Contents

Downloading Logs ................................................................. 100
Overview

SAS Visual Data Mining and Machine Learning 8.4 has general enhancements and improvements for the Data tab and many existing nodes. Two new nodes were added as well.

General Enhancements

- Models imported from SAS Visual Analytics can now be edited.
- A new Insights tab that contains summary reports about the project, champion models, and challenger models is generated. For more information, see Insights Tab on page 94.
- SAS Micro Analytic Service is a supported publishing destination.
- You can specify an individual observation as the type of instance to explain for LIME and ICE.
- Event Classification and Nominal Classification charts are available on the Assessment tab in the Results window for supervised learning models with a class target variable.
- Assessment and Model Interpretability plots include generated detailed descriptions.
- You can specify the number of ROC cutoff values to use in the ROC reports for supervised learning models with a binary target variable.
- Model interpretability properties are now flagged as reporting properties. Therefore, changes to them do not require your model to be retrained.
- The Feature Engineering template has been updated to use the new Target encoding transformation and includes other enhancements.
You can download the data used in the various reports of a node’s results.  
You can download the project Scoring and Assessing logs.  
You can specify a template at project creation.  
You can duplicate a pipeline.  
You can add notes to the header of your pipeline.  
When sharing projects, SAS Administrators must now be included in any group that the project is shared with.  
Internet Explorer 11 is no longer supported.

New Transformation and Imputation Methods for the Data Tab

- You can specify several new transformation methods for class input variables:
  - Level frequency encoding
  - Level proportion encoding
  - Target encoding
  - WOE encoding
- When using the Inverse and Inverse square transformations for interval input variables, an offset is added to the input variables before the transformation is applied. This ensures nonzero values and prevents creating missing values during the transformation when input variable values are zero.
- You can specify Cluster count as the imputation method for class input variables.
- You can specify Cluster mean as the imputation method for interval input variables.

New and Enhanced Nodes

- The Score Data node is a new Miscellaneous node that enables you to score a data table with the score code that was generated by the predecessor nodes in the pipeline. For more information, see Overview of Score Data.
- The Segment Profile node is a new Miscellaneous node that enables you to analyze segmented data in order to identify factors that differentiate each segment from the population. For more information, see Overview of Segment Profile.
- Anomaly Detection node:
  - You can specify whether to include anomaly observations in subsequent nodes when Filter is selected as the Anomaly indicator role.
  - You can specify whether to include class input variables during training.
  - You can specify one of the following options available for the Bandwidth computation method:
- Mean
- Modified mean
- Trace
- User Specify
  - You can specify Fast incremental as your optimization solver.
  - You can specify the Maximum time (minutes) property and the Maximum support vectors property under the Solver Options.

For the Bayesian Network node, you can autotune the following properties:

- Network Structure
- Maximum Parents
- Parenting Method
- Number of Bins

Clustering node:
- You can specify the variable role for the output cluster and cluster distance variables.

Data Exploration node:
- You can specify which data partition (or all input data) to analyze.

Decision Tree node:
- The default value for the Interval bin method property has been updated to Quantile.

Ensemble node:
- You can specify the Voting function to combine models for class targets.

Forest node:
- You can specify the minimum leaf size by specifying either the Count or the Proportion of the training observations with a nonmissing target value.
- The default value for the Interval bin method property has been updated to Quantile.

For the Gradient Boosting node, the following parameters were changed:

- The default value for the L2 regularization parameter has been updated to 1.
- The default value for the Maximum depth parameter has been updated to 4.
- The default value for the Number of interval bins parameter has been updated to 50.
- The default value for the Interval bin method parameter has been updated to Quantile.

Imputation node:
- If a Clustering node immediately precedes the Imputation node, you can specify Cluster count as the imputation method for class variables and you can specify Cluster mean as the imputation method for interval variables.

Model Comparison node:
You can specify **Misclassification at cutoff** for the **Class selection statistic** property.

- **Open Source Code** node:
  - In the Project or User settings, you can specify Python code that automatically prepends code into every **Open Source Code** node when the **Language** property is set to Python.
  - New data items such as `dm_classvar`, `dm_classlevel`, and `dm_predictionvar` have been defined for use in the code editor.

- **SAS Code** node:
  - Separate code editors for training code and scoring code are available.
  - Several new macros and macro variables have been defined and are organized into groups for use in the code editor.

- **Text Mining node**:
  - Kazakh is a new supported language.
  - Stop lists are automatically included and applied for all languages.

- For the **Transformations** node, you can specify several new transformation methods for class input variables:
  - **Level frequency encoding**
  - **Level proportion encoding**
  - **Target encoding**
  - **WOE encoding**

  When you are using **Inverse** transformations for interval input variables, before the transformation is applied, an offset is added to the input variables to ensure nonzero values. This prevents creating missing values during the transformation when input variable values are zero.

- For the **Variable Clustering** node, you can use the **Cluster identifier prefix** property to specify the prefix used to name the variable cluster identifiers and to name the cluster component variables when **Cluster representation** is set to **Cluster component**.

- For the **Variable Selection** node, the following parameters were changed:
  - The default value for the **L2 regularization** parameter has been updated to 1 for **Gradient Boosting Selection**.
  - The default value for the **Maximum depth** parameter has been updated to 4 for **Gradient Boosting Selection**.
  - The default value for the **Number of interval bins** parameter has been updated to 50 for **Gradient Boosting Selection**.
  - The default value for the **Interval bin method** parameter has been updated to **Quantile** for **Decision Tree Selection**, **Forest Selection**, and **Gradient Boosting Selection**.
Getting Started with SAS Visual Data Mining and Machine Learning in Model Studio

Charitable Giving Example ................................................................. 6
  Tutorial Scenario ................................................................. 6
  Create the Project and Import the Input Data .............................. 7
  Modify Variables ........................................................................ 10
  Create a Pipeline ....................................................................... 12
  Generate Descriptive Statistics ................................................... 12
  Replace Missing Values ............................................................. 13
  Automatically Train and Prune a Decision Tree ......................... 14
  Create a Gradient Boosting Model .............................................. 16
  Impute Missing Values .............................................................. 18
  Transform Variables ................................................................ 19
  Create a Logistic Regression ...................................................... 19
  Create a Neural Network ........................................................... 20
  Compare Models ....................................................................... 21
  Interpretability of Champion Model .......................................... 22
  Publish the Champion Model ..................................................... 24

SAS Code Node Examples ............................................................... 24
  Overview .................................................................................. 24
  Create the Project and Import the Input Data .............................. 25
  Modify Variables ....................................................................... 27
  Create a Gradient Boosting Model .............................................. 29
  Perform Variable Selection ......................................................... 33

Integrating Model Studio with SAS Visual Analytics .................... 35
  Overview .................................................................................. 35
  Download the Sample Data ......................................................... 36
  Create the Report ..................................................................... 37
  Create a Forest .......................................................................... 37
  Create a Support Vector Machine .............................................. 38
  Continuing in Model Studio ....................................................... 38

Open Source Code Node Example ................................................. 43
  Overview .................................................................................. 43
  Create the Project and Import the Input Data .............................. 43
  Modify Variables ..................................................................... 46
  Create the Pipeline ................................................................. 47
Charitable Giving Example

Tutorial Scenario

The Charitable Giving tutorial is intended for new Model Studio users. Although prior data mining experience is beneficial, any user can follow the discussion and complete the steps. The tutorial defines the problem, explores and visualizes the input data, performs data preparation, specifies model fit criteria, and then creates, configures, and trains multiple competing statistical modeling algorithms. A champion model is selected, and then score code is generated. The score code performs the trained champion model's analytic task on new data.

The narrative requires sequential data mining steps. Follow the steps in the order of presentation so that your results match the example. Later, you can modify your final pipeline model settings and add new competing models to see how your changes would affect predicted results.

The tutorial uses a training data set with known target variable values named DONOR_DATA. You can download DONOR_DATA from the SAS Visual Data Mining and Machine Learning product page. Click Example Data for Getting Started with SAS Visual Data Mining and Machine Learning in Model Studio. Download the ZIP file and extract DONOR_DATA.SAS7BDAT to a directory that your SAS Visual Data Mining and Machine Learning server can access.

In the tutorial, you are a data analyst at a national charitable organization. Your organization wants to use the results of a previous solicitation for donations to better target its next one. You want to determine which of the individuals in your mailing database are the most generous donors. By approaching only these people, your organization can spend less money on the solicitation effort and more money on charitable concerns. The finished pipeline diagram will resemble the one shown here:
Create the Project and Import the Input Data

This example assumes that you are signed in to Model Studio, which brings you to SAS Drive. In the upper left corner of the SAS Drive window, click the share icon and select **Build Models**.
You are directed to the Projects page. To create the project that you will use in this example:

1. Select **New Project** in the upper right corner of the Projects page.
2. Enter **Charitable Giving Example** for **Name** in the New Project window.
3 Select **Data Mining and Machine Learning** for **Type**.

Note: **Forecasting** and **Text Analytics** are additional options if you have licensed those products.

4 Ensure that **Blank Template** is specified for **Template**.

5 In the **Data** field, select **Browse**. The Choose Data window appears.

6 In the upper left corner of the Choose Data window, select **Import**.
Select Local File and navigate to the folder where DONOR_DATA is stored. Select DONOR_DATA.sas7bdat and click Open.

Click Import Item in the upper right corner of the Choose Data window. When the data is successfully imported, a note appears, saying that the data is ready for use.

Once the data set is successfully imported, click OK in the lower right corner of the Choose Data window. This brings you back to the New Project window.

Click Save in the New Project window. This directs you to the Data tab, where you can modify the variables in your data set.

Modify Variables

On the Data tab, variable roles are indicated in the Role column. To change the role of a variable:

1 Select a variable by clicking the corresponding check box to the left of the Variable Name column. The options panel for the selected variable appears to the right of the variable table on the Data tab. Select the TARGET_D variable.
2. Expand the drop-down list under Role, and select the role type that you want to assign to the selected variable. Changes made to each variable are automatically applied and saved. For TARGET_D, select the role Rejected.

**CAUTION**

To avoid making unwanted changes to variable properties, you must manually deselect each variable that you modify when you are finished making changes to its properties.

3. Ensure that the variables roles are set to the following values:
   - CLUSTER_CODE is set to Rejected.
   - CONTROL_NUMBER is set to ID.
   - TARGET_B is set to Target.
   - TARGET_D is set to Rejected.
   - _dmIndex_ is set to Key.
   - _PARTIND_ is set to Partition. In data mining, a strategy for assessing the quality of model generalization is to partition the data source. A portion of the data, called the training data set, is used for preliminary model fitting. The rest is reserved for empirical validation and is often split into two parts:
validation data and test data. The validation data set is used to prevent a modeling node from overfitting the training data and to compare models. The test data set is used for a final assessment of the model. In this example, the data has been pre-partitioned where 60% of the data is assigned for training, 30% of the data is assigned for validation, and 10% of the data is assigned for testing. Select Map Partition Levels and ensure that the partition levels are set to the following values:

- **Training Level** is set to 1.
- **Validation Level** is set to 0.
- **Test Level** is set to 2.

All other variable roles are set to Input.

4. Set the property Transform to Log10 for the following variables:
- FILE_AVG_GIFT
- LAST_GIFT_AMT
- LIFETIME_AVG_GIFT_AMT

**TIP** It is possible to select multiple variables for editing at one time by selecting the box in front of each variable name. By simultaneously selecting FILE_AVG_GIFT, LAST_GIFT_AMT, and LIFETIME_AVG_GIFT_AMT, you can change the Transform property for all three variables at once.

---

### Create a Pipeline

1. Select the Pipelines tab in the upper left corner.
2. Right-click the Data node and select Run.
3. Once the Data tab has run successfully, continue with the following sections to build your pipeline.

### Generate Descriptive Statistics

To see a statistical summary of the input data:

1. Right-click the Data node and select Add child node ⇒ Miscellaneous ⇒ Data Exploration.
2. Right-click the Data Exploration node and select Run.
3. Once the pipeline has run successfully, right-click the Data Exploration node and select Results from the menu that appears. The following pieces of information about the input data appear either as graphic tiles or rows in a table of projects:
Replace Missing Values

In this example, the variables SES and URBANICITY are class variables for which the value “?” denotes a missing value. SAS denotes a missing value using a blank for character variables and a period for numeric variables. Because this data uses a question mark instead, Model Studio sees it as an additional level of a class variable. However, the knowledge that these values are missing will be useful later in the model-building process. To implement a Replacement node:

1. Right-click the Data node and select Add child node ➔ Data Mining Preprocessing ➔ Replacement.
2. Once created, select the Replacement node.
3. In the options panel, complete the following:
   a. Set Replacement value for unknown class levels to Missing value.
   b. Expand Interval Inputs.
   c. Ensure that the Default limits method property is set to Standard deviation from the mean.
4. Right-click the Replacement node and select Run.
5. Once the node has run successfully, right-click the node and select Results to view detailed information in each of the following sections of interest:
   - Class Variables — Displays the replacement values for class variables.
Interval Variables — Displays the replacement values for interval variables.

Replacement Counts — Displays the number of observations replaced for each variable.

6 Close the Results window.

Automatically Train and Prune a Decision Tree

Decision tree models are advantageous because they are conceptually easy to understand, yet they readily accommodate nonlinear associations between input variables and one or more target variables. They also handle missing values without the need for imputation. Therefore, you decide to first model the data using decision trees. You will compare decision tree models to other models later in this example.

Note: A Model Comparison node is automatically created when you add a Supervised Learning node to your pipeline. Therefore, a Model Comparison node will be added to your pipeline when you add the Decision Tree node.
To insert a **Decision Tree** node:

1. Right-click the **Replacement** node and select **Add child node** ⇒ **Supervised Learning** ⇒ **Decision Tree**.

2. Once created, select the **Decision Tree** node.

3. In the options panel, complete the following:
   
   a. Expand **Splitting Options**.
   
   b. Set **Minimum leaf size** to 8.
   
   c. Expand **Pruning Options**.
   
   d. Set **Subtree method** to **Reduced error**.

4. Right-click the **Decision Tree** node and select **Run**.

5. Right-click the **Decision Tree** node and select **Results** to view components such as the following:

   - **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 pointer over a node of a tree, a text box displays information about the node. Zoom in to display additional details about each node.

   ![Tree Diagram Image]

   - **Treemap** — 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. Select a node to display additional details about that node.
Create a Gradient Boosting Model

The Gradient Boosting node uses a partitioning algorithm to search for an optimal partition of the data for a single target variable. Gradient boosting is an approach that resamples the analysis data several times to generate results that form a weighted average of the resampled data set. Tree boosting creates a series of decision trees that form a single predictive model. Like decision trees, boosting makes no assumptions about the distribution of the data. Boosting is less prone to overfit the data than a single decision tree. If a decision tree fits the data fairly well, then boosting often improves the fit. For more information, see Overview of Gradient Boosting.

To create a gradient boosting model of the data:

1. Right-click the Replacement node and select Add child node ➔ Supervised Learning ➔ Gradient Boosting.

2. Once created, select the Gradient Boosting node.

6. Close the Results window.
3. In the options panel, expand **Tree-splitting Options**. Set **Maximum depth** to 10.

4. In the options panel, enable the **Perform Autotuning** option. Autotuning automatically adjusts the hyperparameters that govern the model training process to the best values. For more information, see **Overview of Autotuning**.

5. Right-click the **Gradient Boosting** node and select **Run**.

6. Right-click the **Gradient Boosting** node and select **Results** to view components such as the following:
   - **Error Plot**

   ![Error Plot Diagram]

   - **Lift Reports** — Located on the **Assessment** tab.

   ![Lift Reports Diagram]

   - **Variable Importance**

   ![Variable Importance Table]

   - **Autotune Best Configuration**
Because of the nondeterministic behavior of SAS Viya, your results might not match these results.

7 Close the Results window.

**Impute Missing Values**

For decision trees, missing values are not problematic. Surrogate splitting rules enable you to use the values of other input variables to perform a split for observations with missing values. In Model Studio, however, models such as regressions and neural networks ignore observations that contain missing values, which reduces the size of the training data set. Less training data can substantially weaken the predictive power of these models. To overcome this obstacle of missing data, you can impute missing values before you fit the models.

*TIP* Impute missing values before fitting a model that ignores observations with missing values if you plan to compare those models with a decision tree. Model comparison is most appropriate between models that are fit with the same set of observations.

To impute missing values:

1 Right-click the Replacement node and select Add child node ⇒ Data Mining Preprocessing ⇒ Imputation.

2 Once created, select the Imputation node.

3 In the options panel, under Interval Inputs, set Default method to Median.

4 Right-click the Imputation node and select Run.

5 Once the Imputation node has successfully run, right-click the node and select Results. Explore the following:

   - Input Variable Statistics
Imputed Variables Summary

<table>
<thead>
<tr>
<th>Imputed Variable</th>
<th>Method</th>
<th>Input Variable</th>
<th>Value</th>
<th>Percent Missing</th>
</tr>
</thead>
<tbody>
<tr>
<td>IMP_REP_DONORBIRTH_YEAR</td>
<td>MEDIAN</td>
<td>REP_DONOR_BIRTH_YEAR</td>
<td>60</td>
<td>25.1656</td>
</tr>
<tr>
<td>IMP_REP_INCOME_GROUP</td>
<td>COUNT</td>
<td>REP_INCOME_GROUP</td>
<td>5</td>
<td>22.6103</td>
</tr>
<tr>
<td>IMP_REP_MONTHS_SINCE_LAST_PROMOTION</td>
<td>MEDIAN</td>
<td>REP_MONTHS_SINCE_LAST_PROMOTION</td>
<td>18</td>
<td>1.1701</td>
</tr>
<tr>
<td>IMP_REP_WEALTH_RATING</td>
<td>COUNT</td>
<td>REP_WEALTH_RATING</td>
<td>9</td>
<td>45.7885</td>
</tr>
</tbody>
</table>

6 Close the Results window.

Transform Variables

At the beginning of this example, you decided to transform the variables FILE_AVG_GIFT, LAST_GIFT_AMT, and LIFETIME_AVG_GIFT_AMT using the Log10 methodology. To execute the transformation of these variables:

1 Right-click the Imputation node and select Add child node ⇒ Data Mining Preprocessing ⇒ Transformations.

2 Right-click the Transformations node and select Run.

Create a Logistic Regression

As part of your analysis, you want to include some parametric models for comparison with the decision trees that you built earlier in this example. Because it is familiar to the management of your organization, you have decided to include a logistic regression as one of the parametric models. To do so:

1 Right-click the Transformation node and select Add child node ⇒ Supervised Learning ⇒ Logistic Regression.

2 Right-click the Logistic Regression node and select Run from the resulting menu.

3 Once the node has successfully run, right-click the Logistic Regression node and select Results. Explore the following:
Create a Neural Network

A neural network model is a stronger predictive model than the other models that you have generated up to this point. Neural networks are a class of parametric models that can accommodate a wider variety of nonlinear relationships between a set of predictors and a target variable than can logistic regression.

You know that the management of your organization prefers a stronger predictive model. You choose to add a neural network model even though it will be more complicated to explain than a regression model or a decision tree.

Building a neural network model involves two main phases. First, define the network configuration. You can think of this step as defining the structure of the model that you want to use. Then, iteratively train the model. The Neural Network node trains a specific neural network configuration, and is best used when you know a lot about the structure of the model that you want to define.

Before creating a neural network, you will reduce the number of input variables with the Variable Selection node. Performing variable selection reduces the number of input variables and saves computer resources. To use the Variable Selection node followed by the Neural Network node:
1 Right-click the Imputation node and select Add child node ⇒ Data Mining Preprocessing ⇒ Variable Selection.

2 Once created, right-click the Variable Selection node and select Run.

3 Right-click the Variable Selection node and select Add child node ⇒ Supervised Learning ⇒ Neural Network.

4 Once created, select the Neural Network node.

5 Take these actions in the options panel:
   a Set Number of hidden layers to 5.
   b Expand Target Layer Options and select Direct connections.

6 Right-click the Neural Network node and select Run.

7 Once the Neural Network node has successfully run, right-click the node and select Results to view components such as the Network Diagram.

8 Close the Results window.

---

**Compare Models**

To use the Model Comparison node to compare the models that you have built in this example and to select one as the champion model:

1 Right-click the Model Comparison node that was created when you first created the Decision Tree node and select Run.
2 Right-click the **Model Comparison** node and select **Results**.

3 In the **Model Comparison** pane, you can see that the Logistic Regression model is selected as the **Champion Model**. In the Model Comparison node, Model Studio selects the champion model based on the value of a single statistic. You can specify which statistic to use for selection in the options pane. Because you did not change the value of this property, the default statistic (Kolmogorov-Smirnov statistic) was used.

**Note:** The selection statistic can also be specified in the **Rules** section within the Project Settings window. To edit the project settings, click the icon in the upper right corner of the window and click **Project settings**. Modifications to the project settings are applied to all applicable areas throughout the project.

4 Close the **Results** window.

---

### Interpretability of Champion Model

Now that you have selected the Logistic Regression model as the Champion Model, you want to bring to management more information about the interpretation of the model. There are several model interpretability plots and reports that can be generated. More details about the model interpretability plots available in Model Studio can be found in **Model Interpretability Plots**.

1 Click the **Logistic Regression** node.

2 Expand the **Global Interpretability** options and select **Variable Importance** and **PD plots**.

3 Expand the **Local Interpretability** options and select **ICE plots** and **LIME**.

4 Right-click the **Logistic Regression** node and select **Run**.

5 Once the **Logistic Regression** node has successfully run, right-click the node and select **Results**. Explore the following on the **Model Interpretability** tab:

- **Surrogate Model Variable Importance**

  ```
<table>
<thead>
<tr>
<th>Variable Label</th>
<th>Variable Name</th>
<th>Relative Importance</th>
<th>Variable Level</th>
</tr>
</thead>
<tbody>
<tr>
<td>Replacement:</td>
<td>REP_RECENT_CARD_RESPONSE_PROS</td>
<td>1</td>
<td>INTERVAL</td>
</tr>
<tr>
<td>RECENT_CARD_RESPONSE_PROP</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Replacement:</td>
<td>REP_MEDIAN_HOME_VALUE</td>
<td>0.4930</td>
<td>INTERVAL</td>
</tr>
<tr>
<td>MEDIAN_HOME_VALUE</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Replacement:</td>
<td>REP_FREQUENCY_STATUS_97%</td>
<td>0.2309</td>
<td>NOMINAL</td>
</tr>
<tr>
<td>FREQUENCY_STATUS_97%</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Replacement:</td>
<td>REP_NUMBER_PROM_12</td>
<td>0.1725</td>
<td>INTERVAL</td>
</tr>
<tr>
<td>NUMBER_PROM_12</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Replacement:</td>
<td>REP_RECENT_CARD_RESOPONSE_PROP</td>
<td></td>
<td></td>
</tr>
<tr>
<td>PD plots</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
  ```

- **Partial Dependence**
Individual Conditional Expectation

Local Explanation for Nominal Variables

Local Explanation for Interval Variables

For more information about the model interpretability plots generated in the Logistic Regression node, see Logistic Regression Results.

Close the Results window.
Publish the Champion Model

Before completing this section, ensure that an administrator has set up a location where you can publish your model. For more information, see SAS Viya Administration: Publishing Destinations.

Publishing a model enables you to execute the model in various run-time engines. You can also manage a published model in SAS Model Manager, if it is licensed. For more information, see Publish Models on page 96.

To publish a model:

1 Navigate to the Pipeline Comparison tab.
2 Ensure that the champion model is selected at the top of the Pipeline Comparison tab.
3 In the upper right corner of the Pipeline Comparison tab, click the icon and select Publish models.
4 The Publish Models window appears. Select the destination where you want your model to be published.
5 Select Publish.
6 The Publishing Results window appears. This window shows the name and published name of your model, as well as the status of your model (publishing, published successfully, and so on).
7 Close the Publishing Results window.

SAS Code Node Examples

Overview

In this section, you use the SAS Code node included with Model Studio to perform two different tasks. In the first example, you create a gradient boosting model with the GRADBOOST procedure. In the second example, you use the FOREST procedure to perform variable selection.

Both examples use the DONOR_DATA data set that is found on the SAS Visual Data Mining and Machine Learning product page. Click Example Data for Getting Started with SAS Visual Data Mining and Machine Learning in Model Studio. Download the ZIP file and extract the contents to a directory that your SAS Visual Data Mining and Machine Learning server can access. This data includes names of those who were solicited for charitable donations and indicates whether they donated.
Download the ZIP file and extract the contents to a directory that your SAS Visual Data Mining and Machine Learning server can access.

Both examples require you to complete the steps in the Create the Project and Import the Input Data and Modify Variables sections. After completing these sections, you can complete the examples in any order.

Create the Project and Import the Input Data

This example assumes that you are signed in to Model Studio, which brings you to SAS Drive. To create the project that you will use in this example:

1. In the upper left corner of the SAS Drive window, click the icon, and select Build Models.

2. Select New Project in the upper right corner of the page.

4 Select **Data Mining and Machine Learning** for **Type**.

5 Ensure that **Blank Template** is specified for **Template**.

6 In the **Data source** field, select **Browse**. The Choose Data window appears.

7 In the upper left corner of the Choose Data window, select **Import**.

8 Select **Local File**, and navigate to the folder where DONOR_DATA is stored. Select **DONOR_DATA.sas7bdat** and click **Open**.

9 Click **Import Item** in the upper right corner of the Choose Data window. When the data is successfully imported, a note appears, saying that the data is ready for use.
Once the data set is successfully imported, click OK in the lower right corner of the Choose Data window. This brings you back to the New Project window.

In the lower left corner of the New Project window, click Save. You are redirected to the Data tab, where you can modify the variables in your data set.

Modify Variables

On the Data tab, variable roles are indicated in the Role column. To change the role of a variable:

1. Select a variable by clicking the corresponding check box to the left of the Variable Name column. The options panel for the selected variable appears to the right of the variable table on the Data tab. Select the TARGET_D variable.

   TARGET_D

   Role:
   Target

   Level:
   Interval

   Order:

   Transform:

   Impute:

   Lower limit:
   Enter a decimal value

   Upper limit:
   Enter a decimal value

2. Expand the drop-down list under Role, and select the role type that you want to assign to the selected variable. Changes made to each variable are automatically applied and saved. For TARGET_D, select the role Rejected.
CAUTION
To avoid making unwanted changes to variable properties, you must manually
deselect each variable that you modify when you are finished making changes to its
properties.

3 Ensure that the variables roles are set to the following values:
   - CLUSTER_CODE is set to Rejected
   - CONTROL_NUMBER is set to ID
   - TARGET_B is set to Target
   - TARGET_D is set to Rejected
   - _dmIndex_ is set to Key.
   - _PARTIND_ is set to Partition.
   In data mining, a strategy for assessing the quality of model generalization is
to partition the data source. A portion of the data, called the training data set,
is used for preliminary model fitting. The rest is reserved for empirical
validation and is often split into two parts: validation data and test data. The
validation data set is used to prevent a modeling node from overfitting the
training data and to compare models. The test data set is used for a final
assessment of the model. In this example, the data has been pre-partitioned
where 60% of the data is assigned for training, 30% of the data is assigned
for validation, and 10% of the data is assigned for testing.
   - All other variable roles are set to Input.

4 On the Data pane, select the _PARTIND_ variable. On the options panel, select
Map Partition Levels and ensure that the partition levels are set to the following
values:
   - Training Level is set to 1.
   - Validation Level is set to 0.
   - Test Level is set to 2.

5 Set the property Transform to Log10 for the following variables:
   - FILE_AVG_GIFT
   - LAST_GIFT_AMT
   - LIFETIME_AVG_GIFT_AMT

TIP It is possible to select multiple variables for editing at one time by
selecting the box in front of each variable name. By simultaneously
selecting FILE_AVG_GIFT, LAST_GIFT_AMT, and
LIFETIME_AVG_GIFT_AMT, you can change the Transform property for
all three variables at once.
Create a Gradient Boosting Model

This example requires you to complete the steps in the Create the Project and Import the Input Data and Modify Variables sections. This example also assumes that you have not created any other pipelines before starting this section. To create the gradient boosting model:

1 Select the Pipelines tab in the upper left corner.
2 Right-click the Data node and select Run.
3 Once the Data tab has run successfully, continue with the following sections to build your pipeline.
4 Right-click the Data node and select Add child node \_miscellaneous\ SAS Code.
5 Right-click the SAS Code node and select Move \_ Supervised Learning. A Model Comparison node is automatically added to the pipeline.
6 Select the SAS Code node. On the options panel, click Open Code Editor.
7 In the Training code editor, enter the following code:

```sas
proc gradboost data=&dm_data
   numBin=20 maxdepth=6 maxbranch=2 minleafsize=5
   minuseinsearch=1 ntrees=10 learningrate=0.1 samplingrate=0.5 lasso=0
   ridge=0 seed=1234;
   %if &dm_num_interval_input %then %do;
      input %dm_interval_input / level=interval;
   %end;

   %if &dm_num_class_input %then %do;
      input %dm_class_input/ level=nominal;
   %end;

   %if "&dm_dec_level"="INTERVAL" %then %do;
      target %dm_dec_target / level=interval ;
   %end;
   %else %do;

      target %dm_dec_target / level=nominal;
   %end;

   &dm_partition_statement;
ods output
   VariableImportance   = &dm_lib..VarImp
   Fitstatistics        = &dm_data_outfit
   ;
   savestate rstore=&dm_data_rstore;
run;

%dmcas_report{dataset=VarImp, reportType=Table, description=%nrbquote(Variable Importance)}
```
%dmcas_report{dataset=VarImp, reportType=BarChart, category=Variable, response=RelativeImportance, description=%rquote(Relative Importance Plot)};

This code uses the following Model Studio macros:

- **DM_DATA** — A macro variable that identifies the CAS training table. If partitioned, the table contains the _partInd_ variable that identifies which observations are used for training, validation, and test. This table is transient and is dropped when the node finishes running.
- **DM_NUM_INTERVAL_INPUT** — A macro variable that identifies the number interval input variables.
- **DM_NUM_CLASS_INPUT** — A macro variable that identifies the number of class input variables.
- **DM_DEC_LEVEL** — A macro variable that identifies the measurement level (binary, interval, ordinal, or nominal) of the target variable.
- **DM_PARTITION_STATEMENT** — A macro variable that identifies partition statement. This variable is blank if the data is not partitioned.
- **DM_LIB** — A macro variable that identifies the SAS library where the variable importance table is saved. This table is named VarImp.
- **DM_DATA_OUTFIT** — A macro variable that identifies the fit statistics data set. This is the data set that is used by the model comparison node to select the best model in the pipeline.
- **DM_DATA_RSTORE** — A macro variable that identifies the remote analytic store that is created by the GRADBOOST procedure. This table is used by Model Studio to score and assess the model.
- **%DM_INTERVAL_INPUT** — A macro that identifies the interval input variables.
- **%DM_CLASS_INPUT** — A macro that identifies the class input variables.
- **%DM_DEC_TARGET** — A macro that identifies the project target variable.
- **%DM_REPORT** — A macro that enables the addition of more reports to the Results window.

8 In the upper right corner of the code editor, click the `>` icon.

9 Click **Close**.

10 Right-click the **SAS Code** node and select **Run**.

   There are two **DM_REPORT** calls to display the contents of the variable importance table VarImp.

   - **reportType=Table** — This call adds the Variable Importance table to the results, as indicated in the **DESCRIPTION** argument.
   - **reportType=BarChart** — This call adds a bar chart that contains Relative Importance for each input.

11 Right-click the **SAS Code** node and select **Results**. Review the following results:

   - **Variable Importance**
Relative Importance Plot

Path EP Score Code — Because you specified that the SAS Code node is a supervised learning node, it automatically creates EP Score Code. More specifically, this code produces an analytic store as its score code.

Lift Reports, ROC Reports, Fit Statistics, and Event Classification — These plots are automatically created because the SAS Code node is a supervised learning node. Click Assessment in the upper left corner of the SAS Code Results window to access these plots.
Click Close in the upper right corner to exit the results.
Perform Variable Selection

This example requires you to complete the steps in the Create the Project and Import the Input Data and Modify Variables sections. This example also assumes that you have not created any other pipelines before starting this section.

1. Select the Pipelines tab in the upper left corner.

2. Right-click the Data node and select Run.

3. Once the Data tab has run successfully, continue with the following sections to build your pipeline.

4. Right-click the Data node and select Add child node ➔ Miscellaneous ➔ SAS Code.

5. Select the SAS Code node. On the options panel, click Open Code Editor.

6. In the Training code editor, enter the following code:

```sas
proc forest data=&dm_data
   minleafsize=5 minuseinsearch=1 seed=12345 loh=0 numbin=20
   ntreees=100 maxdepth=20 inbagfraction=0.6 ;
   partition fraction (valid=0.3 seed=12345);
   %if &dm_num_interval_input %then %do;
      input %dm_interval_input / level=interval;
   %end;

   %if &dm_num_class_input %then %do;
      input %dm_class_input/ level=nominal;
   %end;

   %if "&dm_dec_level"="INTERVAL" %then %do;
      target %dm_dec_target / level=interval ;
   %end;

   %else %do;
      target %dm_dec_target / level=nominal;
   %end;
   grow IGR;
   ODS output VariableImportance = &dm_lib..forestvarimportance ;
run;

%dmcas_report(dataset=forestvarimportance, reportType=BarChart,
   category=Variable, response=RelativeImportance,
   sortDirection=descending, sortBy=RelativeImportance,
   description=&nrbquote(Relative Importance Plot));

filename _frf "&dm_file_deltacode";

data _null_;
   length string $200;
   file _frf;
   set &dm_lib..forestvarimportance ;
   where RelativeImportance >=0.3;
```
string = 'if NAME "'!!kstrip(Variable)!!" then ROLE="REJECTED";';
put string;
run;
filename _frf;

%dmcas_report(file=&dm_file_deltacode, reportType=CodeEditor, description=%nrbquote(Metadata Changes));

This code uses the following Model Studio macros:

- DM_FILE_DELTACODE — A macro variable that identifies the file that contains the DATA step code to modify the columns meta information that is exported by the node.

7 In the upper right corner of the code editor, click the [ ] icon.

This code uses PROC FOREST to identify the relative variable importance for all variables in the input data. Those variables with a relative importance less than 0.3 are assigned the role Rejected. All other variables are kept.

8 Click Close.

9 Right-click the SAS Code node and select Run.

10 Right-click the SAS Code node and select Results.

There are two DMCAS_REPORT calls to display the contents of the variable importance table VarImp.

- reportType=BarChart — This call adds the Relative Importance Plot to the results.

- reportType=CodeEditor — This call adds the Metadata Changes information to the results. All of the variables that are dropped from the analysis are listed here.
There is no score code available for this node because it is not a supervised learning node. There are also no assessment plots.

11 Click **Close** to exit the results.

---

**Integrating Model Studio with SAS Visual Analytics**

**Overview**

The example in this section shows the typical process of creating a model in SAS Visual Analytics, copying it to Model Studio, and then continuing your analysis. This example demonstrates the steps to complete this task and you are encouraged to repeat the process with several additional models.

SAS Visual Analytics enables you to create many different types of reports that demonstrate the efficacy of various statistical models. For more information, see *Working with SAS Visual Statistics* in the SAS Visual Analytics 8.4 documentation. Many of the reports created in SAS Visual Analytics can be sent to Model Studio for comparison against other Model Studio models.

In SAS Visual Analytics, you start by identifying the data that you want to model. Next, you can adjust certain characteristics of that data or create new data items. Then, you add one or more objects to the workspace and assign data items to those objects. Objects vary in complexity from simple tables to more complex statistical models.

Reports in SAS Visual Analytics can range from single-page reports that contain a single object to a multi-page reports that contain several dependencies and inter-object connections. However, when you copy a model from SAS Visual Analytics to Model Studio, a four-node pipeline is always created. The four nodes in the pipeline are as follows:

- The **Data** node
- The **Visual Data Preparation** node
- A model node
- The **Model Comparison** node

The data preparation and model nodes are discussed in more detail in this example. Below is a four-node pipeline that has been created from SAS Visual Analytics. In this case, the modeling algorithm used was Support Vector Machine.
What you cannot determine from this pipeline is that the SAS Visual Analytics report that generated the pipeline contained two pages, each containing its own model. This report, which you will re-create, uses a forest model to create inputs that are used by a support vector machine model. The creation of these inputs is captured in the Visual Data Preparation node, even though the Forest node does not appear in the pipeline.

This example uses the HMEQ data set, which contains 5,960 mortgage applications and indicates whether the applicant defaulted on the loan. The HMEQ data set can be found at [http://support.sas.com/documentation/onlinedoc/viya/examples.htm](http://support.sas.com/documentation/onlinedoc/viya/examples.htm). To complete this example, you must follow the proceeding sections in the order in which they appear.

### Download the Sample Data

1. In a web browser, navigate to the [SAS Viya Example Data Sets](http://support.sas.com/documentation/onlinedoc/viya/examples.htm) page.

2. Download the file `hmeq.csv` to your local machine.
Create the Report

This example assumes that you have already signed in to Model Studio, which brings you to SAS Drive.

To create the report:

1. In the upper left corner of the SAS Drive window, click the Explore and Visualize Data icon, and select Explore and Visualize Data. This opens SAS Visual Analytics, and enables you to choose a data source, create a model, or load an existing project in the Welcome to SAS Visual Analytics window.

2. Click the Data button in the lower left corner of the window to load your data. The Open Data Source window appears, enabling you to select the data source for this project.

3. On the Import tab, click Local File. Navigate to the location where you saved hmeq.csv and select hmeq.csv.

4. In the Open Data Source window, click Import Item. After the table is successfully imported, click OK.

5. By default, the report is named Report 1, which is displayed in the upper left corner. Before continuing with the example, rename the project by saving it.

6. In the Data pane, right-click BAD and select Convert to category.

7. Click in the upper right corner of the window to save the project.

Create a Forest

To create a forest:

1. From the left pane, click the icon to select an object. Drag the icon onto the canvas to create a forest.

2. Click in the right pane. For Response, click Add, and select LOAN.

3. For Predictors, click Add, and select every variable except BAD. Click OK.

4. In the Variable Importance plot, right-click and select Derive predicted. In the New Prediction Items window, review the new data items and click OK.

5. Click in the upper right corner of the window to save the project.
Create a Support Vector Machine

To create a support vector machine:

1. Click the icon to add a new page to the report.

2. From the left pane, click the icon to select an object. Drag the icon onto the canvas to create a support vector machine.

3. Click in the right pane. For Response, click Add, and select BAD.

4. For Predictors, click Add, and select CLAGE, DEBTINC, DELINQ, DEROG, and Predicted: LOAN. Click OK.

5. Click to save the project.

6. In the Variable Importance plot, right-click and select Create pipeline Add to new project. This action copies the model and all data preparation steps to Model Studio. Model Studio automatically opens.

Continuing in Model Studio

After creating the project and copying the necessary information from SAS Visual Analytics to Model Studio, you should see a pipeline that resembles the following image. Your Model Studio project is also named Integration Example, as that title is inherited from SAS Visual Analytics.
As discussed earlier, this pipeline contains four nodes. The **Visual Data Preparation** node contains the score code necessary to create the prediction and residual variables that are used as inputs for the support vector machine. From this point, you can modify the pipeline as if it were created in Model Studio. The only restriction is that you cannot edit the properties of the **Visual Data Preparation** node.

1. In the upper left corner, click **Data** to open the **Data** tab. This tab displays all the information that Model Studio knows about the input data set.

2. Notice that SAS Visual Analytics created two new variables: `_dmIndex_` and `_EVENT_`. The original target variable, **BAD**, has been assigned the role **Rejected**. SAS Visual Analytics creates temporary variables as needed to complete the tasks that you want to perform. In this case, changing BAD from a measure to a category necessitated the creation of `_EVENT_`

   These name changes also require careful consideration if you want to use a holdout data set. That holdout data set must have a target variable name that exactly matches the target variable name created by SAS Visual Analytics. The target variable is used for generating assessment statistics, not scoring.

3. Also, notice that all other variables are assigned the role **Input**. This does not match the support vector machine that you created in SAS Visual Analytics. The **Visual Data Preparation** node handles the role assignments when you run the pipeline.
Note: If you change the target variable information or the partition variable information on the Data tab and try to run your pipeline, it will fail. The score that is generated and applied in the Visual Data Preparation node requires the partition and variable information that was known in SAS Visual Analytics when you created the pipeline. After the project is created (the project advisor code is run), you cannot modify any data item on the Data tab. Therefore, if you altered a data item and your pipeline failed, you need to delete your Model Studio project. Open your SAS Visual Analytics project, create a new pipeline, and try any modifications again.

4 In the upper left corner, click Pipelines to open the Pipelines tab.

The Visual Data Preparation node ensures that all subsequent nodes see the data as it existed in SAS Visual Analytics. This node executes all of the model score code from all of the objects in SAS Visual Analytics that were used to create the copied model. This code can be DS1 code, analytic store code, or some combination of the two. The Visual Data Preparation node ensures that this code runs in the proper order and guarantees that subsequent nodes receive the proper information. This means that the Logistic Regression node that you add in the next step sees the data as it existed in SAS Visual Analytics.

5 Right-click the Visual Data Preparation node and select Add child node ➔ Supervised learning ➔ Logistic Regression. This adds a new supervised learning model to the pipeline. The data preparation steps used in SAS Visual Analytics are applied to the Logistic Regression node.

6 Right-click the Logistic Regression node and select Add child node ➔ Postprocessing ➔ Ensemble.

7 Right-click the Ensemble node and select Add Models ➔ SVM. At this point, your pipeline should resemble the following:
8 Right-click the **Model Comparison** node and select **Run**. This action runs all of the nodes preceding the **Model Comparison** node.

9 Right-click the **Model Comparison** node and select **Results**. This brings you to the **Node** tab, which displays the **Model Comparison** table. The table shows you which model is your champion model. Click **Assessment** in the upper left corner of the Model Comparison Results window to see the **Lift Reports** and **ROC Reports** charts, as well as the **Fit Statistics** table.
Review the results computed for each of the three models in the diagram. The **Ensemble** node was chosen as the champion model.

Click **Close** to exit the Results window.

10 Suppose you want to modify the support vector machine model to improve model performance. Right-click the **SVM** node and select **Enable Properties**. Click **Yes** when the **Enable Properties** window appears.

Note: If you enable the properties of a node created in SAS Visual Analytics, the model is retrained, which might yield new results. Once properties have been enabled, the model cannot be switched back to use the original SAS Visual Analytics score code.

11 Click the **SVM** node. In the **Options** panel, select **Perform Autotuning**. Autotuning automatically chooses the optimal values for the hyperparameters used to build your predictive model. For more information, see **Overview of Autotuning**.

12 Right-click the **Model Comparison** node and select **Run**. This action runs all of the nodes preceding the **Model Comparison** node.

13 Right-click the **Model Comparison** node and select **Results**. Notice the improved misclassification rate of the **SVM** node model.

14 Click **Close** to exit the Results window.
Overview

In this section, you use the **Open Source Code** node to create a Random Forest model in R. You also compare this model with a **Logistic Regression** node. Though a champion is chosen, the intent of the example is not to build the best model. Rather, this example demonstrates how models from Python or R are executed and compared in Model Studio. This example requires the randomForest package in R.

This example uses the HMEQ data set, which has a binary target that indicates whether a mortgage loan defaulted. The inputs include information such as loan amount, reason for loan, years at present job, and debt-to-income ratio.

If you are unsure about whether the HMEQ data set is available on your system, navigate to the **SAS Viya Example Data Sets** page and download **hmeq.csv**. Note where you saved this file.

Create the Project and Import the Input Data

This example assumes that you are signed in to Model Studio, which brings you to SAS Drive. To create the project that you use in this example:

1. In the upper left corner of the SAS Drive window, click the **Build Models** icon, and select **Build Models**.
2. Select **New Project** in the upper right corner of the page.

4 Select Data Mining and Machine Learning for Type.

5 Ensure that Blank Template is specified for Template.

6 In the Data source field, select Browse. The Choose Data window appears.

7 If HMEQ is listed on the Available tab of the Choose Data window, select the HMEQ data set and click OK.
   If the HMEQ data set is not listed on the Available tab, import it:
   a Click the Import tab.
   b Click Local File.
   c In the Open window, navigate to the location where you saved the HMEQ data set and select hmeq.csv.
   d Click Open.
   e In the upper right corner, click Import Item.
   f After the data set is successfully imported, click OK.

8 Click the Advanced button below the Description text box, and the New Project Settings window appears. Select Partition Data in the upper left corner of the window.
9 Ensure that the Create partition variable option is selected, and click Save in the lower right corner of the window. This brings you back to the New Project window.

10 In the lower right corner of the New Project window, click Save. You are redirected to the Data tab, where you can modify the variables in your data set.

Modify Variables

On the Data tab, variable roles are indicated in the Role column. To change the role of a variable:

1 Select a variable by clicking the corresponding check box to the left of the Variable Name column. The options panel for the selected variable appears to the right of the variable table on the Data tab.

2 Expand the drop-down list under Role, and select the role type that you want to assign to the selected variable. Changes made to each variable are automatically applied and saved.

**CAUTION**
To avoid making unwanted changes to variable properties, you must manually deselect each variable that you modify when you are finished making changes to its properties.

3 Using the steps above, adjust the property of Role for each of the following variables:
   - Set BAD to Target.
   - Ensure that all other variables are set to Input.
Create the Pipeline

This example requires you to complete the steps in the previous sections. This example also assumes that you have not created any other pipelines before starting this section. To create the pipeline that contains an open-source model:

1. Navigate to the **Pipelines** tab. This tab should contain a single pipeline with only a **Data** node.

2. Right-click the **Data** node and select **Add child node** ➔ **Supervised Learning** ➔ **Logistic Regression**.

3. Right-click the **Data** node and select **Add child node** ➔ **Data Mining Preprocessing** ➔ **Imputation**. This ensures that all missing data is imputed because some open-source packages cannot handle missing data.

4. Right-click the **Imputation** node and select **Add child node** ➔ **Miscellaneous** ➔ **Open Source Code**.

5. Right-click the **Open Source Code** node and select **Move** ➔ **Supervised Learning**. This ensures that the node performs model assessment and can be compared to the **Logistic Regression** node.

Your current pipeline should resemble the following image.
6 Select the **Open Source Code** node. In the options panel, set the value of **Language** to **R**.

7 In the options panel, click **Open Code Editor**. Enter the following code in the code editor:

```r
library(randomForest)

# RandomForest
dm_model <- randomForest(dm_model_formula, ntree=100, mtry=5, data=dm_traindf, importance=TRUE)

# Score
pred <- predict(dm_model, dm_inputdf, type="prob")
dm_scoreddf <- data.frame(pred)
colnames(dm_scoreddf) <- c("P_BAD0", "P_BAD1")

# Print/plot model output
png("rpt_forestMsePlot.png")
plot(dm_model, main='randomForest MSE Plot')
dev.off()

# Print Variable Importance
write.csv(importance(dm_model), file="rpt_forestIMP.csv", row.names=TRUE)
```
8 In the upper right corner of the code editor, click the icon.

9 Click Close.

10 Right-click the Model Comparison node and select Run.

11 After it has successfully run, right-click the Open Source Code node and select Results. Expand the R code results to view the actual code that was generated by Model Studio and submitted. Notice that this code is a combination of precursor, user, and posterior code. The precursor and posterior code is added based on the node properties and whether the node is in the Preprocessing or Supervised Learning group.

On the Assessment tab, notice that assessment measures such as lift and ROC were computed for the open-source model.

Close the Results window.

12 Right-click the Model Comparison node and select Results. The open-source model was chosen as the champion model based on having a better Kolmogorov-Smirnov statistic.

The Assessment tab lets you compare the results of the open-source model against the logistic regression model that you also created.

13 Close the Results window. The example is now complete.
Managing Projects

Overview of Model Studio Projects

A project is a top-level container for your analytic work in Model Studio. You can view projects in the Model Studio Projects page.

Model Studio projects can be one of three types: Forecasting projects, Data Mining and Machine Learning projects, and Text Analytics projects. The project types that appear in your Model Studio installation depend on the SAS licensing for your site.

Depending on your project filter setting, existing projects in your environment appear either as graphic tiles or rows in a table of projects.
To alternate between table and tile project displays:

- Click the icon near the top of the page to show existing Model Studio projects in a graphic tile matrix.
- Click the icon near the top of the page to show Model Studio projects in a tabular list.

A Model Studio project contains the data source, the pipelines that you create, and related project metadata (such as project type, project creator, share list, and last update history). If you create more than one pipeline in your project, analytic results that compare the performance of multiple pipelines are also stored in the project.

Click the icon in the lower right corner of the project tile to view additional information about the project.
Opening an Existing Project

You use the Model Studio Projects page to access existing projects. If your Model Studio Projects page displays project tiles, click the tile that you want to open. Model Studio opens the selected project.

If your Model Studio Projects page displays a project table, click the row that contains the desired project to open it. Model Studio opens the selected project to the last visited tab.
Creating a New Project

You create new Model Studio projects from the Projects page. To create a new project:

1. Click the **New Project** button in the upper right corner.

2. The New Project window appears.

   ![New Project window](image)

   Enter a name for your new project in the **Name** field.

3. Select a project type from the **Type** list. The choices are **Forecasting**, **Data Mining and Machine Learning**, and **Text Analytics**.

4. Select a template to use for your pipeline. **Templates** are pipelines that are pre-populated with configurations that can be used to create models quickly. The default template is the **Blank template**. The **Blank template** is the baseline pipeline that is pre-populated with only the **Data** node. For more information about the templates available in Model Studio, see **Available Templates on page 81**.
5 You must identify the data source that you want to use. Select the Browse button to open the Choose Data window. Use the Choose Data window to select your data source and click OK. For more information, see Getting Started with the Choose Data Window.

6 If you want to enter information about the project that might be useful to others, enter that content in the Description field.

7 Click the Advanced button to specify additional project creation options. In the New Project Settings window, you can specify the Advisor Options for missing variables, Partition Data settings for system-created partition variables, and Event-based Sampling options to perform over-sampling. There are also Node Configuration options where you can specify configuration code in Python. This Python code is automatically prepended to every Open Source Code that you include in your pipeline when the Language property is set to Python. The Python code can be a maximum of 4000 characters.

8 Click the Save button to create your new project using the name, project type, and data source name that you specified.

After you create your new project, Model Studio takes you to the Data tab of your new project page. Here, you can make adjustments to data source variable names, labels, type, role, and level assignments. For more information about the Data tab, see the Data Management Overview section.

Sharing a Project

After creating a project, you can share it with others in your organization. Model Studio enables you to share projects with user-defined groups.

The Model Studio implementation of sharing is distinct from project sharing as performed in SAS Drive. Any projects that you share using SAS Drive do not retain the same settings for user groups in Model Studio. Also, any projects that you share using Model Studio do not retain the same settings for users in SAS Drive. For more information about the authorization service, see SAS Viya Administration: General Authorization. For more information about SAS Drive, see SAS Drive: Getting Started.

To share a project:

1 Select the desired project by clicking the check box in the project tile, and then click the icon next to the Project page Toolbox.
2 Select Share.

3 The Share Project window appears.

4 Select Share project.

5 Configure the groups by clicking the + icon. Use the Choose Groups window to select which groups you want to share access with.
Once groups have been configured, click **OK**.

6. By default, group members can modify the shared project. To disable this feature, select **Read-Only**.

Note: The following features apply to shared projects:

- Only the owner of a shared project can change shared status of that project.
- Only the owner of a shared project can delete that project.
- If a project is not shared in **Read-Only** mode, then only one person can have the project open at a time. Shared projects that are currently open are indicated with a 📑 icon on the Projects page.
- If a project is shared in **Read-Only** mode, nobody can make changes to the project, including the project owner.
- SAS Administrators must be included in any group that the project is shared with.

7. Once the configurations are set on the Share Project window, click **OK** to share. You can see that your project has been shared on the project tile.
You can also remove sharing of a project. To do this, repeat steps 1 through 3 above, but in the Share Project window select **Private project** and click **OK**. This removes shared access to the project.

**Importing and Exporting a Project**

The source data for a project must already be loaded to the CAS server before you import the project.

To export a project:

1. On the Projects page, select the project that you want to export.
2. Click the **!** icon and select **Export**.

The project files will immediately begin to download. SAS Visual Data Mining and Machine Learning projects are stored as JSON files.

To import a project:

1. Click the **!** icon and select **Import**. If you also have SAS Visual Forecasting or SAS Visual Text Analytics installed, select **Import ⇒ Data Mining and Machine Learning**.
2. In the Import Data Mining Project window, specify the location of the project and an associated data set. When you import a project, you must specify the ZIP file that was saved when you exported the original project.
3. Click **Import**.
Deleting a Project

To delete a project:

1. Select the desired project by clicking the check box in the project tile, and then click the icon next to the Project page Toolbox.

2. Select Delete.

3. The Delete window appears, asking for confirmation of deletion.

4. Click Delete.

Downloading Project Batch API Code

To download project batch API code:
1. Select the desired project by clicking the check box in the project tile, and then click the icon next to the Project page Toolbox.

2. Select **Download Batch API**.

3. The Batch API window appears, which displays the batch API code for the project. Batch API code can be given in the following types:
   - **SAS** — Downloads code to run in a SAS programming environment.
   - **Python** — Downloads code to run in a Python programming environment.
   - **REST** — Downloads a text file with example REST calls that you can use in an application.

4. Select the download type for the batch code, and click **Download**. The code will begin downloading immediately.

To run the batch code, supply the host name of the SAS Visual Data Mining and Machine Learning server and the user name and password.
SAS
Update the host, username, and password macro variables at the end of the code.

```sas
%let protocol = http;
%let host = test.example.com;
%let port = 80;
%let username = my_username;
%let password = my_password;
%let projectId = a0548b2f-a669-4a10-a4dd-052c671c0c00;
```

Python
Set these parameters on the command line. For example, if `download.py` is the file name of the Python batch code, issue the command:

```shell
$ python download.py --host test.example.com --username my_username --password my_password
```

If you have overrides in the project, they might generate conflicts when they are submitted by the batch code. You can update the batch code to automatically resolve these conflicts by adding `autoresolve="true"` to the code.

SAS
Find the PROC HTTP procedure that includes the following IN option with `firstTransaction` and `lastTransaction`.

```sas
in="{"firstTransaction":"@first","lastTransaction":"@last"}"
```

Add the `autoResolve` setting, as follows:

```sas
in="{"firstTransaction":"@first","lastTransaction":"@last","autoResolve":true}"
```

Python
Find the `resubmit_overrides` function:

```python
def resubmit_overrides(env):
    Update the body statement with `autoResolve` assignment.
    body = '{"firstTransaction":"@first","lastTransaction":"@last","autoResolve":true}
```

---

**Specifying Global Settings**

When setting up your Model Studio account, you might want to modify global settings for your account instance. To edit global settings:

1. In the upper right corner of the window, click the user name button, and select **Settings**.
2. The Settings window appears, enabling you to alter global settings.
3 For changing global settings, the following options are available:

- **General** — These settings enable you to set your interface theme, reset messages, and choose a profile picture. To return all settings to their default values, click **Reset**.

- **Region and Language** — These settings enable you to set locales for your browser and Java Runtime Environment. To return all settings to their default values, click **Reset**.

- **Accessibility** — These settings enable you to enable sounds and visual effects for your interface. You can also adjust the display duration for pop-up notifications, invert colorings, display tooltips, and customize the focus indicator. To return all settings to their default values, click **Reset**.

4 For changing Model Studio project settings at a global level, the following options are available for SAS Data Mining users:

- **General** — Select **Reset Column Preferences** to reset table column preferences to their default values.

- **Advisor options** — These settings enable you to specify settings such as the maximum number of class level values and the maximum percentage of missing values. Another option in this group enables you to control the interval cutoff value for numeric variables. If the number of unique levels
exceeds the specified cutoff value, the variable is assigned the measurement level interval. To return all settings to their default values, click **Reset**.

- **Partition Data** — These settings enable you to partition the data set into subsets used for training, validation, and test. You can also specify the partition method. By default, the data is partitioned as follows:

<p>| | |</p>
<table>
<thead>
<tr>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>Training</td>
<td>60%</td>
</tr>
<tr>
<td>Validation</td>
<td>30%</td>
</tr>
<tr>
<td>Test</td>
<td>10%</td>
</tr>
</tbody>
</table>

To return all settings to their default values, click **Reset**.

*Note:* The **Partition Data** settings are active only when a partition variable is not set within the data. Using a data source with a pre-defined partition variable or manually selecting a partition variable overrides these settings. If a variable is assigned the role **Partition** and you modify that variable’s role, these settings remain inactive.

- **Event-based Sampling** — These settings enable you to enable event-based sampling for the model, specifying event and non-event percentages. By default, event-based sampling is disabled. If enabled, the **Event** and **Non-Event** percentages are both 50% by default. To return all settings to their default values, click **Reset**.

- **Node Configuration** — These settings enable you to specify configuration code in Python. This Python code is automatically prepended to every Open Source Code that you include in your pipeline when the **Language** property is set to **Python**. To return all settings to their default values, click **Reset**.

*Note:* The Python code can be a maximum of 4000 characters.

- **Rules** — These settings enable you to specify the rules used for comparing pipeline models, the default binary classification cutoff value, and the number of cutoff values to use for the ROC assessment charts. For more information, see [Overview of Model Comparison](#) in the Model Studio reference documentation. To return all settings to their default values, click **Reset**.

- **Logging** — These settings enable you to enable debug reporting, including options to resolve macro variables, add timings and headers, and retain temporary tables. To return all settings to their default values, click **Reset**.

5 Once the settings are appropriate configured, click **Close**. Settings are automatically saved.
Specifying Project Settings

For certain Model Studio projects, you might need to modify project settings to set up models properly. To edit project settings:

1. Open the project, and then click the  
icon in the upper right corner of the window, under the user name button, and click Project settings.

The Edit Project Settings window contains several properties that might need to be altered for your projects. The following options are available:

- **Partition Data** — These settings enable you to partition the data set into subsets used for training, validation, and test. You can also specify the partition method. By default, the data is partitioned as follows:

<p>| | |</p>
<table>
<thead>
<tr>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>Training</td>
<td>60%</td>
</tr>
<tr>
<td>Validation</td>
<td>30%</td>
</tr>
<tr>
<td>Test</td>
<td>10%</td>
</tr>
</tbody>
</table>

2. The Edit Project Settings window contains several properties that might need to be altered for your projects. The following options are available:

- **Partition Data** — These settings enable you to partition the data set into subsets used for training, validation, and test. You can also specify the partition method. By default, the data is partitioned as follows:
To return all settings to their default values, click **Reset**.

**Note:** The **Partition Data** settings are active only when a partition variable is not set within the data. Using a data source with a pre-defined partition variable or manually selecting a partition variable overrides these settings. If a variable is assigned the role **Partition** and you modify that variable’s role, these settings remain inactive.

- **Event-Based Sampling** — These settings enable you to enable event-based sampling for the model, specifying event and non-event percentages. By default, event-based sampling is disabled. If enabled, the **Event** and **Non-Event** percentages are both 50% by default. To return all settings to their default values, click **Reset**.

- **Node Configuration** — These settings enable you to specify configuration code in Python. This Python code is automatically prepended to every **Open Source Code** that you include in your pipeline when the **Language** property is set to **Python**. To return all settings to their default values, click **Reset**.

  **Note:** The Python code can be a maximum of 4000 characters.

- **Rules** — These settings enable you to specify the rules used for comparing pipeline models, the default binary classification cutoff, and the number of cutoff values to use for the ROC assessment charts. For more information, see **Overview of Model Comparison** in the Model Studio reference documentation. To return all settings to their default values, click **Reset**.

- **Output Library** — These settings enable you to specify the library for output from the model. To return all settings, to their default values click **Reset**.

- **Logging** — These settings enable you to enable debug reporting, including options to resolve macro variables, add timings and headers, and retain temporary tables. To return all settings to their default values, click **Reset**.

3 Once the settings are appropriately configured, click **Save**.

**Note:** The settings configured at the project level override any global settings that you configured for your Model Studio instance.

---

**Importing a Project from SAS Visual Analytics**

Model Studio users can use their SAS Visual Analytics reports to create projects. The following SAS Visual Analytics objects can be used to generate pipelines in Model Studio:
Bayesian Network
Decision Tree
Forest
Generalized Linear Model
Gradient Boosting
Linear Regression
Logistic Regression
Model Comparison
Neural Network
Support Vector Machine

The SAS Visual Analytics objects listed above correspond directly to nodes in Model Studio. To export an object:

1. In SAS Visual Analytics, click the **Create Pipeline** button in the upper left corner of the object canvas, and select either **Add to new project** or **Add to existing project**.

2. If **Add to new project** is selected, the Model Studio interface opens, and a project is created in Model Studio. The name will match the name of the saved SAS Visual Analytics report. If the report in SAS Visual Analytics has no name, the project is named **Interactive Project** in Model Studio. If **Add to existing project** is selected, the Model Studio interface opens, and you are prompted to select an existing project. Only Model Studio projects where the data node has previously been run and where the target variable name, type, and event level match will be available. A pipeline is then created, and named **Interactive-Model Pipeline**.

![Diagram of Model Studio with Interactive Project pipeline](image-url)
This pipeline looks identical to a normal Model Studio pipeline, except for two key differences:

- Model Studio generates a Data Mining Preprocessing node called **Visual Data Preparation**. The **Visual Data Preparation** node runs the score code necessary to perform all the data preparation steps that were performed in SAS Visual Analytics. The properties of this node are not available to edit.

- The SAS Visual Analytics objects are also represented. Here, **Forest** corresponds to the Forest object in SAS Visual Analytics. Supervised learning nodes can be edited in Model Studio. Right-click the supervised learning node, and then select **Enable properties**. If you enable the properties of a node, the model is retrained, which might yield new results. Once properties have been enabled, the model cannot be switched back to use the original SAS Visual Analytics score code. **Model Interpretability** properties do not require retraining the model, so they are automatically enabled and do not affect the original SAS Visual Analytics score code. This pipeline can be run as-is, or nodes can be added to the pipeline to be run for comparison purposes.
Data Management Overview

Model Studio provides several options for managing and modifying data. The Data tab enables you to modify variable assignments and manage global metadata. You can also retrain a model with new data, as long as the target variable in the new data set is the same as the original data set.

Note: Typically, the column names of the input data are not translated. However, certain system-generated UI labels and column names are automatically translated when you are using Model Studio in a non-English language. If your input data contains a column name that matches one of these system-generated UI labels or column names, you might encounter unexpected translations of the column names of your input data.

Importing Data

To add a new data set to the repository for use in a new project:

1. On the Model Studio Projects page, select New Project in the upper right corner.
2 The New Project window appears.

3 Click **Browse** to open the Choose Data window.

4 The Choose Data window appears. Select the **Import** tab.

5 Drag the desired local data set directly to the window. Model Studio parses the data set and pre-populates the window with data set configurations.
6 After configuring the data set import properties, click Import Item. After successful import, the following message appears:

Click OK.

To finish creating a project, follow the steps outlined in Creating a New Project on page 54.

Note: Data set names and variable names can have a maximum length of 32 characters.

User-Defined Formats

Your data sets might include formats that are not natively supported by SAS Visual Data Mining and Machine Learning. To enable SAS Visual Data Mining and Machine Learning to recognize these formats:

1 Upload your format to a CAS format library.

2 Move your format to the compute server machine. If the formats are Windows formats and the machine is UNIX, then you need to use PROC CPORT and PROC CIMPORT to move the format. For more information, see the PROC CPORT and PROC CIMPORT documentation in Base SAS 9.4 Procedures Guide.

3 Add the format to the compute server session search path. For example, you can add the following code to the autoexec_usermods.sas autoexecutable SAS file in /opt/sas/viya/config/etc/compsrv/default:
Retraining Model Data

To retrain a model with new data:

1. Open a project. On the Data tab, click the icon to open the Data Sources pane.

2. In the upper left corner of the Data Source pane, click the icon.

3. The Choose Data window appears. If the desired data set has already been added to the Model Studio repository, select it from the list on the Available tab. If the desired data set is local to your environment, select the Import tab, and follow the instructions in Importing Data on page 69.

4. Click OK. The Data tab now displays details about the new data set.

---

Note: To retrain a model with a new data set, the new data set must use the same target variable as the original data set. For more information, see Managing Global Metadata on page 76.

Managing Variable Assignments

Assigning Variable Metadata

To specify variable properties:

1. On the Data tab, select the desired variables.

2. The right pane enables you to specify several properties of the variables, including the following:
   - Role
   - Level
   - Order
   - Transform
   - Impute
   - Lower Limit
Upper Limit

For the Transform, Impute, Lower Limit, and Upper Limit properties, altering these values in the Data tab does not directly modify the variable. Instead, this sets metadata values for these properties. The Data Mining Preprocessing nodes that use metadata values (Transformations, Impute, Filter, and Replacement) might use these parameters if the corresponding action is requested.

Assigning Variable Metadata Details

The following options are available when specifying variable roles:

- **Assessment** — Specifies that the variable be used for decision processing. The Assessment role is currently not used in Model Studio.

- **Classification** — Specifies that the variable be used for model classification for a class target. For example, if the variable BAD is set as the target variable, the I_BAD variable has the 0 or 1 prediction based on the predicted probabilities and the cutoff used. The classification cutoff is applied only to binary targets.

- **Filter** — Specifies that the variable be used for filtering. For variables with the role of Filter, observations are filtered out when the value = 1 and kept when the value = 0. The Filter role is used in the Filtering node and the Anomaly Detection node.

- **ID** — Specifies that the variable is an ID variable.

- **Input** — Specifies that the variable be used as an input variable in your pipeline.

- **Key** — Specifies that the variable is a unique identifier for all observations. The Key role is used by the Text Mining node and is used in the generation of the observation-based Model Interpretability reports.

- **Offset** — Specifies that the variable is a numeric variable that is used by the GLM node. An offset variable is typically used for a covariate with a known slope. The variable specified is not estimated and is added directly to the model.

- **Partition** — Specifies that the variable be used for partitioning your data set.

- **Prediction** — Specifies that the variable is used during model assessment. This variable is the prediction for an interval target or the posterior probabilities for a class target.

- **Rejected** — Specifies that the variable be excluded from all analysis in your pipeline.

- **Residual** — Specifies that the variable is an error residual. This role is used only for informational purposes.

- **Segment** — Specifies that the variable is a segment variable. Segment variables are created by the Clustering node for cluster IDs created. The segment variable is also used in the Segment Profile node.

- **Target** — Specifies that the variable is the target variable.

- **Text** — Specifies that the variable is a text variable. The Text role is used by the Text Mining node.
- **Time ID** — Specifies that the variable is a time variable. This role is used only for informational purposes.

Nominal variables that are assigned the role **Target** also have the option **Specify the Target Event Level**. This enables you to choose which level to assign as the event level.

The variable role **Partition** can be assigned only to variables with three or fewer than unique levels. You can select **Map Partition Levels** to choose which levels to assign to the Training, Validation, and Test partitions.

---

**Note:** Frequency variables are not supported.

---

**Note:** Interval variables that have a DATE, DATETIME, or TIME format can be assigned only the variable roles **ID** and **Rejected**.

---

- The following options are available when selecting variable levels:
  - Binary
  - Interval
  - Nominal
  - Ordinal

- The following options are available when selecting variable order:
  - Ascending
  - Default
  - Descending
  - Formatted Ascending
  - Formatted Descending

- The following options are available when selecting variable transformations:
  - For class variables, the options are as follows:
    - (none)
    - Bin rare nominal levels
    - Default
    - **Level count encoding** — Encodes the levels of a class input variable with the frequency of the level.
    - **Level encoding** — Encodes the levels of a class input variable by sorting the levels in alphabetical order and then assigning numbers to each level in ascending order.
    - **Level proportion encoding** — Encodes the levels of a class input variable with the proportional frequency of the level.
    - **Target encoding** — Encodes the levels of a class input variable with the event proportion of the level.
    - **WOE encoding** — Encodes the levels of a class input variable with the weight of evidence of the event for that level.

The **Bin rare nominal levels** option is available only for nominal level variables.
For interval variables, the options are as follows:

- (none)
- Best
- Bucket binning
- Centering
- Default
- Exponential
- Inverse
- Inverse square
- Inverse square root
- Log
- Log10
- 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 Squared test for the target variable.

---

**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.

---

The following options are available when you are selecting how to impute missing variable values:

For class variables, the options are as follows:

- (none)
- Cluster count
- Count
- Custom constant value
- Default
- Default constant value
- Distribution

**Cluster count, Count, and Distribution** are not available for unary variables.

For interval variables, the options are as follows:

- (none)
- Cluster mean
Custom constant value
Default
Default constant value
Maximum
Mean
Median
Midrange
Minimum
Trimmed maximum
Trimmed mean
Trimmed midrange
Trimmed minimum
Winsorized maximum
Winsorized mean
Winsorized midrange
Winsorized minimum

You can specify a **Lower limit** and an **Upper limit** to use for the **Metadata limits** method in the **Filtering** node or the **Replacement** node.

---

**Managing Global Metadata**

In Model Studio, *metadata* is defined as the set of variable roles, measurement levels, and other configurations that apply to your data set.

When creating multiple projects using similar data sets (or when using a single data set), you might find it useful to store the metadata configurations for usage across projects. Model Studio enables you to do this by collecting the variables in a repository called Global Metadata. By storing your metadata configurations as global metadata, the configurations apply to new data sets that contain variables with the same name.

To save a variable as global metadata:

1. **On the Data tab**, select the desired variables.
2. **In the right pane**, set the desired variable metadata assignments.
3. **Click the icon** in the upper right corner of the right pane.
4. **A window appears**, confirming that the operation was successful.

To examine and manage the variables designated as global metadata:

1. **Click the icon** in the upper left corner of the window.
2. **The Global Metadata window appears.**
This window displays a table that contains all variables specified as global metadata, as well as their metadata assignments.

3 To remove a variable from the global metadata repository, select the desired variable and click the icon in the upper right corner of the window.

Integration with SAS Visual Analytics

SAS Visual Analytics enables you to transfer certain analytical models from SAS Visual Analytics to Model Studio. To move a model from SAS Visual Analytics to Model Studio, click the Create pipeline button.

This action creates a new project in Model Studio that contains the following elements:

- the active data set
- score code to apply all data processing, filtering, and transformations
- score code to run the model that was exported

The properties of the modeling nodes can be edited, and subsequently models can be retrained in Model Studio. Right-click the modeling node, and then select Enable properties to edit the node. Model interpretability properties are always automatically enabled. You can add and delete nodes in this pipeline as in any other Model Studio pipeline. You can use the Model Studio model comparison and pipeline comparison tools to evaluate your transferred models against any new models.

At this time, the supported models are Bayesian Network, Decision Tree, Forest, Generalized Linear Model, Gradient Boosting, Linear Regression, Logistic Regression, Neural Network, and Support Vector Machine. There also exist exceptions within these models:

- You cannot copy a Decision Tree with a binned measure response.
- You cannot copy a Generalized Linear Model, Linear Regression, or Logistic Regression that uses a frequency, weight, offset, or group by variable.
- You cannot copy a Logistic Regression with a non-binary response variable.
- You cannot copy a Neural Network with a weight variable.

There are a few caveats to note when transferring a model from SAS Visual Analytics to Model Studio.

- In order to add a SAS Visual Analytics model to an existing Model Studio project, the target variable name, type, and event level must match. The data node in the existing Model Studio project must have also been previously run.
Instead of using the variable name that exists in the original data set, SAS Visual Analytics prefers to use the variable label. However, Model Studio prefers to use the variable name as it exists in the original data. Therefore, if the variable names and variable labels in your input data are different, you might experience some unexpected naming issues when a model is transferred. Model Studio displays both the variable name and the variable label in the Variables table layout of the Data pane.

SAS Visual Analytics creates a custom name for target variables. This new variable is indicated with a label in Model Studio.

When you are exporting from the Model Comparison object, only the champion model is exported.

Partition variables must be numeric variables that contain only the values 0 for training data, 1 for validation data, and 2 for testing data. The testing data can be omitted. If the partition information from SAS Visual Analytics differs from Model Studio, then the partitioning of the Model Studio project is used.

Category target variables cannot contain any special characters, including a comma, semicolon, open parenthesis, or close parenthesis. Special characters in the target variable of a Model Studio pipeline cause model creation to fail.

You cannot transfer a model from Model Studio to SAS Visual Analytics. However, you can copy the input data to SAS Visual Analytics for exploration and visualization.

Certain actions that create a data item in SAS Visual Analytics are performed in the Visual Data Preparation node in Model Studio. For example, when you derive a cluster ID in SAS Visual Analytics, a data item is created in the Data pane. If you specify this created data item in a SAS Visual Analytics model that is transferred to Model Studio, it does not appear in the Data pane of your project. Instead, it is re-created when the Visual Data Preparation node runs.

If you enable the properties of a transferred SAS Visual Analytics modeling node in Model Studio, the model is retrained, which might yield new results. Once properties have been enabled, the model cannot be switched back to use the original SAS Visual Analytics score code. Model Interpretability properties do not require retraining the model. Therefore, they are automatically enabled and do not affect the original SAS Visual Analytics score code.

You cannot save your pipeline as a template to the exchange.
Working with Templates

Overview of Templates

Model Studio supports templates as a method for creating statistical models quickly. A template is a special type of pipeline that is pre-populated with configurations that can be used to create a model. A template might consist of multiple nodes or a single node. Model Studio includes a set of templates that represent frequent use cases, but you can also create models themselves and save them as templates in the toolkit.

Creating a New Template from a Pipeline

To create a template from a pipeline:

1. Click the icon next to the pipeline tab in the upper left corner of the canvas.
2. Select Save to The Exchange.
3. In the Save Pipeline to The Exchange window, enter a Name and Description for the new template.
4. Click Save.

You can also create templates from singular nodes. To create a template from a node:

1. Right-click on the desired node. Select Save As. The Save Node to The Exchange window appears.
2 In the Save Node to The Exchange window, enter a **Name** and **Description** for the new template.

3 Click **Save**.

---

**Creating a New Template in The Exchange**

1 Click the 🗯 icon in the upper left corner of the screen. The Exchange page opens. This page enables you to examine all available templates. The Exchange stores node and pipeline templates for SAS Visual Data Mining and Machine Learning, SAS Visual Text Analytics, and SAS Visual Forecasting applications.

2 To create a new template, select the existing template most similar to your desired template. You will duplicate and modify this template.

3 Click the ☝️ icon in the upper right corner of the screen and select **Duplicate**

4 The Save Node to The Exchange window appears. Enter a name and a description for the new template.

5 Click **Save**. Your new template appears in the list of templates.

---

**Modifying an Existing Template**

If you have sufficient permissions, you can modify existing templates. To modify a template:

1 Click the 🗯 icon in the upper left corner of the screen.

2 The Exchange page opens. This page enables you to examine all available templates. The Exchange stores node and pipeline templates for SAS Visual Data Mining and Machine Learning, SAS Visual Text Analytics, and SAS Visual Forecasting applications.
To access a particular template, click the template name. This opens the Node Template or Pipeline Template window. If you do not have Edit privileges for a given template, you will see (Read-Only) displayed in the window.

In the Node Template or Pipeline Template window, you can make changes and configure the nodes in the pipeline. Changes are saved automatically to the template.

Note: While you are editing a template, nodes can be re-configured, but no nodes can be added or deleted.

---

### Available Templates

The following Node templates are included with Model Studio:

<table>
<thead>
<tr>
<th>Node Name</th>
<th>Node Description</th>
<th>Product</th>
</tr>
</thead>
<tbody>
<tr>
<td>Anomaly Detection</td>
<td>Identifies and excludes anomalies (observations) using the support vector data description.</td>
<td>Data Mining and Machine Learning</td>
</tr>
<tr>
<td>Auto-forecasting</td>
<td>Use an ESM, ARIMAX, IDM, or UCM model to generate forecasts.</td>
<td>Forecasting</td>
</tr>
<tr>
<td>Batch Code</td>
<td>Runs SAS batch code.</td>
<td>Data Mining and Machine Learning</td>
</tr>
<tr>
<td>Bayesian Network</td>
<td>Fits a Bayesian network model for a class target.</td>
<td>Data Mining and Machine Learning</td>
</tr>
<tr>
<td>Node Name</td>
<td>Node Description</td>
<td>Product</td>
</tr>
<tr>
<td>--------------------</td>
<td>----------------------------------------------------------------------------------</td>
<td>------------------------------</td>
</tr>
<tr>
<td>Categories</td>
<td>Classifies documents by subject.</td>
<td>Text Analytics</td>
</tr>
<tr>
<td>Clustering</td>
<td>Performs observation-based clustering for segmenting data.</td>
<td>Data Mining and Machine Learning</td>
</tr>
<tr>
<td>Concepts</td>
<td>Extracts specific information from text.</td>
<td>Text Analytics</td>
</tr>
<tr>
<td>Data Exploration</td>
<td>Displays summary statistics and plots for variables in your data table.</td>
<td>Data Mining and Machine Learning</td>
</tr>
<tr>
<td>Decision Tree</td>
<td>Fits a classification tree for a class target or a regression tree for an interval target.</td>
<td>Data Mining and Machine Learning</td>
</tr>
<tr>
<td>Ensemble</td>
<td>Creates a new model by taking a function of posterior probabilities (for class targets) or the predicted values (for interval targets) from multiple models.</td>
<td>Data Mining and Machine Learning</td>
</tr>
<tr>
<td>External Forecasts</td>
<td>Reads forecasts that are generated by an external source.</td>
<td>Forecasting</td>
</tr>
<tr>
<td>Feature Extraction</td>
<td>Generates features based on PCA, robust PCA, SVD, or autoencoders to use as inputs. Note that PCA, SVD, and RPCA use interval inputs only.</td>
<td>Data Mining and Machine Learning</td>
</tr>
<tr>
<td>Filtering</td>
<td>Excludes observations from analysis based on specified criteria.</td>
<td>Data Mining and Machine Learning</td>
</tr>
<tr>
<td>Forest</td>
<td>Fits a forest model, which consists of multiple decision trees based on different samples of the data and different subsets of inputs.</td>
<td>Data Mining and Machine Learning</td>
</tr>
<tr>
<td>GLM</td>
<td>Fits a generalized linear model for an interval target with a specified target distribution and link function.</td>
<td>Data Mining and Machine Learning</td>
</tr>
<tr>
<td>Gradient Boosting</td>
<td>Fits a gradient boosting model, which builds a sequential series of decision trees.</td>
<td>Data Mining and Machine Learning</td>
</tr>
<tr>
<td>Hierarchical Forecasting</td>
<td>Generates forecasts for each level of the specified hierarchy.</td>
<td>Forecasting</td>
</tr>
<tr>
<td><strong>Node Name</strong></td>
<td><strong>Node Description</strong></td>
<td><strong>Product</strong></td>
</tr>
<tr>
<td>---------------------------------------</td>
<td>--------------------------------------------------------------------------------------</td>
<td>--------------------------</td>
</tr>
<tr>
<td>Hierarchical Forecasting (Pluggable)</td>
<td>Generates forecasts using hierarchical forecasting model.</td>
<td>Forecasting</td>
</tr>
<tr>
<td>Imputation</td>
<td>Imputes missing values for class and interval inputs using the specified methods.</td>
<td>Data Mining and Machine Learning</td>
</tr>
<tr>
<td>Linear Regression</td>
<td>Fits an ordinary least squares regression model for an interval target.</td>
<td>Data Mining and Machine Learning</td>
</tr>
<tr>
<td>Logistic Regression</td>
<td>Fits a logistic regression model for a binary or nominal target.</td>
<td>Data Mining and Machine Learning</td>
</tr>
<tr>
<td>Manage Variables</td>
<td>Modifies the metadata of variables.</td>
<td>Data Mining and Machine Learning</td>
</tr>
<tr>
<td>Multistage Model</td>
<td>Generates forecasts using a multistage forecasting model.</td>
<td>Forecasting</td>
</tr>
<tr>
<td>Naive Model</td>
<td>Generates forecasts using naive model.</td>
<td>Forecasting</td>
</tr>
<tr>
<td>Neural Network</td>
<td>Fits a fully connected neural network model.</td>
<td>Data Mining and Machine Learning</td>
</tr>
<tr>
<td>Non-seasonal Model</td>
<td>Generates forecasts using a non-seasonal ESM, ARIMAX, or UCM model.</td>
<td>Forecasting</td>
</tr>
<tr>
<td>Open Source Code</td>
<td>Runs Python or R code.</td>
<td>Data Mining and Machine Learning</td>
</tr>
<tr>
<td>Panel Series Neural Network</td>
<td>Generates forecast using fully connected neural network model.</td>
<td>Forecasting</td>
</tr>
<tr>
<td>Quantile Regression</td>
<td>Fits a quantile regression model for an interval target.</td>
<td>Data Mining and Machine Learning</td>
</tr>
<tr>
<td>Replacement</td>
<td>Replaces data values such as outliers and unknown class levels with specified values.</td>
<td>Data Mining and Machine Learning</td>
</tr>
<tr>
<td>Retired Series</td>
<td>Generates forecasts for retired series using a specified value.</td>
<td>Forecasting</td>
</tr>
<tr>
<td>SAS Code</td>
<td>Runs SAS code.</td>
<td>Data Mining and Machine Learning</td>
</tr>
<tr>
<td>Node Name</td>
<td>Node Description</td>
<td>Product</td>
</tr>
<tr>
<td>---------------------------</td>
<td>----------------------------------------------------------------------------------</td>
<td>----------------------------------</td>
</tr>
<tr>
<td>Save Data</td>
<td>Saves data exported by a node in a pipeline to a CAS library.</td>
<td>Data Mining and Machine Learning</td>
</tr>
<tr>
<td>Score Code Import</td>
<td>Imports SAS score code.</td>
<td>Data Mining and Machine Learning</td>
</tr>
<tr>
<td>Score Data</td>
<td>Scores a table using the score code generated by predecessor nodes and saves the scored table to a CAS library.</td>
<td>Data Mining and Machine Learning</td>
</tr>
<tr>
<td>Seasonal Model</td>
<td>Generates forecasts using a seasonal ESM, ARIMAX, or UCM model.</td>
<td>Forecasting</td>
</tr>
<tr>
<td>Segment Profile</td>
<td>Examines segmented data and enables identification of factors that differentiate the segments from the population.</td>
<td>Data Mining and Machine Learning</td>
</tr>
<tr>
<td>Sentiment</td>
<td>Analyzes attitudes expressed in documents.</td>
<td>Text Analytics</td>
</tr>
<tr>
<td>Stacked Model (NN + TS)</td>
<td>Generates forecasts using stacked model (Neural Network + Time Series).</td>
<td>Forecasting</td>
</tr>
<tr>
<td>SVM</td>
<td>Fits a support vector machine via interior-point optimization for a binary target.</td>
<td>Data Mining and Machine Learning</td>
</tr>
<tr>
<td>Temporal Aggregation Model</td>
<td>Generates forecasts using a temporal aggregation model.</td>
<td>Forecasting</td>
</tr>
<tr>
<td>Text Mining</td>
<td>Parses and performs topic discovery to prepare text data for modeling.</td>
<td>Data Mining and Machine Learning</td>
</tr>
<tr>
<td>Text Parsing</td>
<td>Prepares text for terms analysis.</td>
<td>Text Analytics</td>
</tr>
<tr>
<td>Time Series Regression</td>
<td>Generates forecasts using a regression model.</td>
<td>Forecasting</td>
</tr>
<tr>
<td>Topics</td>
<td>Assigns documents to topics.</td>
<td>Text Analytics</td>
</tr>
<tr>
<td>Transformations</td>
<td>Applies numerical or binning transformations to input variables.</td>
<td>Data Mining and Machine Learning</td>
</tr>
<tr>
<td>Variable Clustering</td>
<td>Performs variable clustering to reduce the number of inputs.</td>
<td>Data Mining and Machine Learning</td>
</tr>
<tr>
<td>Node Name</td>
<td>Node Description</td>
<td>Product</td>
</tr>
<tr>
<td>----------------------------</td>
<td>----------------------------------------------------------------------------------</td>
<td>----------------------------------------------</td>
</tr>
<tr>
<td>Variable Selection</td>
<td>Performs unsupervised and several supervised methods of variable selection to reduce the number of inputs.</td>
<td>Data Mining and Machine Learning</td>
</tr>
</tbody>
</table>

The following Pipeline templates are included with Model Studio:

<table>
<thead>
<tr>
<th>Pipeline Name</th>
<th>Pipeline Description</th>
<th>Product</th>
</tr>
</thead>
<tbody>
<tr>
<td>Advanced template for class target</td>
<td>Extends the intermediate template for class target with neural network, forest, and gradient boosting models, as well as an ensemble.</td>
<td>Data Mining and Machine Learning</td>
</tr>
<tr>
<td>Advanced template for class target with autotuning</td>
<td>Advanced template for class target with autotuned tree, forest, neural network, and gradient boosting models.</td>
<td>Data Mining and Machine Learning</td>
</tr>
<tr>
<td>Advanced template for interval target</td>
<td>Extends the intermediate template for interval target with neural network, forest, and gradient boosting models, as well as an ensemble.</td>
<td>Data Mining and Machine Learning</td>
</tr>
<tr>
<td>Advanced template for interval target with autotuning</td>
<td>Advanced template for interval target with autotuned tree, forest, neural network, and gradient boosting models.</td>
<td>Data Mining and Machine Learning</td>
</tr>
<tr>
<td>Auto-forecasting</td>
<td>Forecasting pipeline with automatic modeling.</td>
<td>Forecasting</td>
</tr>
<tr>
<td>Auto-forecasting (Intermittent)</td>
<td>Forecasting pipeline with automatic, intermittent modeling.</td>
<td>Forecasting</td>
</tr>
<tr>
<td>Base Forecasting</td>
<td>Forecasting pipeline with no modeling components added by default.</td>
<td>Forecasting</td>
</tr>
<tr>
<td>Basic template for class target</td>
<td>A simple linear flow: Data, Imputation, Logistic Regression, Model Comparison.</td>
<td>Data Mining and Machine Learning</td>
</tr>
<tr>
<td>Basic template for interval target</td>
<td>A simple linear flow: Data, Imputation, Linear Regression, Model Comparison.</td>
<td>Data Mining and Machine Learning</td>
</tr>
<tr>
<td>Blank Template</td>
<td>A Data Mining pipeline that contains only a data node.</td>
<td>Data Mining and Machine Learning</td>
</tr>
<tr>
<td>Pipeline Name</td>
<td>Pipeline Description</td>
<td>Product</td>
</tr>
<tr>
<td>-------------------------------</td>
<td>--------------------------------------------------------------------------------------</td>
<td>--------------------------------</td>
</tr>
<tr>
<td>Demand</td>
<td>Forecasting pipeline with demand classification segmentation.</td>
<td>Forecasting</td>
</tr>
<tr>
<td>Classification</td>
<td></td>
<td></td>
</tr>
<tr>
<td>External Forecasts</td>
<td>Forecasting pipeline with external forecasts.</td>
<td>Forecasting</td>
</tr>
<tr>
<td>External Segmentation</td>
<td>Forecasting pipeline with external segmentation.</td>
<td>Forecasting</td>
</tr>
<tr>
<td>Feature engineering template</td>
<td>Data mining pipeline that performs feature engineering.</td>
<td>Data Mining and Machine Learning</td>
</tr>
<tr>
<td>Hierarchical Forecasting</td>
<td>Forecasting pipeline with hierarchical modeling.</td>
<td>Forecasting</td>
</tr>
<tr>
<td>Intermediate template for class target</td>
<td>Extends the basic template with a stepwise logistic regression model and a decision tree.</td>
<td>Data Mining and Machine Learning</td>
</tr>
<tr>
<td>Intermediate template for interval target</td>
<td>Extends the basic template with a stepwise linear regression model and a decision tree.</td>
<td>Data Mining and Machine Learning</td>
</tr>
<tr>
<td>Naive (Moving Average)</td>
<td>Forecasting pipeline with naive, moving average modeling.</td>
<td>Forecasting</td>
</tr>
<tr>
<td>Forecasting</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Naive Forecasting</td>
<td>Forecasting pipeline with naive modeling.</td>
<td>Forecasting</td>
</tr>
<tr>
<td>Non-seasonal Forecasting</td>
<td>Forecasting pipeline with non-seasonal modeling.</td>
<td>Forecasting</td>
</tr>
<tr>
<td>Regression Forecasting</td>
<td>Forecasting pipeline with regression modeling.</td>
<td>Forecasting</td>
</tr>
<tr>
<td>Retired Forecasting</td>
<td>Forecasting pipeline with retired modeling.</td>
<td>Forecasting</td>
</tr>
<tr>
<td>Seasonal Forecasting</td>
<td>Forecasting pipeline with seasonal modeling.</td>
<td>Forecasting</td>
</tr>
<tr>
<td>Text Analytics: Assisted Concept Rule Creation</td>
<td>Use Textual Elements to quickly generate custom concept rules.</td>
<td>Text Analytics</td>
</tr>
<tr>
<td>Text Analytics: Data Access</td>
<td>Text Analytics pipeline that contains a single Data node.</td>
<td>Text Analytics</td>
</tr>
<tr>
<td>Pipeline Name</td>
<td>Pipeline Description</td>
<td>Product</td>
</tr>
<tr>
<td>-------------------------------------------</td>
<td>---------------------------------------------------------------------------------------</td>
<td>--------------------</td>
</tr>
<tr>
<td>Text Analytics: Generate Concepts, Topics,</td>
<td>Text Analytics pipeline for model generation with Concepts, Text Parsing, Sentiment,</td>
<td>Text Analytics</td>
</tr>
<tr>
<td>and Categories</td>
<td>Topics, Categories.</td>
<td></td>
</tr>
<tr>
<td>Text Analytics: Topic Discovery</td>
<td>Text Analytics pipeline that uses text parsing and machine learning to discover topics.</td>
<td>Text Analytics</td>
</tr>
</tbody>
</table>
Overview of Pipelines

Model Studio projects are built around one or more pipelines. A pipeline is a process flow diagram that can be used to represent a sequence of analytical tasks. These analytical tasks are represented as individual nodes in a pipeline.

By default, the initial pipeline for a project uses the template that was specified when the project was created. You can create new pipelines using different templates, and you can make changes to the initial pipeline.
Creating a New Pipeline

In Model Studio, pipelines contain the nodes that process data and create models. A project can contain multiple pipelines.

To create a new pipeline:

1. Navigate to the **Pipelines** tab.

2. Click the + icon next to the current pipeline tab in the upper left corner of the canvas.

   ![Pipeline 1 +]

   The New Pipeline window appears.

3. Give the pipeline a name and an optional description.

4. In the **Template** field, your recently used templates are available. To use a template that you have not used recently, select **Browse templates** and select a template in the Browse Templates window.

5. Click **Save**.

You can also duplicate a pipeline. Click the : icon next the current pipeline tab in the upper left corner of the canvas and click **Duplicate**.

**Note:** The duplicate functionality is not available in SAS Visual Text Analytics 8.4.

Actions on the Pipeline

Click the : icon on the current pipeline tab in the upper left corner of the canvas to perform the following actions:

- **Run** — Runs the entire pipeline.
- **Stop** — Stops the run when the pipeline is running.
- **Duplicate** — Creates a duplicate pipeline. The name is appended with a number. You can rename the duplicate after it is created.

**Note:** The duplicate functionality is not available in SAS Visual Text Analytics 8.4.

- **Rename** — Renames the pipeline.
- **Save to The Exchange** — Saves the pipeline with the nodes and any settings applied to those nodes as a template to The Exchange. The new templates can be used in other projects.
- **Delete** — Deletes the pipeline. This option is available when you have more than one pipeline in your project.
- **Show overview map** — Places a map of the pipeline in the upper left corner of the canvas.
- **Expand header** — Provides a space at the top of the tab to add a description or other text that might be useful. The text can be formatted.

## Modifying a Pipeline

After creating a new pipeline, you are ready to create functionality by adding nodes to the pipeline in either of the following ways:

1. Expand the **Nodes** pane on the left side of the canvas. By default, the **Blank Template** still contains a data node. These steps assume that you are starting with the **Blank Template**.

2. Select a node from the **Data Mining Preprocessing**, **Supervised Learning**, or **Miscellaneous** sections, click and drag it so that the node icon is positioned over the **Data** node, and release the cursor. The new node is added to the canvas, automatically connected to the **Data** node.
Similarly, you can add more nodes to the pipeline, either connected to the Data node, or to the other nodes. Click and drag the new node so that the icon is positioned over the existing node. The new node is added to the pipeline, connected to the node that it was positioned over. See the restrictions (described below) that govern how nodes can be connected to each other.

3 Alternatively, right-click on the Data node, and select Add child node. Select a node from the Data Mining Preprocessing, Supervised Learning, or Miscellaneous options.

   Similarly, you can connect more nodes to existing nodes by either selecting Add child node (creating more successor nodes) or Add parent node (creating predecessor nodes). As with the other method for adding nodes, there are some restrictions as to how nodes can be connected to each other.

You can also delete nodes from your pipeline by right-clicking the node, selecting Delete, and clicking Delete on the Delete window.

Model Studio has a series of rules that govern the positioning of nodes:

1 Data Mining Preprocessing nodes can follow the Data node or other Data Mining Preprocessing nodes. They cannot follow Supervised Learning or Postprocessing nodes.

2 Supervised Learning nodes can follow the Data node or Data Mining Preprocessing nodes. They cannot follow Postprocessing nodes or other Supervised Learning nodes.

3 Postprocessing nodes can follow only Supervised Learning nodes. They are invalid elsewhere.

4 Miscellaneous nodes can follow any Model Studio nodes except for the Model Comparison node. The Model Comparison node is generated when any Supervised Learning node is added to the pipeline, and is automatically connected to follow the added Supervised Learning node.

5 The Data Exploration, Model Comparison, Save Data, and Score Data nodes are terminal nodes—that is, no nodes can follow these nodes in your pipeline.

You can also add notes to your pipeline. This can be useful if multiple users are working in the same project. Click the icon next to the current pipeline tab in the upper left corner of the canvas and click Expand header. Click Collapse header to hide the notes.

---

Creating a Template from a Pipeline

To create a template from a pipeline:

1 Click the icon next to the pipeline tab in the upper left corner of the canvas.

2 Select Save to The Exchange.

3 In the Save Pipeline to The Exchange window, give the template a name and an optional description.

4 Click Save.
Running a Pipeline

There are two ways to run a pipeline:

1. Run all the nodes of the pipeline sequentially, starting with the Data node. This is done by clicking the Run Pipeline button in the upper right corner of the canvas. This can also be done by clicking the icon next to the current pipeline tab in the upper left corner of the canvas and clicking Run.

2. Run one branch of the pipeline, running only the selected node, and all nodes preceding that node by arrows. This is done by right-clicking a node, and selecting Run. For the pipeline to have been fully considered as having run in SAS Visual Data Mining and Machine Learning and SAS Visual Forecasting, you must use the Model Comparison node to run all the nodes in the pipeline.

To interrupt a running pipeline, click the icon next to the current pipeline tab in the upper left corner of the canvas and click Stop.

Node Status

As you look at a pipeline, hold your pointer over each node to see its status.

- When you first create the pipeline, all nodes have a status of Initialized.
- After a node in the pipeline has successfully finished execution, the node shows a status of Successful.
- A node with a status of Pending is waiting for other nodes in the pipeline to complete before starting.
- If a pipeline run fails, check to see which node has a Failed status.
- If you make any changes to the project settings or variable assignments, the nodes change to out-of-date status, and the pipeline must be run again.
- If a node fails for any reason, while a pipeline is running, the status of the subsequent nodes in the pipeline show Canceled.

Comparing Pipelines

Once you have fully run a pipeline, you can compare pipelines with different models to see which model gives the optimal result. You can even “compare” a single pipeline with itself; this displays the results for the single model. To see a pipeline comparison, select the Pipeline Comparison tab. The Pipeline Comparison tab displays the champion model, the algorithm used, and error statistics. The selection
The statistic used to determine the champion model is specified in the Rules section within the Project Settings window. To edit the project settings, click the icon in the upper right corner of the window and click Project settings.

To see alternate statistics, click the icon, and select Manage columns. Use the Manage Columns window to add or remove alternate statistics about the pipelines.

The Pipeline Comparison tab also displays the various results of the champion model. For more information about these results, see the Results section for the given champion model in the SAS Visual Data Mining and Machine Learning: Reference Help documentation.

**Note:** The Pipeline Comparison tab compares only the champion models for each pipeline by default. If you have multiple algorithms in a single pipeline, use the Model Comparison node to compare the performance of each of these individual models. For more information, see Overview of Model Comparison in the Model Studio reference documentation. If you want to add models to the Pipeline Comparison tab, right-click the model node in your pipeline, and select Add challenger model.

Another feature available on the Pipeline Comparison tab is the Compare window. The Compare window enables users to examine the various accuracy statistics of each of the pipelines directly. To compare multiple pipelines:

1. Select at least two pipelines in the left-most column of the comparison table.
2. Click Compare above the table. The Compare window appears. The Compare window contains a table of fit, lift, and ROC statistics. The window also contains line graphs of the statistics, comparing the data roles for each pipeline.

### Insights Tab

You can access a project summary report after successfully running at least one pipeline. To see the report, select the Insights tab. There are six panels with summary information about the project and champion models.

- **Project Summary** — The Project Summary provides information about the project champion model and the most important input variables. Additional project level information is provided:
  - **Project Target** — The target variable specified in the project.
  - **Overall Average** (interval target) — The average of the target variable values in the project data.
  - **Event Percentage** (class target) — The percentage of events in the project data.
  - **Pipelines** — The total number of pipelines in the project.
  - **Project Champion** — The champion model selected in the project.
  - **Created By** — The user ID of the person who originally created the project.
  - **Modified** — The date and time of the most recent modification made to the project.
- **Project Notes** — The Project Notes panel enables you to add notes to the project. This can be useful if multiple users are working in the same project.

  **Note:** The content of the Project Notes panel cannot exceed 4000 characters.

- **Most Common Variables Selected Across All Models** — A bar chart that shows the number of times that an input variable was determined to be an important variable for all models on the Pipeline Comparison tab. This includes the champion model for each pipeline and any challenger models added. Variable importance is calculated using a surrogate model, which is a one-level decision tree for each input variable where the target is the predicted class or value. Input variables with a positive importance value are determined to be important.

- **Assessment for All Models** — A bar chart that shows the values of the selection statistic for all models on the Pipeline Comparison tab. The models are grouped by pipeline.

- **Most Important Variables for Champion Model** — A bar chart that shows the important variables for the project champion model. The relative importance is determined from the actual model when the champion is a decision tree, gradient boosting, or forest model. Otherwise, relative importance is calculated using a one-level decision tree for each input variable to predict the predicted value as a global surrogate model.

- **Cumulative Lift for Champion Model** (class target) — A chart that displays the cumulative lift as a function of the depth for the project champion model. The data is sorted by the predicted probabilities and then divided into 20 quantiles where each quantile contains 5% of the data. The number of events in each quantile is calculated. The cumulative lift for a particular quantile is the ratio of the number of events across all quantiles up to and including the current quantile and the number of events that would be in the quantile at random.

- **Actual vs. Predicted for Champion Model** (interval target) — A chart that displays the actual mean and the predicted mean of the target variable for the project champion model. The data is sorted by the predicted target values and then divided into 20 quantiles where each quantile contains 5% of the data. The actual mean and the predicted mean of the target are plotted for each quantile.

You can download the data that is used to build each chart by clicking the icon. A CSV file with the chart data will immediately begin downloading.

Several charts have detailed descriptions available that explain how the charts were generated. The descriptions also provide information about how to interpret the charts that is custom to your particular data and models. You can access these detailed descriptions in one of the following ways:

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

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

Register Models

To register a model, click the icon in the upper right corner of the Pipeline Comparison tab and select Register models. You cannot register a model when the name of a pipeline is longer than 42 characters.

If SAS Model Manager is installed on the system, this action registers the model in SAS Model Manager. SAS Model Manager is used to store and organize models in a common model repository. It allows for model governance and model change control over time. To view the selected models in SAS Model Manager, click the icon in the upper right corner of the Pipeline Comparison tab and select Manage Models. For more information, see SAS Model Manager: User’s Guide.

If the CASHostAccountRequired group has been created, then Read and Write permissions must be configured for the ModelStore caslib and the ModelPerformanceData caslib. This enables members of the CASHostAccountRequired group to register analytic store models. For more information about how to configure permissions, see File System Directory Permissions.

Note: If you update and re-register a model during the same user session, you must refresh the model in SAS Model Manager to see the updates.

The Model Repository service is always available, so you can still register models if SAS Model Manager is not licensed. For more information, see SAS Viya Administration: Models.

Publish Models

You might want to publish a model so that the model can be executed in various run-time engines. To publish models, you must first create a publishing destination. The types of publishing destinations that are supported are CAS, Hadoop, SAS Micro Analytic Service, or Teradata. Information about how to configure publishing destinations can be found in SAS Viya Administration: Publishing Destinations.

In order to publish analytic stores to SAS Micro Analytic Service, a separate location must be created to store the analytic stores. For more information, see Access to Analytic Store Model Files.

If the CASHostAccountRequired group has been created, then Read and Write permissions must be configured for the ModelStore caslib and the ModelPerformanceData caslib. This enables members of the CASHostAccountRequired group to publish analytic store models. For more information about how to configure permissions, see File System Directory Permissions.
Note: You cannot publish to SAS Micro Analytic Service if you are using user-defined formats.

To publish a model once you have created a publishing destination:

1. Click the icon in the upper right corner of the Pipeline Comparison tab and select Publish models.

2. The Publish Models window appears. Select the model publish destination that you created and the model that you want published.

Note: You cannot publish more than one model at a time.

3. Click Publish. The model is published immediately. If you do not want to reload the table at the same time that you publish content, select Publish without reloading. However, when you publish an item to a CAS destination, you must reload the table in order for the newly published content to be accessible.
When you publish a model, that model is also automatically registered to the Model Repository service.

**Note:** If you are publishing a model to CAS, you must publish the model to the same CAS server where your data is located. Copying data from one CAS server to another is not supported.

### Export Models for Production

Some models in SAS Visual Data Mining and Machine Learning are packaged in a single downloadable DATA step code file. Other models are packaged in two parts, score code and a binary file, for efficiency.

Models can be exported from both the **Pipelines** tab and the **Pipeline Comparison** tab.

- On the **Pipelines** tab, select the pipeline that contains your target model. Right-click the champion node, and select **Download Score Code**.
- On the **Pipeline Comparison** tab, select one pipeline, click the icon in the upper right corner of the **Pipeline Comparison** tab, and then select **Download Score Code**.

Both methods listed above download a ZIP file to the client that contains the model score code. The model score code contains the code generated by the supervised learning node, as well as any data mining preprocessing nodes preceding it.

For models packaged in two parts, the ZIP file contains DS2 code and the analytic stores are saved in the MODELS caslib. The DS2, or DATA step 2, code is also referred to as analytic store code or EP score code. The analytic store score code is a representation of the model pipeline including any pre- or post-processing steps. The second part of the two-part model is the analytical store. Each analytic store is a binary file that contains the state of an analytic procedure after training. There can be multiple analytic stores in a single model. Together, the analytic store code and the analytic store represent your model, and can be used to score new data. The **Download Score Code** action saves the analytic stores in the MODELS caslib. In addition, the analytic store score code is compressed and downloaded to the client. The analytic store score code contains a comment that specifies names of binary files. These binary files are required for use with the score code, and can be found in the MODELS caslib.

You can identify the path associated with the MODELS caslib by using SAS Environment Manager. For more information, see [SAS Viya Administration: Using SAS Environment Manager](https://support.sas.com/)

In the CAS server’s file system, navigate to the location of the MODELS caslib to find the analytic stores for your model. Each analytic store specified in the analytic store score code is in the MODELS directory with an extension of SASHDAT.

### Import Score Code

Some models in SAS Visual Data Mining and Machine Learning are packaged in a single downloadable DATA step code file. Other models are based on a SAS
analytical store. Each analytic store 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 imported into your project.

The 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. If the analytic store is not already available in a caslib, the model can still be imported. To do this, make the analytic store SASHDAT file available on the cas server and load the analytic store into a caslib accessible in Model Studio. The following SAS code demonstrates how to complete this process:

```sas
CAS;
CASLIB _ALL_ ASSIGN;
PROC ASTORE;
/* the cas server directory where the analytic store file was saved */
UPLOAD STORE="/home/myusername/score.sasast"
/* the caslib and table name */
RSTORE=PUBLIC.emforest;
RUN;
/* promote uploaded analytic store so it will be visible across sessions */
PROC CASUTIL;
   PROMOTE CASDATA="emforest"
   CASOUT="em_forest_store"
   INCASLIB="PUBLIC" OUTCASLIB="PUBLIC";
QUIT;
```

To import score code once the score code and any required analytic stores are available in Model Studio:

1. Click the icon in the upper right corner of the Pipeline Comparison tab.
2. Select Import score code. The Import Score Code window appears.
3. In the Import Score Code window, enter a name for the imported model in the Name field.
4. Select either Data step code or Astore code.
5. Click the Browse button next to the appropriate field and navigate to the path that contains your model. Select your model and click Open.
6. Click Import.

Pipeline Comparison then incorporates the imported model into its assessment as if it were another pipeline.

---

Note: When importing a model in Model Studio, the model is assessed using the project data. This requires that the imported model was created using the same target and the same measurement level as the current project. It also requires that the imported score code produces the expected predicted or posterior variables and does not drop the target variable.
Score Holdout Data

To score holdout data:

1. Click the icon in the upper right corner of the Pipeline Comparison tab.
2. Select Score holdout data.
3. The Choose Data window appears. Select the data set that contains the holdout data that you want scored.
4. Click OK.
5. Model Studio scores the holdout data. To see the results of this process, use the Data menu below Pipeline Comparison to select Holdout.

Download Score API

To download the model API:

1. Click the icon in the upper right corner of the Pipeline Comparison tab.
2. Select Download score API. The Scoring API window appears.
3. Select the Download Type. The following choices are available:
   - SAS — Downloads code to run in a SAS programming environment.
   - Python — Downloads code to run in a Python programming environment.
   - REST — Downloads a text file with example REST calls that you can use in an application.
4. After selecting the Download type, click Download. The model API will download immediately.

Downloading Logs

To download project logs:
1 In the project, click the icon in the upper right corner (this is accessible on the Data, Pipelines, and Pipeline Comparison tabs).

2 Select Project logs. You can then select the Project Partitioning logs, the Project Data Advisor log, or the Scoring and Assessing model logs. For pipelines that have not yet run, only the Project Data Advisor log is available.

3 Once the log is selected, click Download log. The log will download immediately, saved as a TXT file.