SAS® Model Manager 15.2: User’s Guide

SAS® Documentation
September 16, 2019

The correct bibliographic citation for this manual is as follows: SAS Institute Inc. 2018. *SAS® Model Manager 15.2: User's Guide*. Cary, NC: SAS Institute Inc.

SAS® Model Manager 15.2: User's Guide

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

September 2019

15.2-P2:mdlmgrug

Contents

1

Introduction to SAS Model Manager

About Managing Models

Using SAS Model Manager, you can store models in a common model repository, and organize them within projects and folders. You can also evaluate models for champion model selection, monitor performance of models, and publish models. All model development and model maintenance personnel, including data modelers, validation testers, scoring officers, and analysts, can use SAS Model Manager.

Here are some of the services SAS Model Manager provides:

- Use a single interface to access all of your business modeling projects. All models are stored in a common model repository. Models can also be accessed in one place using the **Models** category.
- Import models that you develop using a SAS application, such as Model Studio, SAS Visual Analytics, and SAS Studio, as well as SAS code, and PMML models. You can also create a new model with the model's files in a folder or project.
- Compare models to assess candidate models.
- Run tests to validate models for scoring.
- Publish models to CAS, Hadoop, SAS Micro Analytic Service, and Teradata for scoring by external applications or interfaces.
- Create custom workflow definitions to meet your business requirements and to match your business processes. You can then start a workflow process to track the progress of your project.

Data sources are an integral part of the modeling process. Data sources are used for scoring, publishing validation, and performance monitoring. In addition, data sources are used to record adherence to your modeling methodology for audit compliance. Performance data can be created from your operational data, provided that it has the required structure (for example, the data contains a target variable). For information about preparing your data, see *Getting Started with SAS Data Preparation for SAS Viya*.

Sign In to SAS Model Manager

Note: If you are already signed in to SAS Drive, you can access SAS Model Manager by clicking ≡ and selecting **Manage Models**.

To sign in to SAS Model Manager:

1. In the address bar of your web browser, enter the URL for SAS Model Manager and press **Enter**. The **Sign In** page appears.

   Note: If you are in a single sign-on environment, you are not prompted to sign in. Contact your system administrator if you need the URL for SAS Model Manager. The default URL is https://*host_name*/SASModelManager. The last part of the URL (the application name) is case sensitive.

2. Enter a user ID and password.

3. Click **Sign In**.

4. (Optional) If this is your first time signing in to SAS Model Manager, the Welcome to SAS window appears, which enables you to set up a profile. You can select your profile picture and theme, and then watch a short welcome video. You can also set these properties in the Settings window at a later time.

Manage Application Settings

You can use the Settings window to edit user preferences or customize accessibility settings for all SAS web applications. In addition, you can manage model repositories from within the Settings window.

To access the Settings window, click your name in the application bar and select **Settings**.

For information about settings, see the following documentation:

- "Settings" in *General Usage Help for SAS Viya Web Applications*
- "Managing Model Repositories"

High-Level Support Matrix by Model Score Code Type

Here is a summary, arranged by model score code type, of the primary functions that are supported by SAS Model Manager on SAS Viya. There might be additional requirements and restrictions, depending on the function that is being performed. For more information, see the associated section or topic in this document.

Important: The values in the **Model Score Code Type** column within the tables of this topic are associated with the **Score code type** model property. For more information, see "Set Model General Properties" on page 28.

System-Supplied Score Code Types

Note: A model with the score code type of DS2 multi-type can contain code files for a DS2 embedded process, a DS2 package, and one or more analytic stores.

Model Score Code Type	Importing	Scoring	Publishing	Performance Monitoring	Model Comparison and Assessment
Analytic store	Yes	Yes[**]	No	Yes	Yes
DATA step	Yes	Yes	Yes	Yes	Yes
DS2 embedded process	Yes	Yes[**]	Yes[**†]	Yes	Yes
DS2 multi-type	Yes	Yes	Yes	Yes	Yes
DS2 package	Yes	Yes	Yes[††]	Yes[‡]	Yes
PMML[*]	Yes	No	No	Yes[‡]	Yes
SAS program	Yes	Yes[**]	No	Yes	Yes
Additional Score Code Types					
C	Yes	No	No	Yes[‡]	Yes
CAS language	Yes	No	No	Yes[‡]	Yes
Java	Yes	No	No	Yes[‡]	Yes
Lua	Yes	No	No	Yes[‡]	Yes
MATLAB	Yes	No	No	Yes[‡]	Yes
Python	Yes	Yes[***]	No[†††]	Yes[‡]	Yes
R	Yes	No	No	Yes[‡]	Yes

[*] PMML models that are created using PMML 4.2 support DATA step score code. When you are importing valid PMML models, the score code type model property is set to DATA step, instead of to PMML. PMML models with a score code type of DATA step can be scored and published. See "Import Models" on page 21.

[**] Scoring and publishing of decisions that contain a model with a score code type of Analytic store, DS2 embedded process, or SAS program is not supported by SAS Intelligent Decisioning. For more information about scoring or publishing decisions, see *SAS Intelligent Decisioning: User's Guide*.

[***] Models that have a score code type of Python can be scored, if the Python score code is in the correct format. See "DS2 Interface to Python" in *SAS Micro Analytic Service: Programming and Administration Guide*.

[†] Supported only for publishing models to CAS, Hadoop, and Teradata. See "Requirements and Restrictions" on page 47.

[††] Supported only for publishing to SAS Micro Analytic Service. See "Requirements and Restrictions" on page 47.

[†††] Python models with a score code type of DS2 package can be published. See "Preparing Python Models for Publishing" on page 48.

[‡] Supported only for user-provided scored data. See "Monitoring Performance" on page 39.

SAS Viya Models

Models can be built using Model Studio, SAS Visual Analytics, or with SAS Viya modeling procedures within SAS Studio. These models can then be registered to the common model repository.

Product or Tool	Model Score Code Type	Importing	Scoring	Publishing	Performance Monitoring	Model Comparison and Assessment	Retraining
SAS Visual Data Mining and Machine Learning	Analytic store	Yes	Yes[*]	No	Yes	Yes	No
	DATA step	Yes	Yes	Yes	Yes	Yes	Yes[**]
	DS2 multi-type	Yes	Yes	Yes	Yes	Yes	Yes[**]
	Note: The SAS Visual Data Mining and Machine Learning models can be built using Model Studio, SAS Visual Analytics, or with SAS Viya modeling procedures within SAS Studio.						
SAS Visual Statistics	DATA step	Yes	Yes	Yes	Yes	Yes	No
	Note: The SAS Visual Statistics models can be built using SAS Visual Analytics, or with SAS Viya modeling procedures within SAS Studio.						
SAS Visual Text Analytics	SAS program	Yes	Yes[*]	No	Yes	Yes	No
	Note: The SAS Visual Text Analytics models can be built using Model Studio or with SAS Viya modeling procedures within SAS Studio.						

[*] Scoring of decisions that contain a model with a score code type of Analytic store or SAS program is not supported by SAS Intelligent Decisioning. For more information about scoring decisions, see *SAS Intelligent Decisioning: User's Guide*.

[**] Only SAS Visual Data Mining and Machine Learning models that are built using Model Studio can be retrained using SAS Model Manager.

For more information about SAS Viya models, see the following documentation:

- *SAS Visual Data Mining and Machine Learning: User's Guide*
- *SAS Visual Text Analytics: User's Guide*
- *SAS Visual Analytics: Working with SAS Visual Data Mining and Machine Learning*
- *SAS Visual Analytics: Working with SAS Visual Statistics*
- "SAS Viya Procedures" in *SAS Procedures by Name and Product*

Predictive Model Markup and Open-Source Models

You can import models that were created using a programming language other than SAS or an open-source programming language.

Programming Language or Tool	Model Score Code Type	Importing	Scoring	Publishing	Performance Monitoring	Model Comparison and Assessment	Retraining
PMML[*]	DATA step	Yes	Yes	Yes	Yes	Yes	No
	PMML	Yes	No	No	Yes[††]	No	No
Python	Python	Yes	Yes[**]	No[***]	Yes[††]	Yes	No
	DS2 package	Yes	Yes	Yes[†]	Yes[††]	Yes	No

Programming Language or Tool	Model Score Code Type	Importing	Scoring	Publishing	Performance Monitoring	Model Comparison and Assessment	Retraining
R	R	Yes	No	No	Yes[††]	No	No
	DATA step	Yes	Yes	No	Yes	Yes	No

[*] Predictive Model Markup Language (PMML) is an XML-based predictive model interchange format. PMML models that are created using PMML 4.2 support DATA step score code. When importing valid PMML models, the score code type model property is set to DATA step, instead of PMML. PMML models with a score code type of DATA step can be scored and published. See See "Import Models" on page 21.

[**] Models that have a score code type of Python can be scored, if the Python score code is in the correct format. See "DS2 Interface to Python" in *SAS Micro Analytic Service: Programming and Administration Guide*.

[***] Python models with a score code type of DS2 package can be published. See "Preparing Python Models for Publishing" on page 48.

[†] Supported only for publishing to SAS Micro Analytic Service. See "Requirements and Restrictions" on page 47.

[††] Supported only for user-provided scored data. See "Monitoring Performance" on page 39.

SAS 9.4 Models

Models that are created using SAS 9.4 can also be imported into SAS Model Manager 15.2 on SAS Viya 3.4.

Product or Tool	Model Score Code Type	Importing	Scoring	Publishing	Performance Monitoring	Model Comparison and Assessment	Retraining
Base SAS or other code editor	SAS program	Yes	Yes	No	Yes	Yes	No
SAS Enterprise Miner	DATA step	Yes	Yes	Yes	Yes	Yes	No
	PMML[*]	Yes	No	No	No	No	No
	Analytic store	Yes	Yes	No	Yes	Yes	No
	Note: Applies only to SAS analytic store (SASAST) files. SAS package (SPK) files are not supported.						
SAS HPFOREST and HPSVM procedures	Analytic store	Yes	Yes	No	Yes	Yes	No
SAS/STAT linear model procedures	DATA step	Yes	Yes	Yes	Yes	Yes	No

[*] PMML models that are created using PMML 4.2 support DATA step score code. When importing valid PMML models, the score code type model property is set to DATA step, instead of PMML. PMML models with a score code type of DATA step can be scored and published. See See "Import Models" on page 21.

2

Managing Model Repositories

About Model Repositories

You can use model repositories to separate your project and model content, as well as to set permissions for objects within a repository. Some examples are having different repositories for test and production environments, or for different organizations. Model repositories are managed within the Settings window. You can add, delete, and rename a repository. The default repository can be renamed, but it cannot be deleted.

Note: In the May 2019 release of SAS Viya, the Model Repository service default authorization rules for endpoints and repository folders have been modified. Only SAS Administrators and other authorized users can create, update, or delete repositories. In addition, Authenticated Users cannot initially access new custom repositories. A SAS Administrator must grant access for a user or group to a new custom repository. For more information, see “Managing Content” in *SAS Model Manager: Administrator’s Guide*.

To access the **Repository** settings, click your name in the application bar and select **Settings** ⇨ **SAS Model Manager** ⇨ **Repository**.

Note: If you rename the default repository, the Model Repository service must be restarted by a system administrator. For more information, see “Restarting the Model Repository Service” in *SAS Model Manager: Administrator’s Guide*.

Create a New Repository

Note: By default, only SAS Administrators can create new repositories.

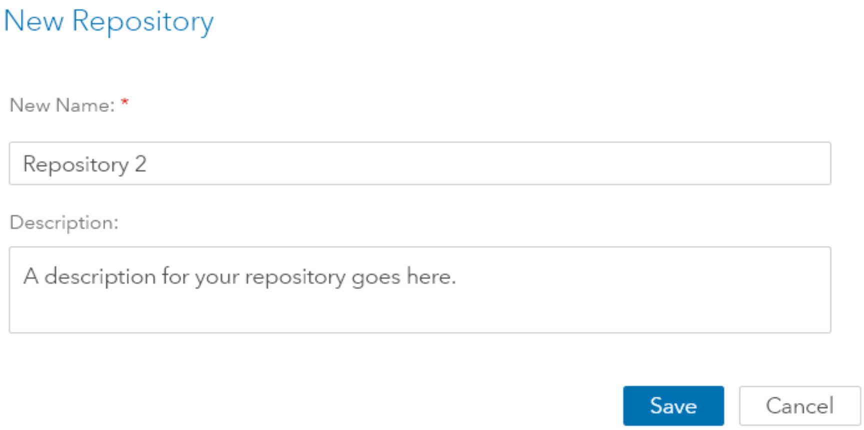
1 Click your name in the application bar and select **Settings** ⇨ **SAS Model Manager** ⇨ **Repository**.

2 Click ✱.

3 Enter a name for the repository.

4 (Optional) Enter a description for the repository.

Note: After you save the new repository, the description cannot be edited.



New Repository

New Name: *

Repository 2

Description:

A description for your repository goes here.

5 Click **Save**.

Rename a Repository

Note: By default, only SAS Administrators can rename repositories.

1 Click your name in the application bar and select **Settings** ⇨ **SAS Model Manager** ⇨ **Repository**.

2 Select a repository, click ⋮ , and select **Rename**.

3 Enter a new name for the repository.

4 Click **Rename**.

Delete a Repository

Note: By default, only SAS Administrators can delete repositories.

1 Click your name in the application bar and select **Settings** ⇨ **SAS Model Manager** ⇨ **Repository**.

2 Select a repository and click 🗑.

3 In the confirmation message, click **Delete**.

3

Working with Projects

About Projects

A project consists of the models, variables, tests, and other resources that you use to determine a champion model. For example, a banking project might include models, data, and tests that are used to determine the champion model for a home equity scoring application. The home equity scoring application predicts whether a bank customer is an acceptable risk for granting a home equity loan.

You create projects within folders. The models within a project are associated with a project version. A project version enables you to group models based on business requirements. The grouping of the models can be for a specific period of time.

Note: The **History** tab of a project shows the primary actions, and the date on which the project was modified and by whom.

Create a New Project

1 Click ⧉ to navigate to the Projects category view.

2 Click **New Project**. The **New Project** window appears.

3 Enter a name for the project.

   Note: The initial version 1.0 is displayed and reflects the level for sequential versions.

4 (Optional) Enter a description for the project.

5 (Optional) Select a model function from the list or enter your own value. The model function indicates the type of output that your predictive model project generates.

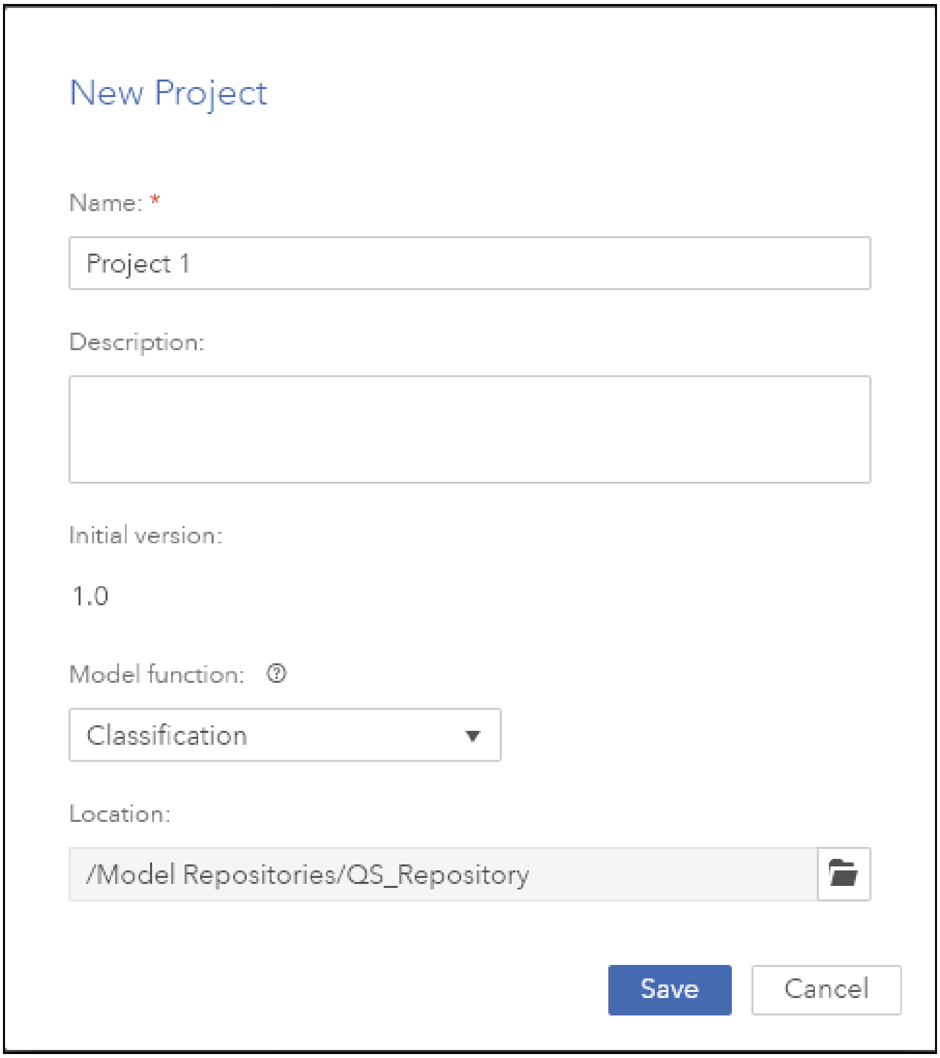
   Note: The value for **Model function** cannot be modified after you click **Save**. If you want to monitor performance of a model, it is recommended to select **Classification** or **Prediction** when creating a project. Only projects with a model function of **Classification** or **Prediction** can be monitored for performance.

6 Accept the default location or select a new location.

   To select a new location, click 📂, select the desired repository or folder, and then click **OK**.

   Note: In the Choose a Location window, you can create a folder within a repository folder to store projects and models for your organization. Repository folders must be created within the **Settings** window. For more information, see "Managing Model Repositories" on page 7.

   It is recommended that the **`/DMRepository`** and **`/VTARepository`** repository folders should be reserved for projects and models that are registered from Model Studio. Also, the **`/VARepository`** repository folder should be reserved for models registered from SAS Visual Analytics. For more information, see "Register Models" in *SAS Visual Data Mining and Machine Learning: User's Guide*.



New Project

Name: *

Project 1

Description:

Initial version:

1.0

Model function:

Classification

Location:

/Model Repositories/QS_Repository

7 Click **Save**.

Creating and Importing Models

After you create a project, you import models into a project version on the **Models** tab. A project can contain multiple versions. You can also copy a model from a folder or another project version. You can view models in all versions or in one selected version on the **Models** tab. After model evaluation, you set one of the candidate models as the champion model and can also set one or more models as challengers.

For more information, see the following:

- "Create a Model within a Project" on page 21
- "Import Models into a Project" on page 23
- "Copy a Model" on page 25
- "Evaluating Models" on page 33

Managing Variables

Input variables and output variables can be added to both project and model objects on the **Variables** tab. The same variable name cannot be used for both an input and output variable.

Add Variables from a Data Source

1 Click the **Variables** tab.

2 Click **Add Variable** and select **Data source**. The **Choose a Data Source** window appears.

Note: If there are no existing variables, click **Add from Data Source**.

3 Select the data source that you want to import the variables from and click **OK**. The **Select Variables** window appears.

4 Select input or output for the variable type.

5 Select the variables that you want to add and click ➡. You can also click ⏩ to add all of the variables from the available items list.

6 Click **OK**.

7 Click 💾.

Add Custom Variables

1 Click the **Variables** tab.

2 Click **Add Variable** and select **Custom variable**. The Add Custom Variables window appears.

Note: If there are no existing variables, click **Add Custom Variables**.

3 Enter a name for the variable.

4 Select a data type and variable type.

5 Expand the **Optional** section to specify a length, measurement, and description for the variable.

6 Click **Add**.

7 Repeat steps 3 through 6 for each variable that you want to add.

8 Click **OK**.

Edit Variables

To edit variables:

1 Click the **Variables** tab.

2 Click on the name of the variable that you want to edit. The Edit Variable window appears.

3 Edit the properties as needed and click **OK**.

4 Click 💾.

Delete Variables

1 Click the **Variables** tab.

2 Select the check box for the variables that you want to delete, click ⋮, and then select **Delete**.

3 Click 💾.

Modifying Project Properties

Project properties contain the project metadata. Project metadata includes information such as the name of the project, the type of project (model function), the project owner, the project identifier, the project location, and of the tables and variables that are used by project processes. The project properties are organized into three types: General, Tags, and User-Defined.

Set Project General Properties

General Properties contains both system-defined properties that you cannot modify, and project specific properties that can be modified, such as the description of the project. None of the project properties are required, except for the name and location.

To set the project general properties, click the **Properties** tab, modify the property values, and then click 💾.

Table 3.1 *List of General Properties*

Property	Description
Name	Specifies the name of the project. A project can be renamed only from the Projects category view.
Description	Specifies the description of the project.
Model function	Specifies the type of output that your predictive model project generates. After it has been declared, the **Model function** property for a project cannot be changed. Ensure that the types of models that you are going to use in the project fit within the selected model function type. For more information, see Table 3.2 on page 15.

Property	Description
Operation status	Specifies the current state of the project: Under Development indicates that the project has started but a champion model is not yet in production. Active indicates that a champion model for this project is in production. Inactive indicates that the champion model is temporarily suspended from production. Retired indicates that the champion model for this project is no longer in production.
Location	Specifies the location of the project in the common model repository.
Champion version	Specifies the project version that contains the champion model.
Champion model name	Specifies the name of the model that is set as the project champion.
Default train table	Specifies the **Default train table** is also used to validate scoring functions or scoring model files when a user publishes the associated project champion model or challenger models to a database. This property is optional.
Target variable	Specifies the name of the target variable that was used to train the model.
Target event value	Specifies the target variable value that defines the desired target variable event.
Target values	For class, nominal, ordinal, or interval targets, the set of possible outcome classes, separated by commas. For example, binary class target values might be **1, 0** or **Yes, No**. Nominal class target values might be **Low, Medium, High**. These values are for information only.
Target level	Specifies the target level of binary, nominal, ordinal, or interval.
Output event probability variable	Specifies the output event probability variable name, when the **Model function** property is set to **Classification**, **Analytical**, **Forecasting**, or **Transformation**. Note: This property is also displayed for user-defined model functions or when the model function property is empty.

Property	Description
Output prediction variable	The output prediction variable name, when the **Model function** property is set to **Prediction**, **Analytical**, **Forecasting**, or **Transformation**. **Note:** This property is also displayed for user-defined model functions or when the model function property is empty.
Output segmentation variable	The output segmentation variable name, when the **Model function** property is set to **Clustering**, **Analytical**, **Forecasting**, or **Transformation**. **Note:** This property is also displayed for user-defined model functions or when the model function property is empty.
UUID	Specifies the universally unique identifier for a project object.
External URL	Specifies a user-defined URL to a project object in another application or to documentation related to the project.
External project ID	Specifies the project ID for a project that was registered from an external application, such as Model Studio.

Table 3.2 Types of Model Functions

Model Function	Description
Analytical	Function for any model that is not Prediction, Classification, or Segmentation.
Classification	Function for models that have target variables that contain binary, categorical, or ordinal values.
Clustering	Function for segmentation or clustering models.
Forecasting	Function for models used to forecast future data based on past data.
Prediction	Function for models that have interval targets with continuous values.
Text analytics	Function for SAS Visual Text Analytics models.
Transformation	Function for models used to determine mathematical functions that can be used to stabilize variances, remove nonlinearity, and correct non-normality in variables to improve the fit of your model.

Add Tags

You can add one or more tags to a project. When you add tags to a project, they are added to a master list of tags that is available to be added to other projects within the same repository.

To add a tag:

1 On the **Properties** tab of a project, select **Tags**.

2 Select an existing tag or enter a name for a tag, and then click ✚.

  Note: The tag name can contain only alphanumeric characters, double-byte characters, the underscore (_), the hyphen (-), and the period (.). Spaces are not allowed.

  Repeat this step for additional tags.

3 Click 💾.

To delete a tag, click 🗑 at the right-side of the row, and then click 💾. The tag is removed from the project.

Add User-Defined Properties

You can add your own project or model properties. The property-value pair is metadata for the project or model.

To add user-defined properties:

1 On the **Properties** tab, select **User-Defined**.

2 Click **Add Property**. The Add Property window appears.

  > **TIP** If user-defined properties already exist, click **Add** above the table.

  a Enter a name for the property.

  b Select a data type for the property.

  c Enter a value for the property.

  d Click **Add** to add the property to the list.

  e Repeat steps a through d for each property that you want to add.

3 Click **Add**.

4 Click 💾.

To edit a property:

1 Click on a property name within the table.

2 Edit the name, the data type, or value of the property.

3 Click **OK**.

4 Click 💾.

To delete properties, select one or more properties in the table, and then click **Delete**.

Managing Project Versions

A project version is a container of models. An initial version is created automatically when you create a project. You can view a list of the project versions on the **Models** tab in the **Version** drop-down list. The latest version is displayed by default. You can also choose to display all versions, create a new version, or manage existing versions from the **Version** drop-down list. When you create a new project version, you can specify a name and description for the version, such as a time interval for a project cycle.

A version is a sequential number that increments by one each time you add a new version. A project can contain multiple editable versions. A project version is used to differentiate collections of models that are meant to solve the project's problem over time-boundaries. Your version might represent a calendar year, a retail season, or a fiscal quarter. A version contains all of the candidate model resources that you need to determine a champion model as well as all champion model resources. For example, you might develop models for a scoring program that determines whether a customer is eligible for a home equity loan.

Create a New Project Version

1. On the **Models** tab, click ▼ in the **Version** drop-down list, and select **New version**. The **New Project Version** window appears.
2. Enter a name for the version, or accept the default name (for example, Version 2).
3. (Optional) Enter a description for the version.
4. Click **Save**.

Manage Project Versions

In addition to creating a new project version, you can edit the description of a version, rename a version, or delete a version.

1. On the **Models** tab, click ▼ in the **Version** drop-down list, and select **Manage versions**. The **Manage Project Versions** window appears.
2. (Optional) Edit the description of a version.
3. Create a new project version.
   a. Click ✱. The **New Project Version** window appears.
   b. Enter a name for the version, or accept the default name (for example, Version 2).
   c. (Optional) Enter a description for the version.
   d. Click **Save**.
4. Rename a project version.
   a. Select a version and click **Rename**. The Rename window appears.
   b. Enter a new name for the version.
5. Delete a project version.

   Note: When only one project version exists, it cannot be deleted. You must also have the appropriate permissions to delete a version.

   a. Select a version and click **Delete**.
   b. Click **Delete** in the confirmation message.
6. Click **Close**.

Delete a Project

Note: You must also have the appropriate permissions to delete a project.

1 In the Projects category view, select one or more projects.

2 Click ⋮ and select **Delete**.

3 In the confirmation message, click **Delete**.

Rename a Project

1 In the Projects category view, select a project, click ⋮, and select **Rename**.

   Note: Open objects cannot be renamed.

2 Enter a new name for the project.

3 Click **Rename**.

Search for Projects

In the **Projects** category view, you can search for projects by name using the search field above the projects list.

You can also search for project objects across applications using the search field in the application bar.

For more information about searching, see "Search" in *General Usage Help for SAS Viya Web Applications*.

4

Working with Models

About Models

You can create new models or import existing models using the SAS Model Manager web application. Models can be stored within a folder or project version in the SAS Model Manager common model repository.

The Models category view enables you to access all of the models in the common model repository in one place. The models can be located in a folder, or a project version. You can import models, create new models, compare models, and export models. You can also search for models. The search field above the list of models enables you to filter the list by model name. The search field in the application bar enables you to search across the web applications that you have access to. You can also filter the search results by object type, modified by, and date modified.

Create a New Model

You can create a new model from one or more model files and store it within a folder or within a project version. When you create a new model in the Models category view, you can choose a repository and folder to store the new model. When you create a model from the **Models** tab of a project, you can select a project version to store the new model.

Create a Model within a Folder

To create a new model in a repository or folder:

1 Click ⛬ to navigate to the Models category view.

2 Click **New Model**. The **New Model** window appears.

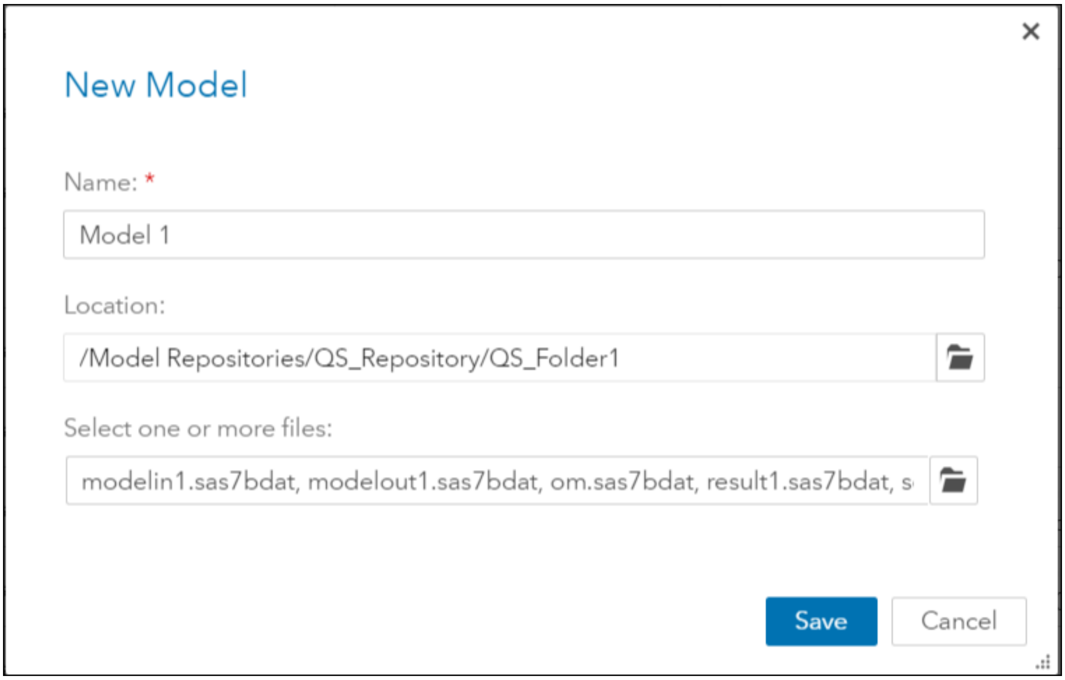
3 Enter a name for the new model.

4 Click 📂 to select a location to store the new model, and then click **OK**.

Note: In the Choose a Location window, you can create a folder within a repository folder to store projects and models for your organization. Repository folders must be created within the **Settings** window. For more information, see "Managing Model Repositories" on page 7.

It is recommended that the `/DMRepository` and `/VTARepository` repository folders should be reserved for projects and models that are registered from Model Studio. Also, the `/VARepository` repository folder should be reserved for models registered from SAS Visual Analytics. For more information, see "Register Models" in *SAS Visual Data Mining and Machine Learning: User's Guide*.

5 (Optional) Click 📂, select one or more files to include within the new model, and then click **Open**.



New Model

Name: *

Model 1

Location:

/Model Repositories/QS_Repository/QS_Folder1

Select one or more files:

modelin1.sas7bdat, modelout1.sas7bdat, om.sas7bdat, result1.sas7bdat, s

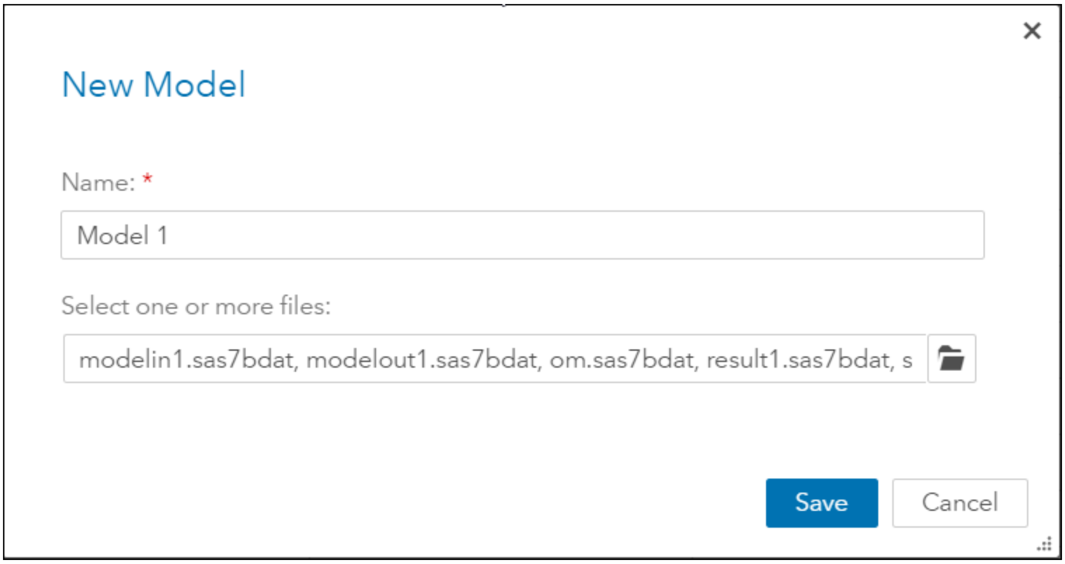
6 Click **Save**. The new model object opens.

Create a Model within a Project

To create a new model within a project version:

Note: By default models are created within the latest project version. You can select a different project version from the **Version** drop-down list.

1. Click 🗐 to navigate to the Projects category view.
2. Open a project.
3. Click **New Model**. The **New Model** window appears.
4. Enter a name for the new model.
5. (Optional) Click 📂, select one or more files to include within the new model, and then click **Open**.



New Model

Name: *

Model 1

Select one or more files:

modelin1.sas7bdat, modelout1.sas7bdat, om.sas7bdat, result1.sas7bdat, s

6. Click **Save**. The new model object opens.

> **TIP** After you are done editing the model content, to return to the project, click 🗐.

Import Models

You can import models into a project or into a folder. Only specific file types can be used to import models into a project or folder. Models that are imported into a folder can then later be moved into a folder or project version.

Note: You cannot score, publish, monitor performance, or run reports for models that are within a folder. Models must be within a project version to perform these tasks.

Restrictions

Here are the file types that can be used to import models:

PMML

a PMML XML file that contains model information. Predictive Modeling Markup Language (PMML) is an XML-based standard for representing data mining results. You can import PMML models that are produced by other applications. PMML 4.2 is supported. Models that are created using PMML 4.2 support DATA step score code. The file extensions can be .xml or .pmml, provided that the file contains valid PMML XML code.

SAS package (SPK) file

a compressed container file that contains a mining result and model component files.

ZIP

an archive file that contains model files. Model files that are associated with a specific model are stored within the ZIP file. The ZIP file can contain model folders at the same level or in a hierarchal folder structure. Each model folder within the ZIP file is imported as a separate model object that contains the contents of the model folder. When you import models from a ZIP file into a project version, the hierarchal folder structure is ignored.

Note: Exporting and re-importing analytic store models that were created using Model Studio is not supported.

Import Models into a Folder

To import models into a folder:

Note: You can import models only into repository folders within the common model repository (**Model Repositories** folder) or folders within a repository folder.

1 Click ⛬ to navigate to the Models category view.

2 Click **Import**. The **Import Models** window appears.

3 Click 📂 to select a location to store the models, and then click **OK**.

   Note: In the Choose a Location window, you can create a folder within a repository folder to store projects and models for your organization. Repository folders must be created within the **Settings** window. For more information, see "Managing Model Repositories" on page 7.

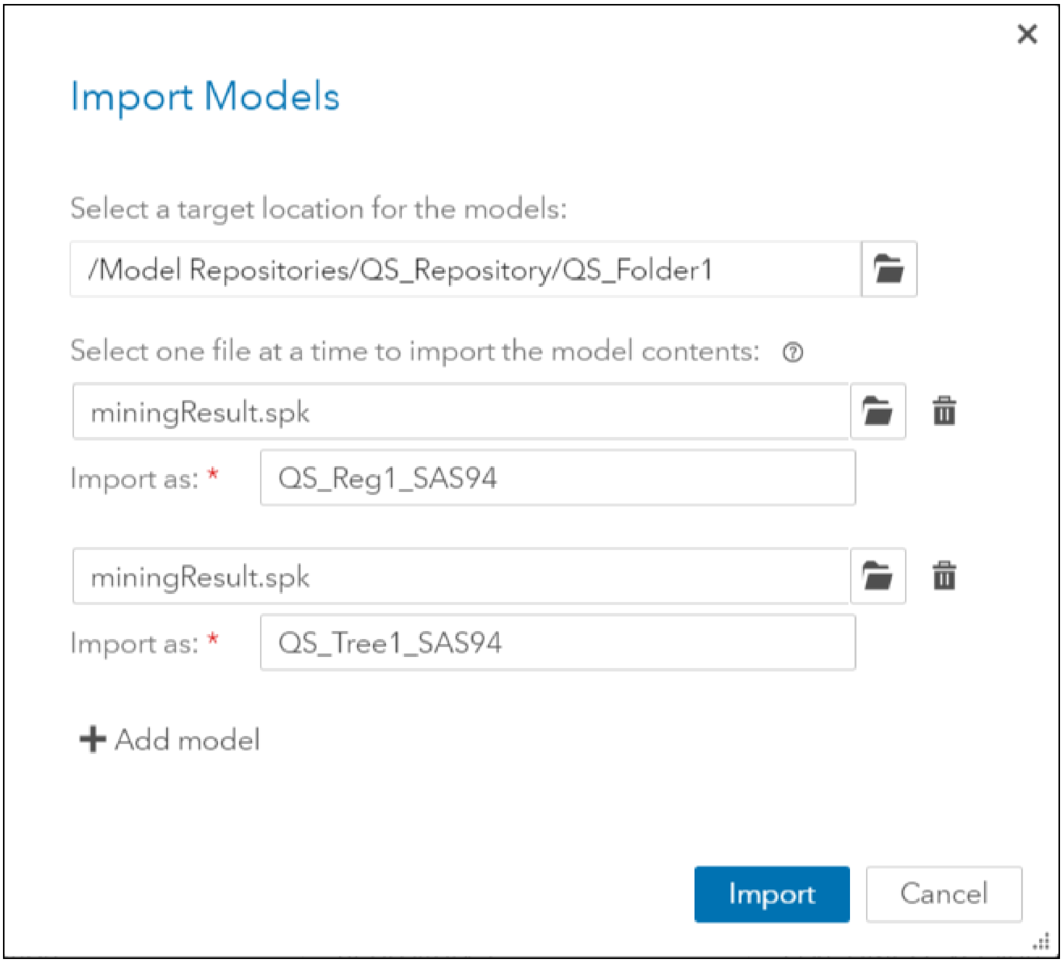
   It is recommended that the `/DMRepository` and `/VTARepository` repository folders should be reserved for projects and models that are registered from Model Studio. Also, the `/VARepository` repository folder should be reserved for models registered from SAS Visual Analytics. For more information, see "Register Models" in *SAS Visual Data Mining and Machine Learning: User's Guide*.

4 Click 📂 to select a file that contains your model contents. Select only one file at a time in the Open window. The name of the selected file is used as the default model name.

   Click **Open**.

5 Click **+ Add model** to add rows so that you can import more models.

6 Repeat steps 4 and 5 until you have selected all of the models that you want to import.



Import Models

Select a target location for the models:

/Model Repositories/QS_Repository/QS_Folder1

Select one file at a time to import the model contents:

miningResult.spk

Import as: * QS_Reg1_SAS94

miningResult.spk

Import as: * QS_Tree1_SAS94

TIP To remove extra lines, click 🗑 before you click **Import**.

7 Click **Import**.

Import Models into a Project

To import model into a project version:

1 Click ⧉ to navigate to the Projects category view.

2 Open a project.

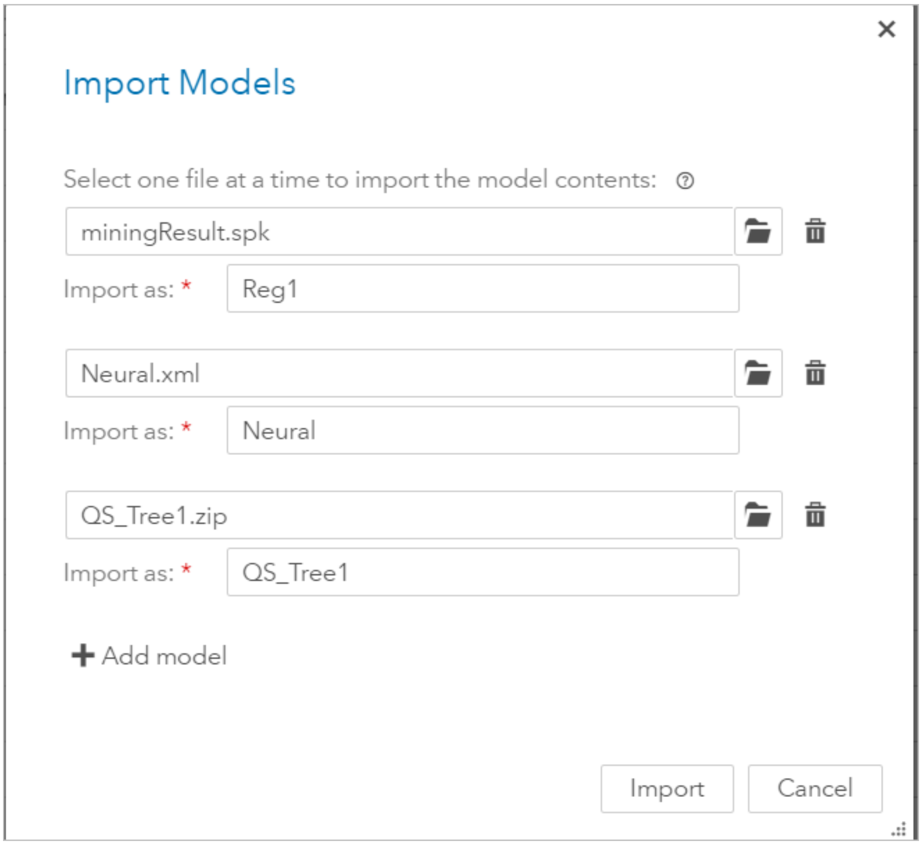
3 Click **Import** and select **Import** from the drop-down list. The **Import Models** window appears.

4 Click 📂 to select a file that contains your model contents. Select only one file at a time in the Open window. The name of the selected file is used as the default model name.

Click **Open**.

5 Click **+ Add model** to add rows so that you can import more models.

6 Repeat steps 2 and 3 until you have selected all of the models that you want to import.



Import Models

Select one file at a time to import the model contents:

miningResult.spk
Import as: * Reg1

Neural.xml
Import as: * Neural

QS_Tree1.zip
Import as: * QS_Tree1

TIP To remove extra lines, click 🗑 before you click **Import**.

7 Click **Import**.

See Also

- "Register a Model" in *SAS Studio: Task Reference Guide*
- %MM_IMPORT_MODEL Macro
- %MM_IMPORT_ASTORE_MODEL Macro

Export a Model

You can export one model at a time from the Models category view or from the **Models** tab of a project.

Note: Exporting and re-importing analytic store models that were created using Model Studio is not supported.

To export a model:

1 Select a model from the list.

2 Click ⋮ and select **Export as ZIP**.

The contents of the model in a ZIP file is downloaded to your local machine.

Move or Copy a Model

You can move a model from a folder to another folder or project version using the Models category view. Only SAS Administrators and users who have Delete permission for the source location where the model resides and Write permission for the target location can move a model. By default, all other users can only copy a model from a folder or another project from the **Models** tab of a project.

For more information, see "Managing Permissions" in *SAS Model Manager: Administrator's Guide*.

Move a Model

To move a model:

1 Click to navigate to the Models category view.

2 Select a model from the list.

3 Click ⋮ and select **Move**. The Choose a Location window appears.

   Note: Only models located within a folder can be moved.

4 Navigate to the folder or project version that you want to move the model to.

5 Click **OK**.

Copy a Model

To copy a model from a folder or another project:

1 Open a project and click the **Models** tab.

2 Click **Import** and select **Copy from** from the drop-down list. The **Choose a Model** window appears.

3 Click › to navigate to a folder or a project version.

4 Click › for the model folder and select the model object. The model object is indicated by the icon .

5 Click **OK**.

Note: Only the latest version of the source model is copied in to the project as a new model object. The initial version for the model is 1.0.

Delete a Model

You can delete one or more models at a time from the Models category view or from the **Models** tab of a project.

Note: Open objects cannot be deleted.

1 Select one or more models.

2 Click ⋮ and select **Delete**.

3 In the confirmation message, click **Delete**.

Rename a Model

You can rename one model at a time from the Models category view or from the **Models** tab of a project.

Note: Open objects cannot be renamed.

To rename a model:

1. Select a model, click ⋮, and select **Rename**.
2. Enter a new name for the model.
3. Click **Rename**.

Manage Model Content and Versions

When you open a model, you can manage model files, add model input and output variables, modify the model properties, and add or view model versions. You can open a model from the Models category view and from the **Models** page of a project.

Managing Model Files

On the **Files** tab of a model, you can add, delete, and download files, as well as assign roles to model files.

You can add any type of file to a model. You can also edit supported file types in the code editor. File types that are not supported by the editor or that are read-only, can be downloaded and viewed by another application.

Add Files

1. Click +. The **Add Model Files** window appears.
2. Click 📂, select one or more files to add to the model, and then click **Open**.
3. Click **Add**.

Assign Model File Roles

You can place your pointer over a model file to view the file properties. Roles might need to be assigned for your model files. To assign a role, select the file and click ☰.

Some roles such as **Score code** are automatically assigned when you import or create new models based on their filenames.

Delete Model Files

Select the file and click 🗑. In the confirmation, click **Delete**.

Download Model Files

Select the file and click ⬇. The model file is downloaded to your local machine.

Managing Variables

Input variables and output variables can be added to both project and model objects on the **Variables** tab. The same variable name cannot be used for both an input and output variable. You can also map model output variables to project output variables.

Add Variables from a Data Source

1. Click the **Variables** tab.
2. Click **Add Variable** and select **Data source**. The **Choose a Data Source** window appears.

   Note: If there are no existing variables, click **Add from Data Source**.
3. Select the data source that you want to import the variables from and click **OK**. The **Select Variables** window appears.
4. Select input or output for the variable type.
5. Select the variables that you want to add and click ➡. You can also click ⏩ to add all of the variables from the available items list.
6. Click **OK**.
7. Click 💾.

Add Custom Variables

1. Click the **Variables** tab.
2. Click **Add Variable** and select **Custom variable**. The Add Custom Variables window appears.

   Note: If there are no existing variables, click **Add Custom Variables**.
3. Enter a name for the variable.
4. Select a data type and variable type.
5. Expand the **Optional** section to specify a length, measurement, and description for the variable.
6. Click **Add**.
7. Repeat steps 3 through 6 for each variable that you want to add.
8. Click **OK**.

Edit Variables

To edit variables:

1. Click the **Variables** tab.
2. Click on the name of the variable that you want to edit. The Edit Variable window appears.
3. Edit the properties as needed and click **OK**.
4. Click 💾.

Delete Variables

1 Click the **Variables** tab.

2 Select the check box for the variables that you want to delete, click ⋮, and then select **Delete**.

3 Click 💾.

Map Output Variables

To set a model as the project champion or as a challenger, run performance, or publish a model, you must map the model output variables to the project output variables. If you do not map the output variables after importing a model, you are prompted to map them when setting a model as the project champion or as a challenger.

Note: You cannot modify the model output variable mappings for a DS2 model that has a score code type of DS2 package, DS2 embedded process, or DS2 multi-type. The names of model output variables must also match the names of the project output variables for a DS2 model. Otherwise, no values are displayed in the **Model Output Variable** column, and you cannot run performance for a model or publish a model.

To map output variables:

1 Click the **Variables** tab of a model.

2 Click ⋮ and select **Map output variables**. The Map Output Variables window appears.

3 Select the model output variables to map with each of the project output variables.

4 Click **OK**.

5 Click 💾.

Modifying Model Properties

Model properties contain the model metadata. Model metadata includes information such as the name of the model, the type of model, the modeler, the model identifier, the name and path of the repository, and of the tables and variables that are used by model processes. The model properties are organized into two types: General and User-Defined.

Set Model General Properties

General Properties contains both system-defined properties that you cannot modify, and model specific properties that can be modified, such as the description of the project.

To set the model general properties, click the **Properties** tab, modify the property values, and then click 💾.

Table 4.1 List of General Properties

Property	Description
Name	Specifies the name of the model. It can be renamed only from the Models category view or the **Models** tab of a project.
Description	Specifies the description of the model.

Property	Description
Location	Specifies the location of the model in the common model repository.
Project name	Specifies the name of the project that contains the model.
Project version	Specifies the project version that contains the model.
Function	Specifies the type of output that your model generates. For more information, see Table 4.2 on page 30.
Score code type	Specifies the type of score code that your model uses. A value must be specified in order for you to be able to publish a model, run a test, or monitor performance for a model. You can select a value from the list or enter your own value. User-defined values are not added to the list. Instead, they are stored within the model properties. **Note:** A model with the score code type of DS2 multi-type can contain code files for a DS2 embedded process, a DS2 package, and one or more analytic stores. Depending on score code type, you can score, monitor, or publish a model. For more information, see "High-Level Support Matrix by Model Score Code Type" on page 2.
Train table	Specifies the **Train table** that is used to validate scoring functions or scoring model files when a user publishes the associated project champion model or challenger models to a database. This property is optional.
Train code type	Specifies the type of train code that your model uses. This property is for informational purposes only. You can select a value from the list or enter your own value. User-defined values are not added to the list. Instead, they are stored within the model properties.
Algorithm	Specifies the computational algorithm that is used for the selected model.
Input variable type	Specifies the type of input variables and whether the variables come from the trainInputVar.json model file or from the inputVar.json model file. Models that are registered from SAS Studio might contain the trainInputVar.json model file. Valid values are "score" and "train". **Note:** If both files are included with the registered model, the property value is set to `score`.
Target variable	Specifies the name of the target variable.
Target event value	Specifies the target variable value that defines the desired target variable event.
Target level	Specifies the target level of binary, nominal, ordinal, or interval.
Output event probability variable	Specifies the output event probability variable name, when the **Model function** property is set to **Classification**, **Analytical**, **Forecasting**, or **Transformation**.

Property	Description
Output prediction variable	The output prediction variable name, when the **Model function** property is set to **Prediction**, **Analytical**, **Forecasting**, or **Transformation**.
Output segmentation variable	The output segmentation variable name, when the **Model function** property is set to **Clustering**, **Analytical**, **Forecasting**, or **Transformation**.
Modeler	Specifies the user ID for the user that built the model.
Tool	Specifies the tool that was used to build the model. An example is Model Studio.
Tool version	Specifies the version number of the tool that is specified in the **Tool** property.
UUID	Specifies the universally unique identifier for a model object.
External model ID	Specifies the model ID for a model that was registered from an external application, such as Model Studio.
External URL	Specifies a user-defined URL to a model object in another application or to documentation related to the model.

Table 4.2 Types of Model Functions

Model Function	Description
Analytical	Function for any model that is not Prediction, Classification, or Segmentation.
Classification	Function for models that have target variables that contain binary, categorical, or ordinal values.
Clustering	Function for segmentation or clustering models.
Forecasting	Function for models used to forecast future data based on past data.
Prediction	Function for models that have interval targets with continuous values.
Text categorization	Function for SAS Visual Text Analytics categorization models.
Text extraction	Function for SAS Visual Text Analytics concepts models.
Text sentiment	Function for SAS Visual Text Analytics sentiment models.
Text topics	Function for SAS Visual Text Analytics topics models.

Model Function	Description
Transformation	Function for models used to determine mathematical functions that can be used to stabilize variances, remove nonlinearity, and correct non-normality in variables to improve the fit of your model.

Add User-Defined Properties

You can add your own project or model properties. The property-value pair is metadata for the project or model.

To add user-defined properties:

1. On the **Properties** tab, select **User-Defined**.
2. Click **Add Property**. The Add Property window appears.

   > **TIP** If user-defined properties already exist, click **Add** above the table.

   a. Enter a name for the property.

   b. Select a data type for the property.

   c. Enter a value for the property.

   d. Click **Add** to add the property to the list.

   e. Repeat steps a through d for each property that you want to add.
3. Click **Add**.
4. Click 💾.

To edit a property:

1. Click on a property name within the table.
2. Edit the name, the data type, or value of the property.
3. Click **OK**.
4. Click 💾.

To delete properties, select one or more properties in the table, and then click **Delete**.

Managing Model Versions

The current version of a model is the latest version in which the model properties and file contents are editable. If you add a new model version manually or perform an action that automatically creates a new model version (such as setting it as the champion model or publishing a champion model from the project level), a snapshot of the model’s contents is taken and a version number is assigned. However, the contents of the new model version that is created can no longer be edited. You can only view the contents of the new model version. Model versions cannot be deleted.

Set the Displayed Version

The displayed version is the version whose information is displayed on the other tabs, such as the **Files**, **Variables**, and **Properties** tabs. The version number for the displayed version appears next to the model name in the object title bar. On the **Versions** tab, a ✓ indicates the displayed version. To change the displayed version, select the version that you want to view, and click **Set Version**.

Note: Here are a few restrictions when creating a new model version.

- The current version of an object is the version that has the highest version number. When you create a new version, SAS Model Manager locks the current version before it creates the new version.
- You cannot save changes to a version that is locked. If you modify a version that is locked and click 🖫, SAS Model Manager asks you if you want to save the changes to the current unlocked version.
- You cannot unlock a version.

Create a New Version

To create a new version:

1. On the **Versions** tab, click **New Version**. The New Version window appears.
2. Select the version type: **Minor** or **Major**. Version numbers follow the format *Major.Minor*. If you select **Major**, the number to the left of the period is incremented. If you select **Minor**, the number to the right of the period is incremented.
3. Click **Save**.

Search for Models

In the **Models** category view, you can search for models by name using the search field above the models list.

You can also search for model objects across applications using the search field in the application bar.

For more information about searching, see "Search" in *General Usage Help for SAS Viya Web Applications*.

5

Evaluating Models

About Evaluating Models

The goal of a modeling project is to identify a champion model that an external scoring application uses to predict an outcome. SAS Model Manager provides tools to evaluate candidate models and declare a project champion model. You can compare and assess models, run a test on a model, and monitor performance of a model. You can also publish models to CAS, Hadoop, SAS Micro Analytic Service, and Teradata, so that you can validate the models within the publishing destination, and the models can be accessed by scoring applications.

Compare Models

You can compare and assess one or more models. When you compare models, the model comparison output includes model properties, user-defined properties, and variables. The model comparison output might also include fit statistics and plots for the models if the required model files are available. The fit statistics, as well as plots for lift and ROC, are produced using the ASSESS procedure. The fit statistics and plots are displayed only for the SAS Visual Data Mining and Machine Learning models that are created using Model Studio. If you were to look at the **Files** tab of a model object, you would see that JSON files (dmcas_fitstat.json, dmcas_lift.json, dmcas_roc.json) are included. These JSON files are used to show the fit statistics and plots when comparing models in SAS Model Manager.

To compare models:

1. Select one or more models.
2. From the Models category view, click ⋮ and select **Compare**.

   On the **Models** tab of project, click **Compare**.

   The **Compare** page appears.
3. Click **Show Differences**. The default is to show all of the comparison model content.
4. Review the differences for the following model information:

   Note: Section titles appear whether the section contains content or not.

   - Model properties
   - User-defined properties
   - Input variables
   - Output variables
   - Target variable
   - Fit statistics
   - Lift and ROC plots
5. Click **Close**.

Test Models

About Testing Models

The purpose of a test is to run the score code of a model and produce scoring results that you can use for scoring accuracy and performance analysis. The test uses the input data source to generate the test output table. If your environment has its own means of executing the score code, then your use of the SAS Model Manager scoring tests is mostly limited to testing the score code. Otherwise, you can use the tests both to test your score code and execute it in a production environment.

Only these models can be scored: those that have the score code model file role assigned, and those that have a score code type of DATA step, SAS program, DS2 package, DS2 embedded process, DS2 multi-type, and

Analytic store. For more information, see “Assign Model File Roles” on page 26 and “Set Model General Properties” on page 28.

When you publish a model, the system creates a publishing validation test. You can edit the publishing validation test to select a test data source and output library. The validation of the model runs within the publishing destination that the model was previously published to. For more information, see “Validate Published Models” on page 37.

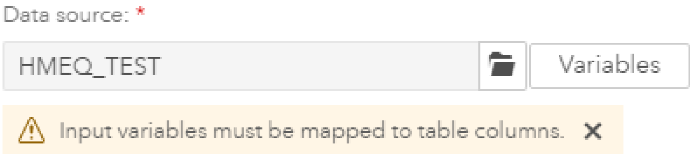
Note: Models that have a score code type of Python can be scored, if the Python score code is in the correct format. For more information, see “DS2 Interface to Python” in *SAS Micro Analytic Service: Programming and Administration Guide*.

Create and Run a New Test

By default, only the user who creates a test definition can view, update, or delete the test definition, as well as run the test and view the test results. For more information, see “Default Permissions” in *SAS Model Manager: Administrator’s Guide*.

1. On the **Scoring** tab of a project, click the **Tests** tab, and then click **New Test**. The New Test window appears.
2. Enter a name for the test if you do not want to use the default name.
3. (Optional) Enter a description for the test.
4. Click **Choose Model** and select a model to test.
5. Click 📂, select the input table for the test, and click **OK**.
6. Map variables.

   Note: SAS Model Manager automatically maps model input variables to the columns in the input table when the names and data types of the variables match those of the table columns. If any input variables cannot be mapped automatically, a warning message is displayed.

   Data source: *

   HMEQ_TEST

   ⚠ Input variables must be mapped to table columns. ✕

   > **TIP** You can change the automatic variable mappings.

   To map variables:

   a. Click **Variables**. The Variable Mappings window appears.

   b. For each input variable, select the table column to which the variable should be mapped.

   c. Click **OK**.
7. (Optional) Expand the **Advanced** section to display the advanced options.
8. (Optional) By default, the library location is the same as that for the input data source. Click 📂 to specify a different library to store the new test output table that is created when the test is run. The input data source and output library must be on the same CAS server.
9. Click **Run** to save and run the test. Alternatively, click **Save** to save the test definition without running it.

   Note: You can also select a check box for a test and click **Run** in the toolbar to rerun a test.

   The status of the test is indicated by the icon in the **Status** column.

Table 5.1 Test Statuses

Icon	Status
●	The test is not ready to run. The test definition is not complete, or it might contain errors.
◉	The test is defined correctly and is ready to run.
⟳	The test is running.
✓	The test completed successfully.
✓⚠	The test completed, but warnings were issued in the SAS log. The URI to the log file is shown on the **Test Results** page.
⊗	The test did not run successfully. Check the SAS log for information. The URI to the log file is shown on the **Test Results** page.

10 Click ▦ in the **Results** column to view the results of the test. The **Test Results** page displays information about the test, including the URIs for the test definition and test results. It also includes URIs to the SAS code that was run by SAS Model Manager, the output data set, and the SAS log that was generated when the code was run.

You can click the **Output**, **Code**, or **Log** pages to view the test result details.

You can also work with the output table in other SAS applications to analyze the data, create and compare models, discover relationships hidden in the data, and generate reports based on the data.

Edit a Test

1 On the **Scoring** tab of a project, click the **Tests** tab.

2 Click on a test name. The Edit Test window appears.

3 Edit the test properties as needed, and then click **Save** or **Run**.

Note: You can also select a check box for a test and click **Run** in the toolbar to rerun a test.

Delete a Test

1 On the **Scoring** tab of a project, click the **Tests** tab.

2 Select one or more tests and click 🗑.

Validate Published Models

About Validating Published Models

When you publish a model, the system creates a publishing validation test. You can edit the publishing validation test to select a test data source and output library. The validation of the model runs within the publishing destination that the model was previously published to.

Only these models can be validated:

- those that have the score code model file role assigned
- those that have a score code type of DATA step, SAS program, DS2 package, DS2 embedded process, DS2 multi-type, and Analytic store

For more information, see "Assign Model File Roles" on page 26 and "Set Model General Properties" on page 28.

Note: Models that have a score code type of Python can be validated, if the Python score code is in the correct format. For more information, see "DS2 Interface to Python" in *SAS Micro Analytic Service: Programming and Administration Guide*.

Edit and Run a Publishing Validation Test

1 Click the **Scoring** tab of a project, and then click the **Publishing Validation** tab.

2 Click on a test name. The Edit Publishing Validation Test window appears.

3 (Optional) Change the name of the test.

4 (Optional) Enter a description for the test.

5 Click 📂, select the input table for the test, and click **OK**.

6 (Optional) Expand the **Advanced** section to display the advanced options.

7 (Optional) By default, the library location is the same as that for the input data source. Click 📂 to specify a different library to store the new test output table that is created when the test is run. The input data source and output library must be on the same CAS server.

8 Click **Run** to run the test. Alternatively, click **Save** to save the test definition without running it.

   Note: You can also select a check box for a test and click **Run** in the toolbar to rerun a test.

   The status of the test is indicated by the icon in the **Status** column. For more information, see Table 5.1 on page 36.

9 Click ▦ in the **Results** column to view the results of the test. The **Test Results** page displays information about the test, including the URIs for the test definition and test results. It also includes URIs to the SAS code that was run by SAS Model Manager, the output data set, and the SAS log that was generated when the code was run.

   You can click the **Output**, **Code**, or **Log** pages to view the test result details.

   You can also work with the output table in other SAS applications to analyze the data, create and compare models, discover relationships hidden in the data, and generate reports based on the data.

Delete a Publishing Validation Test

1. On the **Scoring** tab of a project, click the **Publishing Validation** tab.
2. Select one or more publishing validation tests, click ⋮, and select **Delete**.

Duplicate a Publishing Validation Test

1. On the **Scoring** tab of a project, click the **Publishing Validation** tab.
2. Select one of the publishing validation tests, click ⋮, and select **Duplicate**.

Set Champion and Challenger Models

The champion model is the best predictive model that is chosen from a pool of candidate models. Before you identify the champion model, you can evaluate the structure, performance, and resilience of candidate models. When a champion model is ready for production scoring, you set the model as the champion model. The project version that contains the champion model becomes the champion version for the project. You can publish the champion model to CAS, Hadoop, SAS Micro Analytic Service, and Teradata.

You use challenger models to test the strength of champion models. The champion model for a project can have one or more challenger models. A model can be flagged as a challenger model only after a champion model for the project has been selected. A challenger model can be located in any version of a project.

Set a Champion Model

1. Click the **Models** tab of a project.
2. Select a model, click ⋮, and select **Set as champion**.
3. If the **Select Project Output Variables** window appears, select the model output variables to use as project level output variables. You can use the same variable names or specify different names for the project output variables.

   Note: You cannot modify the names of the project output variables for DS2 models. In addition, in order to monitor performance of SAS Visual Data Mining and Machine Learning 8.4 DATA step models, the project output variables must have the same name as the model output variables, if they are registered from Model Studio.

   Click **Save**.
4. If the model input variables are not project input variables, you are prompted to add the input variables to the project.

   In the confirmation message, click **Yes**.

   Note: If you click **No**, the model is not set as the project champion.

Set a Challenger Model

1. Click the **Models** tab of a project.

2. Select a model, click ⋮, and select **Set as challenger**.

3. If the **Select Project Output Variables** window appears, select the model output variables to use as project level output variables. You can use the same variable names or specify different names for the project output variables.

   Note: You cannot modify the names of the project output variables for DS2 models.

   Click **Save**.

4. If the model input variables are not project input variables, you are prompted to add the input variables to the project.

   In the confirmation message, click **Yes**.

   Note: If you click **No**, the model is not set as a challenger model.

Clear Model Role

1. Click the **Models** tab of a project.

2. Select the champion model or a challenger model, click ⋮, and select **Clear role**.

Monitoring Performance

About Monitoring Performance

To ensure that a champion model in a production environment is performing efficiently, you can collect performance data that has been created by the model at intervals that are determined by your organization. A performance data set is used to assess model prediction accuracy. It includes all of the required variables as well as one or more actual target variables. For example, you might want to create performance data sets monthly or quarterly and then use SAS Model Manager to create a performance definition that includes each time interval.

SAS Model Manager enables you to monitor and evaluate model performance. Performance monitoring can be performed on champion, challenger, and candidate models. Model performance can sometimes be improved by tuning or refitting the model, or by using a new champion model. To monitor performance, you create a performance definition and then you execute it. The output from executing a performance definition includes several charts, such as Variable Distribution, Characteristic, Stability, Lift, Gini, ROC, Kolmogorov-Smirnov (KS), and Average Squared Error (ASE) charts.

You can also create the performance monitoring output by writing your own SAS program using the performance monitoring macros that are provided with SAS Model Manager. You can then submit your program using SAS Studio. The performance results tables that are produced using the macros can then be viewed in SAS Studio, SAS Environment Manager, or SAS Visual Analytics. For more information, see “Performance Monitoring Macros” in *SAS Model Manager: Macro Reference*.

Note: Models with a score code type of **SAS program**, **DATA step**, **DS2 embedded process**, **DS2 multi-type**, or **Analytic store** can be monitored for performance. For more information, see “Set Model General Properties” on page 28.

For more information and examples of the performance monitoring reports, see “Concepts: Performance Monitoring” on page 57.

Set Project Properties

1 Click the **Properties** tab of a project.

2 On the **General** page, specify the following properties:

Classification project

- Target variable
- Target event value
- Target level
- Output event probability variable

Prediction project

- Target variable
- Target level
- Output prediction variable

3 Click 💾.

Create a New Performance Definition

1 Click the **Performance** tab of a project and then click **New Definition**. The New Performance Definition window appears.

2 Select an option from the **Data scoring method** drop-down list.

System scores data
When you choose this data method, the model score code is used to score the data before generating the performance results.

User provides scored data
When you choose this data method, the data tables must contain the predicted values for the scored model.

3 Select an input data method.

Use a single data source
Click 📂, select a data source and click **OK**.

Use a library that contains tables with a specified prefix

- Click 📂, select the data source server, and select a library. Click **OK**.
- Enter a prefix.

Note: The value for the *prefix* cannot contain underscores, and spaces are not recommended in the prefix name or the table name.

Here are rules for data table names:

- They must have at least three levels. This is true for the name of the data table that you use as a data source. It is also true for the name of the data tables that are located in the selected library.
- The second level must be a number.
- An underscore is treated as a delimiter between the levels within the name.
- The sequence number and time label must be unique across all of the data table names.

Use one of the following formats for the name of the data tables:

- *prefix_sequenceNumber_timeLabel*
- *prefix_sequenceNumber_timeLabel_modelUUID*
- *prefix_sequenceNumber_timeLabel_modelUUID_modelRole*

Note: When you select both **User provides scored data** and **Use a library that contains tables with a specified prefix**, your data table names must contain the UUID of the model. Valid values for the model role are **`champion`** or **`challenger`**.

Here are the descriptions and restrictions for the different levels of a table name:

prefix
: The prefix is the first level of the table name and specifies which data tables in a data library to use for performance. The prefix cannot contain underscores, and spaces are not recommended in the prefix name.

sequence number
: The sequence number is the second level of the table name and specifies the order in which the data tables should be used for performance monitoring. The sequence number must be the second level of the table name. The sequence number must be unique across all of the data table names. It is recommended that you start with the number *1* and increase from there.

time label
: The time label is the third level of the table name and specifies the label to use to represent the data in the performance charts. The time label must be unique across all of the data table names. The label can be a period of time such as Q1 or Q12019, or another meaningful label. Spaces are not recommended in the label name.

model UUID
: The model UUID is the fourth level of the table name. If you provide user-scored data and select a library to use for performance monitoring, the data table names must contain the UUID of the model.

model role
: The model role is the fifth level of the table name, and is optional. Valid values are **`champion`** or **`challenger`**.

4 Click **OK**. The **Definition** page appears.

5 Specify the definition details for the chosen data processing method.

- System scores data
- User provides scored data

Specify Definition Details for System Scores Data

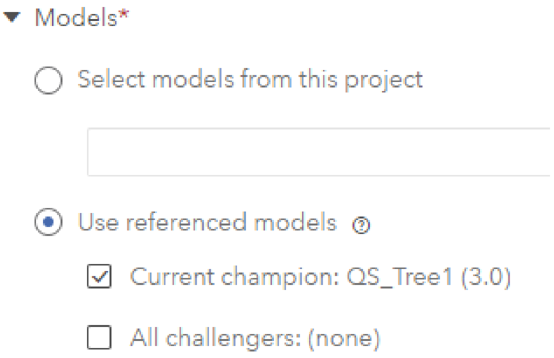
1 Enter a name for the performance definition (for example, *My_Performance_Definition1*).

2 (Optional) Enter a description.

3 (Optional) Click the **Tables** section title to expand it. The single data source or the library and prefix that you previously specified are displayed.

4 Click the **Models** section title to expand it and select the **Use referenced models** option.

5 Accept the default option to use the current champion.



▼ Models*

○ Select models from this project

● Use referenced models

☑ Current champion: QS_Tree1 (3.0)

□ All challengers: (none)

6 (Optional) Click the **Report Settings** section title to view the input and output variables to include in the report results.

> **TIP** By default, all of the input variables and output variables are selected. Click **Choose Variables** to modify which variables are included in the report results.

7 Click the **Project Properties** section title to verify that the required properties have been set.

8 (Optional) Click the **Output Settings** section title to verify the CAS library for the output tables. The default value is `cas-shared-default/modelPerformanceData`. The input data source and the output data library must be on the same CAS server.

9 Click 💾. You are returned to the **Performance** tab.

10 Run performance and view results.

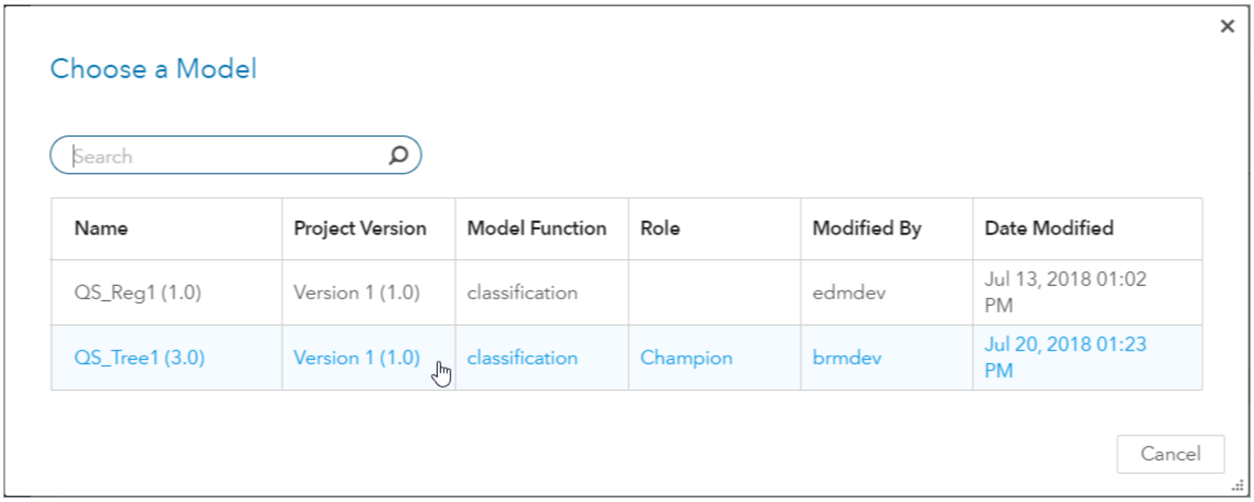
Specify Definition Details for User Provides Scored Data

1 Enter a name for the performance definition (for example, *My_Performance_Definition1*).

2 (Optional) Enter a description.

3 (Optional) Click the **Tables** section title to expand it. The single data table or the library and prefix that you previously specified are displayed.

4 Click the **Model** section title to expand it, click **Choose Model**, and select a model.



Choose a Model

Search

Name	Project Version	Model Function	Role	Modified By	Date Modified
QS_Reg1 (1.0)	Version 1 (1.0)	classification		edmdev	Jul 13, 2018 01:02 PM
QS_Tree1 (3.0)	Version 1 (1.0)	classification	Champion	brmdev	Jul 20, 2018 01:23 PM

5 (Optional) Click the **Report Settings** section title to view the input and output variables to include in the report results.

> **TIP** By default, all of the input variables and output variables are selected. Click **Choose Variables** to modify which variables are included in the report results.

6 Click the **Project Properties** section title to verify that the required properties have been set.

7 (Optional) Click the **Output Settings** section title to verify the CAS library for the output tables. The default value is `cas-shared-default/modelPerformanceData`. The input data source and the output data library must be on the same CAS server.

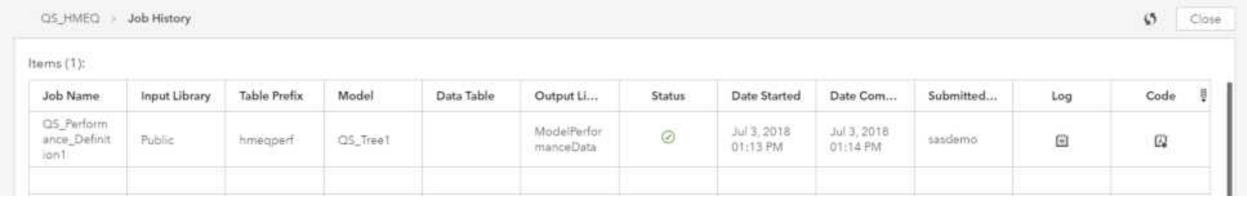
8 Click 💾. You are returned to the **Performance** tab.

9 Run performance and view results.

Run Performance and View Results

1 Click **Run** on the **Performance** tab.

2 Click **View Job History** to view the current status of the performance job.

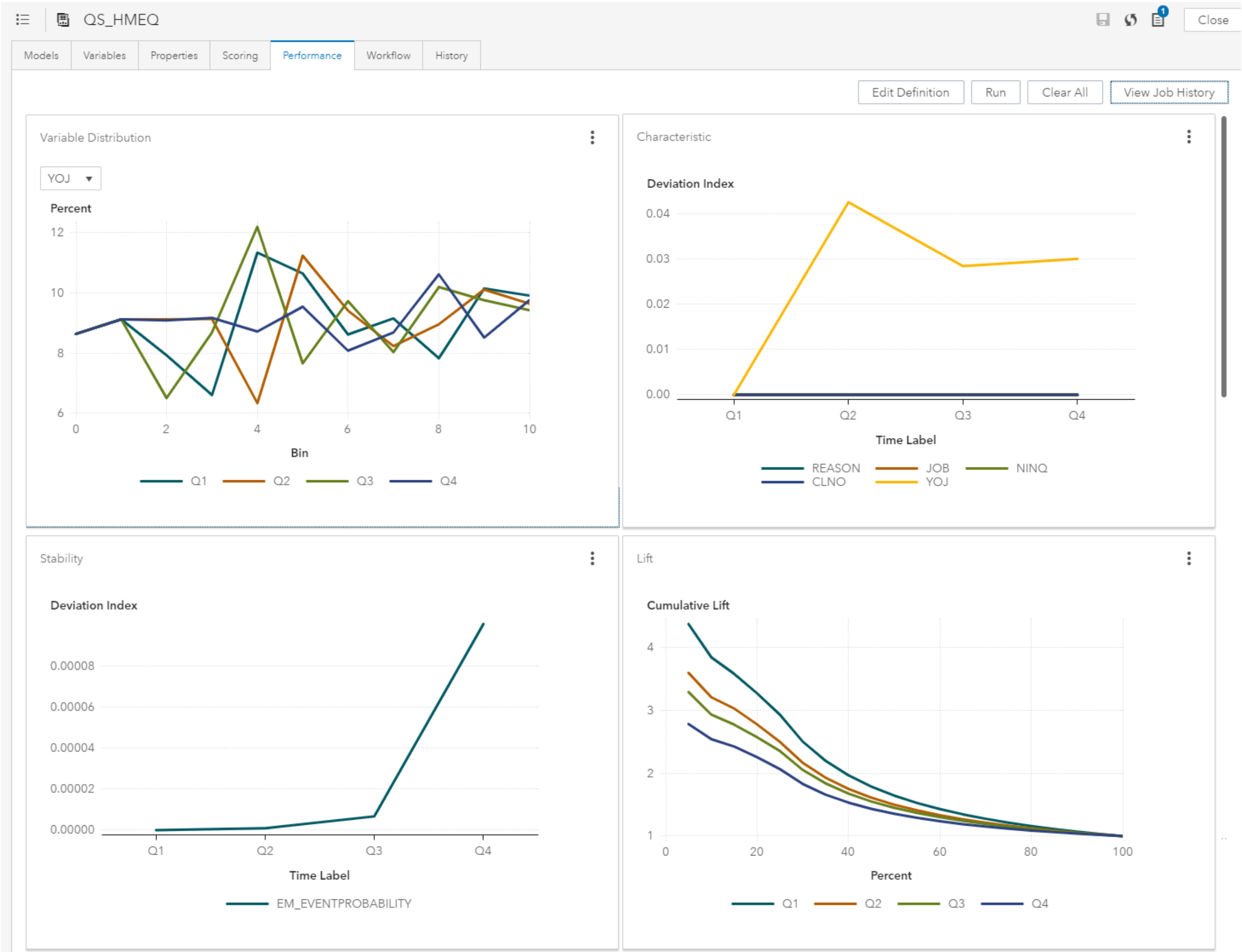


Items (1):

Job Name	Input Library	Table Prefix	Model	Data Table	Output Li...	Status	Date Started	Date Com...	Submitted...	Log	Code
QS_Performance_Definition1	Public	hmeqperf	QS_Tree1		ModelPerformanceData	✓	Jul 3, 2018 01:13 PM	Jul 3, 2018 01:14 PM	sasdemo		

3 When the job is complete, click **Close** to return to the **Performance** tab and view the results.

Figure 5.1 Example of Performance Results for a Classification Model with a Binary Target



4 (Optional) Click ⋮ and select **Explore and Visualize Data**.

Note: SAS Visual Analytics opens within the same browser window, and the performance data for the associated chart is shown. For more information, see *SAS Visual Analytics: Working with Report Content*.

To return to your project, click ≡ and select **Manage Models**.

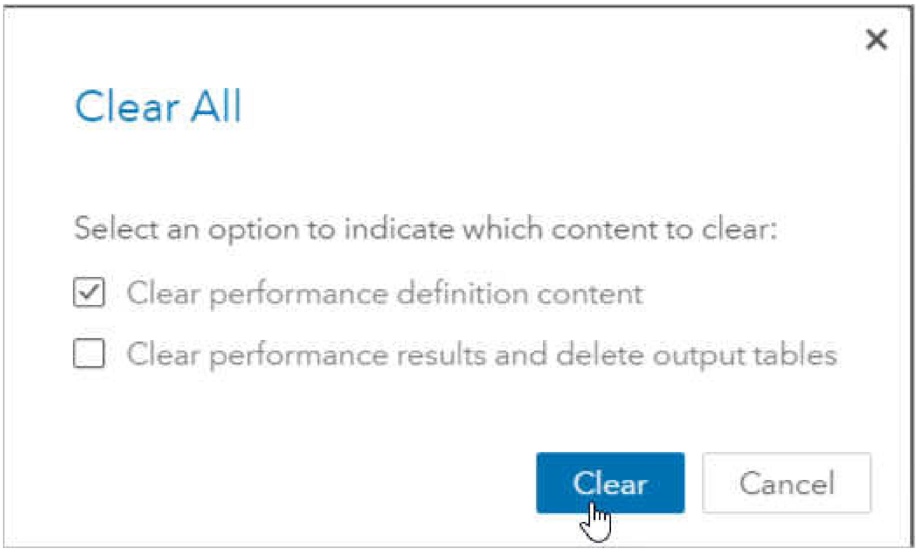
5 Click **Close**.

Clear Definition Content or Performance Results

You can clear the content of an existing performance definition, or clear performance results and delete output tables.

1 Click **Clear All** on the **Performance** tab.

2 Select the check box for the definition content or data that you want to clear.



Clear All

Select an option to indicate which content to clear:

☑ Clear performance definition content

□ Clear performance results and delete output tables

3 Click **Clear**.

Retrain a Project from Model Studio

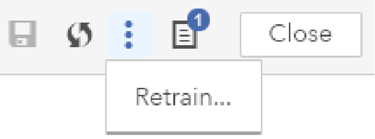
Only SAS Visual Data Mining and Machine Learning projects that have been registered from Model Studio into the SAS Model Manager common model repository can be retrained. Model studio projects and their models are stored in the `DMRepository` repository folder when they are registered.

To send a retrain request for a Model Studio project:

1 Click ⧉ to navigate to the Projects category view.

2 Open a project that was registered from Model Studio.

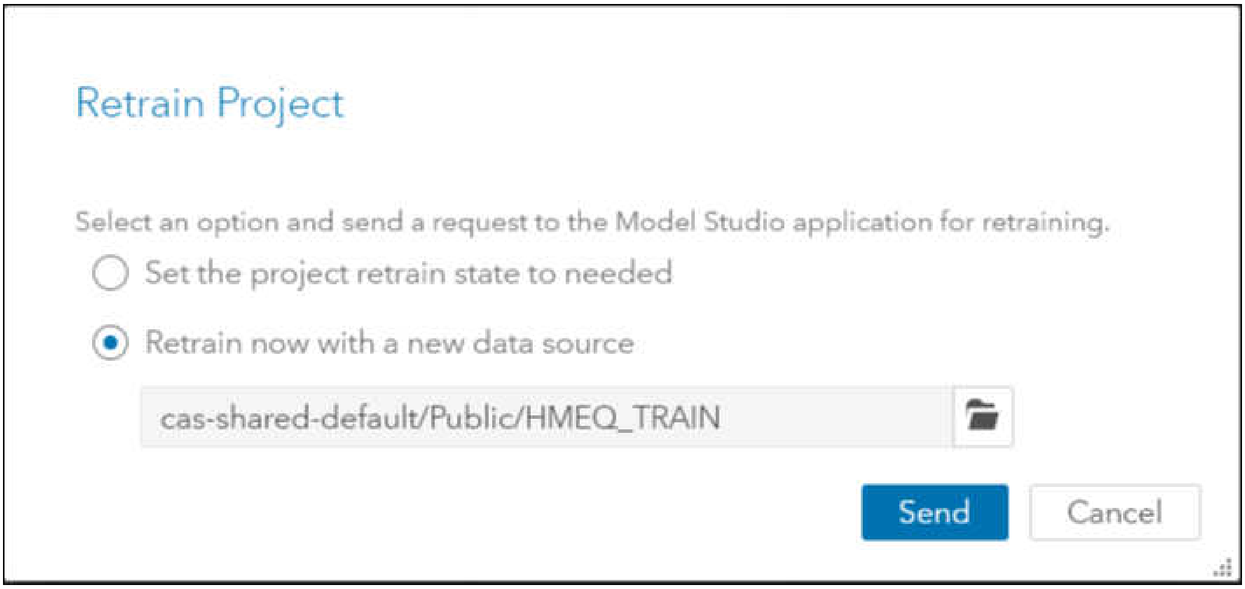
Note: If a project has been registered from Model Studio, an actions menu button appears to the left of the open items icon.



3 Click ⋮ and select **Retrain**.

4 Select an option to send a retrain request to Model Studio.

- Select the **Set the project retrain state to needed** option if you want to indicate that the project and its models are ready for retrain.
- Select the **Retrain now with a new data source** option, if you want to select a new data source and send a request to Model Studio to retrain the project now. Click 📂 to choose a data source.



Retrain Project

Select an option and send a request to the Model Studio application for retraining.

○ Set the project retrain state to needed

● Retrain now with a new data source

cas-shared-default/Public/HMEQ_TRAIN

5 Click **Send**.

Note: When you select the **Retrain now with a new data source** option, all pipelines in the associated Model Studio project are rerun. When the pipelines are finished running in Model Studio, a new champion model is selected during pipeline comparison. The new champion model is then auto-registered into a new project version within the associated SAS Model Manager project.

6

Publishing Models

About Publishing Models

You can publish models to a publish destination, so that it can be used by other applications for tasks such as scoring. Models can be published to destinations that are defined for CAS, Hadoop, SAS Micro Analytic Service, and Teradata.

Models can be published from different locations within the SAS Model Manager web application. The project champion model can be published from the Projects category view. Models can also be published from the Models category view or the **Models** tab of a project.

Publish destinations are defined by a SAS Administrator. For more information, see *SAS Viya Administration: Publishing Destinations*

Note: When you publish a model from a project version, the system creates a publishing validation test. You can edit the publishing validation test to select a test data source and output library. The validation of the model runs within the publishing destination that the model was published to. For more information, see “Validate Published Models” on page 37.

Requirements and Restrictions

Before you can publish a model, you must first set the score code type for a model. Only models with a score code type of DATA step, DS2 package, DS2 embedded process, DS2 multi-type, and Analytic store can be published.

Note: A model with the score code type of DS2 multi-type can contain code files for a DS2 embedded process, a DS2 package, and one or more analytic stores.

Depending on score code type, you can publish a model:

- Models with a score code type of **DATA step**, **DS2 embedded process**, or **DS2 multi-type** can be published to CAS, Hadoop, and Teradata.

- Models with a score code type of **DATA step**, **DS2 package**, or **DS2 multi-type** can be published to the SAS Micro Analytic Service.

  Note: You can convert Python source code to DS2 package score code, so that you can publish a Python model to SAS Micro Analytic Service. For more information, see "Preparing Python Models for Publishing" on page 48.

For more information, see "Set Model General Properties" on page 28.

Preparing Python Models for Publishing

You can prepare Python models for publishing. As part of this process the system converts the Python source code to DS2 package score code, so that you can publish your Python model to SAS Micro Analytic Service. The score code type for the model is changed from `Python` to `DS2 package`, and the score.sas file with the DS2 package code is created. The Python source code file is also reassigned the role of `Score resource`.

Here are the prerequisites for preparing a Python model for publishing:

- The score code type for the model must be Python.
- The Python source code (.py) file must have the role of `Score code`.
- The input variables for the Python model must match the input variables within the Python source code (.py) file.
- The output variables for the Python model must match the output variables within the Python source code (.py) file.

For more information, see"Manage Model Content and Versions" on page 26.

To preparing a Python model for publishing:

1 Open a model from the Models category view or from the **Models** tab of a project.

2 From the **Publish** menu button, click ▾ and select **Prepare for publishing**.

You can now publish your Python model to SAS Micro Analytic Service. For more information, see "Publish Models" on page 49.

Publish a Project Champion Model

1 Click ⧉ to navigate to the Projects category view.

2 Select a project, click ⋮, and select **Publish**. The **Publish Models** window appears.

3 Select a publish destination from the list of destinations.

   Note: If you have Read and Write permissions to the caslib that is specified in a publishing destination, the destination is shown in the list. For more information, see "CAS Authorization: How to (Authorization Window)" in *SAS Viya Administration: Cloud Analytic Services Authorization* and *SAS Viya Administration: Publishing Destinations*.

4 (Optional) In the **Items to Publish** section, edit the **Published Name** if you do not want to use the default name for the published module. The maximum length and character restrictions differ depending on your destination.

Table 6.1 Requirements and Restrictions for Published Names

Destination	Maximum Length	Requirements and Restrictions
SAS Micro Analytic Service	32	The published name must start with a letter or underscore. It cannot contain spaces, special characters, or multi-byte characters.
Teradata	128	The published name must start with a letter or an underscore. It cannot contain spaces, special characters, or multi-byte characters.
SAS Cloud Analytic Services (CAS)	128	The published name cannot contain single or double quotation marks.
Hadoop	128	The published name cannot contain colons (:) or double quotation marks.

5 (Optional) If you have previously published the project champion model, select the check box in the **Replace** column in order to replace the previously published item of the same name in the same destination.

6 Click **Publish**. The Publishing Results window appears. The status of the publishing request is displayed in the **Status** column.

Note: When you select a CAS destination and click **Publish**, the CAS destination table is automatically reloaded and the newly published item is made available to other applications. If the table contains models that are currently in use by SAS Model Manager or another application, you might not want to reload the table at the same time that you publish content. In that case, select **Publish without reloading**. However, you must manually reload the table in order for the newly published content to be accessible.

When you are publishing to a SAS Micro Analytic Service destination, the **Micro Analytic Module** column is also displayed with a URL to the published model.

7 When the status changes to `Published successfully`, click **Close**.

Note: A publishing validation test is created after a model has been published. For more information, see "Validate Published Models" on page 37.

Note: Models can also be published from the **Models** tab of a project.

Publish Models

In the **Models** category view, you can publish a model that is located within a folder or within a project. You can also publish a model from the **Models** tab of a project, including the champion model.

Important: Before you can publish a Python model to SAS Micro Analytic Service, you must convert the Python source code to DS2 score code. For more information, see "Preparing Python Models for Publishing" on page 48.

Publish Models from the Models Category

1 Click ∴ to navigate to the Models category view.

2 Select a model, click ⋮, and select **Publish**. The **Publish Models** window appears.

Note: Alternatively, you can open a model and click **Publish**.

3. Select a publish destination from the list of destinations.

   Note: If you have Read and Write permissions to the caslib that is specified in a publishing destination, the destination is shown in the list. For more information, see "CAS Authorization: How to (Authorization Window)" in *SAS Viya Administration: Cloud Analytic Services Authorization* and *SAS Viya Administration: Publishing Destinations*.

4. (Optional) In the **Items to Publish** section, edit the **Published Name** if you do not want to use the default name for the published module. The maximum length and character restrictions differ depending on your destination.

   Table 6.2 Requirements and Restrictions for Published Names

Destination	Maximum Length	Requirements and Restrictions
SAS Micro Analytic Service	32	The published name must start with a letter or underscore. It cannot contain spaces, special characters, or multi-byte characters.
Teradata	128	The published name must start with a letter or an underscore. It cannot contain spaces, special characters, or multi-byte characters.
SAS Cloud Analytic Services (CAS)	128	The published name cannot contain single or double quotation marks.
Hadoop	128	The published name cannot contain colons (:) or double quotation marks.

5. (Optional) If you have previously published a model, select the check box in the **Replace** column in order to replace the previously published item of the same name in the same destination.

6. Click **Publish**. The Publishing Results window appears. The status of the publishing request is displayed in the **Status** column.

   Note: When you select a CAS destination and click **Publish**, the CAS destination table is automatically reloaded and the newly published item is made available to other applications. If the table contains models that are currently in use by SAS Model Manager or another application, you might not want to reload the table at the same time that you publish content. In that case, select **Publish without reloading**. However, you must manually reload the table in order for the newly published content to be accessible.

   When you are publishing to a SAS Micro Analytic Service destination, the **Micro Analytic Module** column is also displayed with a URL to the published model.

7. When the status changes to `Published successfully`, click **Close**.

   Note: For models that are within a project version, a publishing validation test is created after a model has been published. For more information, see "Validate Published Models" on page 37.

Publish Models from the Projects Category

1. Click ⧉ to navigate to the Projects category view.

2. Open a project.

3. Select a model, click ⋮ , and select **Publish**. The **Publish Models** window appears.

   Note: Alternatively, you can open a model and click **Publish**.

4. Select a publish destination from the list of destinations.

Note: If you have Read and Write permissions to the caslib that is specified in a publishing destination, the destination is shown in the list. For more information, see "CAS Authorization: How to (Authorization Window)" in *SAS Viya Administration: Cloud Analytic Services Authorization* and *SAS Viya Administration: Publishing Destinations*.

5 (Optional) If you have previously published a model, in the **Items to Publish** section, select the check box in the **Replace** column for each model that you want to replace.

6 (Optional) Update the published name.

7 Click **Publish**. The Publishing Results window appears. The status of the publishing request is displayed in the **Status** column.

Note: When you select a CAS destination and click **Publish**, the CAS destination table is automatically reloaded and the newly published item is made available to other applications. If the table contains models that are currently in use by SAS Model Manager or another application, you might not want to reload the table at the same time that you publish content. In that case, select **Publish without reloading**. However, you must manually reload the table in order for the newly published content to be accessible.

When you are publishing to a SAS Micro Analytic Service destination, the **Micro Analytic Module** column is also displayed with a URL to the published model.

8 When the status changes to `Published successfully`, click **Close**.

Note: For models that are within a project version, a publishing validation test is created after a model has been published. For more information, see "Validate Published Models" on page 37.

7

Using SAS Workflow with SAS Model Manager

About Using Workflows

SAS Model Manager uses the **Workflow** tab within a project and the Tasks category view to interface with SAS Workflow. A workflow is an instance of a workflow definition. A workflow can be used to track the progress of objects, such as projects. An authorized user can use SAS Workflow Manager to create workflow definitions and to make them available to SAS Model Manager for use. Workflow definitions contain the set of tasks, participants, and data objects that comprise a business task. The status that you select when completing a task determines the next task in the workflow. All users can access the Tasks category view.

For information about creating workflow definitions, see *SAS Workflow Manager: User's Guide*.

Requirements

Before users can use SAS Workflow with SAS Model Manager, a system administrator must complete the following tasks:

- Configure a user account for a workflow client. This enables the account to make service task calls. For more information, see "Configure SAS Model Manager" in *SAS Viya for Linux: Deployment Guide*.
- Add users or a user group to the Application Administrators group. As a result, users or groups can import and enable a workflow definition within SAS Workflow Manager, and can start a workflow in SAS Model Manager. For more information, see *SAS Workflow Manager: Administrator's Guide*.
- Specify the SAS Model Manager client identifier and enable a workflow definition to make it available to SAS Model Manager. For more information, see "Creating and Editing Definitions" in *SAS Workflow Manager: User's Guide* and *SAS Workflow Manager: Quick Start Tutorial*.

Start a New Workflow

When you start a new workflow, it is associated with the project. For a specific project, only one workflow can be in progress at a time. The tasks within a workflow must be completed or the in-progress workflow process must be terminated, before a new workflow can be started.

To start a workflow:

1 Open a project and click the **Workflow** tab.

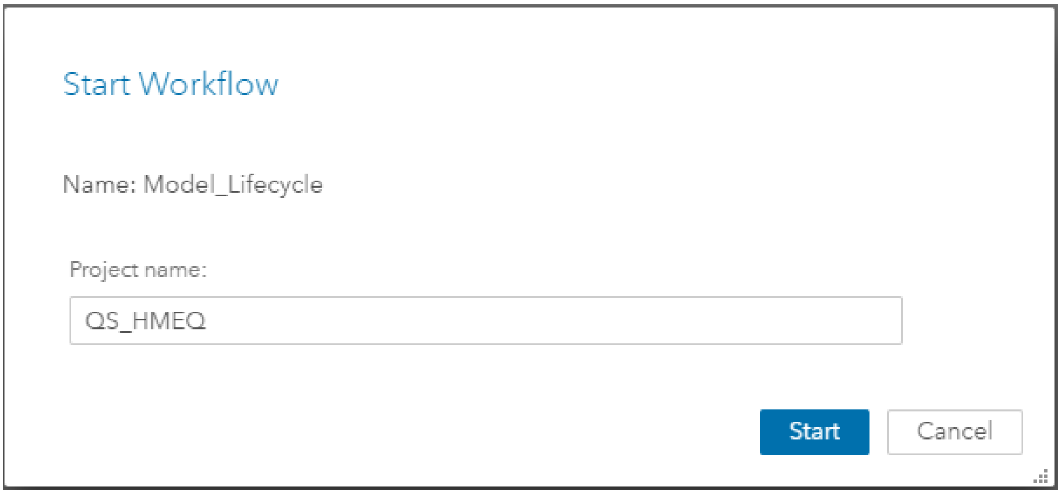
Note: The **Workflow** tab appears only for users who are members of the Application Administrators group when workflow definitions are available.For more information, see "Requirements" on page 53.

2 Click **Start Workflow** and select a workflow definition from the list. The Start Workflow window appears.

3 Specify values for any prompts that are displayed.

Note: What is displayed in the **Start Workflow** window depends on what is configured in the workflow definition start node. If prompts are not configured for the start node, the default text is "Are you sure you want to start this workflow?".

For example, specify a value for the project name.

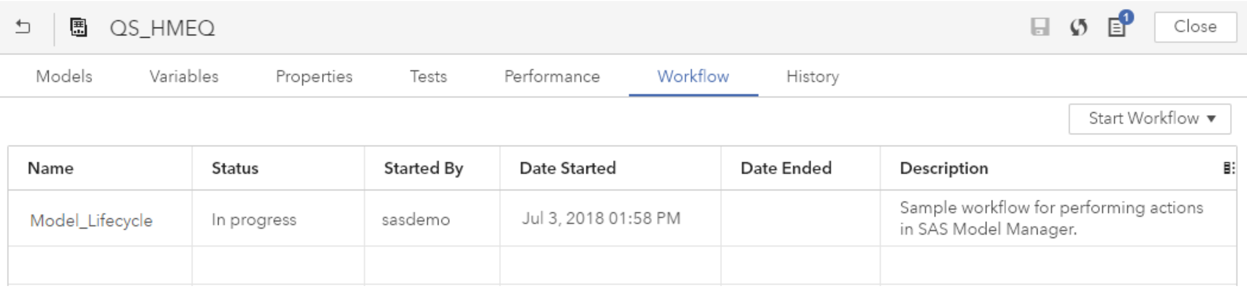


Start Workflow

Name: Model_Lifecycle

Project name: QS_HMEQ

4 Click **Start**. The workflow is added to the list with a status of "In progress".



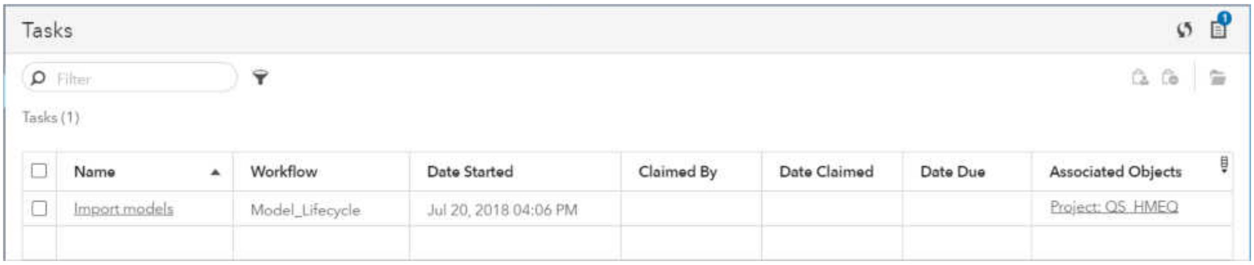
QS_HMEQ

Name	Status	Started By	Date Started	Date Ended	Description
Model_Lifecycle	In progress	sasdemo	Jul 3, 2018 01:58 PM		Sample workflow for performing actions in SAS Model Manager.

Working with Tasks

About Tasks

The Tasks category view displays the tasks for workflows that are in progress, and that you have been assigned to as a potential owner or that have been claimed by you.



Tasks

Filter

Tasks (1)

□	Name ▲	Workflow	Date Started	Claimed By	Date Claimed	Date Due	Associated Objects
□	Import models	Model_Lifecycle	Jul 20, 2018 04:06 PM				Project: QS_HMEQ

In the Tasks category view, you can perform the following:

- claim a task
- open a task
- release a task
- view the object that is associated with a task

Complete a Task

1. Click on a task to open it.
2. Click to claim the task.
3. Specify values for any prompts that are displayed on the **Prompts** tab.
4. Click the **Properties** tab to view task properties, including the associated object.
5. Click on the name of the associated object to open it.
6. Complete any actions that are associated with the task. An example is importing models into a project.
7. Click to switch back to the task object.
8. Click **Complete**.
9. Click **Complete** in the confirmation message.

Note: Alternatively, you can select one or more tasks and can click to open them or click to claim them.

Release a Task

1. Select one or more tasks.
2. Click .

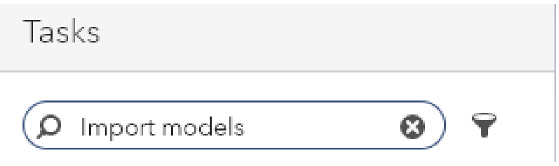
Note: Alternatively, if you already have the task open, you can click 🔓.

Filter Tasks

From the Tasks category view, you can filter the tasks that are displayed in the list. Here are the two options available for filtering tasks:

- Enter a value in the **Filter** field above the list to filter the list by task name.

  Figure 7.1 Example of Filtering by Task Name

  Tasks

  Import models

- Click ▼ to filter the list by workflow name, date started, date claimed, or date due.

8

Concepts

Concepts: Performance Monitoring

Performance Monitoring Reports

About Performance Monitoring Reports

You can monitor the performance of models by analyzing the performance results. You can create a performance definition on the **Performance** tab of a project. When you run performance for models, the performance results that are generated are based on the default performance index warnings and alerts.

The following types of changes are included in the performance results:

Data Composition Reports

The Variable Distribution chart shows you the distributions for a variable in one or more time periods, which enables you to see the differences and changes over time. The Characteristic and Stability reports detect and quantify shifts in the distribution of variable values that occur in input data and scored output data over time. By analyzing these shifts, you can gain insights on scoring input and output variables.

Model Monitoring Reports

The model monitoring reports are a collection of performance assessment reports that evaluate the predicted and actual target values. The model monitoring reports create several charts:

- Lift
- Gini
- ROC (Receiver Operating Characteristic)
- KS
- ASE (Average Squared Error) for prediction models

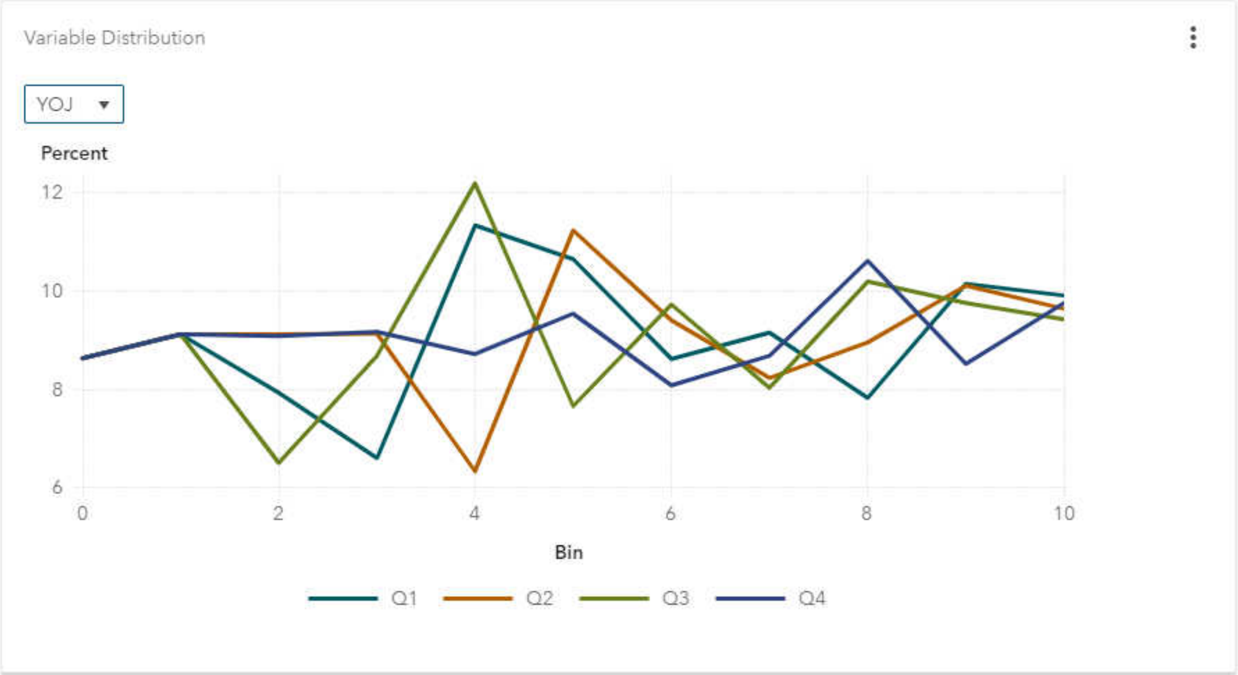
Data Composition Reports

Variable Distribution

On the **Performance** tab of a project, you can view the variable distribution of a model. The variable distribution chart is a graphical representation of distributions over a period of time for the selected variable. Each line plot represents the data for a specific period of time. The Y axis is the percentage of observations in a bin that is proportional to the total count.

To change the variable that appears in the chart, select a variable from the drop-down list.

Here is an example of a Variable Distribution chart.



Characteristic and Stability

Together, the Characteristic and Stability reports detect and quantify shifts that can occur in the distribution of model performance data, scoring input data, and the scored output data that a model produces.

Note: For each time period that you run performance, SAS Model Manager creates a new point on the charts. Line segments between points in time do not appear on the charts unless you specify at least three data sources and collection dates as part of the performance definition.

Characteristic Report

The Characteristic report detects and quantifies the shifts in the distribution of variable values in the input data over time. These shifts can point to significant changes in customer behavior that are due to new technology, competition, marketing promotions, new laws, or other influences.

To find shifts, the Characteristic report compares the distributions of the variables in these two data sets:

- the training data set that was used to develop the model
- a current data set

If large enough shifts occur in the distribution of variable values over time, the original model might not be the best predictive or classification tool to use with the current data.

The Characteristic report uses a deviation index to quantify the shifts in a variable's values distribution that can occur between the training data set and the current data set. The deviation index is computed for each predictor variable in the data set, using this equation:

$$\text{Deviation\_Index} = \sum \left(\%\,Actual - \%\,Expected \right) \times ln \left(\frac{\%\,Actual}{\%\,Expected} \right)$$

Numeric predictor variable values are placed into bins for frequency analysis. Outlier values are removed to facilitate better placement of values and to avoid scenarios that can aggregate most observations into a single bin.

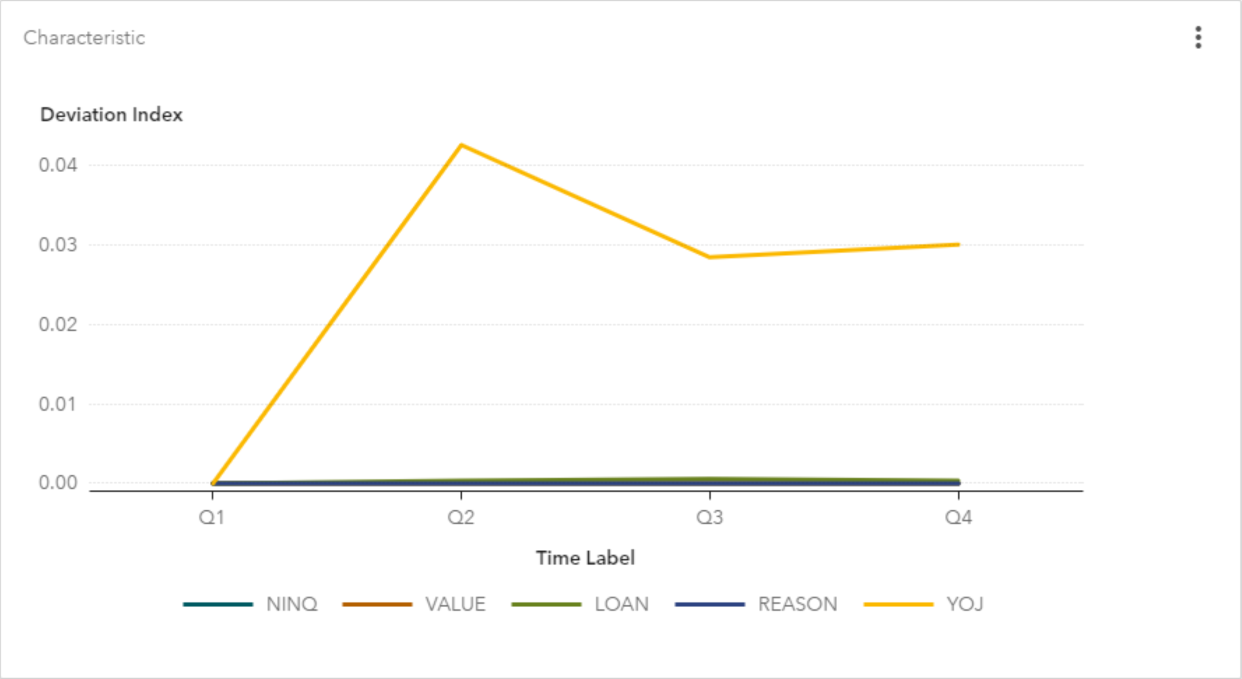
If the training data set and the current data set have identical distributions for a variable, the variable's deviation index is equal to 0. A variable with a deviation index value that is P1>2 is classified as having a mild deviation. The Characteristic report uses the performance measure P1 to count the number of variables that receive a deviation index value that is greater than 0.1.

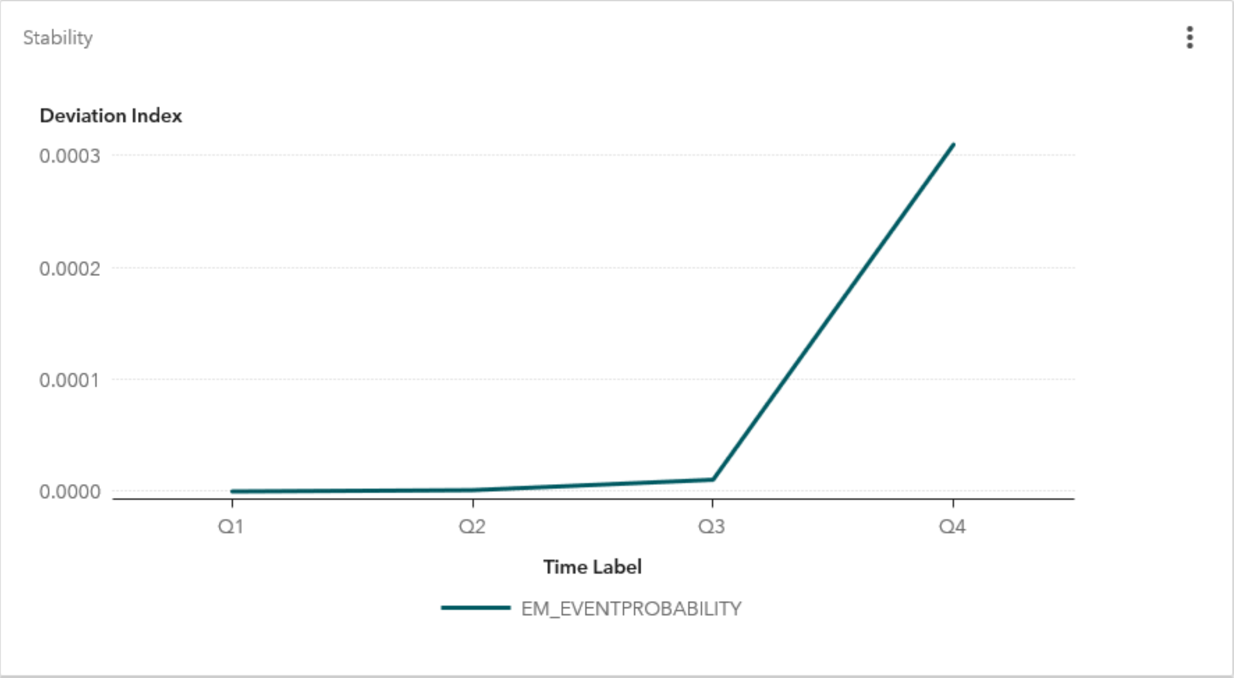
A variable that has a deviation index value that is P1>5 or P25>0 is classified as having a significant deviation. A performance measure P25 is used to count the number of variables that have significant deviations, or the number of input variables that receive a deviation index score value that is greater than or equal to 0.25.

Stability Report

The Stability report evaluates changes in the distribution of scored output variable values as models score data over time, and detects and quantifies shifts in the distribution of output variable values in the data that is produced by the models. If an output variable from the training data set and the output variable from the current data set have identical distributions, then that output variable's deviation index is equal to 0. An output variable with a deviation index value that is greater than 0.10 and less than 0.25 is classified as having a mild deviation. A variable that has a deviation index value that is greater than 0.30 is classified as having a significant deviation. Too much deviation in predictive variable output can indicate that model tuning, retraining, or replacement might be necessary.

Here are examples of the Characteristic and Stability charts.





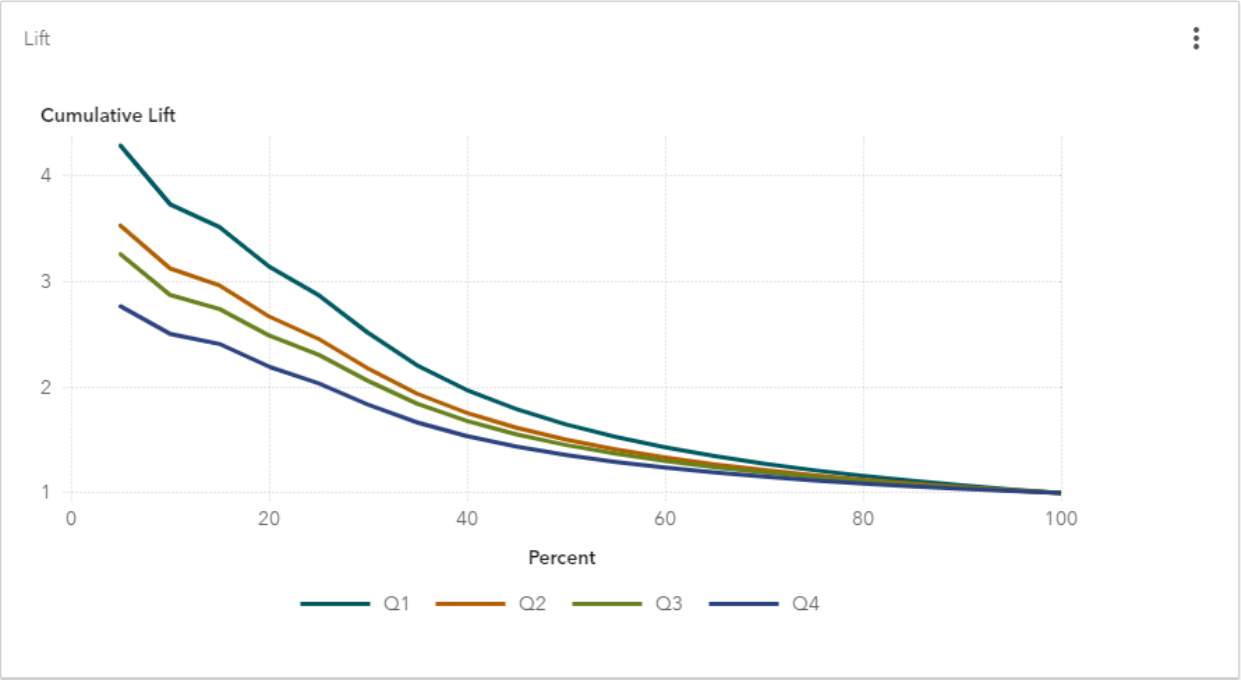
Model Monitoring Reports

Lift

The Lift report provides a visual summary of the usefulness of the information that is provided by a model for predicting a binary outcome variable. Specifically, the report summarizes the utility that you can expect by using the champion model as compared to using baseline information only. Baseline information is the prediction accuracy performance of the initial performance monitoring definition or batch program using operational data.

A monitoring Lift report can show a model's cumulative lift at a given point in time or the sequential lift performance of a model's lift over time. The Lift performance indexes Lift5Decay, Lift10Decay, Lift15Decay, and Lift20Decay are used to detect model performance degradation. The performance indexes are not displayed in the Lift chart, but are available in the mm_model_indicator performance results data table. The data that underlies the Lift chart is contained in the mm_lift performance results data table.

Here is an example of a monitoring Lift chart.



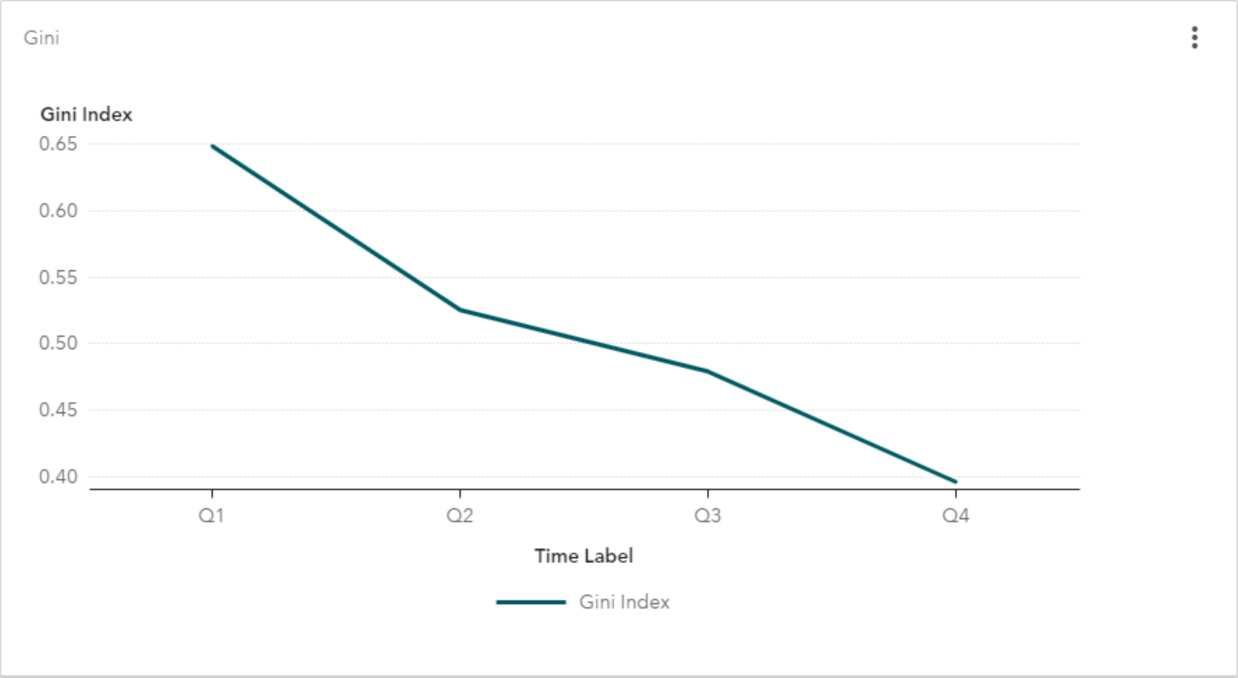
Gini

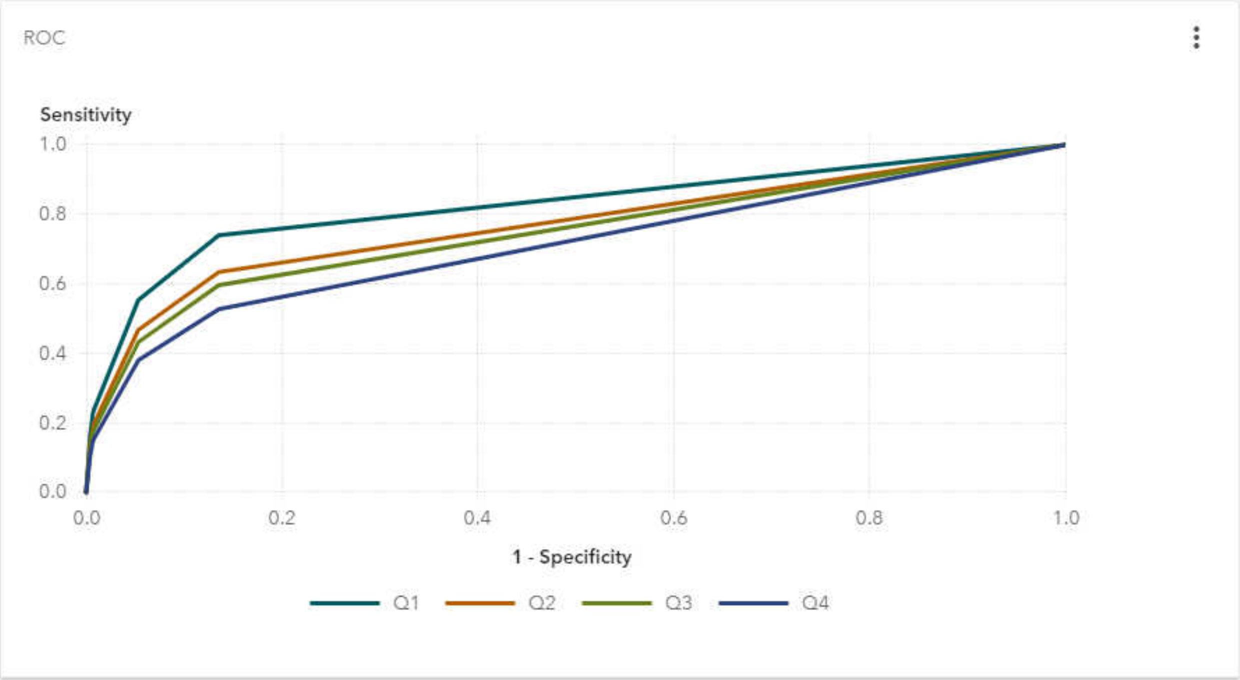
The Gini and ROC reports show you the predictive accuracy of a model that has a binary target. The plot displays sensitivity information about the Y axis and 1-Specificity information about the X axis. Sensitivity is the proportion of true positive events. Specificity is the proportion of true negative events. The Gini index is calculated for each ROC curve. The Gini coefficient is a benchmark statistic that can be used to summarize the predictive accuracy of a model, and is directly related to the area under the ROC curve (2*AUC-1).

Use the monitoring Gini report to detect degradations in the predictive power of a model.

The data that underlies the monitoring Gini and ROC reports are contained in the mm_roc performance results data table.

Here are examples of the monitoring **Gini** and **ROC** charts.





KS

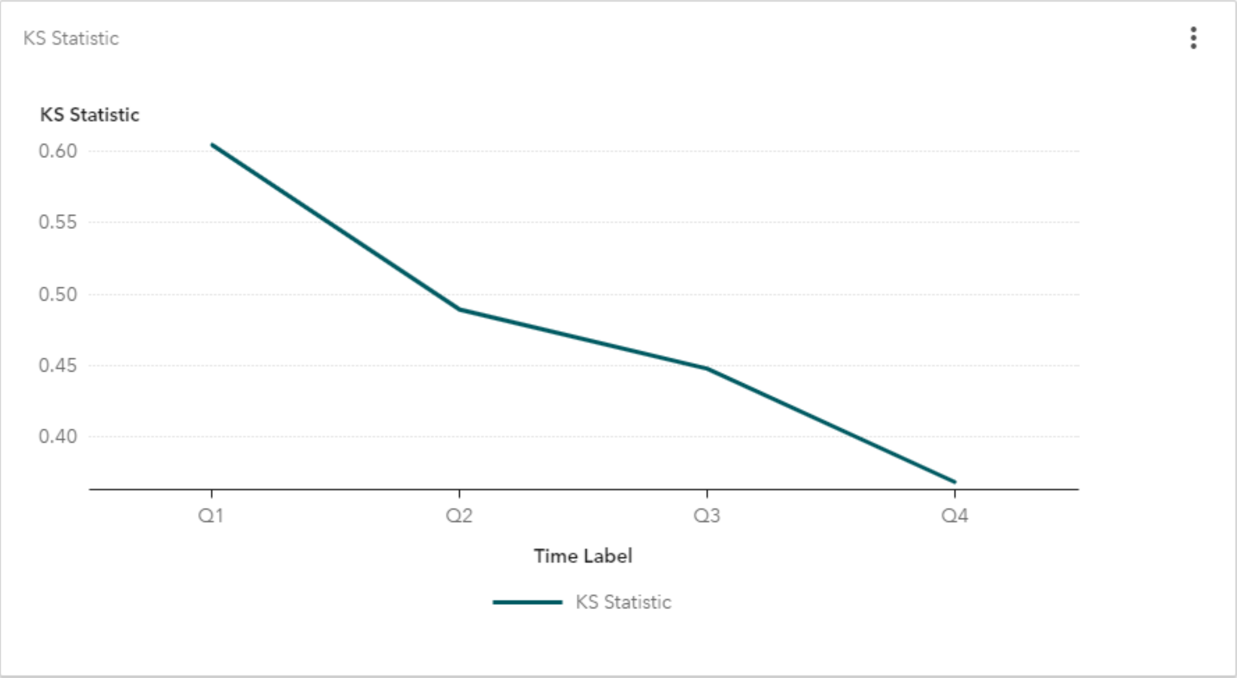
The KS report contains the Kolmogorov-Smirnov (KS) test plots for models with a binary target. The KS statistic measures the maximum vertical separation, or deviation between the cumulative distributions of events and non-events. This trend report uses a summary data set that plots the KS statistic and the KS probability cutoff values over time.

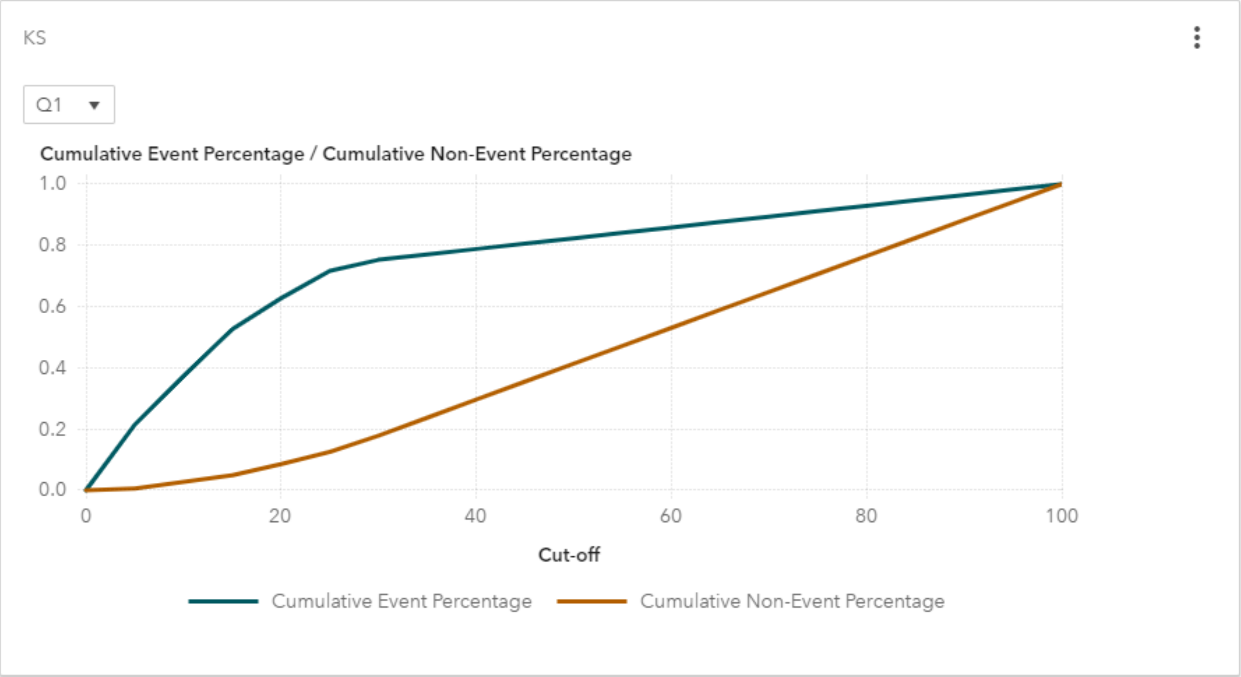
Use the KS report to detect degradations in the predictive power of a model. To scroll through a successive series of KS performance depictions, select a time interval from the **Time Interval** list box. If model performance is declining, it is indicated by the decreasing distances between the KS plot lines.

The ksDecay performance index detects model performance degradation. The ksDecay performance index is not displayed on the KS chart, but is available in the mm_model_indicator performance results data table.

The data that underlies the KS chart is contained in the mm_ks performance results data table.

Here are examples of the KS charts.

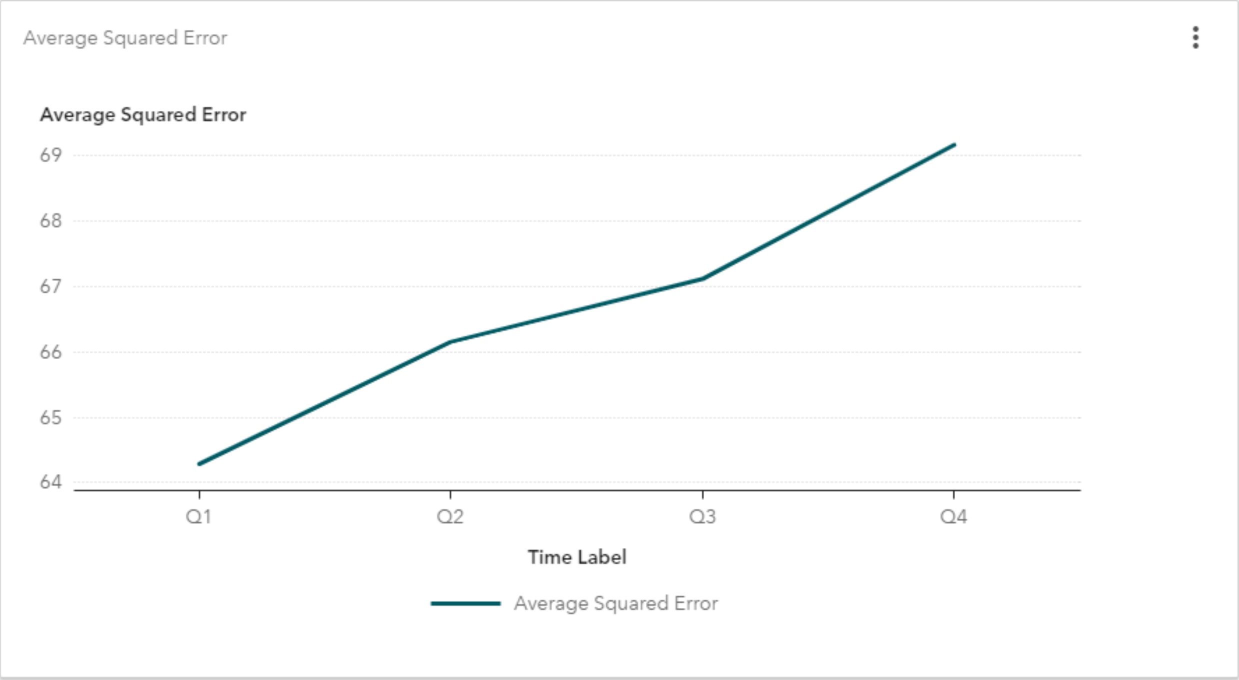




Average Squared Error

The Average Squared Error (ASE) report checks the accuracy of a prediction model with an interval target by comparing the estimation derived from the test data and the actual outcomes that are associated with the test data for different time periods.

Here is an example of the Average Squared Error chart.



Performance Index Warnings and Alerts

The production model performance reports use performance measurement thresholds to benchmark and gauge the performance of a predictive model. When one of the performance measurements exceeds one or more specified indexes or thresholds, warning and alert events occur.

The performance of a model is monitored from three categories.

Characteristic

The P1 and P25 performance indexes represent the count of input variables with deviation index scores exceeding 0.1 and 0.25, respectively. Here is an example of alert and warning thresholds:

```
alertCondition='p1>5 or p25>0';
warningCondition='p1>2';
```

Stability

The output deviation index scores represent the deviation levels in the distribution of the model's scored output variables. Here is an example of alert and warning thresholds:

```
alertCondition='outputDeviation>0.03';
warningCondition='outputDeviation>0.01';
```

Model Assessment

For the Lift, Gini, ROC, and KS reports, here are the threshold values for the decay statistics.

lift5Decay
is the lift performance decay based on the top 5% of the target population of interest from time A to time B.

lift10Decay
is the lift performance decay based on the top 10% of the target population of interest from time A to time B

lift15Decay
is the lift performance decay based on the top 15% of the target population of interest from time A to time B.

lift20Decay
is the lift performance decay based on the top 20% of the target population of interest from time A to time B.

giniDecay
is the performance decay of the Gini index from time A to time B.

ksDecay
is the performance decay of the KS statistic from time A to time B.

For the prediction model the aseDecay statistic is the performance decay of the ASE statistic from time A to time B.

Here is an example of alert and warning thresholds:

```
alertCondition='(lift5Decay>0.15 and lift10Decay>0.12)
                     or giniDecay>0.1 or ksDecay>0.1';
warningCondition='lift5Decay>0.05';
```

Concepts: PMML Support

Overview

PMML is an XML markup language that was developed to exchange predictive and statistical models between modeling systems and scoring platforms. Users can import the majority of standard-compliant PMML models and score them within a SAS environment via the SAS PSCORE procedure.

PROC PSCORE Functionality

The SAS PSCORE procedure generates SAS DATA step score code that is functionally equivalent to the PMML model. The generated score code can be executed on all platforms that are supported by SAS to score the data sets. You can submit the score code in SAS Enterprise Miner via the Program Editor, SAS Enterprise Miner Project code, or within a SAS Enterprise Miner Process Flow Diagram, via the SAS Code node. However, the SAS Enterprise Miner UI environment is not necessary to run the score code.

Note: The PSCORE procedure generates both DATA step code and DS2 code. However, only DATA step model score code is generated when you are registering a PMML model into SAS Model Manager.

Supported Versions

PROC PSCORE currently supports the use of PMML 4.2. Earlier versions of PMML are not supported for use with PROC PSCORE.

Supported PMML Models

SAS PROC PSCORE supports the following types of PMML models:

- Regression
- Trees
- Neural Networks
- Clustering models
- Scorecard
- Vector Machine

- Naïve Bayes
- Baseline models

The following models are supported on an experimental basis:

- Time Series
- General Regression

Requirements for PROC PSCORE

In order to use PROC PSCORE, you must have SAS 9.4 or later, a well formed PMML modeling file, and Write access to the output directory for the DATA step score file. A SAS Enterprise Miner license is not necessary to run PROC PSCORE.

PROC PSCORE Usage

```
PROC PSCORE PMML FILE = “<full-pathname-of-PMML-file>"
     DS FILE = "<full-pathname-of-output-DS-file>"
```

PROC PSCORE Example

/*Run the PSCORE procedure on a generated PMML file*/

```
PROC PSCORE PMML FILE = "C:\temp\heart_pmml1.xml"
     DS FILE = "C:\temp\ds_heart_score.sas";
  run;
```

See Also

SAS Enterprise Miner 15.1 PMML Support