SAS® Visual Data Mining and Machine Learning 8.3: User's Guide

SAS® Documentation
September 10, 2019

The correct bibliographic citation for this manual is as follows: SAS Institute Inc. 2018. *SAS® Visual Data Mining and Machine Learning 8.3: User's Guide*. Cary, NC: SAS Institute Inc.

SAS® Visual Data Mining and Machine Learning 8.3: User's Guide

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

September 2019

8.3-P1:vdmmlug

Contents

1

Getting Started with SAS Visual Data Mining and Machine Learning in Model Studio

Charitable Giving Example

Tutorial Scenario

This example tutorial is intended for new Model Studio users. The target audience ranges from new users to experienced data scientists. The analytic narrative is authored for individuals who are familiar with fundamental data mining concepts. 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. Prior data mining experience is highly beneficial. However, the tutorial is designed so that general users can follow and comprehend the analytic narrative and complete the stepwise example from start to end.

The analytic narrative reviews intermediate computational results and statistics throughout the data mining process. During the tutorial, SAS code is displayed in intermediate result windows to provide information about input and target variables generated during successive analytic steps. SAS programming knowledge is not necessary to perform any task outlined in this book. However, being able to browse through SAS output and recognize process performance and statistical operation results adds learning value. Following the tutorial narrative is a good way to become familiar with SAS Visual Data Mining and Machine Learning software, as well as learning more about a typical data mining problem solving approach.

The Charitable Giving tutorial uses example data named DONOR_RAW_DATA. The DONOR_RAW_DATA is the training data with known target variable values. The DONOR_SCORE_DATA represents potential donors with no target variable included in the data set. This data set is not used in this example. You can download an archived file containing all of the required tutorial data here: Download Charitable Giving Example Data (zip).

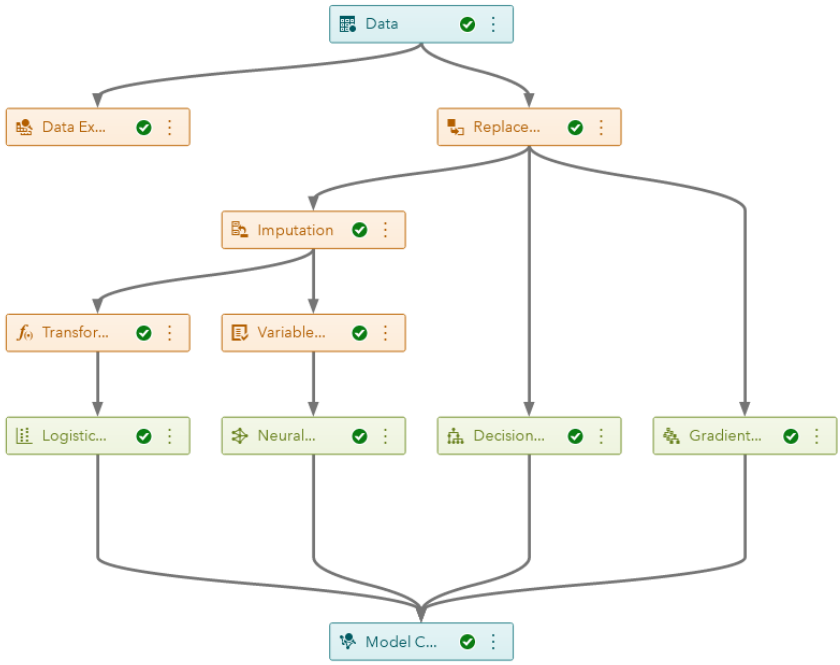
Download the zip archive file and extract the contents to a directory that your SAS Visual Data Mining and Machine Learning server can access. You will use only the DONOR_RAW_DATA for this example.

This tutorial is a fund raiser solicitation example that will familiarize you with Model Studio features as you follow a structured data mining narrative. You will perform tasks to build and configure an analytic pipeline in SAS Visual Data Mining and Machine Learning. The pipeline will perform analytic data preparation, data replacement, and train competing models to find the best algorithm to choose from a data pool of candidates for charitable solicitation.

The analytic narrative for the Charitable Giving tutorial requires sequential data mining steps. Follow the chapters and the steps within the chapters in the order in which they are presented. You should be able to reproduce the results of each tutorial step on your own workstation before continuing on to the next step. If you deviate from the analytic process and configuration detailed in the example, your intermediate and final results might not be valid. After you successfully complete the example as described, feel free to modify your final pipeline model settings. You can

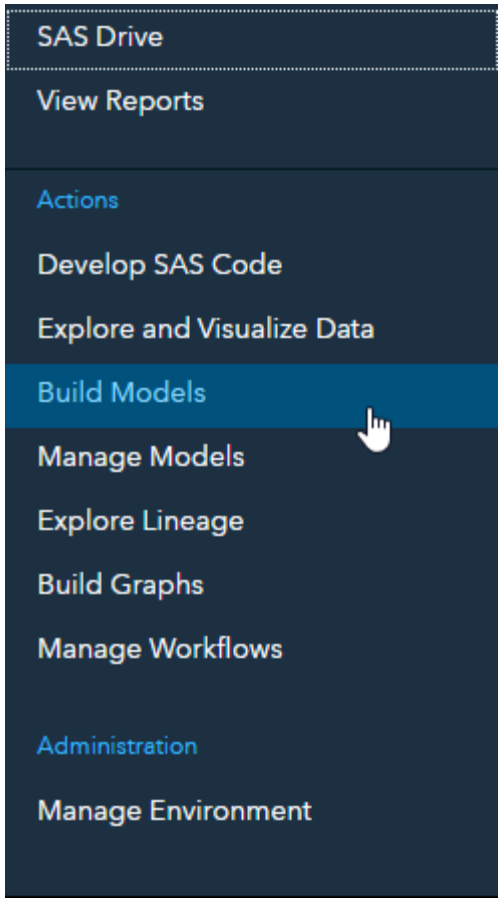
add new competing models to experiment and see how your changes would affect predicted results.

In the tutorial, you are a data analyst at a national charitable organization. Your organization seeks to use the results of a previous postcard mail solicitation for donations to better target its next one. In particular, you want to determine which of the individuals in your mailing database are the most generous donors. By soliciting only these people, your organization can spend less money on the solicitation effort and more money on charitable concerns. When you have finished building the pipeline as outlined in this example, the diagram will resemble the one shown here:



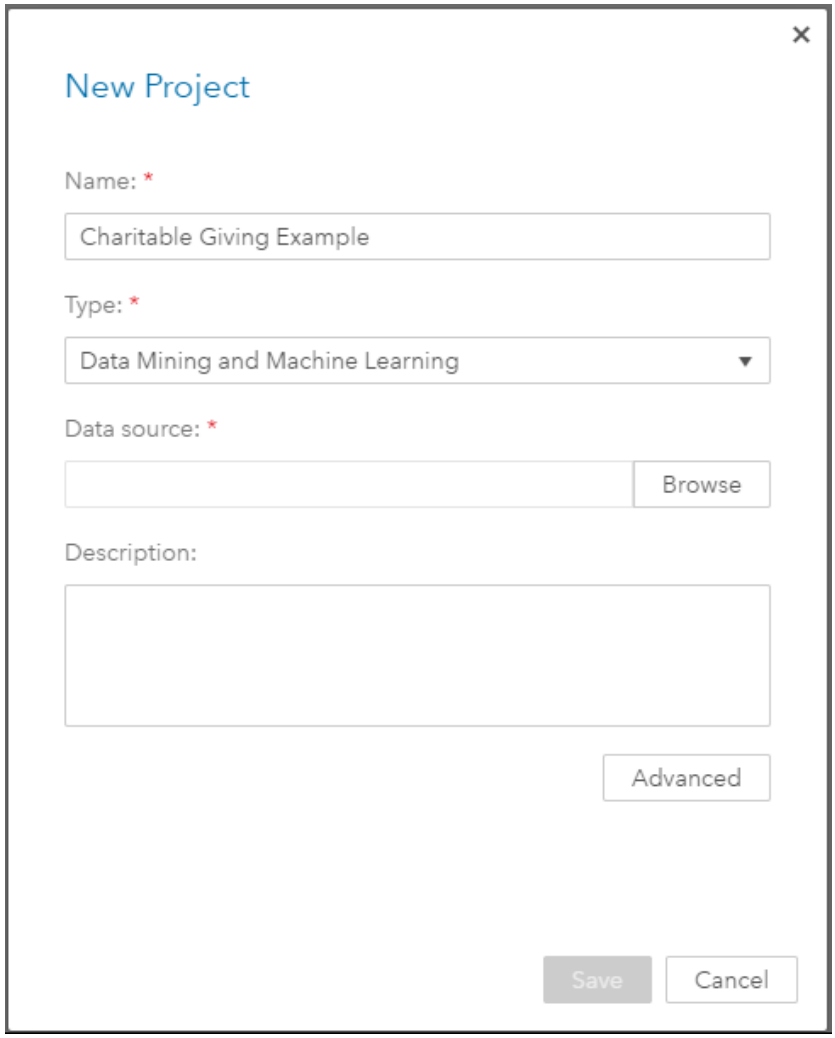
Create the Project and Import the Input Data

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 ☰ icon and select **Build Models**.



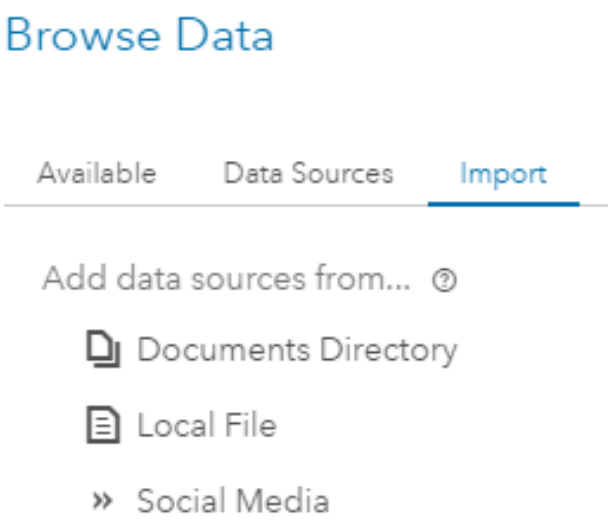
You are directed to the Projects page. To create the project that you will use in this example, complete the following steps:

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**.



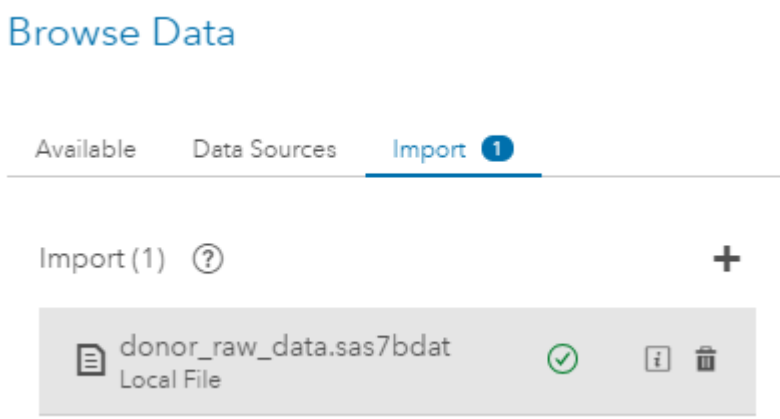
4 In the **Data source** field, select **Browse**. The Browse Data window appears.

5 In the upper left corner of the **Browse Data** window, select **Import**.



6 Select **Local File** and navigate to the folder where DONOR_RAW_DATA is stored. Select **DONOR_RAW_DATA.sas7bdat** and click **Open**.

7 Click **Import Item** in the upper right corner of the Browse Data window. When the data is successfully imported, the icon appears next to the imported data file in the **Import** tab.

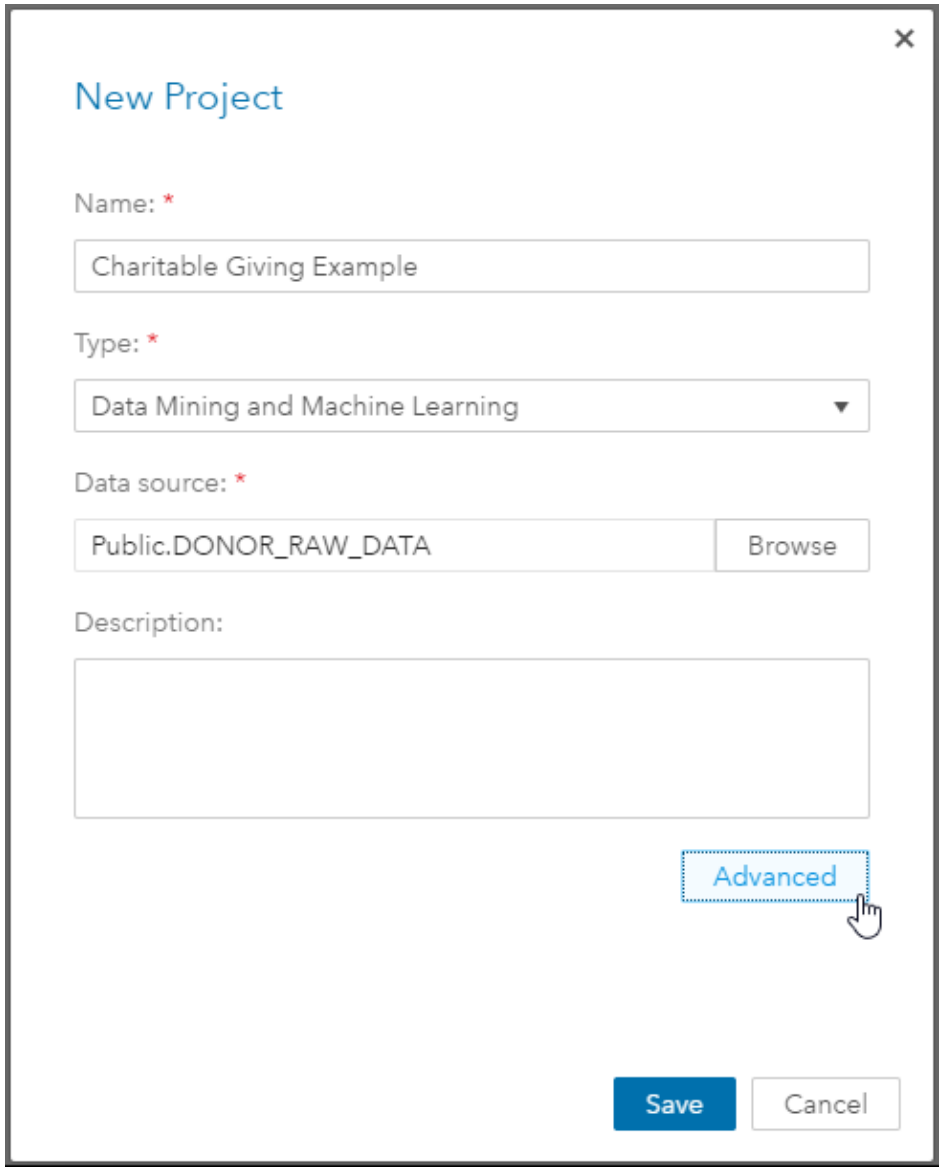


8 Once the data set is successfully imported, click **OK** in the lower right corner of the Browse Data window. This brings you back to the New Project window.

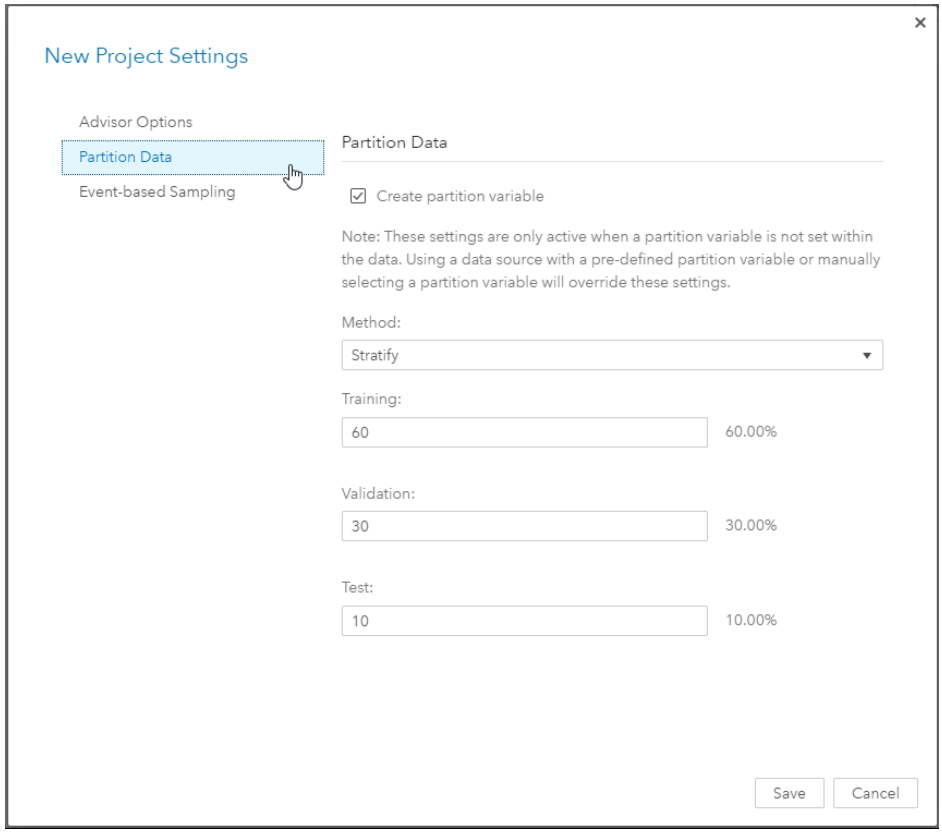
Partition the Data

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. To partition the data, complete the following steps:

1 Select **Advanced** under the **Description** text box.

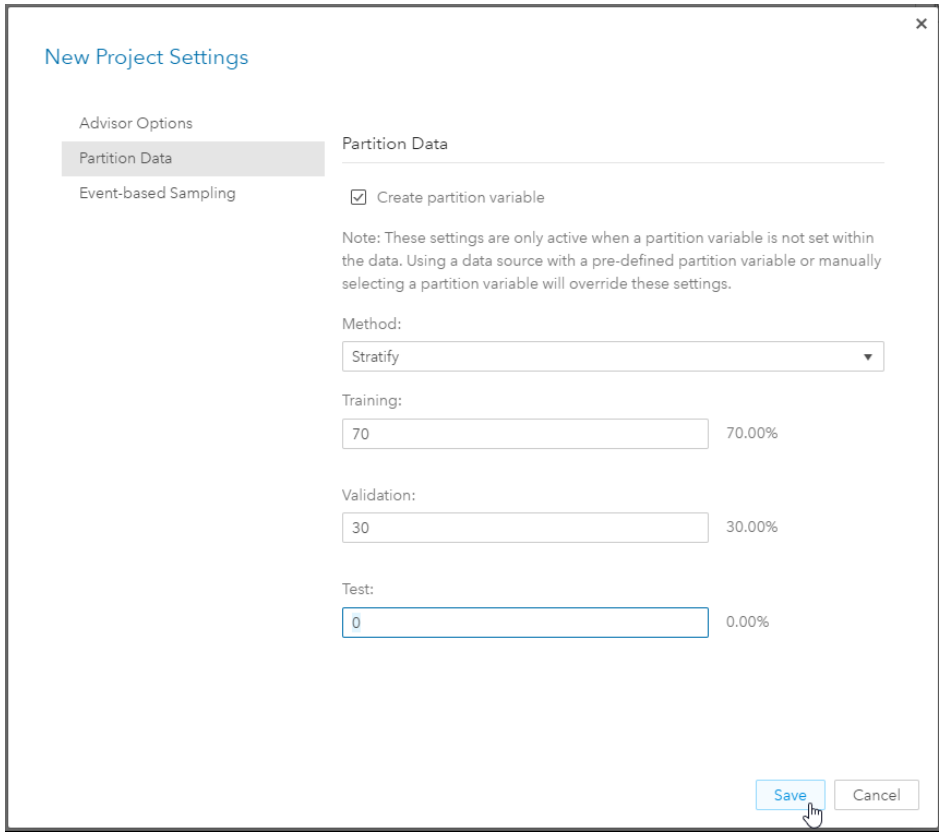


2 The New Project Settings window appears. Select **Partition Data** from the list in the upper left corner of the window.



3 Ensure that the **Create partition variable** option is selected. Then, adjust the settings to match the following:

- Set **Method** to **Stratify**
- Set **Training** to `70`
- Set **Validation** to `30`
- Set **Test** to `0`



These properties define the percentage of input data that is used in each type of mining data set. In this example, you use a training data set and a validation data set, but you do not use a test data set.

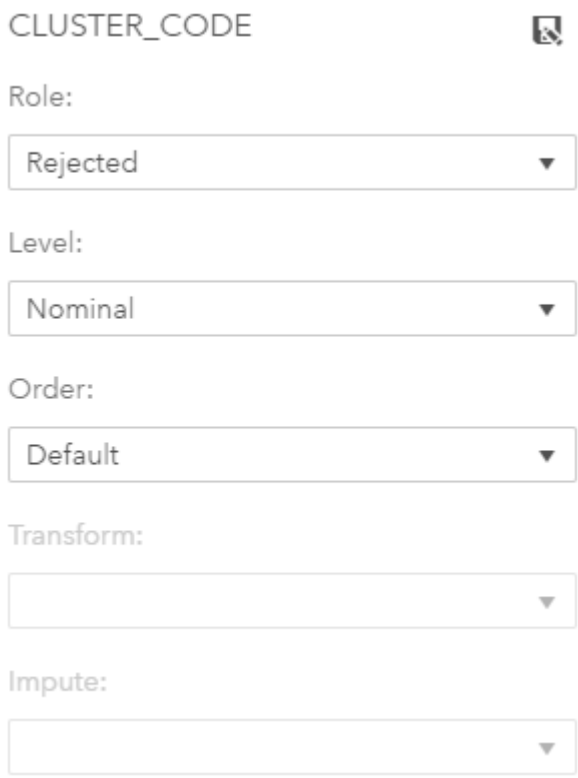
4 Click **Save** in the New Project Settings window, and you will return to the New Project window.

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

Modify Variables

In the **Data** tab, variable roles are indicated in the **Role** column. To change the role of a variable, complete the following steps:

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 in 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 CLUSTER_CODE to **Rejected**
- Set CONTROL_NUMBER to **ID**
- Set TARGET_B to **Target**
- Set TARGET_D to **Rejected**
- Ensure that all other variables 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 simultaneously selecting FILE_AVG_GIFT, LAST_GIFT_AMT, and LIFETIME_AVG_GFT_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, complete the following steps:

1. Right-click on the **Data** node and select **Add below** ⇨ **Miscellaneous** ⇨ **Data Exploration**.
2. Right-click the **Data Exploration** node and select **Run**.
3. Once the pipeline has run successfully, right-click on the **Data Exploration** node and select **Results** from the menu that appears. The following pieces of information about the input data are represented in either graphical or tabular fashion:
   - **Important Inputs**
   - **Class Variable Summaries**
   - **Class Variable Distributions**
   - **Interval Variable Moments**
   - **Interval Variable Summaries**
   - **Interval Variable Distributions**
   - **Missing Values**
   - **Target by Input Variable Crosstabulations**
   - **Node Score Code**
   - **Properties**
   - **Output**
4. Click **Close**.

Replace Missing Values

In this example, the variables SES and URBANICITY are class variables for which the value ? denotes a missing value. Because a question mark does not denote a missing value in the terms that SAS defines a missing value (that is, a blank or a period), 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, complete the following steps:

1. Right-click on the **Data** node and select **Add below** ⇨ **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. Set **Default limits method** to **Standard deviation from the mean**.
4. Right-click the **Replacement** node and select **Run**.
5. Once the node has run successfully, right-click on the node and select **Results** to view detailed information in each of the following windows:
   - **Class Variables**

Class Variables

Name	Variable Label	Replace Variable	Type	Replacement Value...
CARD_PROM_12		REP_CARD_PROM_12	N	.
DONOR_GENDER		REP_DONOR_GENDER	C	
FREQUENCY_STATUS_97NK		REP_FREQUENCY_STATUS_97NK	N	.
HOME_OWNER		REP_HOME_OWNER	C	
INCOME_GROUP		REP_INCOME_GROUP	N	.
IN_HOUSE		REP_IN_HOUSE	N	.
		REP_OVERLAY_SOUR		

   - **Replacement Counts**

Replacement Counts

Name	Variable Label	Train	Role	Variable Level
CARD_PROM_12		0	INPUT	NOMINAL
DONOR_GENDER		0	INPUT	NOMINAL
FREQUENCY_STATUS_97NK		0	INPUT	NOMINAL
HOME_OWNER		0	INPUT	BINARY
INCOME_GROUP		0	INPUT	NOMINAL
IN_HOUSE		0	INPUT	BINARY
OVERLAY_SOURCE		0	INPUT	NOMINAL
REP_STAR		0	INPUT	BINARY

6. Click **Close**.

Automatically Train and Prune a Decision Tree

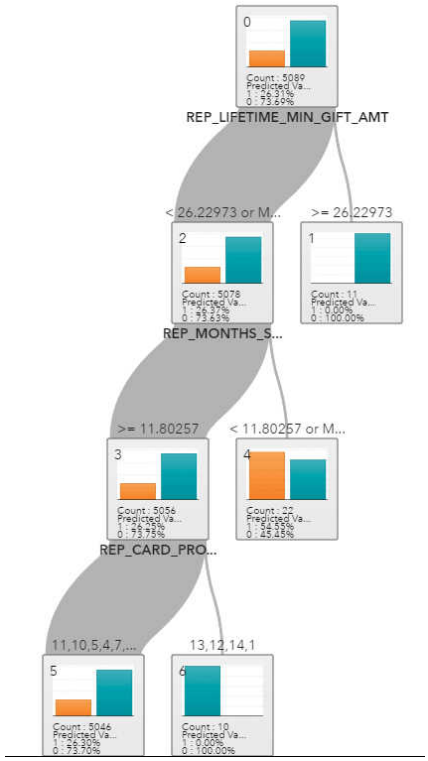
Decision tree models are advantageous because they are conceptually easy to understand, yet they readily accommodate nonlinear associations between input

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.

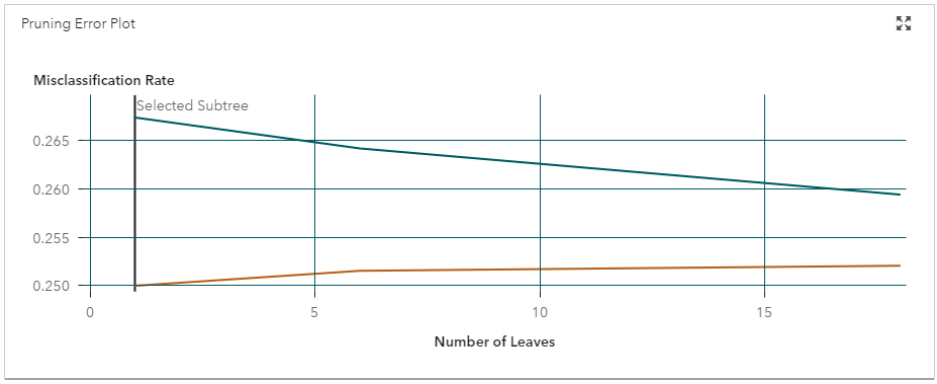
Note: When creating a **Decision Tree** node, a **Model Comparison** node is automatically created.

To insert a **Decision Tree** node, complete the following steps:

1. Right-click the **Replacement** node and select **Add below** ⇨ **Supervised Learning** ⇨ **Decision Tree**.
2. Select the **Decision Tree** node.
3. In the options panel, complete the following:
   - **a** Expand **Splitting Options**.
   - **b** Set **Maximum depth** to **10**.
   - **c** Set **Minimum leaf size** to **8**.
   - **d** Set **Surrogate rules** to **4**.
   - **e** Expand **Pruning Options**.
   - **f** 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**. Explore the following:
6. In the Decision Tree Results window, explore the following:
   - The **Tree Diagram** and **Tree Map**



   - **Pruning Error Plot**



- **Score Outputs** and **Score Inputs**
- **Fit Statistics**

Fit Statistics

Data Role	Partition Indicator	Formatted Partition	Sum of Frequencies	Average Squared Er...
TRAIN	1	1	13,560	0.1878
VALIDATE	0	0	5,812	0.1878

Due to the nondeterministic behavior of SAS Viya, your results might not be an identical match to these results.

7 Click **Close**.

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 about the Gradient Boosting node, see the Model Studio help documentation.

To create a gradient boosting model of the data, complete the following steps:

1 Right-click on the **Replacement** node and select **Add below** ⇨ **Supervised Learning** ⇨ **Gradient Boosting**.

2 Select the **Gradient Boosting** node. In the options panel, expand **Tree-splitting Options**.

3 Set **Maximum depth** to **10**.

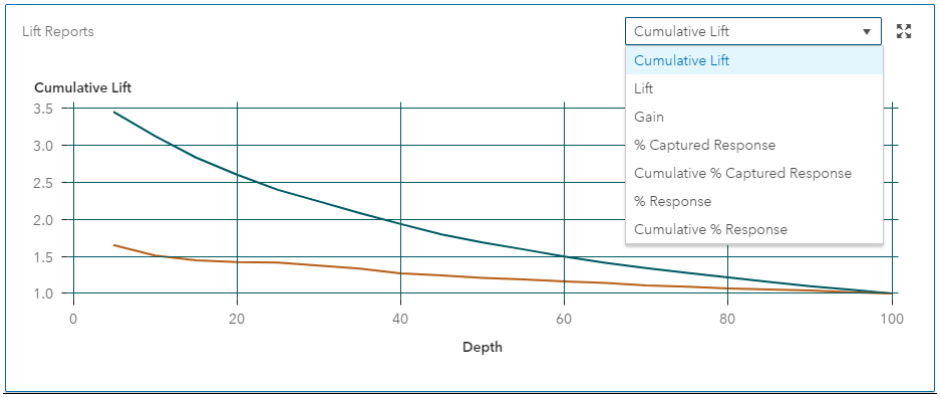
4 Enable the **Perform Autotuning** option.

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

6 Right-click the **Gradient Boosting** node and select **Results**. Explore the following:

- **Error Plot**

- **Lift Reports**



- **Variable Importance**

Variable Importance

Variable Name	Train Importance	Importance Standard Devi...	Relative Importance
REP_RECENT_RESPONSE_COUNT	15.9557	8.2965	1
REP_WEALTH_RATING	15.8483	0.9320	0.9933
REP_INCOME_GROUP	10.5391	0.8322	0.6605
REP_FREQUENCY_STATUS_97NK	8.1112	4.7264	0.5084
REP_CARD_PROM_12	8.1024	2.3709	0.5078
REP_MEDIAN_HOME_VALUE	7.1707	1.5514	0.4494

7 Click **Close**.

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, complete the following steps:

1. Right-click the **Replacement** node and select **Add below** ⇨ **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 on the node and select **Results** . Explore the following:
   - **Input Variable Statistics**

Input Variable Statistics

Input Variable	Variable Level	Number of Missing ...	Percent Missing	Imputable
REP_CARD_PROM_12	NOMINAL	0	0	0
REP_DONOR_AGE	INTERVAL	3,362	24.7935	1
REP_DONOR_GENDER	NOMINAL	0	0	0
REP_FILE_AVG_GIFT	INTERVAL	0	0	0
REP_FILE_CARD_GIFT	INTERVAL	0	0	0
REP_FREQUENCY_STATUS_97NK	NOMINAL	0	0	0

   - **Imputed Variables Summary**

Imputed Variables Summary

Imputed Variable	Method	Input Variable	Value	Percent Missing
IMP_REP_DONOR_AGE	MEDIAN	REP_DONOR_AGE	60	24.7935
IMP_REP_INCOME_GROUP	COUNT	REP_INCOME_GROUP	5	22.6401
IMP_REP_MONTHS_SINCE_LAST_PROM_R	MEDIAN	REP_MONTHS_SINCE_LAST_PROM_RESP	18	1.4012
IMP_REP_WEALTH_RATING	COUNT	REP_WEALTH_RATING	9	45.3171

6 Click **Close**.

Transform Variables

At the beginning of this example, you opted to transform the variables FILE_AVG_GIFT, LAST_GIFT_AMT, and LIFETIME_AVG_GIFT_AMT using the **Log10** methodology. In order to execute the transformation of these variables, complete the following steps:

1 Right-click the **Imputation** node and select **Add below** ⇨ **Data Mining Preprocessing** ⇨ **Transformations**.

2 Right-click on 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, complete the following steps:

1 Right-click the **Transformation** node and select **Add below** ⇨ **Supervised Learning** ⇨ **Logistic Regression**.

2 Right-click on 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:

- **Regression Fit Statistics**

Regression Fit Statistics

Statistic	Description	Training	Validation
M2LL	-2 Log Likelihood	14,773.8238	6,378.9986
AIC	AIC (smaller is better)	14,793.8238	6,398.9986
AICC	AICC (smaller is better)	14,793.8400	6,399.0365
SBC	SBC (smaller is better)	14,868.9726	6,465.6702
ASE	Average Square Error	0.1808	0.1823

- **Parameter Estimates**

Parameter Estimates

Effect	Parameter	t Value	Sign	Estimate
REP_MEDIAN_HOME_VALUE	REP_MEDIAN_HOME_VALUE	7.6204	+	0.0002
REP_FREQUENCY_STATUS_97NK	REP_FREQUENCY_STATUS_97NK_1	6.5869	-	-0.4897
REP_MONTHS_SINCE_LAST_GIFT	REP_MONTHS_SINCE_LAST_GIFT	5.0952	-	-0.0273
REP_MONTHS_SINCE_FIRST_GIFT	REP_MONTHS_SINCE_FIRST_GIFT	4.4142	+	0.0030
Intercept	Intercept	4.3463	-	-0.6184
REP_RECENT_CARD_RESPONSE_PROP	REP_RECENT_CARD_RESPONSE_PROP	4.2717	+	0.5773

- **Fit Statistics**

Fit Statistics

Data Role	Partition Indicator	Formatted Partition	Sum of Frequencies	Average Squared Er...
TRAIN	1	1	13,560	0.1808
VALIDATE	0	0	5,812	0.1824

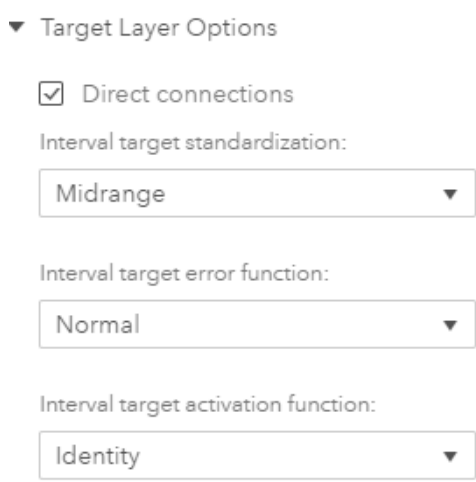
4 Click **Close**.

Create a Neural Network

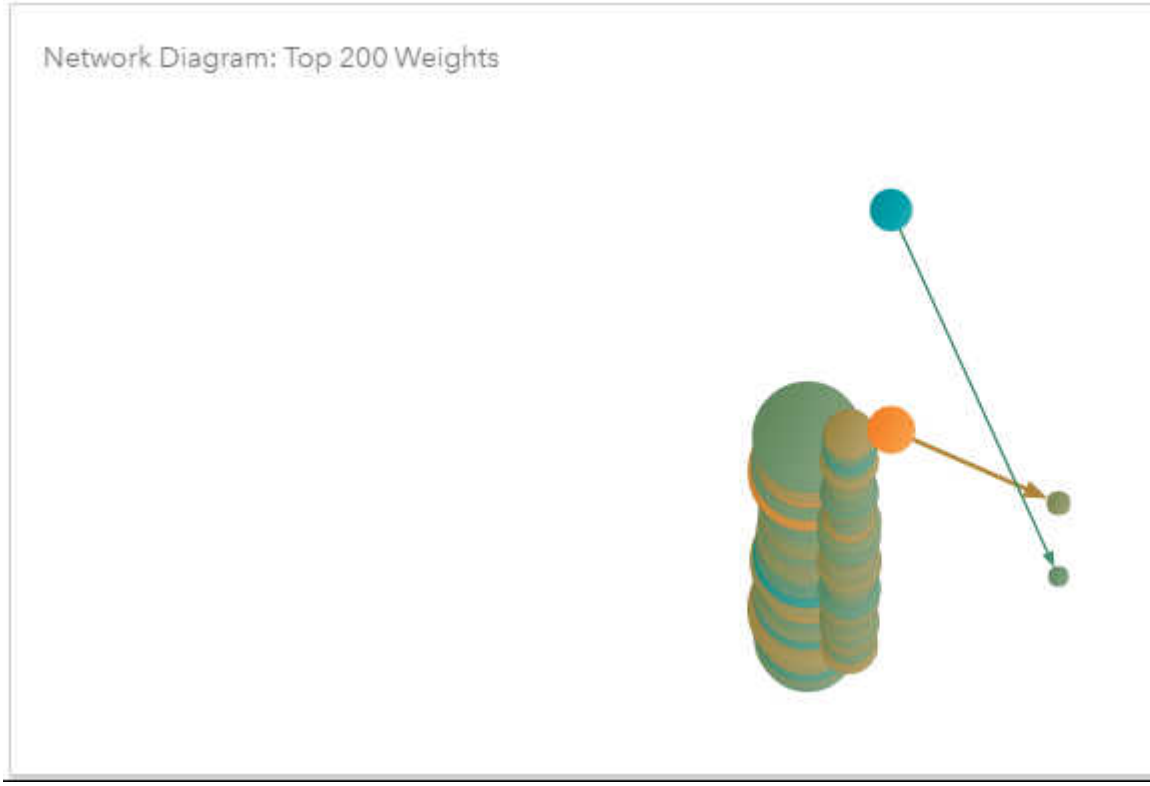
Neural networks are a class of parametric models that can accommodate a wider variety of nonlinear relationships between a set of predictors and a target variable than can logistic regression. Building a neural network model involves two main phases. First, define the network configuration. You can think of this step as defining the structure of the model that you want to use. Then, iteratively train the model. A neural network model will be more complicated to explain to the management of your organization than a regression or a decision tree. However, you know that the

management prefers a stronger predictive model, even if it is more complicated. So, you run a neural network model, which you compare to the other models later in the example. 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 to reduce the number of input variables that are used in a neural network, complete the following steps:

1. Right-click the **Imputation** node and select **Add below** ⇨ **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 below** ⇨ **Supervised Learning** ⇨ **Neural Network**.
4. Select the **Neural Network** node.
5. In the options panel, complete the following steps:
   a. Set **Number of hidden layers** to **5**.
   b. Expand **Target Layer Options** and ensure that **Direct connections** is selected.



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



8 Click **Close**.

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, complete the following steps:

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 properties pane. Because you did not change the value of this property, the default statistic was used.

4 Close the **Results** window.

Publish the Champion Model

Before completing this section, you should review how to configure publishing destinations in *SAS Viya Administration: Publishing Destinations*.

To publish a model, complete the following steps:

1. Go 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 that you want your model to be published in.
5. Select **Publish**.
6. The Publishing Results window appears. This window shows name and published name of you model, as well as the status of your model (publishing, published successfully, and so on.)

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 PROC GRADBOOST. In the second example, you use the FOREST procedure to perform variable selection.

Both examples use the DONOR_RAW_DATA that is found in the Charitable Giving Example Data (zip). This data includes a list of people that were solicited for donation by a charity, including whether they donated. The ZIP file also includes a second data set, DONOR_SCORE_DATA, that does not indicate whether a patron donated. These examples do not use DONOR_SCORE_DATA.

Download the zip archive file and extract the contents to a directory that your SAS Visual Data Mining and Machine Learning server can access. You will use only the DONOR_RAW_DATA for this example.

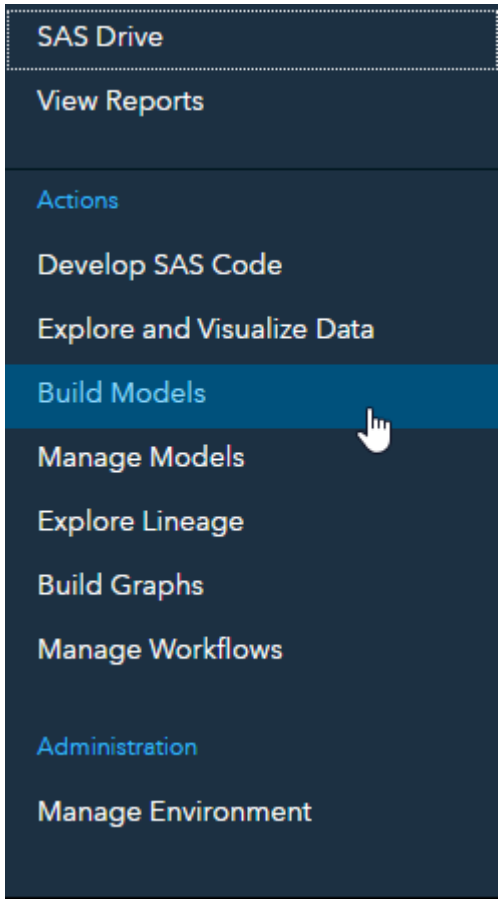
Both examples require you to complete the steps in the Create the Project and Import the Input Data on page 21 and Modify Variables on page 24 sections. After completing these sections, you can complete the examples in any order.

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, complete the following steps:

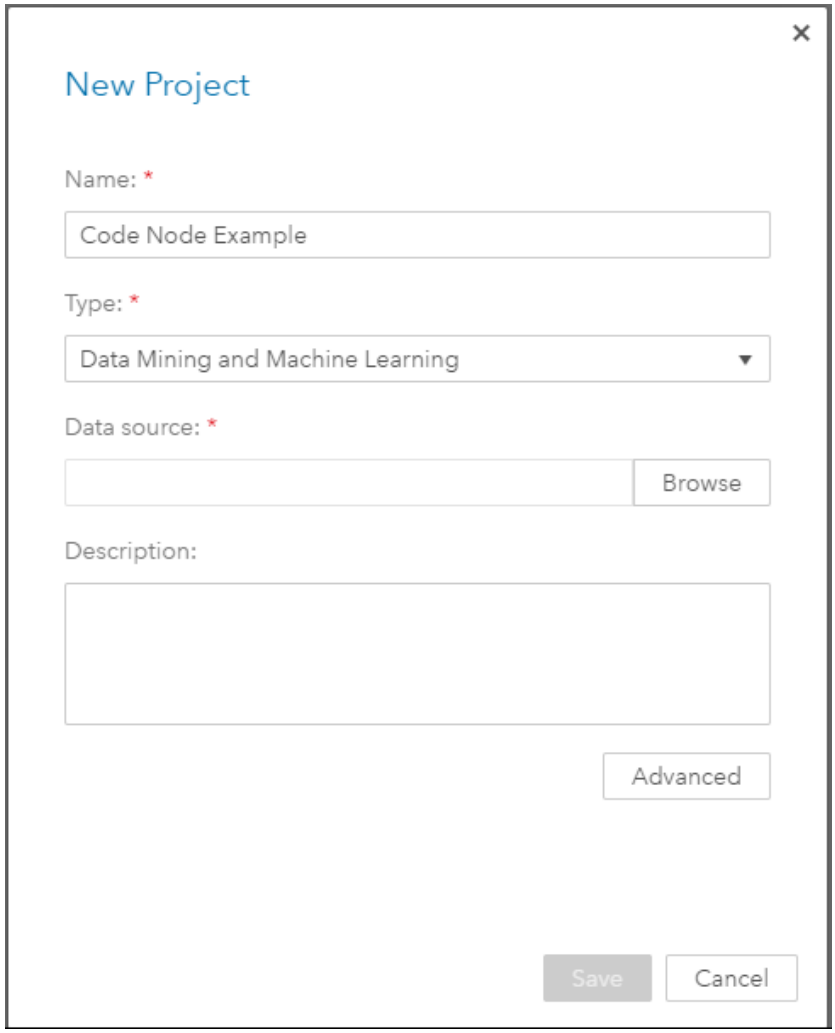
1. In the upper left corner of the SAS Drive window, click the ≡ icon, and select **Build Models**.

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



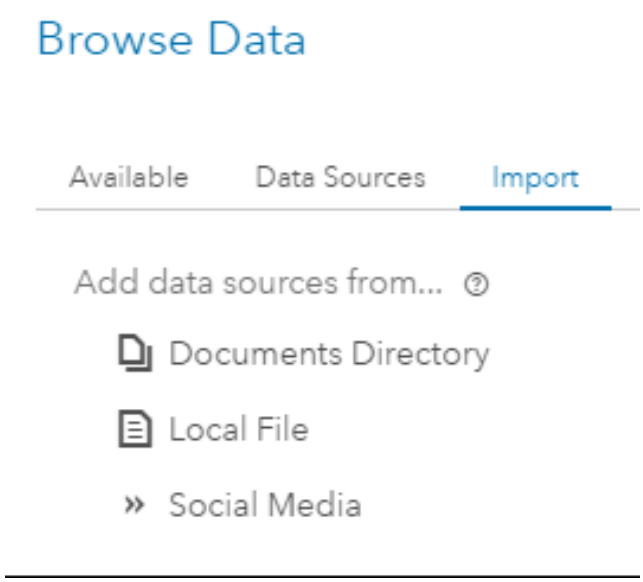
3 Enter `Code Node Example` for **Name** in the New Project window.

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



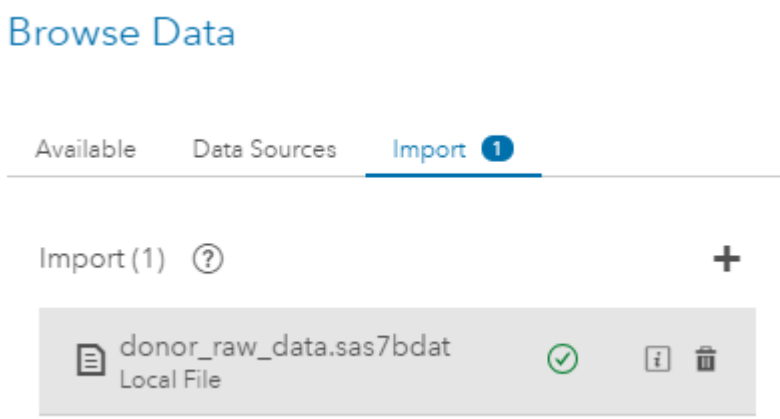
5 In the **Data source** field, select **Browse**. The Browse Data window appears.

6 In the upper left corner of the Browse Data window, select **Import**.



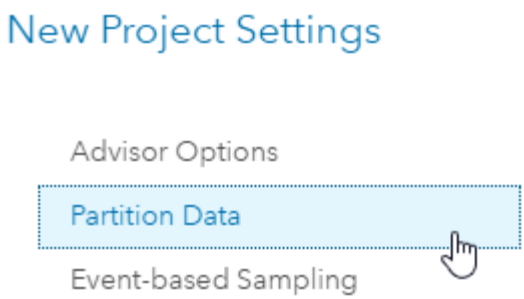
7 Select **Local File**, and navigate to the folder where DONOR_RAW_DATA is stored. Select **DONOR_RAW_DATA.sas7bdat** and click **Open**.

8 Click **Import Item** in the upper right corner of the Browse Data window. When the data is successfully imported, the ⊘ icon appears next to the imported data file in the **Import** tab.



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

10 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.



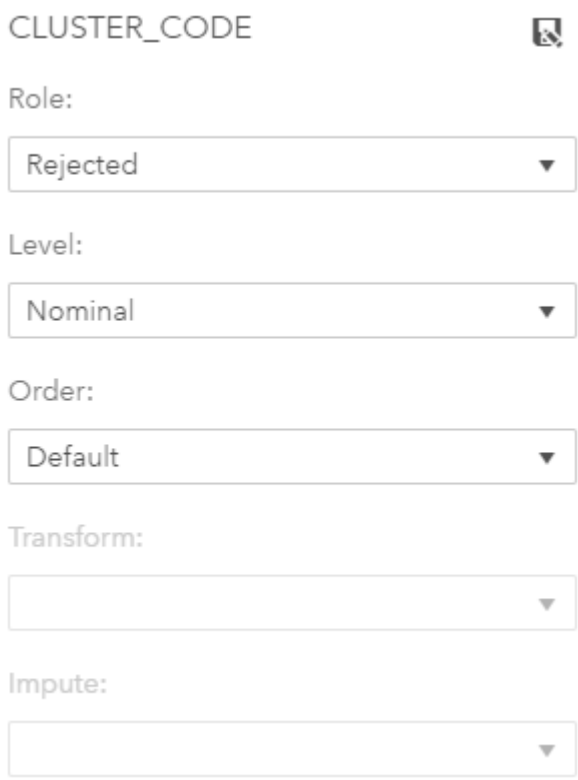
11 Ensure that the **Create partition variable** option is selected, and click **Save** in the lower right corner of the window. This brings you back to the New Project window.

12 In the lower left corner of the New Project window, click **Save**. You are redirected to the **Data** tab.

Modify Variables

In the **Data** tab, variable roles are indicated in the **Role** column. To change the role of a variable, complete the following steps:

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 in 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 CLUSTER_CODE to **Rejected**
- Set CONTROL_NUMBER to **ID**
- Set TARGET_B to **Target**
- Set TARGET_D to **Rejected**
- Ensure that all other variables 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 simultaneously selecting FILE_AVG_GIFT, LAST_GIFT_AMT, and

LIFETIME_AVG_GFT_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 on page 21 and Modify Variables on page 24 sections. This example also assumes that you have not created any other pipelines before starting this section. To create the gradient boosting model, complete the following steps:

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 on the **Data** node and select **Add below** ⇨ **Miscellaneous** ⇨ **SAS Code**.
5. Right-click on 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**.
7. In the code editor, enter the following code:

```
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 Impo

%dmcas_report(dataset=VarImp, reportType=BarChart, category=Variable,
    response=RelativeImportance, description=%nrbquote(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:

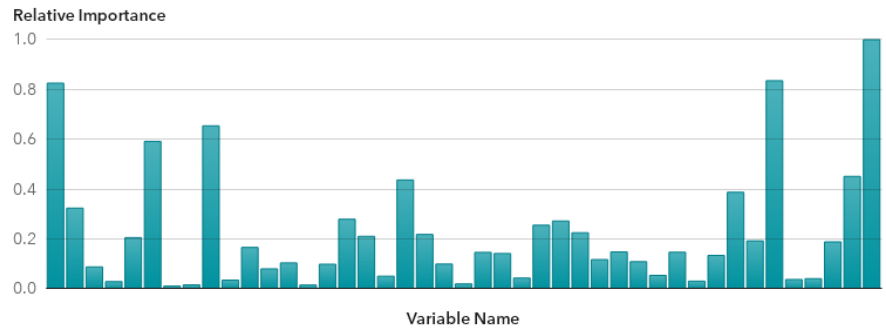
- **Variable Importance**

Variable Importance

Variable Name	Train Importance	Importance Standard...	Relative Importance
WEALTH_RATING	23.6037	1.9204	1
RECENT_RESPONSE_COUNT	19.7174	8.5559	0.8353
CARD_PROM_12	19.4883	3.2649	0.8256
INCOME_GROUP	15.4433	1.3822	0.6543
FREQUENCY_STATUS_97NK	13.9647	11.1356	0.5916
URBANICITY	10.6587	1.9227	0.4516
MONTHS_SINCE_LAST	10.3161	3.4411	0.4371

- **Relative Importance Plot**

Relative Importance Plot

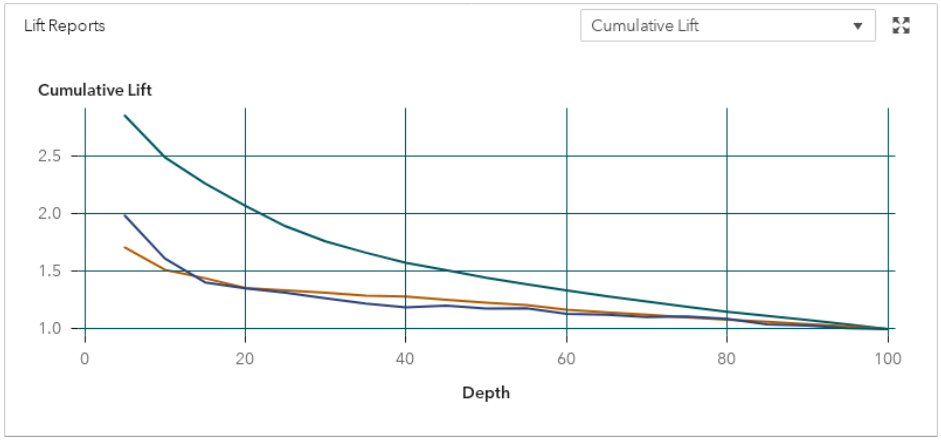


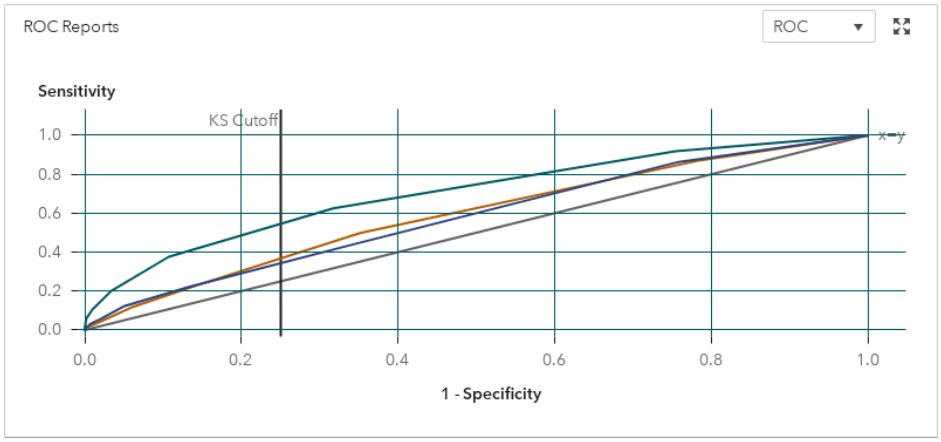
- **EP Score Code** — Because you specified that the **SAS Code** node is a supervised learning node, it automatically creates EP Score Code. More specifically, this code produces an analytic store as its score code.

EP Score Code

```
data sasep.out;
    dcl package score _8WE3ZCUMVCOJ78998LILOVCGI();
    dcl double "DONOR_AGE";
    dcl double "FILE_AVG_GIFT";
    dcl double "FILE_CARD_GIFT";
    dcl double "LAST_GIFT_AMT";
    dcl double "LIFETIME_CARD_PROM";
    dcl double "LIFETIME_GIFT_AMOUNT";
    dcl double "LIFETIME_GIFT_COUNT";
    dcl double "LIFETIME_MIN_GIFT_AMT";
    dcl double "LIFETIME_PROM";
    dcl double "MEDIAN_HOME_VALUE";
    dcl double "MEDIAN_HOUSEHOLD_INCOME";
```

- **Lift Reports**, **ROC Reports**, and **Fit Statistics** — 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.





12 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 on page 21 and Modify Variables on page 24 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 on the **Data** node and select **Add below** ⇨ **Miscellaneous** ⇨ **SAS Code**.

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

6 In the code editor, enter the following code:

```
proc forest data=&dm_data
   minleafsize=5 minuseinsearch=1 seed=12345 loh=0 numbin=20
   ntrees=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
```

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 columnsmeta 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.

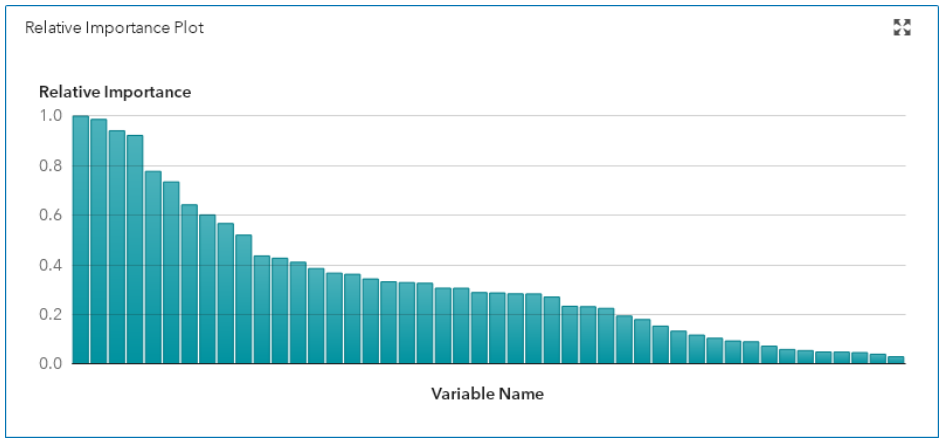
8 Click **Close**.

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.

Metadata Changes

```
if NAME "FREQUENCY_STATUS_97NK" then ROLE="REJECTED";
if NAME "RECENT_RESPONSE_COUNT" then ROLE="REJECTED";
if NAME "RECENT_CARD_RESPONSE_COUNT" then ROLE="REJECTED";
if NAME "CARD_PROM_12" then ROLE="REJECTED";
if NAME "WEALTH_RATING" then ROLE="REJECTED";
if NAME "MONTHS_SINCE_LAST_GIFT" then ROLE="REJECTED";
if NAME "INCOME_GROUP" then ROLE="REJECTED";
if NAME "MONTHS_SINCE_LAST_PROM_RESP" then ROLE="REJECTED";
if NAME "NUMBER_PROM_12" then ROLE="REJECTED";
if NAME "RECENT_CARD_RESPONSE_PROP" then ROLE="REJECTED";
if NAME "FILE_CARD_GIFT" then ROLE="REJECTED";
if NAME "RECENT_RESPONSE_PROP" then ROLE="REJECTED";
if NAME "PCT_ATTRIBUTE4" then ROLE="REJECTED";
if NAME "DONOR_AGE" then ROLE="REJECTED";
```

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. The

purpose of this example is to demonstrate the steps necessary to complete this task and you are encouraged to repeat the process with several additional models.

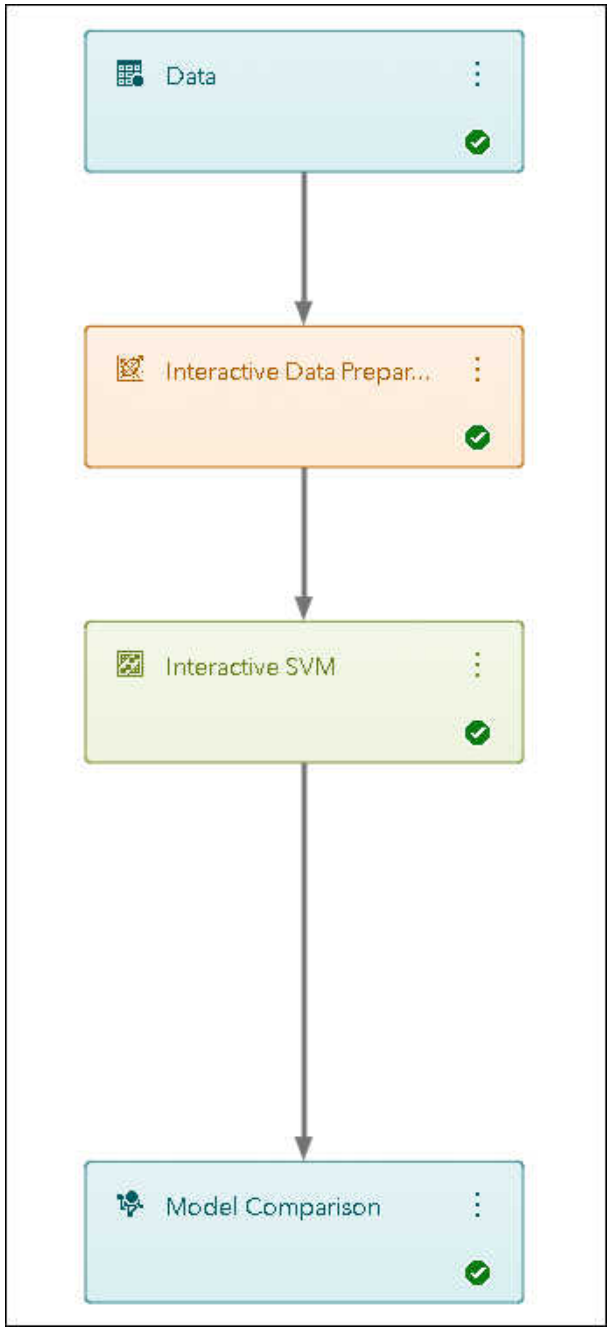
SAS Visual Analytics is a data visualization tool that enables you to create many different types of reports that demonstrate the efficacy of various statistical models. For more information, see Working with SAS Visual Statistics in the SAS Visual Analytics 8.2 documentation. Many of the reports created 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 **Interactive Data Preparation** node
- An interactive model node
- The **Model Comparison** node

The two interactive nodes will be 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. The pipeline below shows that each of the components has already successfully run, indicated by the green check mark in the component.



What you cannot determine from this pipeline is that 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 followed by Gradient Boosting model to create inputs that are used by a Support Vector Machine. The creation of these inputs is captured in the Interactive Data Preparation node, even though the Forest and Gradient Boosting nodes do not appear in the pipeline.

This example uses the HMEQ data set, which contains 5,960 mortgage applications and whether the applicant defaulted on the loan. The HMEQ data set can be found at 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 http://support.sas.com/documentation/onlinedoc/viya/examples.htm.
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.

Complete these steps to create the report:

1. In the upper left corner of the SAS Drive window, click the ≡ 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.

   Click the ⋮ icon in the upper right corner of the page, and then select **Save**. This opens the Save As window. In the **My Folder** pane, navigate to a location where you have Write permission. In the **Name** field, enter `Integration Example`, and click **Save**.

   Typically, you can save your work in `My Folder`.
6. On 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

Complete these steps 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 to save the project.

Create a Support Vector Machine

Complete these steps 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**.

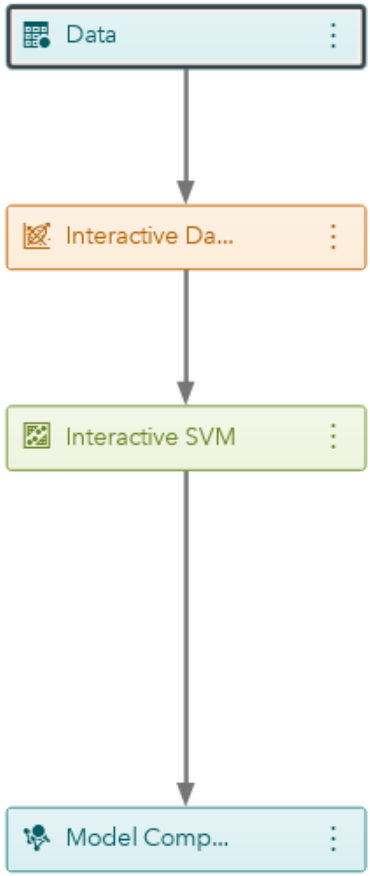
4 For **Predictors**, click **Add**, and select **CLAGE**, **DEBTINC**, **DELINQ**, **DEROG**, and **Predicted: LOAN**. Click **OK**.

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 completes, 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 **Interactive 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 **Interaction Data Preparation** node or the **Interactive SVM** 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 **Interactive Data Preparation** node handles the role assignments when you run the pipeline.

   Note: If you change the target or partition variable information about the **Data** tab and try to run your pipeline, it will fail. The score that is generated and applied in the **Interactive Data Preparation** node requires the partition and

variable information that was known in SAS Visual Analytics when you created the pipeline. After you run the pipeline, 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.

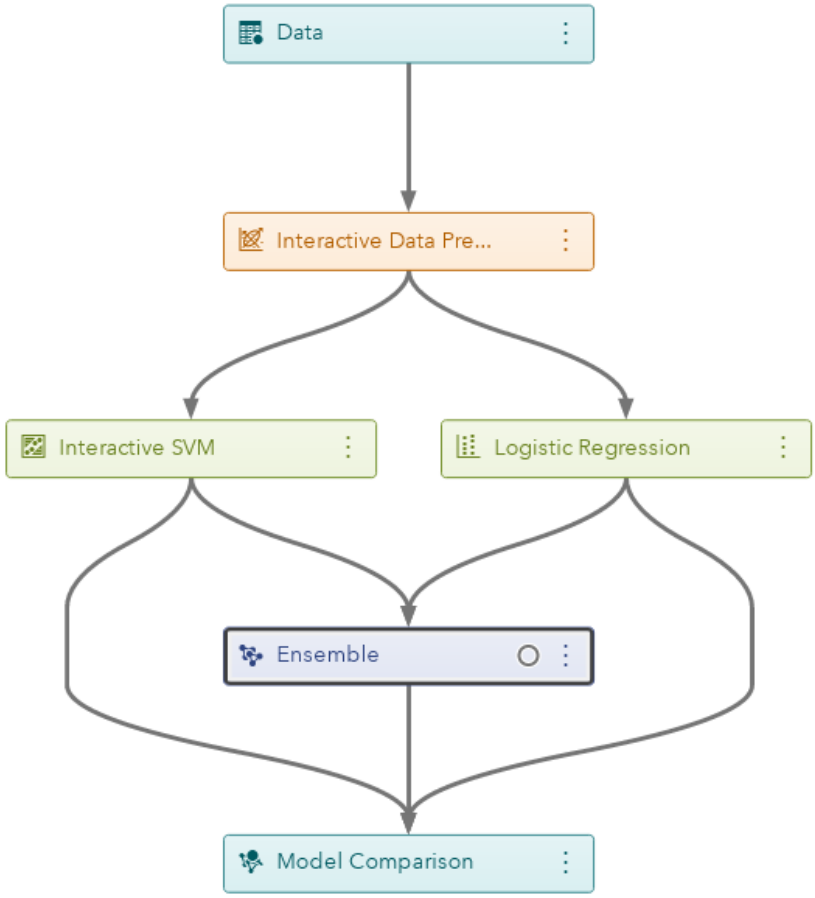
4 In the upper left corner, click **Pipelines** to open the **Pipelines** tab.

The primary purpose of the Interactive Data Preparation node is to ensure that all subsequent nodes see the data as it existed in SAS Visual Analytics. This node is responsible for the execution of 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 Interactive 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 that you add in the next step sees the data as it existed in SAS Visual Analytics.

5 Right-click the **Interactive Data Preparation** node and select **Add below** ⇨ **Supervised learning** ⇨ **Logistic Regression**. This adds a new support vector machine to the pipeline, but does not apply the data preparation steps used in SAS Visual Analytics.

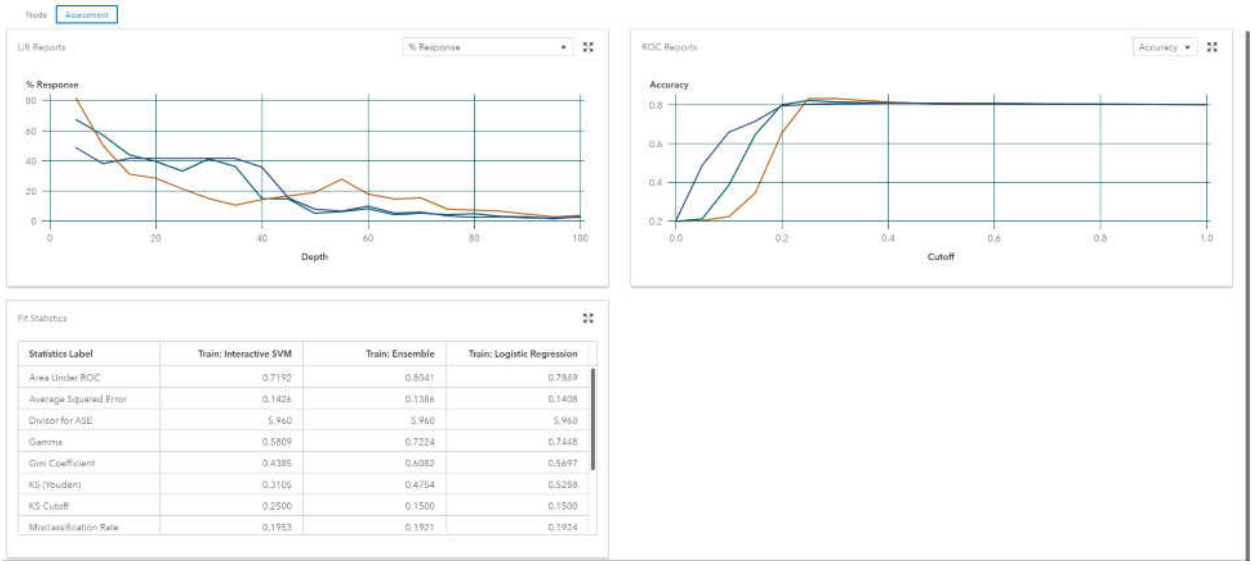
6 Right-click the **Logistic Regression** node and select **Add below** ⇨ **Postprocessing** ⇨ **Ensemble**.

7 Right-click the **Ensemble** node and select **Add models** ⇨ **Interactive 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, which 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 page.

Open Source Code Node Example

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 SAMPSIO.HMEQ data set. The HMEQ data set 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.

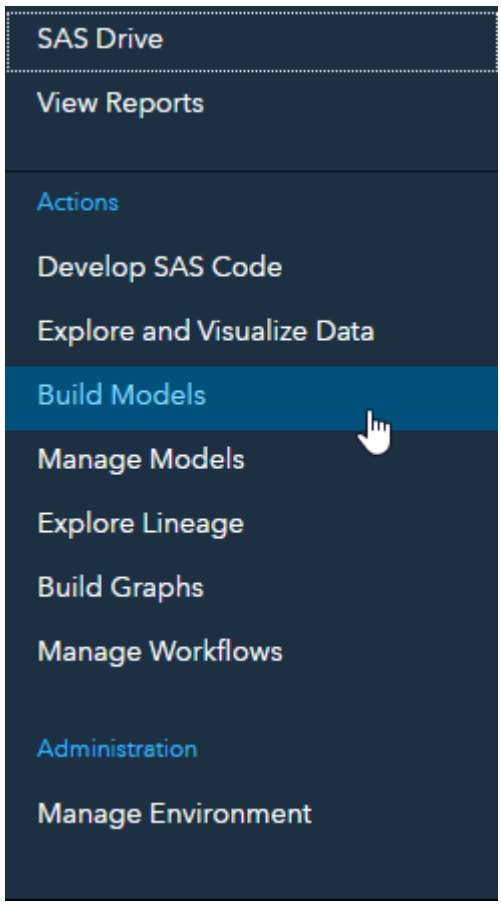
If you are unsure about whether the HMEQ data set is available on your system, navigate to the SAS Viya Example Data Sets page and download **hmeq.csv**. Note where you saved this file.

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, complete the following steps:

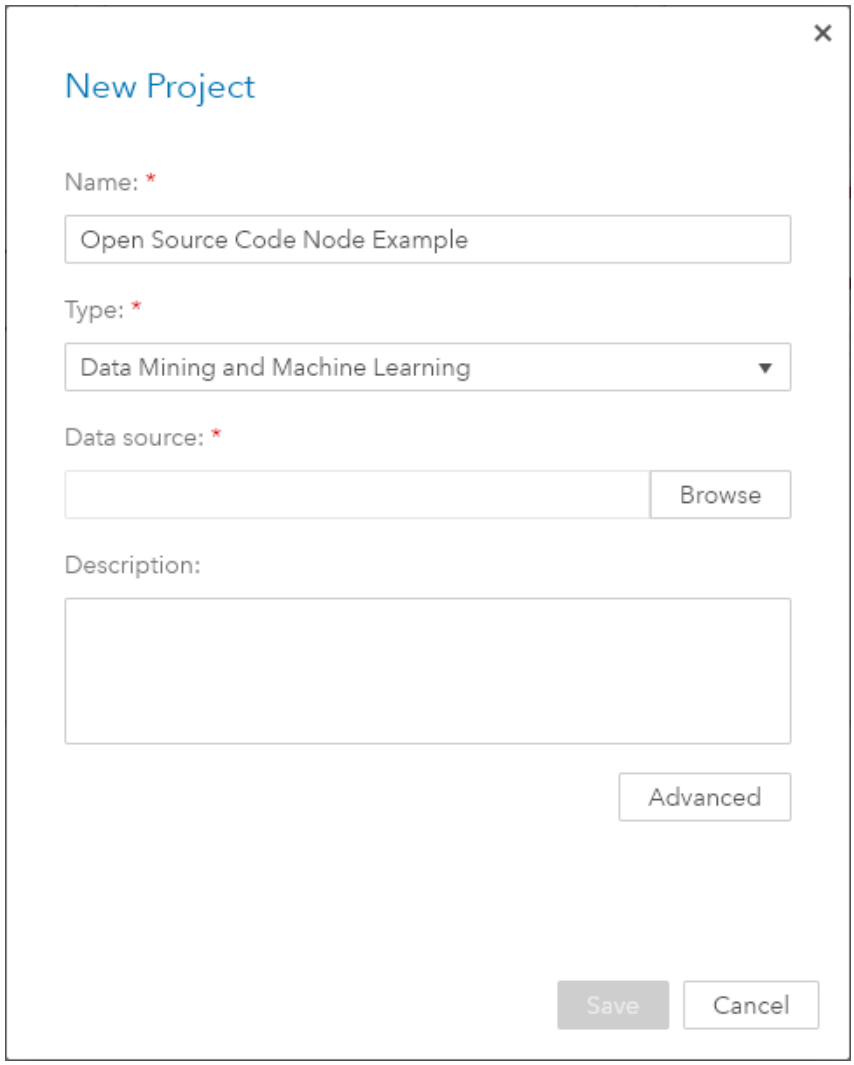
1 In the upper left corner of the SAS Drive window, click the ≡ icon, and select **Build Models**.

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



3 Enter `Open Source Code Node Example` for **Name** in the New Project window.

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



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

6 If **HMEQ** is listed in the **Available** tab of the Browse Data window, select the HMEQ data set and click **OK**.

If the HMEQ data set is not listed in the **Available** tab, complete the following steps:

a Click the **Import** tab.

b Click **Local File**.

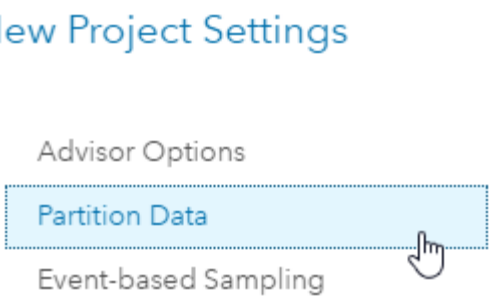
c In the Open window, navigate to the location where you saved the HMEQ data set and select **hmeq.csv**.

d Click **Open**.

e In the upper right corner, click **Import Item**.

f After the data set is successfully imported, click **OK**.

7 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.



8 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.

9 In the lower right corner of the New Project window, click **Save**. You are redirected to the **Data** tab.

Modify Variables

In the **Data** tab, variable roles are indicated in the **Role** column. To change the role of a variable, complete the following steps:

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 in 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, complete the following steps:

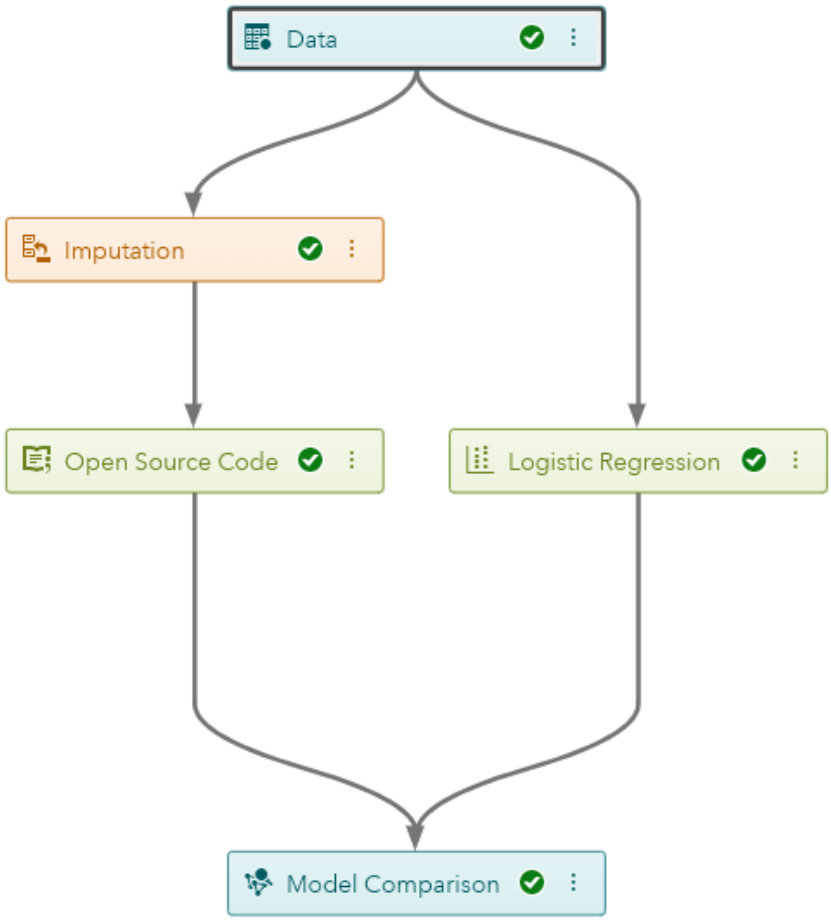
1 Navigate to the **Pipelines** tab. This tab should contain a single pipeline with only a **Data** node.

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

3 Right-click the **Data** node and select **Add below** ⇨ **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 below** ⇨ **Miscellaneous** ⇨ **Open Source Code**.

Your current pipeline should resemble the following image.



5 Select the **Open Source Code** node. On the properties panel, set the value of **Language** to **R**.

6 In the properties panel, click **Open**. Enter the following code in the code editor:

```
library(randomForest)

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

# 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)
```

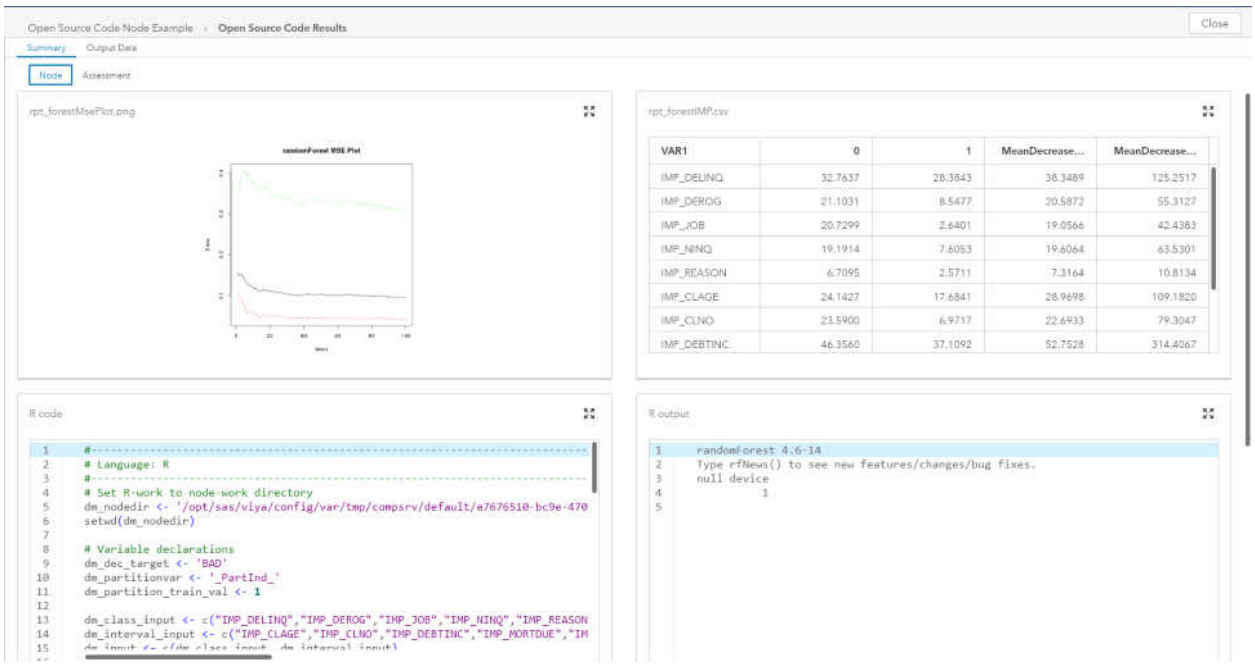
7 In the upper right corner of the code editor, click the icon.

8 Click **Close**.

9 Right-click the **Open Source Code** node and select **Move** ⇨ **Supervised Learning**. This ensures that the node will perform model assessment and can be compared to the **Logistic Regression** node.

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 misclassification rate.

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

Close the results window. The example is now complete.

2

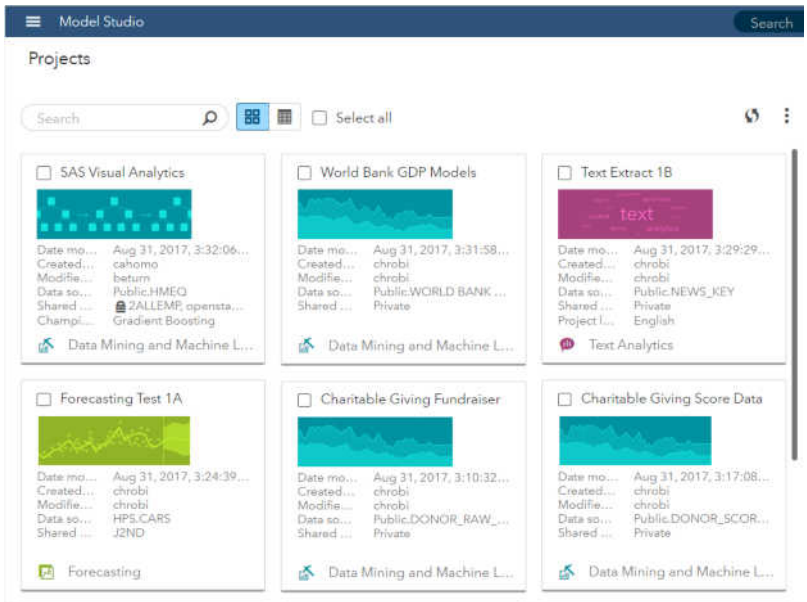
Managing Projects

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 depends on the SAS licensing for your site.

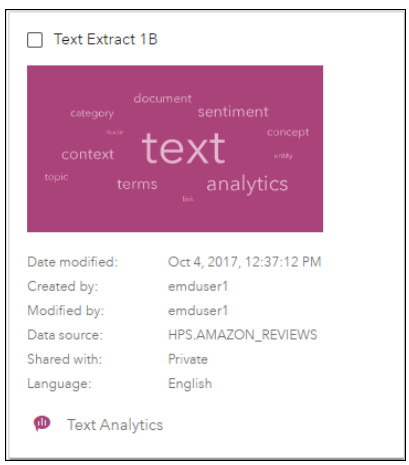
Depending on your project filter setting, existing projects in your environment appear either as graphic tiles or rows in a table of projects.



To alternate between table and tile project displays, complete the following steps:

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

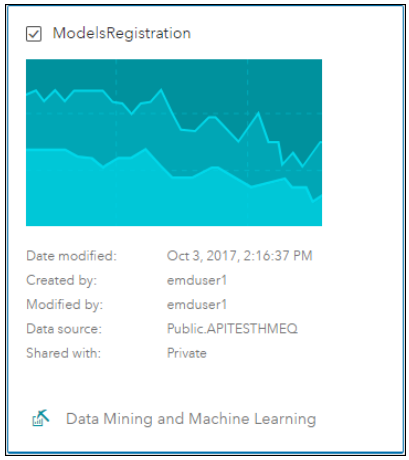
A Model Studio project contains the data source, the pipelines that you create, and related project metadata (such as project type, project creator, share list, and last update history). If you create more than one pipeline in your project, analytic results that compare the performance of multiple pipelines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



Opening an Existing Project

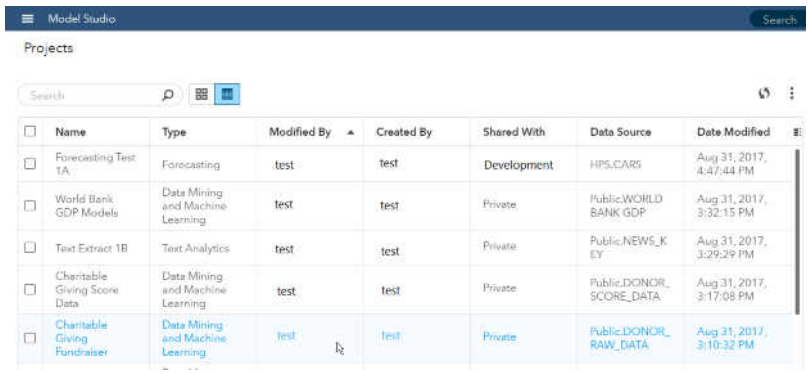
You use the Model Studio Projects page to access existing projects. If your Model Studio Projects page displays project tiles, simply click on the tile that you want to

open. Model Studio will open the selected project. Alternatively, you can click the check box in a project tile to select it, and then right-click the ⋮ icon next to the Project page Toolbox, and choose *Open*. Either approach yields the same results.



The ⋮ icon is at top right on the project page.

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



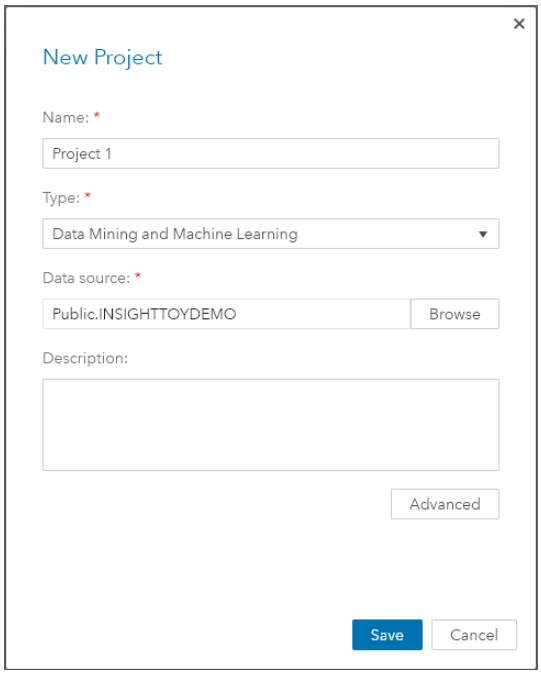
Alternatively, you can click the check box in a project's first column, and then right-click the ⋮ icon next to the Project page Toolbox. Then, choose *Open* to open the project. Either approach yields the same results.

Creating a New Project

You create new Model Studio projects from the Projects page. To create a new project, complete the following steps:

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

2 The New Project window appears.



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 Next you must identify the data source that you want to use. Select the **Browse** button to open the Browse Data window. Use the Browse Data window to select your data source and click **OK**. For more information about using the Browse Data window to choose your data source, see Getting Started with the Choose Data Window

5 If you would like to enter information about the project that might be useful to others, enter that content in the **Description** field.

6 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.

7 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 on page 63 section.

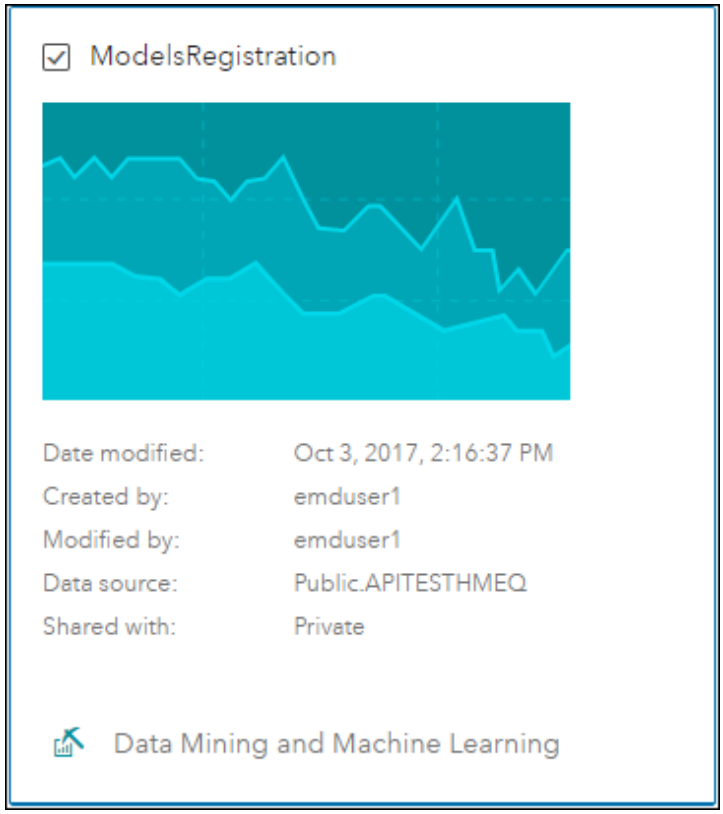
Sharing a Project

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 separate and 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.

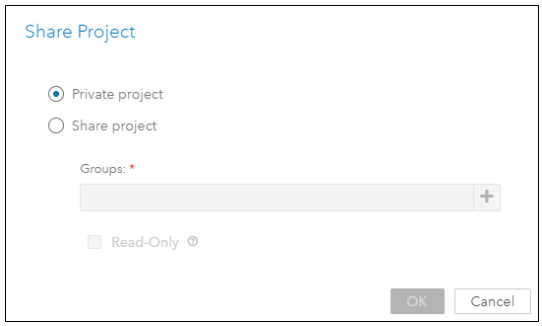
To share a project:

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.

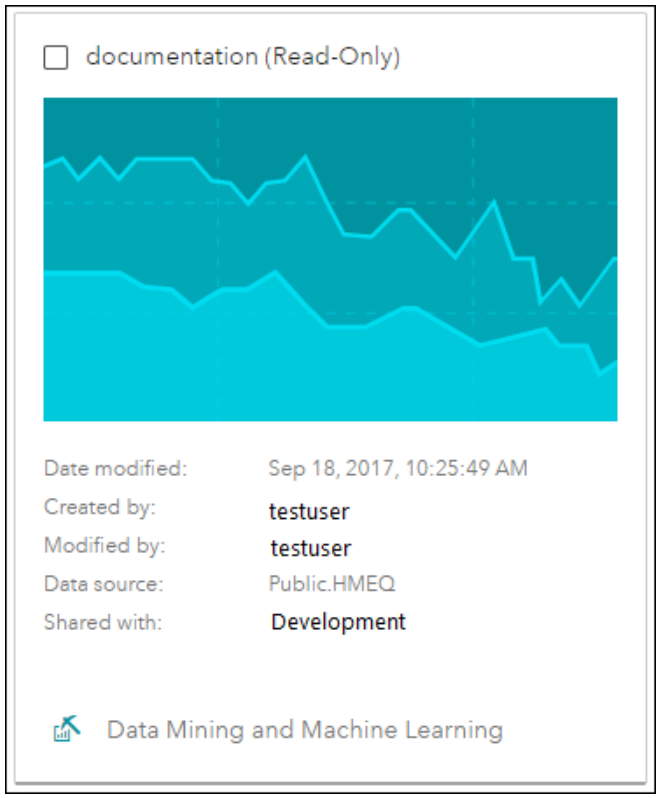
Once groups have been configured, click **OK**.

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 **Read-Only**, then only one person can have the project open at a time. Shared projects that are currently open are indicated with a icon the Projects page.
- If a project is shared **Read-Only**, nobody can make changes to the project, including the project owner.

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 1–3 above, but in Share Project window select **Private project** and click **OK**. This will remove shared access to the project.

Importing and Exporting a Project

To import or export a project, you must belong to the SAS Admin group.

To export a project, complete the following steps:

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.

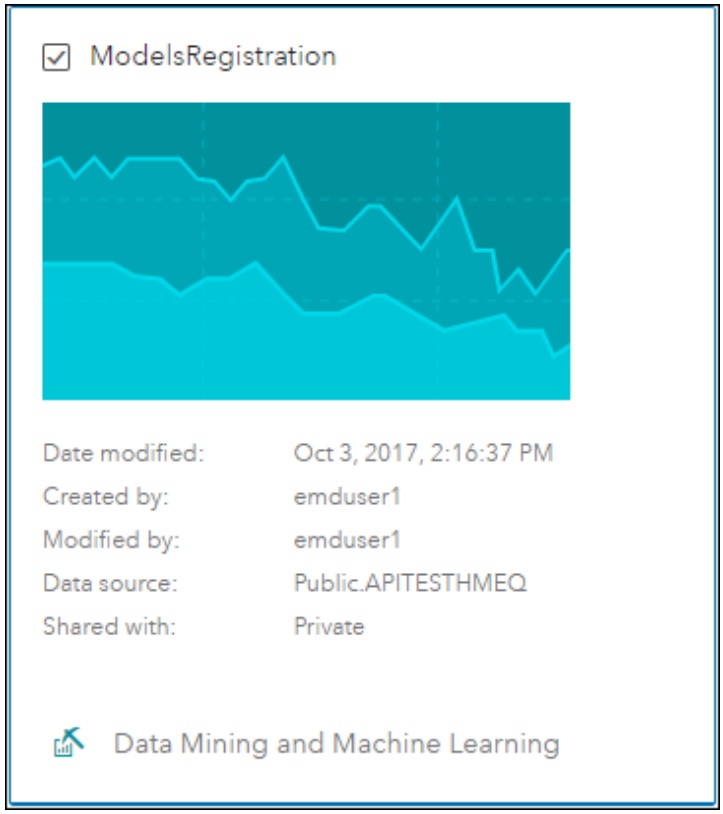
To import a project, complete the following steps:

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

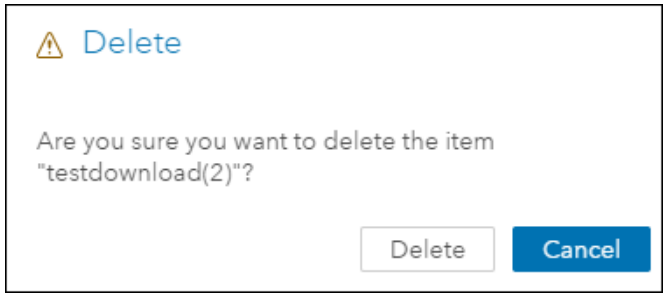
To delete a project, complete the following steps:

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 will appear, asking for confirmation of deletion.

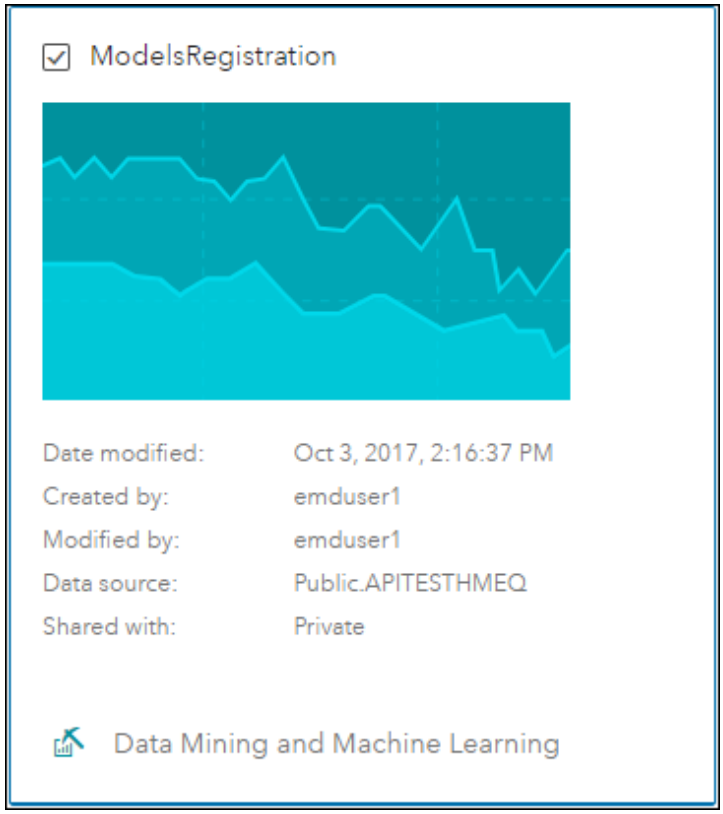


4 Click **Delete**. The projects page will reload, no longer displaying the deleted project.

Downloading Project Batch API Code

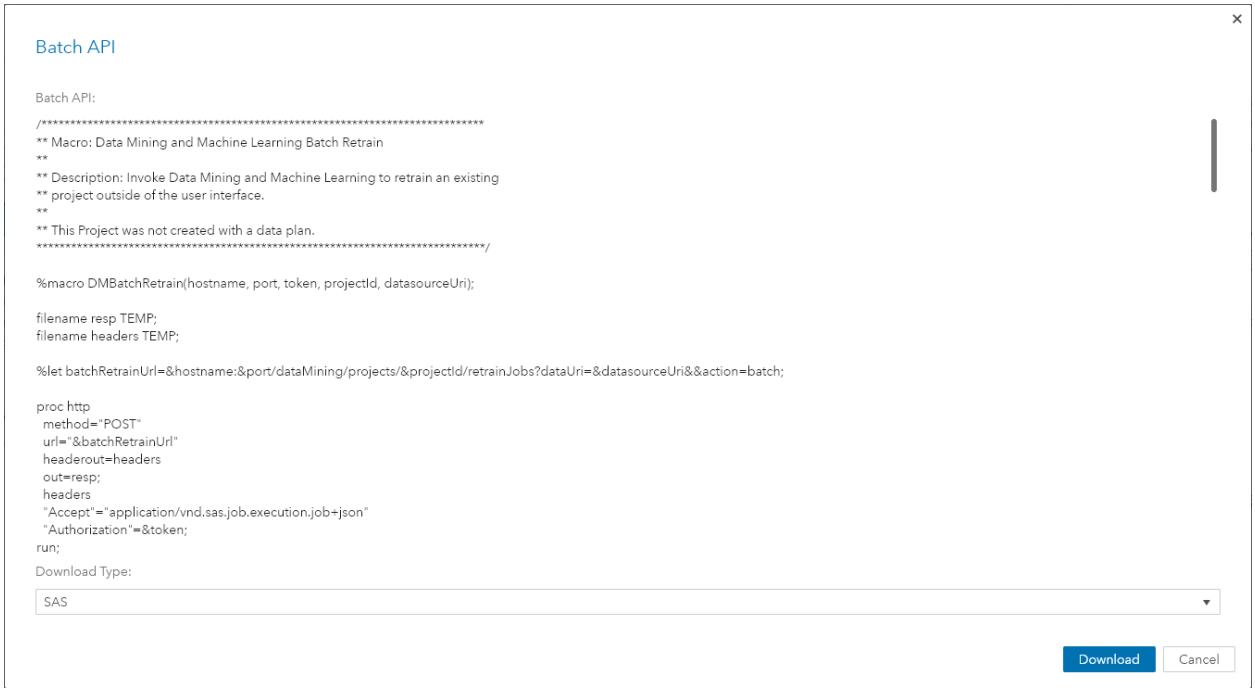
To download project batch API code, complete the following steps:

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 Python, SAS, or REST code.



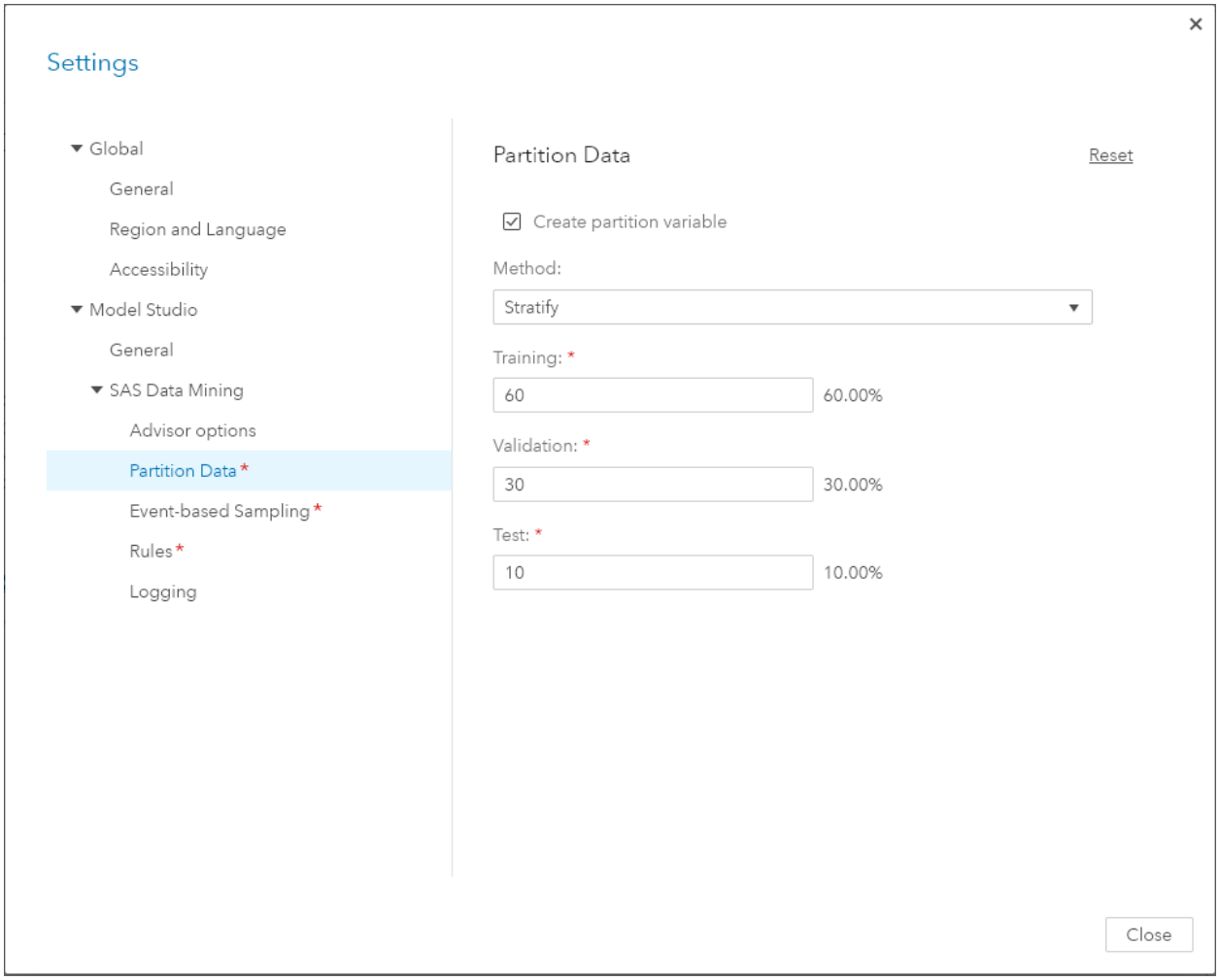
4 Select the download type for the batch code, and click **Download**. The code will begin downloading immediately.

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, complete the following steps:

1 In the upper right corner of the window, click your user name, 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 invert colorings for your interface. 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** — this setting enables you to reset table column preferences to their default values.
- **Advisor options** — these settings enable you to specify the maximum number of class level values, the maximum percentage of missing values, and the interval cutoff value.
- **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:

Training	60%
Validation	30%
Test	10%

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

- **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**.
- **Rules** — these settings enable you to specify the rules used for comparing pipeline models or the default binary classification cutoff value. 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**.

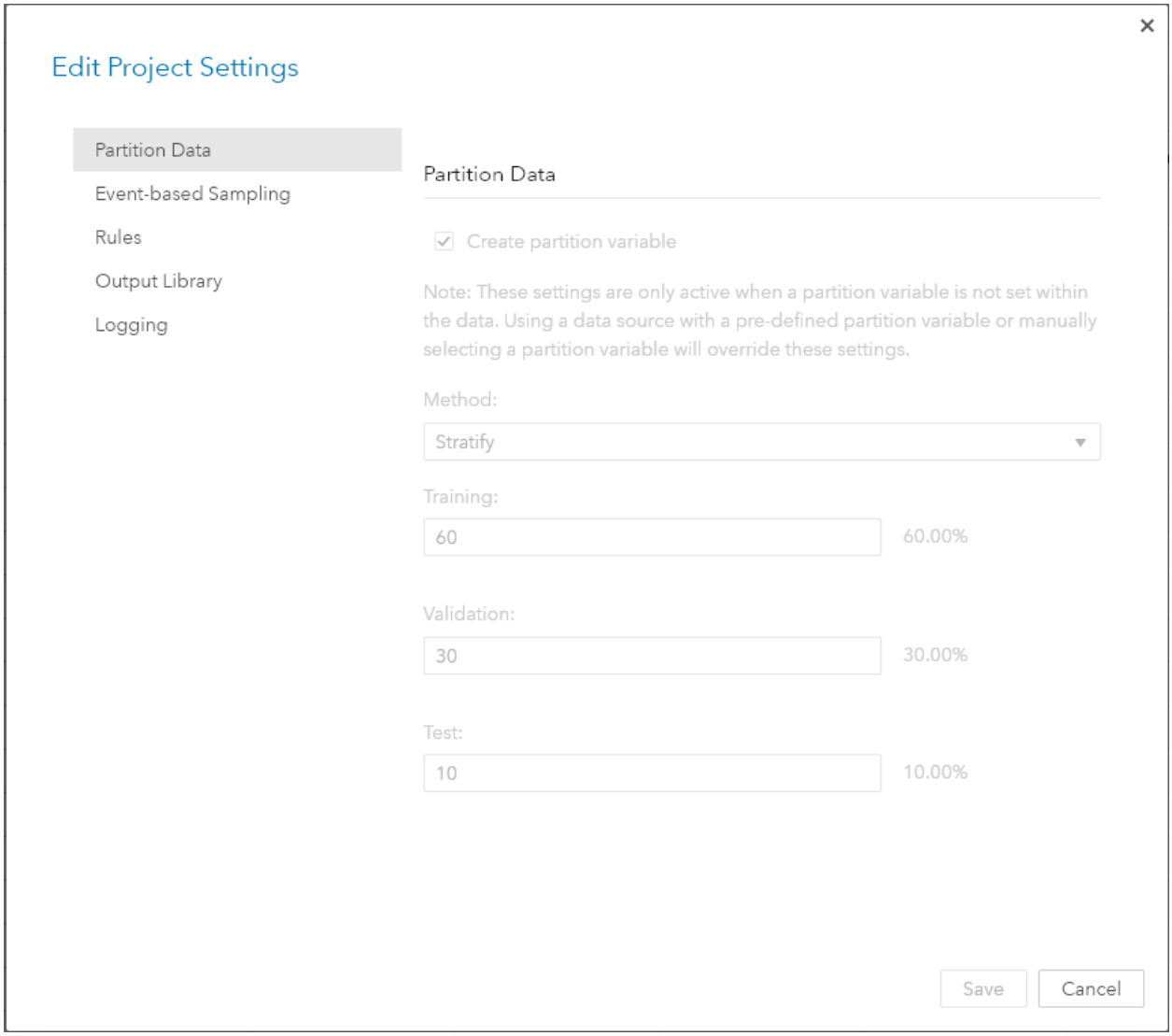
5 Once the settings are appropriate configured, click **Close**. Settings will be automatically saved.

Specifying Project Settings

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

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

2 Select **Project settings**. Selecting this will open the Edit Project Settings window.



3 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:

Training	60%
Validation	30%
Test	10%

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

- **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**.

- **Rules** — these settings enable you to specify the rules used for comparing pipeline models or the default binary classification cutoff. 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**.

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

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

Importing a Project from SAS Visual Analytics

Model Studio users can leverage their SAS Visual Analytics reports to create projects. The following SAS Visual Analytics objects can be used to generate pipelines in Model Studio:

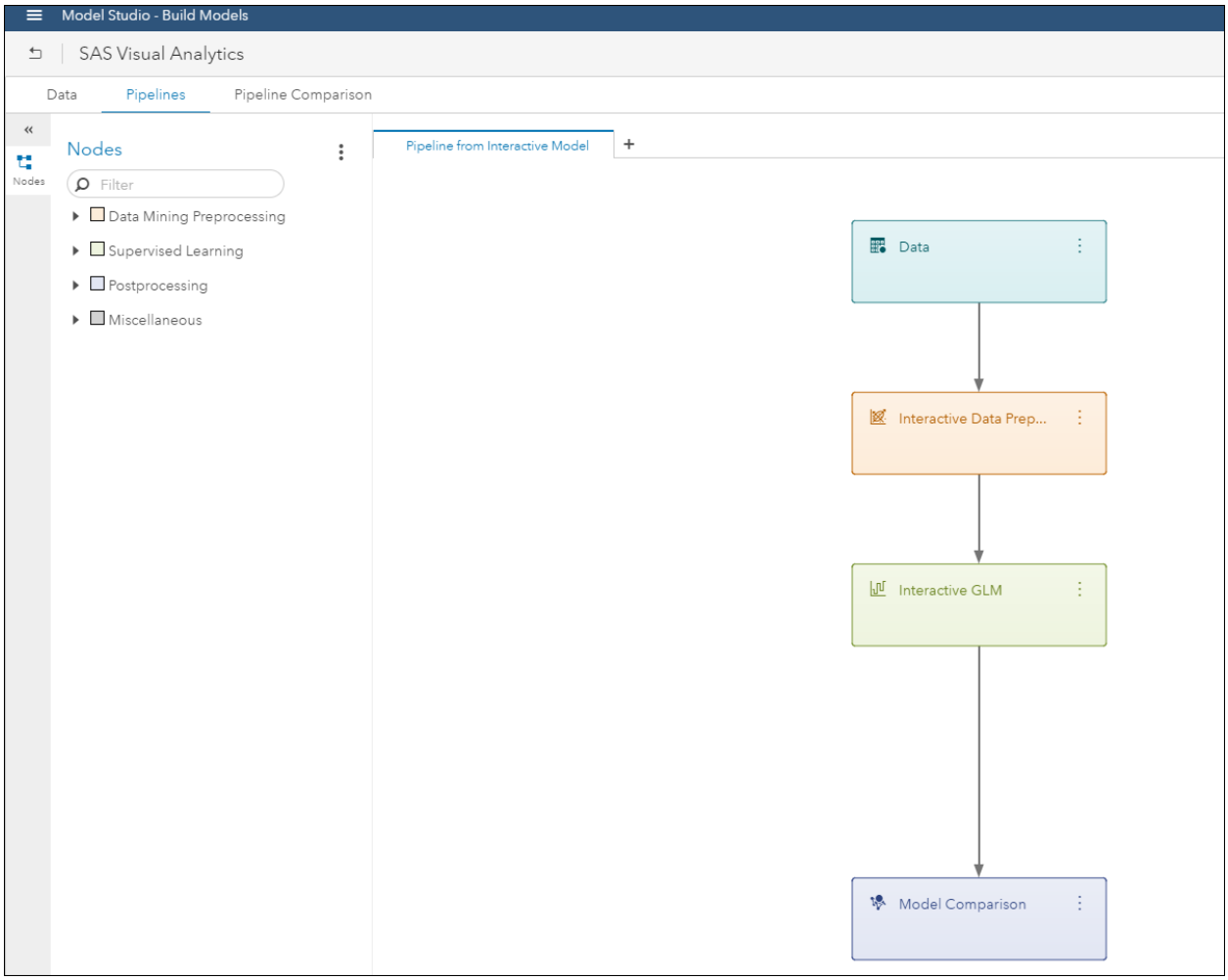
- **Bayesian Network**
- **Decision Tree**
- **Forest**
- **Generalized Linear Model**
- **Gradient Boosting**
- **Linear Regression**
- **Logistic Regression**
- **Model Comparison**
- **Neural Network**
- **Support Vector Machine**

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

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 existing project** is selected, the Model Studio interface will open, and a project will be created in Model Studio named the name of the saved SAS Visual Analytics report. If the report in SAS Visual Analytics has no name, the project will be named *Interactive Project* in Model Studio. If **Add to existing project** is selected, the Model Studio interface will open, and you will be prompted to select an existing project. Only Model Studio projects where the

target variable name, type, and event level match will be available to select from. A pipeline will be created, named *Pipeline from Interactive Model*.



This pipeline looks identical to a normal Model Studio pipeline, except for two key differences:

- Model Studio generates a Data Mining Preprocessing node called **Interactive Data Preparation**. The **Interactive 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, **Interactive GLM** corresponds to the Generalized Linear Model object in SAS Visual Analytics. As with the **Interactive Data Preparation** node, the properties of this node are not available to edit. This pipeline can be run as is, or additional nodes can be added to the pipeline to be run for comparison purposes.

3

Working with Data

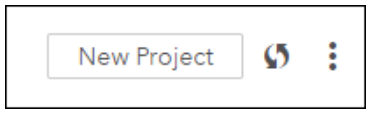
Data Management Overview

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.

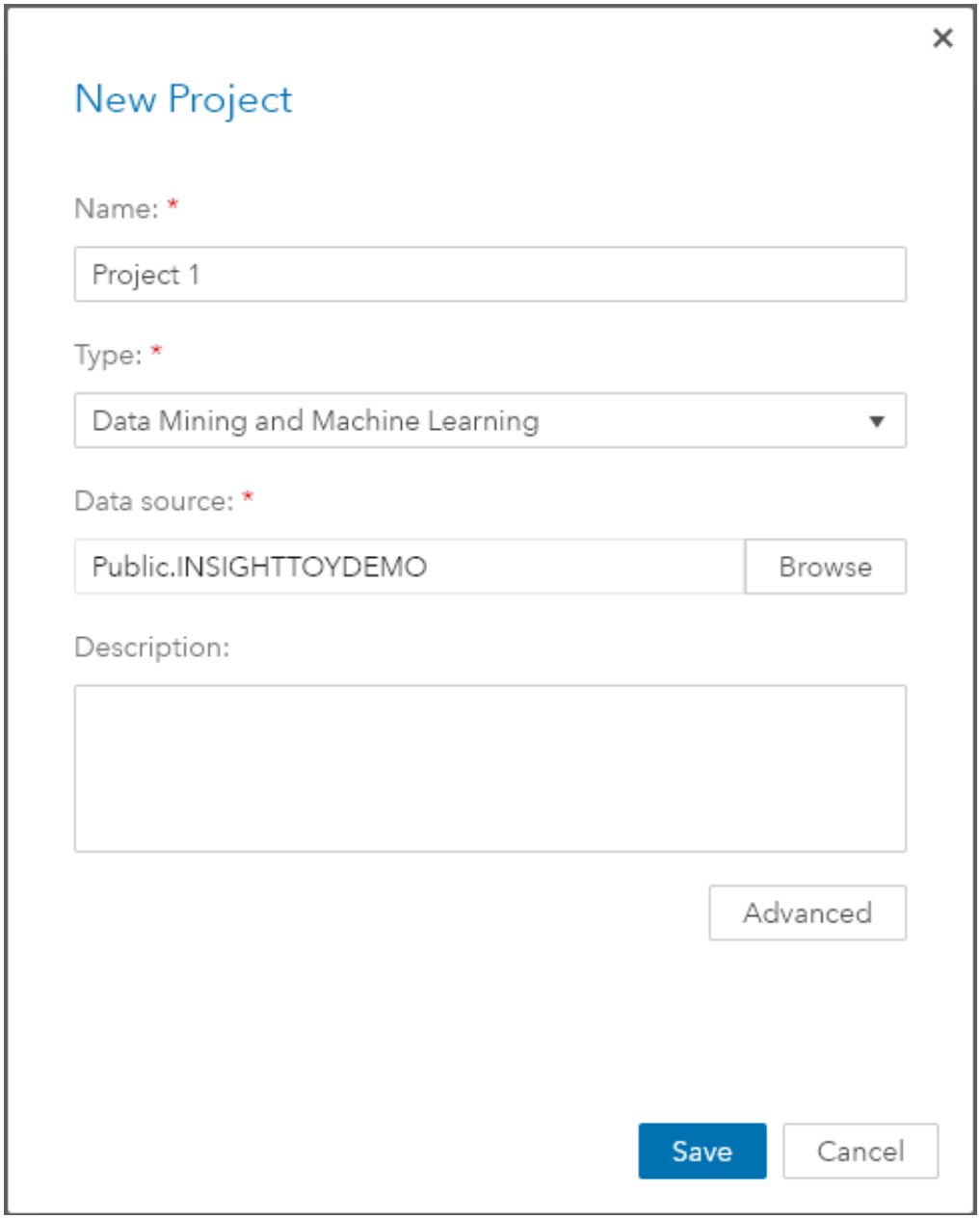
Importing Data

To add a new data set to the repository for use in a new project, complete the following steps:

1 On the Model Studio Projects page, select **New Project** in the upper right corner.



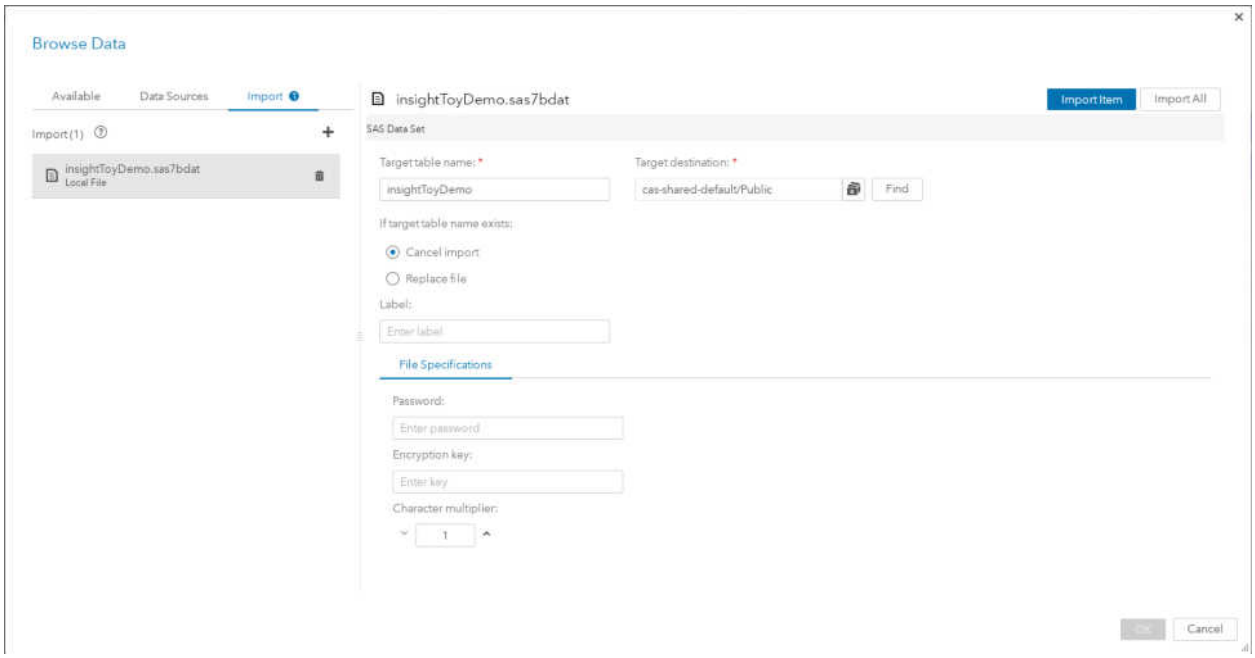
2 The New Project window appears.



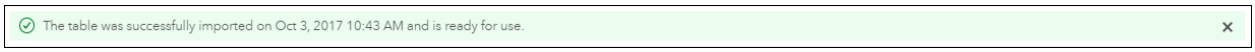
3 Click **Browse** to open the Browse Data window.

4 The Browse 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 will appear:



Click **OK**.

To finish creating a project, follow the steps outlined in Creating a New Project on page 49 .

Note: Data set 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, complete the following steps:

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`**:

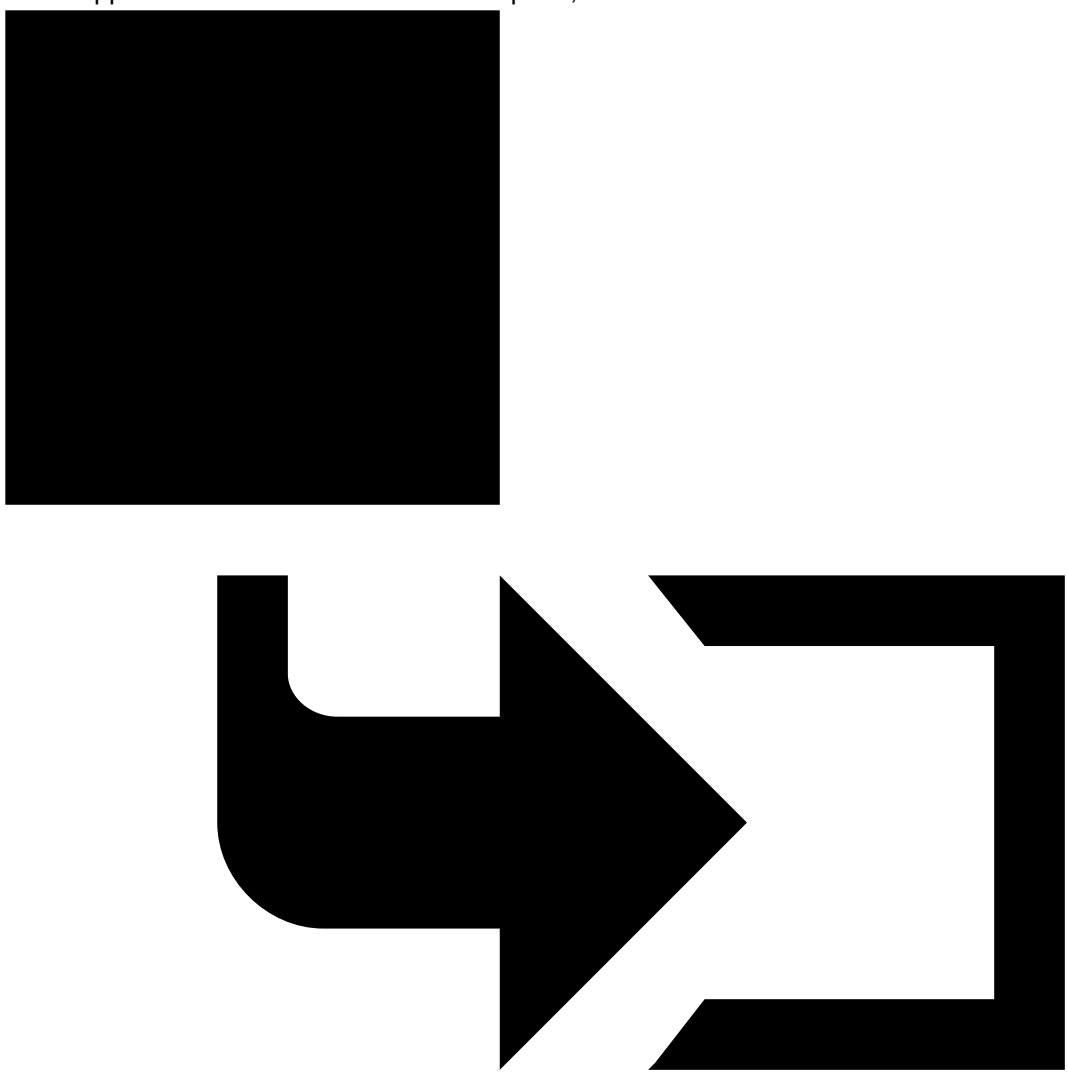
```
libname format '</home/filepath/casuser/>';
            options fmtsearch=(format.emfmt);
```

See Managing User-Defined Formats in SAS Viya in the SAS Viya Administration documentation for more information.

Retraining Model Data

To retrain a model with new data, complete the following steps:

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 Browse 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 contained in Importing Data on page 63.

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 about metadata, see Managing Global Metadata on page 70.

Managing Variable Assignments

Assigning Variable Metadata

To specify variable properties, complete the following steps:

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 fewer than 254 unique values.

- 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:
    - **Bin rare nominal levels**
    - **Default**
    - **Level encoding**
    - **None**

    The **Bin rare nominal levels** option is available only for nominal level variables.
  - The following options are available when selecting transformations for interval variables:
    - **Best**
    - **Bucket binning**
    - **Centering**
    - **Default**
    - **Exponential**
    - **Inverse**
    - **Inverse square**
    - **Inverse square root**
    - **Log**
    - **Log10**
    - **None**
    - **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.
- The following options are available when selecting how to impute missing variable values:
  - The following options are available when selecting how to impute missing class variable values:
    - **Count**
    - **Custom constant value**
    - **Default**
    - **Default constant value**
    - **Distribution**

- **None**

- The following options are available when selecting how to impute missing interval variable values:
  - **Custom constant value**
  - **Default**
  - **Default constant value**
  - **Maximum**
  - **Mean**
  - **Median**
  - **Midrange**
  - **Minimum**
  - **None**
  - **Trimmed maximum**
  - **Trimmed mean**
  - **Trimmed midrange**
  - **Trimmed minimum**
  - **Winsorized maximum**
  - **Winsorized mean**
  - **Winsorized midrange**
  - **Winsorized minimum**

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, complete the following steps:

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 as successful.

To examine and manage the variables designated as global metadata, complete the following steps:

1 Navigate to the Projects page. Click **Open The Exchange** to access the Exchange.

2 In the upper right corner of the Exchange, click **Global Metadata**.

3 The Global Metadata window appears.

This window displays a table that contains all variables specified as global metadata, as well as their metadata assignments.

4 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. In the Delete window that appears, select **Delete** to remove the variable.

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

While the individual nodes in your Model Studio pipeline are Read-only, you can add and delete nodes in this pipeline as in any other Model Studio pipeline.

At this time, the supported models are Bayesian Network, Decision Tree, Generalized Linear Model, Linear Regression, Logistic Regression, Forest, Gradient Boosting, 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.

- 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.
- The nodes that are created in Model Studio contain Read-only score code. This means that you cannot edit or retrain your SAS Visual Analytics models in Model Studio. However, you can edit the pipeline to connect new nodes to the transferred nodes. You can use the Model Studio model comparison and pipeline comparison tools to evaluate your transferred models against any new models.
- 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.
- 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 causes 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 **Interactive 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 **Interactive Data Preparation** node runs.
- Within Model Studio, it is not possible to edit the property settings or options that were used to create the SAS Visual Analytics models that were copied into Model Studio.

4

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, complete the following steps:

1. Select the icon in the upper right corner of the canvas. The Template window will appear.
2. In the Save Pipeline to The Exchange window, enter a **Name** and **Description** for the new template.
3. Click **Save**.

You can also create templates from singular nodes. To create a template from a node, complete the following steps:

1. Right-click on the desired node. Select **Save As**. The Save Node to The Exchange window will appear.

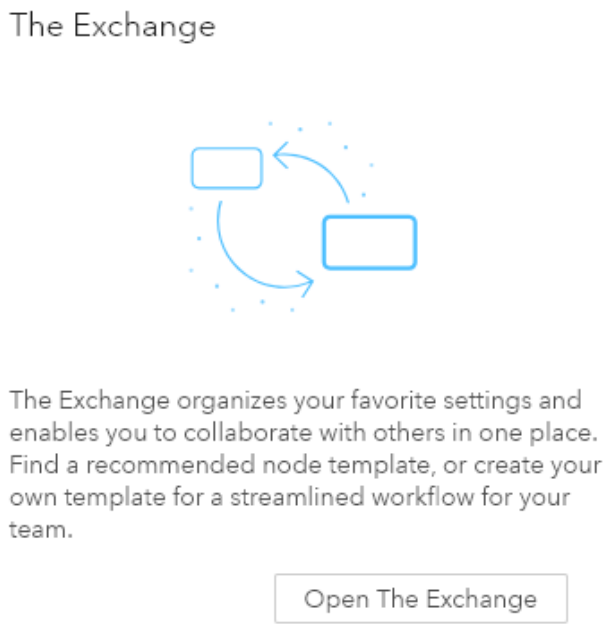
2 In the Save Node to The Exchange window, enter a **Name** and **Description** for the new template.

3 Click **Save**.

Creating a New Template in the Exchange

Templates in the Exchange can be duplicated and subsequently modified. This method enables you to modify the contents of read-only templates without having Edit access to the template itself. To duplicate a template, complete the following steps:

1 Navigate to the Projects page. In the Exchange pane, click **Open The Exchange**.



The Exchange opens. This page enables you to examine all available templates. The Exchange stores node and pipeline templates, as well as templates for 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**

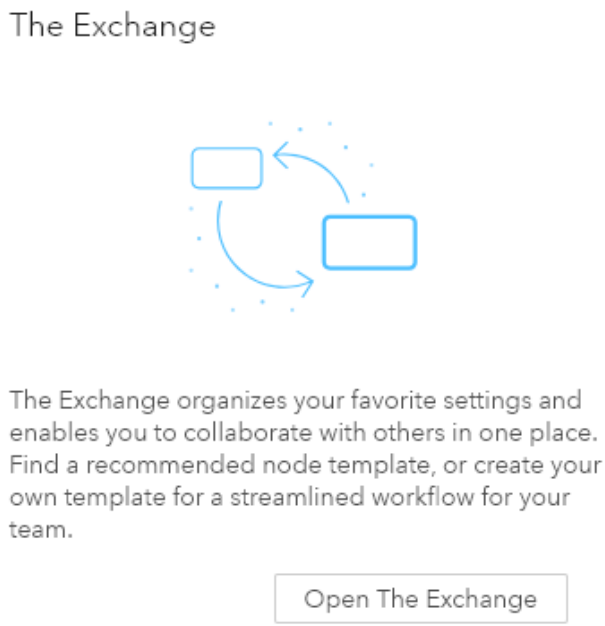
4 The Save Node to The Exchange window appears. In this window, enter a **Name** and **Description** for the new template.

5 Click **Save**. Your new template should appear in the list of templates. Follow the directions in Modifying an Existing Template on page 75 to edit this template.

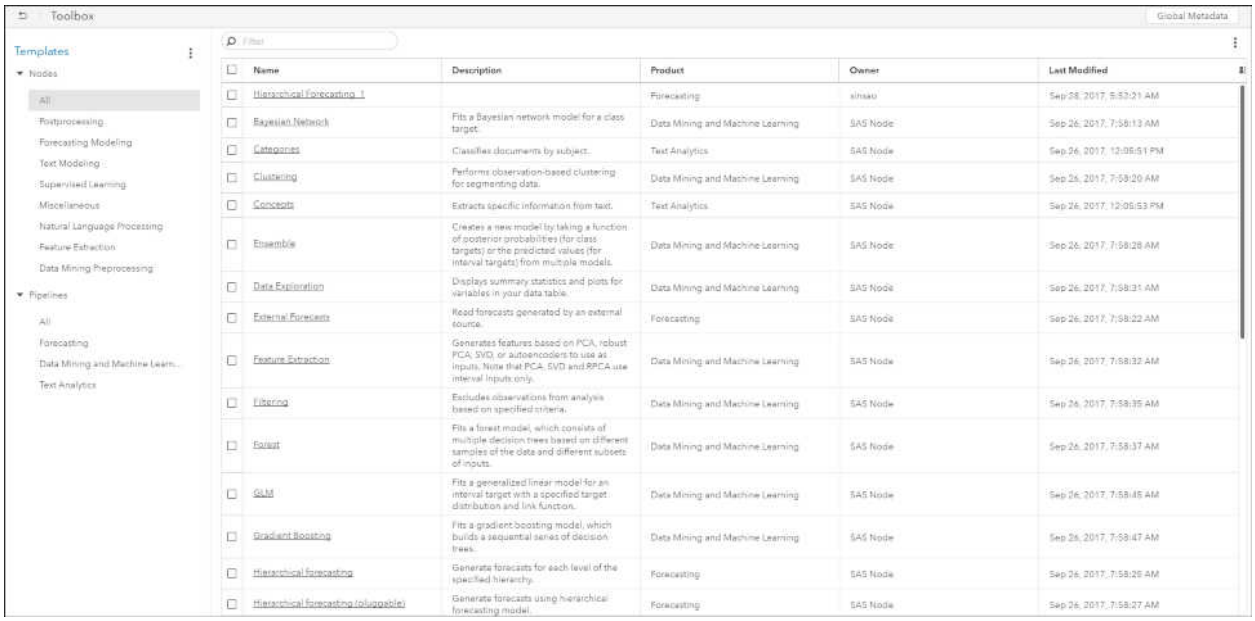
Modifying an Existing Template

If you have sufficient permissions, you can modify existing templates. To modify a template, complete the following steps:

1 Navigate to the Projects page. In the Exchange pane, click **Open The Exchange**.



2 The Exchange opens. This page enables you to examine all available templates. The Exchange stores node and pipeline templates, as well as templates for SAS Visual Text Analytics and SAS Visual Forecasting applications.



3 To access a particular template, click the template name. This will open 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 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:

Node Name	Node Description	Product
Anomaly Detection	Identifies and excludes anomalies (observations) using the support vector data description.	Data Mining and Machine Learning
Auto-forecasting	Generates a simple forecast with ESM or ARIMAX model.	Forecasting
Batch Code	Runs SAS batch code.	Data Mining and Machine Learning
Bayesian Network	Fits a Bayesian network model for a class target.	Data Mining and Machine Learning
Categories	Classifies documents by subject.	Text Analytics
Clustering	Performs observation-based clustering for segmenting data.	Data Mining and Machine Learning
Concepts	Extracts specific information from text.	Text Analytics
Data Exploration	Displays summary statistics and plots for variables in your data table.	Data Mining and Machine Learning
Decision Tree	Fits a classification tree for a class target or a regression tree for an interval target.	Data Mining and Machine Learning
Ensemble	Creates a new model by taking a function of posterior probabilities (for class targets) or the predicted values (for interval targets) from multiple models.	Data Mining and Machine Learning
External Forecasts	Reads forecasts generated by an external source.	Forecasting

Node Name	Node Description	Product
Feature Extraction	Generates features based on PCA, robust PCA, SVD, or autoencoders to use as inputs. Note that PCA, SVD, and RPCA use interval inputs only.	Data Mining and Machine Learning
Filtering	Excludes observations from analysis based on specified criteria.	Data Mining and Machine Learning
Forest	Fits a forest model, which consists of multiple decision trees based on different samples of the data and different subsets of inputs.	Data Mining and Machine Learning
GLM	Fits a generalized linear model for an interval target with a specified target distribution and link function.	Data Mining and Machine Learning
Gradient Boosting	Fits a gradient boosting model, which builds a sequential series of decision trees.	Data Mining and Machine Learning
Hierarchical Forecasting	Generates forecasts for each level of the specified hierarchy.	Forecasting
Hierarchical Forecasting (Pluggable)	Generates forecasts using hierarchical forecasting model.	Forecasting
Imputation	Imputes missing values for class and interval inputs using the specified methods.	Data Mining and Machine Learning
Linear Regression	Fits an ordinary least squares regression model for an interval target.	Data Mining and Machine Learning
Logistic Regression	Fits a logistic regression model for a binary or nominal target.	Data Mining and Machine Learning
Manage Variables	Modifies the metadata of variables.	Data Mining and Machine Learning
Multistage-forecasting	Generates forecasts using multistage forecasting model.	Forecasting
Naive Model Forecasting	Generates forecasts using naive model.	Forecasting
Neural Network	Fits a fully connected neural network model.	Data Mining and Machine Learning
Open Source Code	Runs Python or R code.	Data Mining and Machine Learning
Panel Series Neural Network	Generates forecast using fully connected neural network model.	Forecasting

Node Name	Node Description	Product
Quantile Regression	Fits a quantile regression model for an interval target.	Data Mining and Machine Learning
Replacement	Replaces data values such as outliers and unknown class levels with specified values.	Data Mining and Machine Learning
SAS Code	Runs SAS code.	Data Mining and Machine Learning
Save Data	Saves data exported by a node in a pipeline to a CAS library.	Data Mining and Machine Learning
Score Code Import	Imports SAS score code.	Data Mining and Machine Learning
Sentiment	Analyzes attitudes expressed in documents.	Text Analytics
Stacked Model (NN + TS) Forecasting	Generate forecasts using stacked model (Neural Network + Time Series).	Forecasting
SVM	Fits a support vector machine via interior-point optimization for a binary target.	Data Mining and Machine Learning
Text Mining	Parses and performs topic discovery to prepare text data for modeling.	Data Mining and Machine Learning
Text Parsing	Prepares text for terms analysis.	Text Analytics
Topics	Assigns documents to topics.	Text Analytics
Transformations	Applies numerical or binning transformations to input variables.	Data Mining and Machine Learning
Variable Clustering	Performs variable clustering to reduce the number of inputs.	Data Mining and Machine Learning
Variable Selection	Performs unsupervised and several supervised methods of variable selection to reduce the number of inputs.	Data Mining and Machine Learning

The following Pipeline templates are included with Model Studio:

Pipeline Name	Pipeline Description	Product
Advanced template for class target	Extends the intermediate template for class target with neural network, forest, and gradient boosting models, as well as an ensemble.	Data Mining and Machine Learning

Advanced template for class target with autotuning	Advanced template for class target with autotuned tree, forest, neural network, and gradient boosting models.	Data Mining and Machine Learning
Advanced template for interval target	Extends the intermediate template for interval target with neural network, forest, and gradient boosting models, as well as an ensemble.	Data Mining and Machine Learning
Advanced template for interval target with autotuning	Advanced template for interval target with autotuned tree, forest, neural network, and gradient boosting models.	Data Mining and Machine Learning
Auto-forecasting	Forecasting pipeline with automatic modeling.	Forecasting
Automated feature engineering template	Template to perform automated feature engineering.	Data Mining and Machine Learning
Base Forecasting	Forecasting pipeline with no modeling components added by default.	Forecasting
Basic template for class target	A simple linear flow: Data, Imputation, Logistic Regression, Model Comparison.	Data Mining and Machine Learning
Basic template for interval target	A simple linear flow: Data, Imputation, Linear Regression, Model Comparison.	Data Mining and Machine Learning
Blank Template	A Data Mining pipeline that contains only a data node.	Data Mining and Machine Learning
External Forecasts	Forecasting pipeline with external forecasts.	Forecasting
External Segmentation	Forecasting pipeline with external segmentation.	Forecasting
Hierarchical Forecasting	Forecasting pipeline with hierarchical modeling.	Forecasting
Intermediate template for class target	Extends the basic template with a stepwise logistic regression model and a decision tree.	Data Mining and Machine Learning
Intermediate template for interval target	Extends the basic template with a stepwise linear regression model and a decision tree.	Data Mining and Machine Learning
Text Analytics: Assisted Concept Rule Creation	Use Textual Elements to quickly generate custom concept rules.	Text Analytics
Text Analytics: Data Access	Contains a single Data node.	Text Analytics

Text Analytics: Generate Concepts, Topics, and Categories	Text Analytics pipeline for model generation with Concepts, Text Parsing, Sentiment, Topics, Categories.	Text Analytics
Text Analytics: Topic Discovery	Text Analytics pipeline that uses text parsing and machine learning to discover topics.	Text Analytics

5

Working with Pipelines

Overview of Pipelines

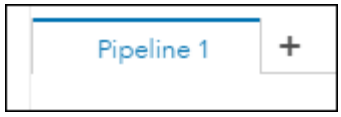
Model Studio projects are built around one or more pipelines. These pipelines are grouped together in a top-level container that also includes the data set that you want to model and a pipeline comparison tool.

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, complete the following steps:

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



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. For more information about templates, see Overview of Templates on page 73 .

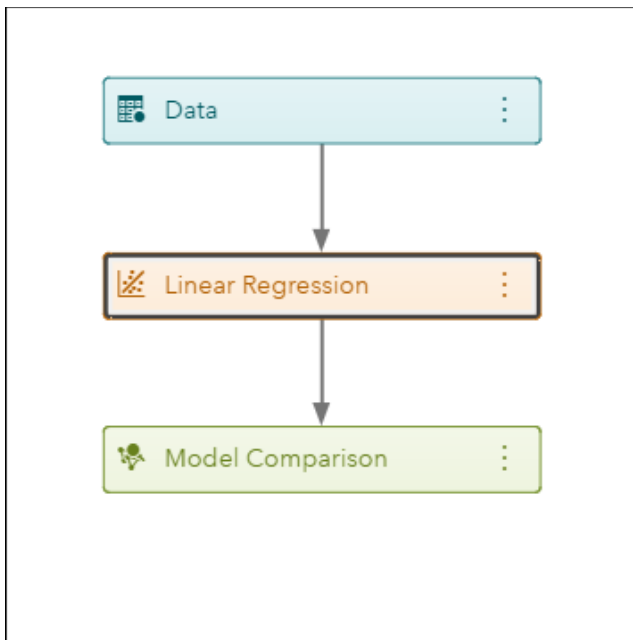
5 Click **Save**.

Modifying a Pipeline

After creating a new pipeline, you are ready to create functionality by adding nodes to the pipeline. There are two ways to add a node to the pipeline:

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 will be added to the canvas, automatically connected to the **Data** node.



In a similar manner, 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 will be added to the

pipeline, connected to the node that it was positioned over. There are some restrictions as to how nodes can be connected to each other. These restrictions are discussed below.

3 Alternatively, right-click on the **Data** node, and select **Add below**. 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 below** (creating more successor nodes) or **Insert above** (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; this will be discussed below.

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.

Creating a Template from a Pipeline

To create a template from a pipeline, complete the following steps:

1 Select the icon in the upper right corner of the canvas. The Save Pipeline to The Exchange window will appear.

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

3 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 selecting the ► icon in the upper right corner of the canvas.
2. Run one branch of the pipeline, only running the selected node, and all nodes preceding that node by arrows. This is done by right-clicking on a node, and selecting **Run**. For the pipeline to have been fully considered as having run, you must use the **Model Comparison** node to run all the nodes in the pipeline.

To interrupt a running pipeline, select the ■ icon in the upper right corner of the canvas.

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 will display the results for the single model. To see a pipeline comparison, select the **Pipeline Comparison** tab. The **Pipeline Comparison** tab will display the champion model, the algorithm used, and error statistics.

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 will also display 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. 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.

Another feature available in the **Pipeline Comparison** tab is the Compare window. TheCompare window allows users to examine the various accuracy statistics of each of the pipelines directly. To compare multiple pipelines, complete the following steps:

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

Managing Models

Register Models

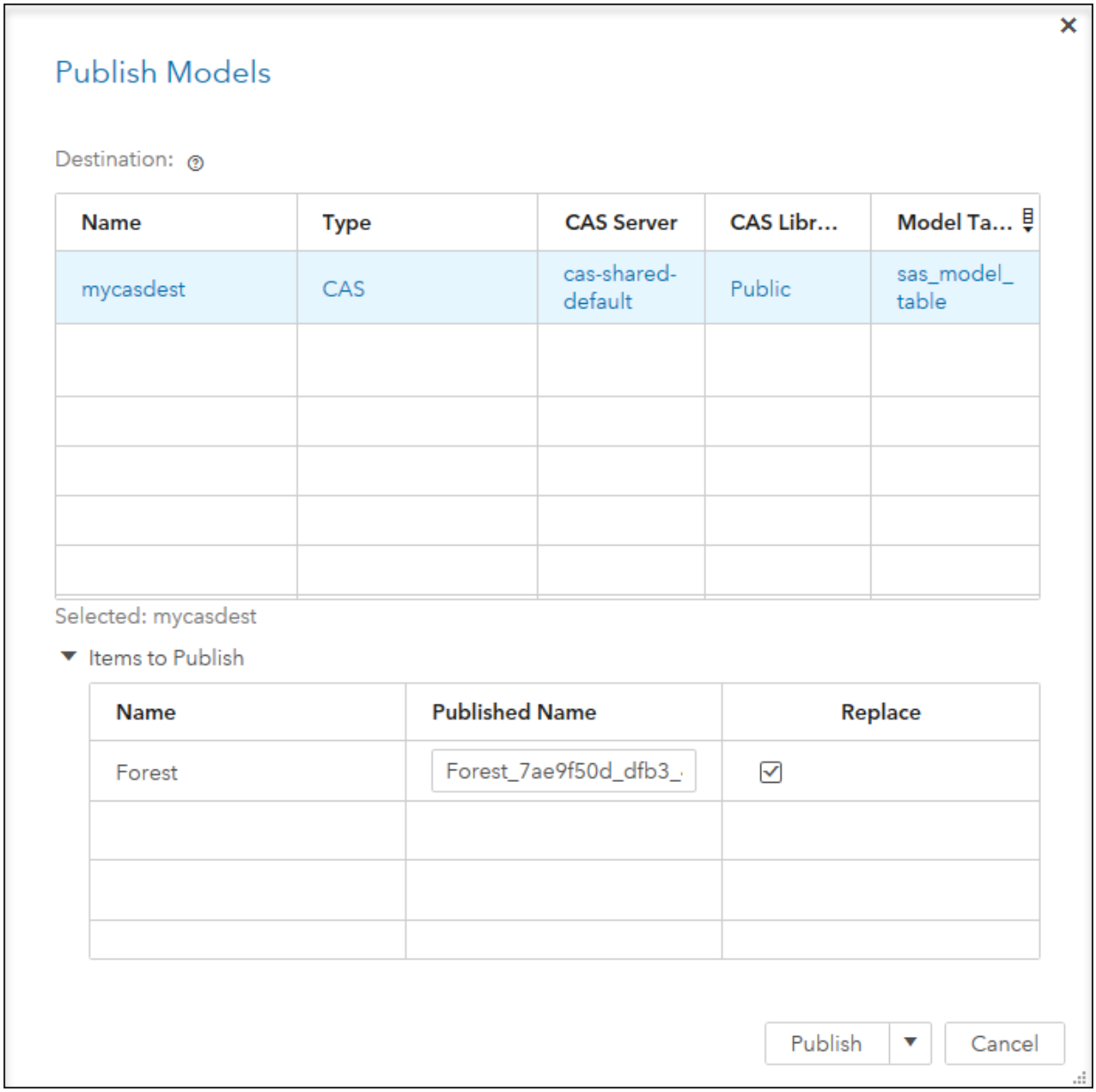
To register a model in SAS Model Manager, click the ⋮ icon in the upper right corner of the **Pipeline Comparison** tab and select **Register models**. This registers the model in SAS Model Manager. SAS Model Manager is used to store and organize models in a common repository. It allows for model governance and model change control over time. See SAS Model Manager: User's Guide for more information.

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 supported are CAS, Hadoop, or Teradata. Information about how to configure publishing destinations can be found in *SAS Viya Administration: Publishing Destinations*.

Once you have created a publishing destination, complete the following steps:

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 models that you want published.



3 Click **Publish**. The model will publish 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.

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.

- In the **Pipelines** tab, choose the pipeline that contains your target model. Right-click the champion node, and select **Download Score Code**.
- In the **Pipeline Comparison** tab, you can select one pipeline, and then select **Download Score Code**.

Both methods listed above will 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 will contain DS2 code and an analytic store. 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 about SAS Environment Manager, see SAS Viya Administration: Using SAS Environment Manager.

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 will be 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:

```
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;
```

Once the score code and any required analytic stores are available in Model Studio to import score code, complete the following steps:

1. Select 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. Select 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**.

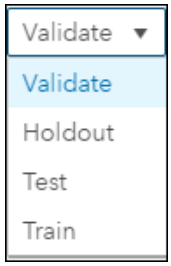
Pipeline Comparison will then incorporate the imported model into its assessment as if it were another pipeline.

Note: When importing a model in Model Studio, the model will be 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, complete the following steps:

1. Select the ⋮ icon in the upper right corner of the **Pipeline Comparison** tab.
2. Select **Score holdout data**.
3. The Browse Data window appears. Select the data set that contains the holdout data that you want scored.
4. Click **OK**.
5. Model Studio will score 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, complete the following steps:

1. Select 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
   - Python
   - REST
4. After selecting the **Download type**, click **Download**. The model API will download immediately.

Downloading Logs

To download project logs, complete the following steps:

1. In the project, select 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 either the **Partition log** or the **Advisor log**. For pipelines that have not yet run, only the **Advisor log** is available.
3. Once the log is selected, click **Download logs**. The log will download immediately, saving as a TXT file.

6

SAS Visual Data Mining and Machine Learning Accessibility

Accessibility

For information about the accessibility of this product, see Model Studio: Accessibility Features.