variables, whether they were selected or rejected by each method, and the final output role from the **Combination criterion** option. This result table is available when more than one model is selected, including the unsupervised model if combined with supervised models.
- **Cumulative Variance Explained** — Displays a bar chart giving the proportion of variance explained as each variable is considered. The parameters are ordered by decreasing incremental variance. This result table is available for both Unsupervised Selection and Fast Selection.
- **Variance Explained Values** — Displays the variance explained values at each iteration, as another variable or class variable level (parameter) is considered. This result table is available for both Unsupervised Selection and Fast Selection.
- **Parameter Estimates — Linear Regression Selection** — Displays the statistical values obtained as a result of linear regression. This table includes effect, parameter, t-value, estimate, the p-value, and the standard error.
- **Tree Variable Importance** — Displays a bar chart giving the relative importance of variables obtained via the decision tree model. This result table is available when Decision Tree is the only model selected.
- **Tree Variable Importance Values** — Displays importance statistics of variables obtained via the decision tree method. This table includes variable name, train importance, importance standard deviation, relative importance, and count.
- **Forest Variable Importance** — Displays a bar chart giving the relative importance of the variables obtained via the forest model. This result table is available when Forest is the only model selected.
- **Forest Variable Importance Values** — Displays importance statistics of variables obtained via the forest method. This table includes variable name, train importance, importance standard deviation, and relative importance.
- **Gradient Boosting Variable Importance** — Displays a bar chart giving the relative importance of the variables obtained via the gradient boosting model. This is available when Gradient Boosting is the only model selected.
- **Gradient Boosting Variable Importance Values** — Displays importance statistics of variables obtained via the gradient boosting model. This table includes variable name, train importance, importance standard deviation, and relative importance.
- **Properties** — Specifies the various properties that you selected before running the node.
- **Output** — Displays the SAS output of the variable selection run. The output given varies based on the chosen variable selection method.

PART 2

Risk Modeling

<i>Chapter 34</i>	
<i>Interactive Grouping</i>	285
<i>Chapter 35</i>	
<i>Reject Inference</i>	295
<i>Chapter 36</i>	
<i>Scorecard</i>	299

Interactive Grouping

<i>Overview of Interactive Grouping</i>	285
<i>Interactive Grouping Properties</i>	286
Predefined Groupings	286
Interval Variable Binning Options	286
Special Code Options	287
Grouping Options	287
Variable Selection Options	289
Output Options	290
<i>Interactive Grouping Results</i>	290
<i>Using the Interactive Grouping Editor</i>	291
<i>Using a Special Codes Data Set</i>	292

Overview of Interactive Grouping

Note: Risk Modeling Add-on for SAS Visual Data Mining and Machine Learning is not included with the base version of SAS Visual Data Mining and Machine Learning. If your site has not licensed Risk Modeling Add-on for SAS Visual Data Mining and Machine Learning, the **Interactive Grouping** node does not appear in your SAS Visual Data Mining and Machine Learning software.

The **Interactive Grouping** node is a Data Mining Preprocessing node that performs initial variable screening and variable grouping. The variable groups can be used as input variables for predictive modeling. Input variables are referred to as characteristics and each possible value of a characteristic is referred to as an attribute.

Grouping offers the following advantages:

- It offers an easier way to deal with rare classes and outliers with interval variables.

- It makes it easy to understand relationships and therefore gain far more knowledge of the portfolio. A chart that displays the relationship between attributes of a characteristic and performance is a much more powerful tool than a simple variable strength statistic. It enables you to explain the nature of this relationship, in addition to the strength of the relationship.
- Nonlinear dependencies can be modeled with linear models.
- It enables user control over the development process. By shaping the groups, you shape the final composition of the scorecard.
- The process of grouping characteristics enables you to develop insights into the behavior of risk predictors and to increase knowledge of the portfolio, which can help in developing better strategies for portfolio management.

The **Interactive Grouping** node first performs binning on the interval characteristics. You can specify either quantile or bucket binning. After the characteristics have been pre-binned, a decision tree model is fitted for each characteristic to create the variable groups.

The weight of evidence (WOE) measure is calculated to assess the predictive power of each attribute. The information value and the Gini statistic are calculated to assess the predictive power of the characteristic. WOE variables and grouping variables are generated, and these can be used as input variables for predictive models.

The **Interactive Grouping** node provides an interactive grouping editor that enables you to manually adjust the variable groups that are created by the node.

Interactive Grouping Properties

Predefined Groupings

- **Use frozen groupings** — Specifies whether frozen groupings should be used or if new groupings should be created during training. Variable groupings are automatically saved each time the **Interactive Grouping** node is run. You can also save groupings in the Interactive Grouping editor. If **Use frozen groupings** is selected, then the groupings that are saved are used. Automatic grouping is performed only on variables without saved groupings. If **Use frozen groupings** is deselected, then previously defined groups are ignored, and new groupings are created based on the current values of the node properties.

Interval Variable Binning Options

- **Apply level rule** — Specifies whether the number of unique levels of an interval input variable should be taken into consideration when determining how the variable should be treated. If selected, the number of unique levels are compared to the value of the **Number of bins** property. If the number of levels is

less than or equal to the number of bins specified, the input variable is treated as an ordinal input variable. If the number of levels exceeds the number of bins specified, the input variable is treated as an interval input variable. If this option is deselected, all interval input variables are treated as interval input variables, regardless of the number of unique levels

- **Binning method** — Specifies the method that is used to bin the interval input variables. Select **Bucket** to divide input variables into evenly spaced intervals based on the difference between maximum and minimum values. Select **Quantile** to divide input variables into approximately equal sized groups. The default value is **Quantile**.
- **Number of bins** — Specifies the number of bins to use for interval input variables. Bin size is $(\text{maximum value} - \text{minimum value}) / (\text{Number of Bins})$. The default value is 20.

Special Code Options

- **Use special codes** — Specifies whether a special codes mapping data set exists for training.
- **Special codes data set** — Specifies the name of the data set that is used to map values to the corresponding special missing value. See [Using a Special Codes Data Set on page 292](#) for more information.

.....

Note:

You cannot use variations of the DOLLAR, FRACT, and PERCENTN formats in your special codes. Those special codes are not supported.

.....

Grouping Options

- **Interval grouping method** — Specifies the method by which interval input variables are grouped. Here are the possible values:
 - **Constrained optimal** — Creates groups based on pre-defined constraints.
 - **Optimal criterion** — Creates groups based on the **Class target criterion** property.
 - **Quantile** — Creates groups that are approximately equal size.

The default value is **Optimal criterion**.

- **Ordinal grouping method** — Specifies the method by which ordinal input variables are grouped. Here are the possible values:
 - **Constrained optimal** — Creates groups based on pre-defined constraints.
 - **Optimal criterion** — Creates groups based on the **Class target criterion** property.
 - **Quantile** — Creates groups that are approximately equal size.

The default value is **Optimal criterion**.

■ Tree-Based Grouping Options

- **Class target criterion** — Specifies the criterion for evaluating candidate splitting rules. **Entropy** uses the reduction in entropy measure. **Chi-square** uses the p-value of the Pearson chi-square statistic for the target. The default value is **Entropy**.

Note: The **Chi-square** criterion is currently not supported. The grouping is always run with the **Entropy** criterion.

- **Missing values** — Specifies how splitting rules handle observations that contain missing values for a variable. Here are the possible values:

- **Largest branch**
- **Separate branch**

The default value is **Largest branch**.

■ Constrained Optimal Options

- **Apply WOE monotonicity** — Specifies whether a monotonic distribution of the Weight of Evidence (WOE) should be imposed.
- **Apply minimum number of groups** — Specifies whether the **Minimum number of groups** constraint should be applied during the generation of groups.
- **Minimum number of groups** — Specifies the minimum number of groups that should be created.

Note: The value that is specified for the **Maximum number of groups** property takes precedence over the value that is specified for the **Minimum number of groups** property if conflicting values are specified.

- **Apply minimum non-event** — Specifies whether the **Minimum non-event count** constraint should be applied during the generation of groups.
- **Minimum non-event count** — Specifies the minimum number of nonevents in each group that is created.
- **Apply minimum event** — Specifies whether the **Minimum event count** constraint should be applied during the generation of groups.
- **Minimum event count** — Specifies the minimum number of events in each group that is created.
- **Apply maximum total count** — Specifies whether the **Maximum total count** constraint should be applied during the generation of groups.
- **Maximum total count** — Specifies the maximum total number of observations in each group that is created.
- **Apply minimum WOE difference** — Specifies whether the **Minimum WOE difference** constraint should be applied during the generation of groups.
- **Minimum WOE difference** — Specifies the minimum difference in the WOE between the created groups.
- **Manage Advanced Options** — Invokes the Advance Constrained Optimal Options editor. You can customize the constraints for individual input variables.

Note: The Advance Constrained Optimal Options window displays all input variables. To edit the constraints of a variable, you must specify **Constrained optimal** as the **Interval grouping method** for interval input variables, and **Ordinal grouping method** for ordinal input variables.

- **Maximum number of groups** — Specifies the maximum number of groups to be created.

Note: The value that is specified for the **Maximum number of groups** property takes precedence over the value that is specified for the **Minimum number of groups** property if conflicting values are specified.

- **Bounds display precision** — Specifies the numerical precision to which interval variable bounds are displayed.
- **Apply restrictions** — Specifies whether to apply a restriction of minimum group size when automatic grouping is performed.
- **Type** — Specifies whether to base the minimum distribution restriction on a percentage or count.
- **Percent** — Specifies the percent of the distribution to be used when **Apply restrictions** is selected and the specified **Type** is percent.
- **Count** — Specifies the count to be used when **Apply restrictions** is selected and the specified **Type** is count.
- **Adjust WOE** — Specifies whether to activate the WOE adjustment for a group in which all observations have the same target value. If selected, the adjustment factor that you specify is added to the number of events and the number of nonevents in order to calculate WOE values.
- **Adjustment factor** — Specifies the adjustment factor to use when calculating WOE values if **Adjust WOE** is selected.
- **Group level** — Specifies the level that is assigned to the exported group variables. Possible values are **Ordinal** and **Nominal**.

Variable Selection Options

- **Variable selection method** — Specifies the statistic to use for variable selection. If the value of the statistic that is specified is less than the corresponding cutoff value, the exported variable is assigned the role **Rejected**. Here are the possible values:

- (none)**
- Gini statistic**
- Information value**

The default value is **Information value**


- **Gini cutoff** — Specifies the cutoff value to be used when **Variable selection method** is set to **Gini statistic**. The default value is 20.
- **Information value cutoff** — Specifies the cutoff value to be used when **Variable selection method** is set to **Information value**. The default value is 0.1.

Output Options

- **Number of variables** — Specifies the maximum number of variables to plot in the Event Rate Plot. Only the variables that exceed the active variable selection method cutoff are plotted.



Interactive Grouping Results

After running the node, open the Results window by right-clicking the node and selecting **Results**.

You can download the data that is used to build the charts and tables in the Results window by clicking . A CSV file with the chart or table data will immediately begin downloading.

In the **Output Data** tab, you can view, save, and explore the data that is generated by the node. For more information, see [Explore and Visualize](#).

Several charts have detailed descriptions available that explain how the charts are generated. The descriptions also provide information about how to interpret the charts that is custom to your particular data and models. To access these detailed descriptions:

- Expand an individual chart by clicking . The detailed description is in the right panel.
- Select  on an individual chart. A window appears with the detailed description.

.....

Note: The detailed descriptions are available only in the English language.

.....

Summary

- **Output Variables** — Displays all input variables and their Gini statistic, information value, variable level, and calculated role. The properties that are specified under **Variable Selection Options** determine which variables are rejected by the node.
- **Statistics Plot** — Displays the statistic that is specified in the **Variable selection method** property for all input variables.
- **Event Rate Plot** — Displays the event rate for each group of attributes of a variable.
- **Statistics Table** — Displays several statistics such as the event count, nonevent count, and WOE for each variable group.
- **Fine Detail Table** — Displays several statistics such as the event count, nonevent count, and WOE for each variable bin.
- **Node Statistics** — Displays statistics of the node run. Statistics include the time at which the node run finished executing, the setup time (in seconds), the run





duration (in seconds), whether the run was initiated from the pipeline, and the user who ran the node.

- **Node Score Code** — Displays the SAS score code that was created by the node. The SAS score code can be used outside the Model Studio environment to score new data.
- **Properties** — Specifies the various properties that you selected before running the node.
- **Output** — Displays the SAS output of the interactive grouping run.

Using the Interactive Grouping Editor

To launch the Interactive Grouping editor, you must first run the **Interactive Grouping** node. Then right-click the node, and select **Manage Group Bins**. The **Select variable** drop-down list enables you to select the variable that is displayed in the editor. You can click **←** to select the previous variable in the list and **→** to select the next variable in the list.

The following charts and tables are displayed in the editor:


- **Interactive Binning** chart — Displays each variable bin's event count and WOE. Click  to maximize the chart. When the chart is maximized, you can change options such as the event count and event metric, and you can select whether to display the WOE in the chart. You can also adjust these options by right-clicking in the chart and selecting **Options**.
- **Grouping** chart — Displays each variable group's event count and WOE.
- **Variable Bins** table — Displays the group, cutoff, event count, nonevent count, total count, event rate, and weight of evidence (WOE) for each variable bin. You can manually adjust the variable bins for interval variables in the following ways:
 - Select  to split the selected bin. In the Split bin window, enter a **New Cutoff Value** for the bin.
 - Select  to merge two or more selected bins.
 - Select  to change the group of one or more selected bins.

Each time that you adjust a variable bin, the Interactive Binning chart, Grouping chart, Gini statistic, and information value are automatically adjusted as well. The old and current values of the Gini statistic and information value are displayed. This enables you to easily see whether your manual adjustments improve the predictive power of the variable.

If you select **Group** for the **Table level**, then the Variable Groups table displays the same details as the Variable Bins table, but at the group level. In the Variable Groups table, you can manually override the WOE. The Grouping chart displays the new WOE.

Note: If you manually override the WOE, you cannot press Enter or the down arrow on the keyboard after typing the new value in the last row of the table. This results in the new WOE not being saved. To exit the box and save the new WOE, either press

Tab, the right arrow, or the left arrow on the keyboard or click elsewhere in the editor.

Click  to open the Choose a Variable window. Information about each variable's level, role, original Gini statistic and information value, and current Gini statistic and information value is available. You can interactively adjust the variable role to either **Input** or **Rejected**.

Click **Reset** to reverse all interactive changes. The groupings revert to the last time that they were saved.

Using a Special Codes Data Set

The **Interactive Grouping** node can handle mixed-type data and special coded or holdout values. To use special codes, set the **Use Special Codes** property to **Yes** and specify the **Special Codes Data Set**.

Your special codes data set must be defined with the following columns:

- **REPLACEMENT** — A special missing value to which the corresponding holdout value is mapped. The variable type must be character.
- **VARIABLE** — The name of the input variable or variable group to be mapped. VARIABLE must have a length of 32. Here are the available variable groups:
 - _ALL_** — Applies the variable mapping to any input variable that contains the corresponding value.
 - _NUMERIC_** — Applies the variable mapping to any numeric input variable.
 - _CHARACTER_** — Applies the variable mapping to any character input variable.
 - _INTERVAL_** — Applies the variable mapping to any interval input variable.
 - _ORDINAL_** — Applies the variable mapping to any ordinal input variable.
 - _NOMINAL_** — Applies the variable mapping to any nominal input variable.
 - _BINARY_** — Applies the variable mapping to any binary input variable.

Variable mappings are applied in the following order:

- 1 **_ALL_**
- 2 **_NUMERIC_ and _CHARACTER_**
- 3 **_INTERVAL_, _ORDINAL_, _NOMINAL_, and _BINARY_**
- 4 **Individual variable mappings**

Variable mappings that appear later in the list supersede variable mappings that appear earlier in the list. For example, if a variable is both **_NUMERIC_** and **_INTERVAL_**, then the **_INTERVAL_** variable mapping supersedes the **_NUMERIC_** variable mapping.

- **VALUE** — The original holdout or special coded value. The variable type must be character and must contain the formatted value of the variable if a format is used. For example, suppose a numeric variable has integer values 1, 2, and 3

and a 4.2 format is applied. Then the values in this column should be 1.00, 2.00, and 3.00 in order to map to the unformatted values 1, 2, and 3.

Note: You cannot use variations of the DOLLAR, FRACT, and PERCENTN formats in your special codes. Those special codes are not supported.

For `_NUMERIC_` and `_INTERVAL_` variable mappings, the VALUE column must contain a number, although VALUE is still a character variable.

The **Interactive Grouping** node first processes all values that are not included in the special codes data set, creating a default set of bins based on property values. It then creates an additional bin for each unique replacement value that is specified in the special codes data set.

Reject Inference

<i>Overview of Reject Inference</i>	295
<i>Reject Inference Properties</i>	296
General Properties	296
Hard Cutoff	297
Parceling	297
<i>Reject Inference Results</i>	298

Overview of Reject Inference

Note: Risk Modeling Add-on for SAS Visual Data Mining and Machine Learning is not included with the base version of SAS Visual Data Mining and Machine Learning. If your site has not licensed Risk Modeling Add-on for SAS Visual Data Mining and Machine Learning, the **Reject Inference** node does not appear in your SAS Visual Data Mining and Machine Learning software.

The **Reject Inference** node is a Data Mining Preprocessing node. Use the **Reject Inference** node to remedy selection bias in your data. The sample data that is used to develop a credit scoring model is structurally different from the "through-the-door" population to which the credit scoring model is applied. The number of events and nonevents in the target variable that is created for the credit scoring model is based on the records of applicants who were all accepted for credit. However, the population to which the credit scoring model is applied includes applicants who would have been rejected under the scoring rules that were used to generate the initial model.

The reject inference approach uses the model that was trained using the accepted applications to score the rejected applications. The observations in the rejects data set are classified as *inferred nonevents* and *inferred events*. The inferred observations are then added to the accepts data set, which contains the actual nonevent and event records, to form an augmented data set. This augmented data set, which represents the "through-the-door" population, serves as the training data set for a second scorecard model.

The **Reject Inference** node provides two different methods that you can use to classify the observations in the rejects data set as either *inferred nonevents* or *inferred events*:

- Hard cutoff
- Parceling

The hard cutoff method uses a cutoff score to classify observations as nonevent or event. In this context, the score refers to the number of scorecard points that are calculated by the **Scorecard** node. Any score that is below the hard cutoff value is classified as an event.

The parceling method distributes the scored rejects into equal sized buckets that are defined by the **Score Range Method** that you specify. The scored rejects are then randomly classified as a nonevent or event within each bucket. By default, the rejects are classified as being an event or nonevent in proportion to what is observed in the accepts data within each score bucket. You can adjust this proportion with the **Event Rate Increase** property. The proportion of observations that are classified as events is determined by the observed event rate in the accepts data multiplied by the value of the **Event Rate Increase** property.

An **Interactive Grouping** node and a **Scorecard** node must precede a **Reject Inference** node.

Note: The variable types in the accepts data and rejects data must match. Otherwise, the node run fails.

If you specify your own partitioning variable in the accepts data, that variable must also be in the rejects data. Otherwise, the node run fails.

Reject Inference Properties

General Properties

- **Score data table name** — Specifies the score data to be added to the training data.
- **Inference method** — Specifies how to classify the rejects data set observations. Here are the possible values:
 - Hard cutoff**
 - Parceling**

The default value is **Hard cutoff**.

- **Rejection rate** — Specifies the probability of rejection in the population. The **Reject Inference** node uses the **Rejection rate** property to generate a weight that is used to adjust the number of sampled rejects to accurately reflect the number of rejects in the population. Possible values range from 0.0001 to 1. The default value is 0.3.

Note: The **Rejection rate** property has no effect unless your data table includes a variable with the role **Frequency**. Currently, Model Studio does not support frequency variables.

- **Event rate increase** — Specifies the adjustment value for the reject event rate within each score bucket. By default, the rejects are classified as being an event or nonevent in proportion to what is observed in the accepts data within each score bucket. When **Event rate increase** is specified, the observed event rate for the accepts is multiplied by the specified value to determine the event rate for the rejects data. Possible values range from 0 to 100. The default value is 1. The **Event rate increase** property is available only when **Parceling** is specified for the **Inference method**.

Hard Cutoff

- **Cutoff score** — Specifies the cutoff value of the scores to classify the rejects as good or bad. An observation with a score below the cutoff is assigned the status of bad (event). Otherwise, the observation is classified as good (nonevent). The default value is 200.

Parceling


- **Score range method** — Specifies how to define the range of scores for the rejects data to be bucketed. Here are the possible values:
 - **Accepts** — Distributes the rejected applications into equal-sized buckets based on the score range that is in the accepts data set.
 - **Manual** — Distributes the rejected applicants into equal-sized buckets based on the values that you specify for the **Min score** and **Max score** options.
 - **Rejects** — Distributes the rejected applications into equal-sized buckets based on the score range that is in the rejects data set.
 - **Scorecard** — Distributes the rejected applicants into equal-sized buckets based on the score range that is in the augmented data set.

The default value is **Accepts**.

- **Min score** — Specifies the lower bound of the score range when the **Score range method** is set to **Manual**.
- **Max score** — Specifies the upper bound of the score range when the **Score range method** is set to **Manual**.
- **Score buckets** — Specifies the number of buckets to use for parceling during the classification of good and bad. Possible values range from 1 to 100. The default value is 25.
- **Random seed** — Specifies the seed for generating random numbers. With parceling, the rejects are randomly assigned as event and nonevent within each score bucket.



Reject Inference Results

After running the node, open the Results window by right-clicking the node and selecting **Results**.

You can download the data that is used to build the charts and tables in the Results window by clicking . A CSV file with the chart or table data will immediately begin downloading.

In the **Output Data** tab, you can view, save, and explore the data that is generated by the node. For more information, see [Explore and Visualize](#).

Several charts have detailed descriptions available that explain how the charts are generated. The descriptions also provide information about how to interpret the charts that is custom to your particular data and models. To access these detailed descriptions:

- Expand an individual chart by clicking . The detailed description is in the right panel.
- Select  on an individual chart. A window appears with the detailed description.

Note: The detailed descriptions are available only in the English language.

Summary

- **Interval Variables** — Displays the distribution of the scorecard points, probability of the event, and probability of the nonevent for the accepts and inferred data sets.
- **Classification Chart** — Displays the distribution of the predicted event and nonevent for the accepts and inferred data sets.
- **Node Statistics** — Displays statistics of the node run. Statistics include the time at which the node run finished executing, the setup time (in seconds), the run duration (in seconds), whether the run was initiated from the pipeline, and the user who ran the node.
- **Properties** — Specifies the various properties that you selected before running the node.
- **Output** — Displays the SAS output of the scorecard run.

Scorecard

<i>Overview of Scorecard</i>	299
<i>Scorecard Properties</i>	300
General Properties	300
Scaling Options	300
Adverse Characteristic Options	301
Logistic Regression	302
Post Training Properties	306
<i>Scorecard Results</i>	308
<i>Using the Scorecard Node</i>	312

Overview of Scorecard

Note: Risk Modeling Add-on for SAS Visual Data Mining and Machine Learning is not included with the base version of SAS Visual Data Mining and Machine Learning. If your site has not licensed Risk Modeling Add-on for SAS Visual Data Mining and Machine Learning, the **Scorecard** node does not appear in your SAS Visual Data Mining and Machine Learning software.

The **Scorecard** node is a Miscellaneous node that calculates a credit score that is used to reflect the odds of an applicant being a good or bad credit risk. An **Interactive Grouping** node must precede a **Scorecard** node. If your pipeline contains a **Reject Inference** node, then another **Interactive Grouping** node must follow the **Reject Inference** node. Otherwise, you cannot add another **Scorecard** node to your pipeline.

Input variables are referred to as *characteristics* and each possible value of a characteristic is referred to as an *attribute*. The **Scorecard** node calculates a credit score in two steps:

- 1 Either the weight of evidence (WOE) or group variables are used as input variables to a logistic regression model. This estimates the natural log of the odds ($\ln(\text{odds})$) as a linear function of the characteristics.

- 2 A linear transformation is applied to the predicted $\ln(\text{odds})$ to compute a score for each attribute of each characteristic. The score for an attribute of a characteristic is derived from the following equation: $\text{score} = \ln(\text{odds}) * \text{factor} + \text{offset}$.

Several scaling options are available that enable you to control the factor and offset values of the score equation. Scaling enables you to control the range of the scores as well as the rate of change in the odds for a given increase in the score.

Scorecard Properties

General Properties

- **Analysis variables** — Specifies whether the weight of evidence (WOE) variables or the group variables are used in the scorecard computation.
- **Scorecard type** — Specifies the type of scorecard to generate. Here are the possible values:
 - **Detailed** — Displays the information contained in the **Summary** scorecard and contains additional information such as group number, WOE, population percentages, and regression coefficients for each variable.
 - **Summary** — Displays the variable names and the scorecard points for each variable.

The default value is **Summary**.

Scaling Options

- **Intercept-based scorecard** — Specifies whether to generate a scorecard with scorecard points for the intercept as well as all variables in the model.
- **Reverse scorecard** — Specifies whether the generated scorecard points should be reversed. Scorecard points decrease as the event rate decreases.
- **Odds** — Specifies the nonevent to event odds that correspond to the score value that you specify in the **Scorecard points** property.
- **Scorecard points** — Specifies a specific score that is associated with the odds that are specified in the **Odds** property.
- **Points to double odds** — Specifies the increase in score points that results in the score that corresponds to twice the odds.
- **Precision** — Specifies how many decimal places are used to calculate the scorecard points of an attribute. Possible values are integers between 0 and 4. The default value is 0, which produces integer scorecard points.

- **Bucketing method** — Specifies the method that is used for creating the buckets of scorecard points. The bucketing method affects the appearance of plots and tables in the node results. Here are the possible values:
 - **Min/max distribution**
 - **Quantiles**

The default value is **Min/max distribution**.
- **Number of buckets** — Specifies the number of bins into which the score range is divided.
- **Revenue accepted good** — Specifies the assumed average revenue from accepting applicants who have good credit status. This value is used to provide horizontal reference lines in a trade-off plot as well as part of the calculations in the Gains table.
- **Cost accepted bad** — Specifies the assumed average cost of accepting applicants who have a bad credit status. This value is used to provide horizontal reference lines in a trade-off plot as well as part of the calculations in the Gains table.
- **Current approval rate** — Specifies the percentage of applicants who are approved. This value is used to provide horizontal reference lines in a trade-off plot in the node results.
- **Current event rate** — Specifies the percentage of applicants who have bad credit status. This value is used to provide horizontal reference lines in a trade-off plot in the node results.

Adverse Characteristic Options

- **Method** — Specifies the method to use to identify adverse characteristics. Here are the possible values:
 - **Neutral score** — the value corresponds to the scorecard points when the WOE is equal to zero.
 - **Weighted average score** — the value corresponds to the weighted average of the scorecard points of the observations in the groups.

The default value is **Neutral score**.
- **Display value** — Specifies whether the adverse characteristic value should be displayed in the generated scorecard.
- **Generate report** — Specifies whether adverse characteristics should be reported in the node results.
- **Number of characteristics** — Specifies the number of adverse characteristics to include in the generated report.
- **Manage Adverse Variables** — Enables you to identify whether a variable is included in the adverse variable calculations and reports.

Logistic Regression

- **Selection method** — Specifies a model selection method. Here are the possible values:
 - **(none)** — Specifies not to perform variable selection.
 - **Backward** — Specifies that training is done by least squares regression with all candidate effects in the model. Features are removed until the stop criterion is met.
 - **Fast backward** — Specifies that training is done by least squares regression that starts with all effects in the model. Features are deleted without refitting the model.
 - **Forward** — Specifies that training is done by least squares regression that starts with no candidate effects in the model. Effects are added until the Entry significance level or the stop criterion is met.
 - **LASSO** — Specifies that training is done using the group LASSO method, adding and removing effects by a sequence of LASSO steps.
 - **Stepwise** — Specifies that training is done by least squares regression. The training starts as in the forward model, but it might remove effects already in the model.

The default value is **Stepwise**.

- **Selection Options**
 - **Effect-selection criterion** — Specifies the criterion that the procedure uses to determine the order in which effects enter or leave at each step of the selection method. Here are the possible values:
 - **AIC** — Specifies Akaike's Information Criterion. AIC is based on the Kullback-Leibler information measure of discrepancy between the true distribution of the response variable and the distribution specified by the model. Smaller values indicate better models, and AIC values can become negative.
 - **AICC** — Specifies Corrected Akaike's Information Criterion. AICC adjusts the AIC value by accounting for sample size. Extra features penalize AICC more than AIC. As the sample size increases, AICC and AIC converge.
 - **SBC (BIC)** — Specifies Schwarz Bayesian Criterion (SBC), also known as the Bayesian Information Criterion (BIC). Unexplained variations in the response variable and the number of features increase the value of the SBC. Smaller values indicate better models. SBC penalizes total number of model parameters more strongly than AIC, and therefore tends to choose more sparse models.
 - **Significance level** — Specifies the standard statistical significance level criterion.

If the **Selection method** above is **LASSO**, this option is unavailable. The default value is **SBC (BIC)**.

- **Selection-process stopping criterion** — Specifies the criterion to stop the selection process. Here are the possible values:

- **AIC** — Specifies Akaike's Information Criterion. AIC is based on the Kullback-Leibler information measure of discrepancy between the true distribution of the response variable and the distribution specified by the model. Smaller values indicate better models, and AIC values can become negative. Selection stops at the step where AIC starts to increase.
- **AICC** — Specifies Corrected Akaike's Information Criterion. AICC adjusts the AIC value by accounting for sample size. Extra features penalize AICC more than AIC. Smaller values indicate better models. As the sample size increases, AICC and AIC converge. Selection stops at the step where AICC starts to increase.
- **SBC (BIC)** — Specifies Schwarz Bayesian Criterion (SBC), also known as the Bayesian Information Criterion (BIC). Unexplained variations in the response variable and the number of features increase the value of the SBC. SBC penalizes the total number of model parameters more strongly than AIC and therefore tends to choose more sparse models. Selection stops at the step where SBC starts to increase.
- **Significance level** — Specifies the standard statistical significance level criterion. Selection stops at the step where significance level exceeds the specified level. The default value is 0.05.
- **Validation ASE** — Average square error (ASE) of the model is computed by using the validation data, and selection stops at the step where the validation ASE starts to increase. This method requires partitioned data.

The default value is **SBC (BIC)**.

- **Model-selection criterion** — Specifies the criterion to choose the model (from the list of models at each step of the selection process) that yields the best value of the specified criterion. If the optimal value of the specified criterion occurs for models at more than one step, then the model that has the smallest number of parameters is chosen. Here are the possible values:
 - **AIC** — Specifies Akaike's Information Criterion. AIC is based on the Kullback-Leibler information measure of discrepancy between the true distribution of the response variable and the distribution specified by the model. Smaller values indicate better models, and AIC values can become negative. The model that has the minimal AIC value is chosen.
 - **AICC** — Specifies Corrected Akaike's Information Criterion. AICC adjusts the AIC value by accounting for sample size. Extra features penalize AICC more than AIC. As the sample size increases, AICC and AIC converge.
 - **SBC (BIC)** — Specifies Schwarz Bayesian Criterion (SBC), also known as the Bayesian Information Criterion (BIC). Unexplained variations in the response variable and the number of features increase the value of the SBC. SBC penalizes total number of model parameters more strongly than AIC and therefore tends to choose more sparse models. The model that has the minimal SBC value is chosen.
 - **Validation ASE** — Average square error (ASE) of the model is computed by using the validation data, and the model that has the minimal validation ASE is chosen. This method requires partitioned data.

The default value is **SBC (BIC)**.

- **Manage Variable Ordering** — Specifies the order in which variables are passed into the model if the **Minimum number of effects** option is set to a value greater than zero.
- **Entry significance level** — Specifies the significance level for adding variables in the forward and stepwise directions. The default value is 0.05.
- **Stay significance level** — Specifies the significance level for removing variables in backward and stepwise directions. The default value is 0.05.
- **Maximum number of effects** — Specifies the maximum number of effects in any model that is considered during the selection process. If a model at some step of the selection process contains the specified maximum number of effects, then no additional effects are considered. If **Maximum number of effects** is set to 0 (the default value), this option is ignored.
- **Minimum number of effects** — Specifies the minimum number of effects in any model that is considered during the forward and backward selection process. If **Minimum number of effects** is set to 0 (the default value), this option is ignored.
- **Maximum number of steps** — Specifies the maximum number of selection steps that are performed. If **Maximum number of steps** is set to 0 (the default value), this option is ignored.
- **Optimization Options**
 - **Optimization technique** — Specifies the optimization method used when fitting a model. Here are the possible values:
 - (none)
 - Conjugate-gradient
 - Double-dogleg
 - Dual quasi-Newton
 - Nelder-Mead simplex
 - Newton-Raphson
 - Newton-Raphson with ridging
 - Trust-region

The default value is **Newton-Raphson with ridging**. For more information, see the PROC NL MIXED documentation in *SAS/STAT User's Guide*.

Note: Optimization options are not available for the **LASSO** selection method. LASSO uses the Nesterov algorithm as the default optimization technique.

- **Maximum number of iterations** — Specifies the maximum number of iterations of any optimization. The default value depends on the optimization technique used:

Optimization Technique	Default Value
Conjugate-gradient	400
Double-dogleg	200

Dual quasi-Newton	200
Nelder-Mead simplex	1000
Newton-Raphson	50
Newton-Raphson with ridging	50
Trust-region	50

To use the default value, leave **Maximum number of iterations** blank or use a dot.

- **Maximum number of function evaluations** — Specifies the maximum number of function calls of any optimization. The default value depends on the optimization technique used:

Optimization Technique	Default Value
Conjugate-gradient	1000
Double-dogleg	500
Dual quasi-Newton	500
Nelder-Mead simplex	3000
Newton-Raphson	125
Newton-Raphson with ridging	125
Trust-region	125

To use the default value, leave **Maximum number of function evaluations** blank or use a dot.

- **Maximum CPU time in seconds** — Specifies an upper limit of CPU time (in seconds) for the optimization process. The default value is the largest floating-point double representation of your computer. To use the default value, leave **Maximum CPU time in second** blank or use a dot.
- **Minimum number of iterations** — Specifies the minimum number of iterations in any optimization. The default value is 1. To use the default value, leave **Minimum number of iterations** blank or use a dot.
- **Normalize objective function** — Specifies whether the objective function should be normalized during optimization by the reciprocal of the used frequency count. By default, this option is selected.
- **Convergence Options**
 - **Absolute function convergence** — Specifies the threshold for absolute function convergence. The default value is the negative square root of the

largest double-precision value. To use the default value, leave **Absolute function convergence** blank or use a dot.

- **Absolute function difference convergence** — Specifies the threshold for absolute function difference convergence. The default value is 0. To use the default value, leave **Absolute function difference convergence** blank or use a dot.
- **Absolute gradient convergence** — Specifies the threshold for absolute gradient convergence. The default value is 1E-5. To use the default value, leave **Absolute gradient convergence** blank or use a dot.
- **Relative function difference convergence** — Specifies the relative function difference convergence. The default value is the machine precision times 2. To use the default value, leave **Relative function difference convergence** blank or use a dot.
- **Relative gradient convergence** — Specifies the threshold for relative gradient convergence. The default value is 1E-8. To use the default value, leave **Relative gradient convergence** blank or use a dot.
- **Use the exact percentile method for lift calculations** — Specifies whether to use the exact percentile method for calculating lift and related assessment measures. When this property is deselected, the iterative method is used. The exact method should be used if there are convergence errors when using the iterative method.
- **Binary Classification Cutoff**
 - **Specify node binary classification cutoff** — Specifies whether to use the binary classification cutoff specified below for the node. If this property is deselected, the project binary classification cutoff is used for determining the predicted value for a binary target based on the posterior probabilities. By default, this property is deselected.
 - **Node binary classification cutoff** — Specifies the cutoff to use in the node for determining the predicted value for a binary target based on the posterior probabilities. This option is available only if **Specify node binary classification cutoff** is selected. The default value is 0.5.

.....

Note: The **Rules** settings in the Project Settings window enable you to specify a new cutoff value by selecting **Override the default classification cutoff**. If **Specify node binary classification cutoff** is deselected, then the value that you specified in the Project Settings window is used. If **Specify node binary classification cutoff** is selected, then the value that you specified in **Node binary classification cutoff** is used.

.....

Post Training Properties

Changing the following properties will not retrain the model.

Model Interpretability — Specifies the method of model interpretation in order to show how the model responds to changing inputs.

- **Global Interpretability**


- **Variable importance** — Specifies whether to display a table of the relative importance values for the input variables. The surrogate variable importance values are calculated using a one-level decision tree for each input to predict the predicted value as a global surrogate model. For tree-based supervised learning models, the model variable importance table is displayed as well.
- **PD plots** — Specifies whether to display partial dependency (PD) plots that show the relationship between model input variables and their predictors. PD plots are generated for only up to the top ten variables in the relative importance table. Here is how to determine the variables that are displayed in the PD plot: for tree-based models (forest, decision tree, and gradient boosting), the model variable importance table is used. For other supervised learning models, the surrogate variable importance table is used.
- **Local Interpretability**
 - **ICE plots** — Specifies whether to display Individual Conditional Expectation (ICE) plots. ICE plots can reveal interesting subgroups and interactions between model variables. ICE plots are generated for only up to the top ten variables in the relative importance table. Here is how to determine the variables that are displayed in the ICE plot: for tree-based models (forest, decision tree, and gradient boosting), the model variable importance table is used. For other supervised learning models, the surrogate variable importance table is used.
 - **LIME** — Specifies whether to use the Locally Interpretable Model-Agnostic Explanation (LIME) method to explain the model predictions.
 - **Kernel SHAP** — Specifies whether to use the Kernel Shapley additive explanations (Kernel SHAP) method to explain the model predictions.
 - **Maximum number of Kernel SHAP variables** — Specifies the maximum number of Kernel SHAP variables to plot. Possible values range from 1 to 100. The default value is 20.
 - **Specify instances to explain** — Specifies the individual observations that you want to explain. The **Random** observations option provides explanations for five randomly selected observations. The **Specify up to 5** observations option enables you to provide IDs for the specific observations in the data. The IDs should be unique values of the variable that has the role **Key** in the metadata. If no variable is assigned the role **Key**, then you can use the variable `_dmIndex_` to specify the individual observations.
 - **LIME/Kernel SHAP Tables**
 - **Explainer Information table** — Specifies whether the Explainer Information tables for the LIME and Kernel SHAP methods should be included in the node results. The Explainer Information table contains information about what settings were used to perform the LIME and Kernel SHAP methods.
 - **Explainer Fidelity table** — Specifies whether the Explainer Fidelity tables for the LIME and Kernel SHAP methods should be included in the node results. The Explainer Fidelity table contains information about the accuracy of the local explanation predictions, compared to the black-box machine learning model.
- **PD/ICE Options**
 - **Maximum number of variables** — Specifies the maximum number of input variables to use to generate the PD and ICE plots. The input variables are

selected based on their relative importance. Possible values range from 1 to 10. The default value is 5.

- **Number of observations** — Specifies the maximum number of observations to sample for the PD and ICE plots. The default value is 1000.
- **Number of tick points** — Specifies the number of tick points for an interval analysis variable to generate for the PD and ICE plots. Possible values range from 3 to 100. The default value is 50.
- **Truncate lower tail** — Specifies under what percentage to discard the values of the interval analysis variables for the PD and ICE plots. Possible values range from 0 to 99. The default value is 5%.
- **Truncate upper tail** — Specifies above what percentage to discard the values of the interval analysis variables for the PD and ICE plots. Possible values range from 0 to 99. The default value is 95%.
- **Seed** — Specifies the seed for generating random numbers. The seed is used for selecting individual observations to explain, selecting observations to generate PD and ICE plots, and simulating data for LIME and Kernel SHAP.



Scorecard Results

After running the node, open the Results window by right-clicking the node and selecting **Results**.

You can download the data that is used to build the charts and tables in the Results window by clicking . A CSV file with the chart or table data will immediately begin downloading.

In the **Output Data** tab, you can view, save, and explore the data that is generated by the node. For more information, see [Explore and Visualize](#).

Several charts have detailed descriptions available that explain how the charts are generated. The descriptions also provide information about how to interpret the charts that is custom to your particular data and models. To access these detailed descriptions:

- Expand an individual chart by clicking . The detailed description is in the right panel.
- Select  on an individual chart. A window appears with the detailed description.

Note: The detailed descriptions are available only in the English language.

Node

- **t Values by Parameter** — Displays bar charts for the t values in the final model. The bars are color-coded to indicate the algebraic signs of the coefficients.
- **Parameter Estimates** — Displays a table of the various statistics related to estimates for the parameters. These statistics include the t value, sign, estimate, absolute estimate, the p-value, chi-square value, and standard error.

- **Selection Summary** — Displays a table of the iterations, adding in and removing effects. This table also includes the SBC and optimal SBC values.
- **Regression Fit Statistics** — Displays a table of data on the logistic regression fit statistics. The table includes the statistic, a corresponding qualitative description, and the training, validation, and testing values.
- **Scorecard** — Displays the number of scorecard points that correspond to each variable group. If the **Scorecard type** property is set to **Detailed**, then the scorecard also displays the variable regression coefficient, the variable group weight of evidence (WOE), the event rate, and the percentage of population.
- **Empirical Odds Plot** — Displays the empirical odds plotted against the average values of the scorecard points.
- **Captured Event Plot** — Displays the proportion of the total event count that is captured across the proportion of the total population count.
- **Scorecard Table** — Displays the detailed scorecard information in addition to any manual WOE changes.
- **Node Statistics** — Displays statistics of the node run. Statistics include the time at which the node run finished executing, the setup time (in seconds), the run duration (in seconds), whether the run was initiated from the pipeline, and the user who ran the node.
- **Node Score Code** — Displays the SAS score code that was created by the node. The SAS score code can be used outside the Model Studio environment to score new data.
- **Path Score Code** — Displays the SAS score code that was created by the node, as well as all preceding nodes up to the most recent **Reject Inference** node. All score code is reset when a **Reject Inference** node is added to the pipeline. The SAS model score code can be used outside the Model Studio environment to score new data.
- **DS2 Package Code** — Displays the SAS DS2 package code that was created by the node. The SAS score code can be used outside the Model Studio environment in custom user applications. DS2 package code is used when publishing models to SAS Micro Analytic Service.
- **Score Inputs** — Displays a table of data on the input variables for scoring calculations. The table includes the variables' name, role, level, type, label, format, and length.
- **Score Outputs** — Displays a table of data on the predicted response variable for scoring calculations. The table includes the variables' name, role, type, format and length, as well as the creator, creator GUID, and function.
- **Training Code** — Displays the SAS code that Model Studio used to train the node.
- **Properties** — Specifies the various properties that you selected before running the node.
- **Output** — Displays the SAS output of the scorecard run.

Assessment

- **Score Distribution** — Displays the distribution of the scorecard points, the event odds, and the log of the event odds.
- **Gains Table** — Displays several statistics for each score bucket. You can control the number of score buckets that are generated with the **Number of buckets** property.

- **Trade-off Plots** — Displays the following statistics across the range of scorecard points:
 - Cumulative Event Rate and Cumulative Approval Rate
 - Average Marginal Profit and Cumulative Approval Rate
 - Average Total Profit and Cumulative Approval Rate
- **Statistics Table** — Displays several statistics such as the Gini statistic and information value for each input variable.
- **Event Frequency Charts** — Displays various event counts and rates across the range of scorecard points. The following plots are available:
 - Bucket
 - Event Count
 - Cumulative Event Count
 - Marginal Event Rate
 - Cumulative Event Rate
- **Average Predicted Probability** — Displays the average predicted probability of an event across the range of scorecard points.
- **Adverse Characteristics** — Displays the number of times each input variable is cited for each adverse reason. The output data includes additional details about the reasons for denial for each applicant.
- **Lift Reports** — Displays the cumulative lift as a function of the depth for the model. The cumulative lift is given for each of the data roles. To examine other statistics as a function of depth, use the drop-down menu in the upper right corner. Other statistics include lift, gain, captured response percentage, cumulative captured response percentage, response percentage, and cumulative response percentage. This result is displayed only if the target is a class variable.
- **ROC Reports** — Displays the ROC (receiver operating characteristic) chart for a model, giving the sensitivity as a function of 1-specificity. The sensitivity is given for each of the data roles. To examine other statistics, use the drop-down menu in the upper right corner. Other statistics include accuracy and F1 score. This result is displayed only if the target is a class variable.
- **Fit Statistics** — Displays a table of the fit statistics for the model, broken down by data role.
- **Event Classification** — Displays the confusion matrix at various cutoff values for each partition. The confusion matrix contains true positives (events that are correctly classified as events), false positives (non-events that are classified as events), false negatives (events that are classified as non-events), and true negatives (non-events that are correctly classified as non-events). The classification cutoffs that are used are the default value of 0.5, the Kolmogorov-Smirnov cutoff value for each partition, and the **Node binary classification cutoff** that is specified for binary targets. Use the drop-down menu to view the information in the chart as percentages or counts. You can also view the information summarized in a table. This result is displayed only if the target is a class variable.
- **Nominal Classification** — Displays either the percentage of observations or the number of observations that predict each level of the target variable. The plot is segmented by target level and partition level. The target level with the greatest

predicted probability is the predicted outcome. This result is displayed only if the target is a nominal variable.


Model Interpretability

- **Surrogate Model Variable Importance** — Displays a table of the relative importance of the input variables, including variable level and label. Relative importance is calculated using a one-level decision tree for each input variable to predict the predicted value as a global surrogate model.
- **PD Plot** — Displays the functional relationship between the input variable and the model prediction. For interval input variables, the 95% confidence interval for the average target prediction is indicated by the shaded band around the line. The X axis includes a heat map that shows the distribution of the input variable. The extreme values of the input variables are removed by truncating the lower and upper tails of its distribution. The amount of truncation can be controlled by specifying values for the **Truncate lower tail** and the **Truncate upper tail** properties.
- **PD and ICE Overlay Plot** — Displays the functional relationship between the input variable and the model prediction as well as the functional relationship between individual observations and the model prediction. For each individual observation, the corresponding ICE curve displays values of the input variable on the X axis and the corresponding predicted probability or prediction of the target variable on the Y axis, holding the other input variable values constant at that observation. The X axis includes a heat map that shows the distribution of the input variable. The extreme values of the input variables are removed by truncating the lower and upper tails of its distribution. The amount of truncation can be controlled by specifying values for the **Truncate lower tail** and the **Truncate upper tail** properties.
- **LIME Explanations** — The LIME algorithm calculates a more easily interpretable linear model around an individual observation. The chart displays the regression estimates for the input variables that are selected in the local surrogate linear regression model. The input variables are ordered by significance such that the most important variable for the local regression model is at the bottom of the chart. The LASSO technique is used to select the most significant predictors from the set of input variables that are used to train the model. Each nominal input variable is binary-encoded based on whether it matches the level of the individual observation. A positive estimate indicates that the observed value of the input variable increases the predicted probability of the event.
- **Kernel SHAP Values** — For each individual observation, an input variable's Kernel SHAP value is the contribution of the observed value of the input variable to the predicted probability or prediction of the target variable. The Kernel SHAP values are the regression coefficients that are obtained by fitting a weighted least squares regression. The Kernel SHAP values of all input variables sum to one. The input variables are ordered by significance such that the most important variable according to the absolute Kernel SHAP values is at the top of the chart. Each nominal input is binary-encoded based on whether it matches the individual observation. Interval inputs are binary encoded based on their proximity to the individual observation.
- **LIME Information Table** — Displays information about the settings used to perform the LIME method.
- **Kernel SHAP Information Table** — Displays information about the settings used to perform the Kernel SHAP method.


- **LIME Fidelity Table** — Displays information about the accuracy of the local explanation predictions that are calculated using the LIME method, compared to the black-box machine learning model. The model prediction, the explainer prediction, and the explainer root mean square error (RMSE) are displayed.
- **Kernel SHAP Fidelity Table** — Displays information about the accuracy of the local explanation predictions that are calculated using the Kernel SHAP method, compared to the black-box machine learning model. The model prediction, the explainer prediction, and the explainer root mean square error (RMSE) are displayed.

Using the Scorecard Node

After you run the **Scorecard** node, you can manually adjust the scorecard points and the score ranges.

To manually adjust the scorecard points, right-click the node, and then select **Manage Scorecard Points**. In the Scorecard Points window, you can manually adjust the number of points that each variable group contributes to the total score. Select **Reset** to reset the scorecard points back to their original values. Click  to save your scorecard point adjustments.

Note: You cannot adjust the scorecard points for interval variables that have double quotation marks in the variable name.

To manually adjust the score ranges, right-click the node, and then select **Manage Score Ranges**. In the Scorecard Ranges window, you can manually adjust the score ranges for each score bucket. Select **Reset** to reset the scorecard ranges back to their original values. Click  to save your scorecard range adjustments.

To indicate that you want your scorecard model to be assessed and processed by the **Model Comparison** node:

- 1 Right-click the node, and select **Move**.
- 2 Select **Supervised Learning**. The **Scorecard** node is now connected to a **Model Comparison** node.

Note: If you move your **Scorecard** node to the Supervised Learning group, you cannot add a successive **Reject Inference** node.

To move your **Scorecard** node out of the Supervised Learning group:

- 1 Right-click the **Scorecard** node, and select **Move**.
- 2 Select **Preprocessing**.