# Workflow Environment Perspective

## User Guide

Version: 4.1.10

www.worldprogramming.com

# Contents

# Introduction

The Workflow Environment is the collective name for a set of features that help with data mining, predictive modelling tasks, and provide a range of Machine Learning capabilities.

The Workflow Environment operates in a completely different way to how WPS Workbench operates when you develop a SAS language; program execution, and log and result output are all handled differently. For this reason, WPS Workbench is supplied with two perspectives:

- The *SAS Language Environment*, which is dedicated to classic SAS language programming and output handling.
- The *Workflow Environment*, dedicated to the **Workflow Editor** view and **Data Profiler** view.

The **Workflow Editor** view provides a palette of drag-and-drop interactive blocks that you combine to create workflows that connect to a data source, filter and manipulate the data into a smaller subset using the **Data Preparation** blocks. You can save the data to an external dataset for later use. You can use the Machine Learning capabilities available in the **Model Training** blocks to discover predictive relationships in your filtered and manipulated data.

Once created, a workflow is re-usable, so can be used with different input datasets to generate output datasets filtered or manipulated in the same way.

A workflow can auto-generate error-free code from the models, ready for deployment and execution in production.

The **Data Profiler** view is a graphical tool that enables you to explore WPS datasets used in a workflow, or external to a workflow. You can use the **Data Profiler** view to interact with and explore your data through graphical views and predictive insights.

Many of the Data Science features available in the **Workflow Editor** view are enabled by World Programming's built-in SAS language capabilities. Although coding is not a prerequisite for creating a workflow, those in the team who have the requisite programming skills can carry out more advanced tasks in a workflow, using not only the SAS language code block, but also R, Python and soon, SQL.

## Who should use the Workflow Environment

If you are familiar with the Cross-Industry Standard Process for Data Mining (CRISP-DM), you can use workflows to follow this process in the preparation and modelling of data sources of all sizes. The Workflow Environment tools enable a team of people with different skill sets to work on the same data in a collaborative environment. For example, a single workflow could be created that enables:

- Data analysts to blend the prepared data through joining, transformation, partitioning, and so on, to create raw datasets of varying sizes.
- Data scientists to use machine learning algorithms to build, explore and validate reproducible predictive models, including scorecards, from proven datasets.

# Workflow Environment perspective

The Workflow Environment perspective contains the views and tools required to create and manage workflows.

Workflow files are created and managed in projects using the **Project Explorer** view, or on a host to which you have access using the **File Explorer** view.

The **Workflow Editor** view is used to create and run workflows. The **Data Profiler** view is used to inspect the contents of a dataset in the workflow.

The **Workflow Link Explorer** view and **File Explorer** view enable you to connect to a remote host and specify a default Workflow Engine on a remote host to run your workflows.

The perspective also enables you to create and view tasks or bookmarks in a workflow. Other views and functionality available in WPS Workbench are visible in this release, but are not currently supported by the Workflow Environment perspective.

The SAS language library feature for creating datasets, such as the temporary `WORK` library, is not available in a workflow. *Working datasets* are created as part of a workflow, and the **Data Profiler** view is used to explore the dataset in the workflow.

To open the Workflow Environment perspective, click the **Window** menu, click **Perspective** click **Open Perspective** and then click **Workflow Environment**.

## Project Explorer view

The **Project Explorer** view is used to edit and manage projects and their associated workflows and datasets.

The **Project Explorer** view only displays projects that are in your current workspace. You can create several projects in a workspace, and use each project for a specific task.

If you change the default Workflow Engine to a remote host, the location of the projects and workflow files does not change but remains on local host. Any references to datasets in the workspace or project cannot be used with a remote Workflow Engine, and must be moved to the remote host using the **File Explorer** view, and any references in the workflow changed to the appropriate filepath for the remote host.

Selecting an item in the **Project Explorer** view displays information about that item in the **Properties** view.

### Displaying the view

To display the **Project Explorer** view:

1. Click the **Window** menu, click **Show View** and then click **Project Explorer**.

## Create a new project

A project is a Workbench folder that contains workflows and related datasets. Projects are accessed through a workspace. You can only have one workspace open at a time, but a workspace can contain multiple projects.

To create a new project:

1. Click the **File** menu, click **New** and click **Project**.
2. In the **New** dialog box, expand the **General** group, select **Project** and click **Next**.
3. In the **Project** pane, specify a **Project name** and click **Finish**.

# File Explorer view

The **File Explorer** view is used to access the local file system, and the file systems of connected remote hosts.

The **File Explorer** view can be used to manage workflows and associated datasets on remote and local file systems. Because the **File Explorer** view does not manage content through projects, you cannot use Workbench functionality to import or export projects or archives.

The **File Explorer** view enables access to all files that are available on your local file system, and remote hosts with connections visible in the **Workflow Link Explorer** view

### Displaying the view

To display the **File Explorer** view:

1. Click the **Window** menu, click **Show View** and then click **File Explorer**.

### Connections and shortcuts

It is only possible to view files on remote systems when you have enabled the required remote host connections. A remote host connection is created and enabled in the **Workflow Link Explorer** view.

### Working with files and folders

You can use **File Explorer** to:

- Create a new folder shortcut, folder or file.
- Select files and folders to be moved, copied or deleted.
- Rename a selected file or folder.

### File explorer preferences

To display hidden files and folders in the **File Explorer** view:

1. Click the **Window** menu, and then click **Preferences**.
2. In the **Preferences** dialog box, expand the **WPS** group and select **File Explorer**.
3. In the **File Explorer** view, select **Show hidden files and folders** and click **OK**.

# Workflow Editor view

The **Workflow Editor** view provides a visual programming canvas that enables you to prepare data, export datasets and train models.

A workflow is defined by a single file, identified with a `.workflow` extension, and these files are managed in either the **Project Explorer** view or **File Explorer** view.

The **Workflow Editor** view contains the **Workflow** tab and the **Settings** tab.

## Workflow tab

The **Workflow** tab in the **Workflow Editor** view contains the canvas on which a workflow is created.

The **Workflow** tab canvas has a palette of operations, such as data import tasks, data preparation tasks, machine learning operations, coding blocks, scoring, and so on. To select the required operation, drag the corresponding block onto the canvas, where you can connect it with other blocks already on the canvas to create a workflow.

Most types of blocks that can be added to a workflow require you to specify properties. Properties are set in the configuration dialog box that can be accessed through the shortcut menu for a block. To view or specify block properties, right-click the required block in the canvas and click **Configure**.

By default, the workflow automatically runs as blocks are added or edited. You can also right click on a block and choose to run the workflow up to that position. Working datasets created as part of the workflow are deleted when the **Workflow Editor** view is closed. If you automatically run a workflow, reopening a saved workflow automatically recreates all working datasets in the workflow.

### Adding comments to a workflow and blocks

You can add comments to a workflow and these can be used to post notes that describe the purpose of a workflow or identify the individual steps in a workflow when the workflow is shared.

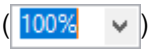
- To create a new comment for the workflow, right-click in the **Workflow Editor** view and click **Add comment**.
- To add a comment to a block, right-click the required block and click **Add comment**.

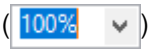
# Workflow navigation

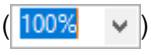
The **Workflow Editor** view enables you to zoom in and out of a workflow, to relocate blocks in a workflow and move the workflow within the editor view.

Zoom-in and zoom-out features in the **Workflow Editor** view can be controlled from the **Zoom** tools, keyboard, or mouse. The zoom tools are grouped in the Workbench toolbar:


To change the scale of the workflow view:

- Click **Zoom in** (  ) to increase the scale of the workflow view by 10% of the current scale. Alternatively, press **Ctrl**+**Plus Sign** to do the same.
- Click **Zoom out** (  ) to decrease the scale of the workflow view by 10% of the current scale. Alternatively, press **Ctrl**+ **Minus Sign** to do the same.
- Click the zoom list ( 

) and then select **Page** to scale the workflow view to fit the available screen area in the **Workflow Editor**.
- Press **Ctrl**+**0** (zero) to set the scale of the workflow view to 100%.
- Click the zoom list ( 

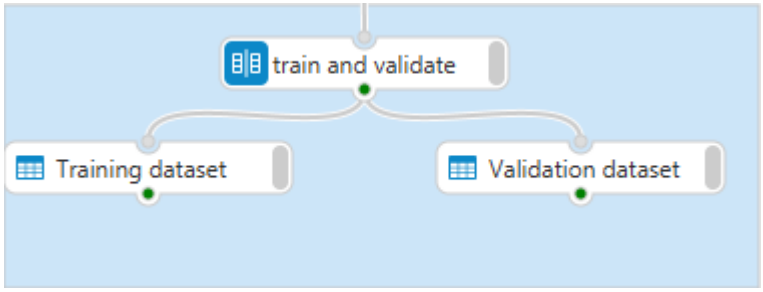
) and then select **Height** to scale the workflow view to fit the height in the available screen area in the **Workflow Editor**. If the workflow design is too wide for the scale, some blocks may be partly-visible or not visible at all after scaling.
- Click the zoom list ( 

) and then select **Width** to scale the workflow view to fit the width in the available screen area in the **Workflow Editor** view. If the workflow design is too tall for the scale, some blocks may be partly-visible or not visible at all after scaling.

## Moving workflow blocks in the view

The position of multiple blocks can be changed in the workflow view. If blocks are connected to other blocks in the workflow, the connections are maintained between the block. To relocate multiple items in the workflow:

1. Click the **Workflow Editor** view.
2. Press **Ctrl** and drag the pointer to cover all the blocks:




The selected blocks are highlighted (their outlines become blue).

3. Click one of the highlighted blocks, and drag the group of blocks to the required place in the workflow.

   After dragging, only the block you selected remains highlighted.

### Move the visible area of the workflow

You can move the canvas to enable you to focus on a specific part of the workflow. To move the canvas, click on a blank area and drag until the required area of the workflow is in view.

## Saving a workflow

After modifying the workflow, save it to ensure no loss of data.

A workflow can be saved in a project in the active workspace, or to a location on a local or remote host. If you use the **Project Explorer** view to manage workflow files, the workflow is saved to a project folder in the current workspace. If you use the **File Explorer** view to manage workflow files, the workflow can be saved to any local or remote host location in which you have permission to create files.

To save a workflow, click the **File** menu and then click **Save**. The workflow file is updated with any changes you have made.

### Save a workflow to a different file

You can save a workflow to an alternative file either in the current workspace or to a folder on any host to which you have access.

- If you use the **Project Explorer** view to manage files, you can save the workflow as a different file in any project in your current workspace, but cannot save a workflow to a folder that can only be accessed through the **Project Explorer** view.
- If you use the **Project Explorer** view to manage files, you can save the workflow as a different file to any available folder structure that you can access on each host, but cannot save a workflow to a project in the current workspace.

# Export to SAS language program

The **Workflow Editor** view allows you to export a Workflow as SAS language code, either to the clipboard or to a specified file.

**Note:**
To export a Workflow to SAS language code, the Workflow must not contain any failed or out of date blocks.

To export a Workflow as SAS language code:

1. Ensure that your workflow is displayed and runs successfully.
2. Right click on the Workflow background and then click **Export to SAS Language Program**.

   An **Export workflow to SAS language program** dialog box appears.
3. In **Temporary library name**, enter a name for the library to be defined at the start of the SAS language code.
4. In **Directory for temporary datasets**, enter a directory for the library to be defined at the start of the SAS language code.
5. Click **Next**.

   SAS language code for the workflow is displayed.
6. You can now either copy the code to the clipboard or save it to a file:
   - To copy the code to the clipboard, click **Copy Code**. You can now close the dialog box by clicking **Cancel**.
   - To save the code to a file, click **Next**, choose either your **Workspace** or an **External** location, and then click **Browse** to choose the location and type a filename, including the file extension (.sas or .wps).
7. Click **Finish**

# Settings tab

The **Workflow Editor** view **Settings** tab contains the library definitions stored with the workflow, and enables you to manage these.

Library references enable you to access datasets stored in the supported database types. The libraries can be created in the project, referenced from WPS Hub, or created and stored in the workflow. The **Settings** tab displays data source references that are:

- A link to a reference created in the **Database** view, in either the project or WPS Hub.
- A copy of a reference created in the **Database** view, in either the project or WPS Hub.
- A reference created specifically for the workflow.

To create a link to a reference in the **Database** view, select the required reference, drag to the **Settings** tab and in the **Database Operations** dialog box select **Link to databases**.

To create a copy of a reference in the **Database** view, select the required library, drag to the **Settings** tab and in the **Database Operations** dialog box select **Copy databases**.

To create a new reference, click **Add** on the **Settings** tab and use the **Add Database** wizard to create the new library reference. For more information about creating a library reference, see *Create a new database reference* (page 16).

When a new data source has been added successfully, you can use the **Test connection** button to ensure the connection works. The test result is displayed in the **Errors** column of the **Databases** list.

The **Settings** tab can be used to edit references copied from the **Database** view or created for the workflow. The **Add database** wizard is also used to edit an existing reference.

To remove a reference select the required library in the **Databases** list and click **Remove**.

# Workflow Link Explorer view

The **Workflow Link Explorer** view enables you to create remote host connections, a new Workflow Engine instance and to set options for the connection.

A Workflow Engine is an instance of the WPS processing engine running on a local or remote host. You can have multiple connections to remote hosts active in the **Workflow Link Explorer** view; each connection can have one Workflow Engine defined for it. You can have only one active Workflow Engine in the Workflow Environment perspective.

## Displaying the view

To display the **Workflow Link Explorer** view:

From the **Window** menu, click **Show View** and then click **Workflow Link Explorer**.

## Objects displayed

There are two node types visible in this view, a connection node (1) and a Workflow Engine (2):


A *connection node* represents a connection to a host machine. A Workflow Engine represents the WPS installation that will run the currently-active workflow. Each engine can have a number of instances enabling you to run the branches of large workflows in parallel.

### Managing the connections and servers

From this view you can:

- Open or close a connection.
- Specify a default Workflow Engine.
- Define a new connection and Workflow Engine.
- Specify system options for a Workflow Engine.

# Create a new remote host connection

A remote host connection enables you to access a remote host's file system using the **File Explorer** view and to link to the Workflow Engine installed on remote host to run workflows.

Before creating a new connection, you will need SSH access to the server machine that has a licensed WPS Analytics installation. You might need to contact the administrator to obtain this information, and to ensure that you have access.

To create a new remote host connection:

1. In the **Workflow Link Explorer** view click **Create a new remote host connection**.

   The **New Server Connection** dialog is displayed.
2. Select **New SSH Connection (3.2 and later -- UNIX, MacOS, Windows)**
3. Click **Next**.

   The **New Remote Host Connection (3.2 and later)** dialog box is displayed.
4. In **Hostname**, enter the name or IP address of the remote server machine.

   Unless modified by your system administrator, the default **Port** value (*22*) should be left unchanged.
5. In **Connection name** enter a unique name to be displayed in **Workflow Link Explorer** view for this connection.
6. In **User name**, enter your user ID for the remote host.
7. Select the required option:

| Option | Description |
|---|---|
| **Enable compression** | Controls whether or not data sent between the Workbench and the remote connection is compressed. |
| **Verify hostname** | Confirms whether the host you have specified in the **Hostname** entry exists. |
| **Open the connection now** | Whether the connection is automatically opened immediately |

| Option | Description |
| --- | --- |
| **Open the connection automatically on Workbench startup** | Controls whether the connection is automatically opened when the Workbench is started. |

8. Click **Next**.

   A dialog that enables you to define the connection directory shortcuts is displayed.

9. Click **Add**.

   The **Directory Shortcut** dialog box is displayed.

10. In **Directory Name**, enter the shortcut name to be displayed.
11. In **Directory Path**, enter the full path of the target directory.
12. Click **OK** to save the changes.
13. Click **Finish**.

    All changes are saved. If you have not previously validated the authenticity of the remote host, the **SSH2 Message** dialog box is displayed. If you have previously validated the authenticity of the remote host, the remote host connection is created.

14. If the identity of the host in the **SSH2 Message** dialog is correct, click **Yes**.

    The **Password Required** dialog is displayed.

15. Enter your password for the remote machine.
16. Click **OK**.

    The remote host connection is created.

# Define a new remote Workflow Engine

Workflows are run using a Workflow Engine. You can define a new Workflow Engine, if required.

Before creating a new Workflow Engine, you need to know the installation directory path for the licenced WPS installation on the remote host.

To define a new Workflow Engine:

1. In the **Workflow Link Explorer** view, right-click the required remote host connection node and click **New Engine** in the shortcut menu.

   The **New Remote Engine** dialog is displayed.

2. In **Engine Name**, enter a name. This name is displayed in the **Workflow Link Explorer** view.
3. In **Base WPS install directory**, enter the absolute path to the WPS installation directory on the remote host.
4. Click **Finish** to create the Workflow Engine.

You can specify the number of instances available to the Workflow Engine when a workflow is run, and will automatically use up to the maximum number of available instances.

Right-click the default engine and click **Properties** in the shortcut menu to display the Workflow Engine properties dialog. In the **Engine Instances** panel of the properties dialog you can modify the **Maximum number of engine instances** available when running a workflow.

# Specify a default Workflow Engine

You can specify which Workflow Engine is used by default when running workflows.

The pre-defined default Workflow Engine is the engine on the local host connection. Changing the default engine to a different host might require changes to the workflow to enable the workflow to run on a different Workflow Engine.

After you have specified a different default engine, you need to restart WPS Workbench. You should, therefore, save your workflow before specifying a default. To specify the default:

1. Save and close any open workflows in the **Workflow Editor** view.
2. In the **Workflow Link Explorer** view, right-click the required Workflow Engine and click **Set as Default Engine** in the shortcut menu.

   The **Restart required** dialog box is displayed.
3. Click **Yes**.

When WPS Workbench restarts, the specified default Workflow Engine is shown in bold in the **Workflow Link Explorer** view.

After changing the default Workflow Engine, you need to check that the remote WPS engine can support the current workflow; for example:

- Ensure that filepaths used to import datasets can be accessed from the remote host.
- If you use the **Database Import** block, ensure that the remote WPS engine has the required database client software installed.

# Specify startup options for a Workflow Engine

You can specify the settings applied to a Workflow Engine and use these startup options to change the behaviour of the engine.

The startup options for Workflow Engine are WPS system options. You can control items such as system resource usage or language settings. Not all system options can be set as startup options. A list of system options that can be set when a Workflow Engine starts, their effect, and their supported values, can be found in the *WPS Reference for Language Elements*.

All startup options are set in WPS Workbench in the same way. For example, to set the value of the `ENCODING` startup option for a Workflow Engine:

1. In the **Workflow Link Explorer** view, right-click the required Workflow Engine and click **Properties**.

   The **Properties for Workflow Engine** dialog box is displayed.
2. Expand **Startup**, and then click **System Options**.
3. In the **System Options** panel, click **Add**.

   The **Startup Option** dialog box is displayed.
4. Enter `ENCODING` in the **Name** field.

   If you do not know or are unsure of the name of a system option you can click **Select**, which displays the **Select Startup Option** dialog box from which you can select a system option. Click **OK** in this dialog box to select the option.
5. Enter `UTF-8` in the **Value** field.
6. Click **OK** to save the changes and when prompted, click **Yes** to restart the Workflow Engine for the changes to take effect.

# Database view

The **Database** view enables you to create a reference to a database and to access references created and stored in WPS Hub.

A *reference* is an object that specifies the details required to access to database. If you have multiple workflows that require access to the same data, you can create a library reference and import data into a workflow using the **Database Import** block.

## Displaying the view

To display the **Database** view:

1. Click the **Window** menu option, select **Show View** and then click **Database view**.

## Objects displayed

There are two node types visible in this view, a host node (1) and a data source node (2):


A *host node* represents the location where the data source connection information is stored, either **Hub** or **Workbench**.

- The **Hub** node displays all references available in the authorisation domain in WPS to which you have connected. For more information, see *Using WPS Hub with workflows* (page 33).
- The **Workbench** node contains all references you have created. These references are stored in the current project, but can be shared with a workflow through the workflow **Settings** view.

A *database node* represents the reference element specifying a connection to a specific database. Previously-created references can be imported from a data source definition file. The settings for any node created in the **Workbench** host can be exported to a database definition file.

# Create a new database reference

A database reference enables you to access datasets stored in either a local or remote database.

Before creating a new reference, you must ensure the database client connector for the database type is installed and accessible from the Workflow Engine used to run the workflow.

To create a new database reference:

1. In the **Database** view click **Add database** to display the **Add database** wizard.
2. Select the database type from the list: DB2, MySQL, Oracle, ODBC, PostgreSQL, or SQL Server; then click **Next**.
3. Step two of the wizard varies according to the database you are connecting to:
   - For **MySQL Settings**, **PostgreSQL Settings**, and **SQL Server Settings**:

     Enter the **Host name**, **Port**, and choose the **Authentication** method as either **Credentials** or **Auth Domain**.

     If you have chosen **Credentials**, then enter a **Username** and **Password** and click **Connect**. If you have chosen **Auth domain**, then choose an authorisation domain from the **Auth domain** drop down list.
   - For **DB2 Settings**:

     Enter the **Database** name and choose the **Authentication** method as either **Credentials** or **Auth Domain**.

     If you have chosen **Credentials**, then enter a **Username** and **Password** and click **Connect**. If you have chosen **Auth domain**, then choose an authorisation domain from the **Auth domain** drop down list.
   - For **Oracle Settings**:

     Enter the **Net service** name (for example, the path to the Oracle database) and choose the **Authentication** method as either **Credentials** or **Auth Domain**.

     If you have chosen **Credentials**, then enter a **Username** and **Password** and click **Connect**. If you have chosen **Auth domain**, then choose an authorisation domain from the **Auth domain** drop down list.
   - For **ODBC Settings**:

Enter the **Datasource Name** and choose the **Authentication** method as **Credentials**, **Auth Domain**, or **Use ODBC credentials**.

If you have chosen **Credentials**, then enter a **Username** and **Password** and click **Connect**. If you have chosen **Auth domain**, then choose an authorisation domain from the **Auth domain** drop down list.

4. Click **Connect** to attempt the connection. If the connection is successful, a **Connection successful** window is displayed. If the connection fails, a **Connection failed** window is displayed; clicking **Details** on this window will display a connection log showing what went wrong.
5. Step three depends on what database you are connecting to:
   - **DB2** and **Oracle**: Choose from the **Schema** drop down list and click **Next**.
   - **MySQL**: Choose from the **Database** drop down list and click **Next**.
   - **SQL Server** and **PostgreSQL**: Choose the **Database** first, then a **Schema** from the filtered drop down list, and finally click **Next**.
   - **ODBC**: Optionally and if supported, choose from the **Schema** drop down list and click **Next**.
6. In the **Name** box, enter a name to use to refer to the database.
7. Click **Finish**.

   A **Configure Database Import** window is displayed.
8. Click **OK** to return to the workflow.

# Data Profiler view

The **Data Profiler** view enables you to view content and details about a dataset.

To open the **Data Profiler** view, select the required working dataset, right-click **Open With** and then click **Data Profiler** in the shortcut menu.

# Summary View

The **Summary View** lists information about the dataset, including number of observations and variables in the dataset and variable characteristics.

The **Summary View** contains two panels, **Summary** and **Variables**.

## Summary

The **Summary** panel displays summary information about the dataset: the dataset name, the total number of observations, and the total number of variables in the dataset.

## Variables

The **Variables** panel displays information about the variables in the dataset, such as the variable name, label type, length, and so on. The following information is displayed:

**Variable**
The name of the variable in the dataset.

**Label**
The alternative display name for the variable.

**Type**
The type of the variable. The type can be either *Numeric* for numbers and date and time values, or *Character* for character and string data.

**Classification**
The category of the variable; this can be one of:

- *Categorical*. A variable that can contain a limited number of possible values, and the limit is below the specified classification threshold.
- *Continuous*. A variable that can contain an unlimited number of possible values. This classification is used for numeric variables where the number of distinct values in the variable is greater than the specified classification threshold
- *Discrete*. A variable that can contain a limited number of possible values. This classification is used for character variables where the number of distinct values in the variable is greater than the specified classification threshold.

The classification threshold is specified on the **Data** panel of the **Preferences** dialog box.

**Length**
The size required to store the variable values. For character types, the number represents the maximum number of characters found in a variable value. For numeric types, the number represents the maximum number of bytes required to store the value.

**Format**
How the variable is displayed when output. For more information about formats see the section *Formats* in the *WPS Reference for Language Elements*.

**Informat**
The formatting applied to the variable when imported into WPS. For more information about formats see the section *Informats* in the *WPS Reference for Language Elements*.

**Distinct Values**
The number of unique values in the variable. If the variable classification is Continuous, the display indicates there are more unique values than the specified classification threshold.

**Missing Values**
The number of missing values in the variable.

**Frequency Distribution**
For each value in a non-continuous variable, displays a chart showing the number of occurrences for each value in the variable.

# Data

The **Data** panel lists all observations in a dataset and enables you to view, filter, or sort the observations

The **Data** panel in the **Data Profiler** view is a read-only version of the **Dataset Viewer**. You can modify the view, sort and filter data in the **Data** panel. If you want to edit values in the dataset, you need to use the **Dataset Viewer**. For more information about filtering and sorting observations, and about the **Dataset Viewer** see *Dataset Viewer* (page 27).

# Univariate View

The **Univariate view** tab displays summary statistics for all numeric univariate variables.

The columns displayed in the **Univariate View** tab are determined using the **Calculate Statistics** dialog box. Click **Configure Statistics** (Σ) to open **Preferences** and select the statistics you want displayed:

**Quantiles**
Lists quantile points.

**Variable Structure**
Lists each of the selected principal statistics describing the variable, such as number of missing values, and minimum and maximum values.

**Measures of Central Tendency**
Lists each of the measures used to identify the central point in the values of the variable

**Measures of Dispersion**
Lists each of the selected measures used to show the variation from the central value in the variable.

**Others**
Lists other statistics that can be displayed in the statistics table.

# Univariate Charts

The **Univariate Charts** tab enables you to view the frequency distribution table and graphs for a selected dataset variable.

The information displayed is determined by the variable selected in the **Variable Selection** list. Each section in the tab displays the frequency of values occurring in the selected variable.

## Frequency Table

Displays the frequency of the values in the specified variable, the percentage of the total number of observations for each value, and cumulative frequencies and percentages.

If the variable type is categorical or discrete, the table displays one row for each potential value. If the variable type is continuous, the variable is binned and the frequency information is displayed for each bin.

## Frequency Chart

The frequency distribution of the specified variable can be displayed as a histogram, line chart or pie chart. In all cases, the charts display the frequency of values as the percentage of the total number of observations for the variable.

The chart can be edited and saved. Click **Edit chart**  to open the Chart Editor from where the chart can be saved to clipboard.

# Correlation Analysis

The **Correlation Analysis** tab enables you to view the strength of the relationship between numeric variables in the dataset.

## Options

Specifies the coefficient type to use when comparing variables, and which numeric variables in the dataset are to be compared.

**Coefficient**

Specifies the type of coefficient used to compare values.

**Pearson**

Specifies Pearson correlation coefficient, defined as:

$$r_{x,y} = \frac{\sum_{i=1}^{n}(x_i - \bar{x})(y_i - \bar{y})}{\sqrt{\sum_{i=1}^{n}(x_i - \bar{x})^2 \sum_{i=1}^{n}(y_i - \bar{y})^2}}$$

where

$\bar{x} = \frac{1}{n}\sum_{i=1}^{n} x_i$ is the mean of variable $x$

$\bar{y} = \frac{1}{n}\sum_{i=1}^{n} y_i$ is the mean of variable $y$.

**Spearman's Rho**

Specifies Spearman's rank correlation coefficient, defined as:

$$r_s = \frac{\sum_{i=1}^{n}(Rx_i - \overline{R}_x)(Ry_i - \overline{R}_y)}{\sqrt{\sum_{i=1}^{n}(Rx_i - \overline{R}_x)^2 \sum_{i=1}^{n}(Ry_i - \overline{R}_y)^2}}$$

where

- $Rx_i$ is the rank of $x_i$
- $\overline{R}_x = \frac{1}{n}\sum_{i=1}^{n} Rx_i$ is the mean of the ranked variable $R_x$.
- $Ry_i$ is the rank of $y_i$
- $\overline{R}_y = \frac{1}{n}\sum_{i=1}^{n} Ry_i$ is the mean of the ranked variable $R_y$.

The values of the variables are first ranked and the ranks compared. Using this coefficient may therefore be more robust to some outliers in the data.

**Kendall's Tau**

Specifies the Kendall rank correlation coefficient, defined as:

$$\tau_b = \frac{n_c - n_d}{\sqrt{(n_0 - n_1)(n_0 - n_2)}}$$

Where

- $n_0 = \frac{n(n-1)}{2}$
- $n_1 = \sum_{i=1}^{n} \frac{t_i(t_i - 1)}{2}$
- $n_2 = \sum_{j=1}^{n} \frac{u_j(u_j - 1)}{2}$
- $n_c$ is the number of concordant pairs.
- $n_d$ is the number of discordant pairs.
- $t_i$ is the number of tied values in the *i*th group of tied values for the first variable.
- $u_j$ is the number of tied values in the *j*th group of tied values for the second variable.
- The difference in the number of concordant and discordant pairs can be expressed as:

$$n_c - n_d = \sum_{i<j} sign(x_i - x_j)\ sign(y_i - y_j)$$

The values of the variables are first ranked and the ranks compared. Using this coefficient may therefore be more robust to some outliers in the data.

**Select Variables**

Displays a list of numeric variables in the dataset, from which you can select the variables to compare.

## Correlation Coefficient Matrix

The matrix is shown in colour blocks, the size and colour of the blocks indicates the relationship between the compared variables. The scale for the items displayed ranges from 1 a strong positive correlation to -1 where there is a strong negative correlation.

Using variables that show a strong positive correlation may lead to the model becoming unstable as a small change in the variables may lead to a large change in the model.

If you are using the model for prediction, variables showing a strong positive correlation may be good predictors.

### Correlation Statistics

Displays the correlation statistics and scatter plot for the item selected in the Correlation Coefficient Matrix. The table displays the variables being compared, the value for the specified coefficient type and the P-value for the comparison. The scatter plot displays all values in the dataset, using the same colour scale as the matrix, as either a plot or heat map if the number observations is very high.

## Predictive Power

The **Predictive Power** tab enables you to view the predictive power of independent variables in the dataset in relation to the selected dependent variable.

The information displayed is determined by the variable selected as the **Dependent Variable**. Each section displays the relationship between the independent variables in the dataset and the specified **Dependent Variable**.

### Statistics table

The **Statistics Table** displays all the variables in the dataset and a series of predictive power statistics to enable you to identify the most effective independent variables for the specified dependent variable. The statistics displayed are:

- **Entropy Variance**. Displays the *Entropy Variance* value for each variable in relation to the specified **Dependent Variable**.
- **Chi Sq**. Displays the *Chi-Squared* value for each variable in relation to the specified **Dependent Variable**.
- **Gini**. Displays the *Gini Variance* value for each variable in relation to the specified **Dependent Variable**.

For more information about how these values are calculated, see *Predictive power criteria* (page 24)

### Entropy Variance Chart

Displays the bar chart of independent variables' entropy variance in relation to the specified **Dependent Variable**. The variables are displayed in order from the highest entropy variance to the lowest. The number of variables displayed in the chart is determined by the preferences set in the **Data Profiler** panel of the **Preferences** dialog box.

The chart can be edited and saved. Click **Edit chart**  to open the Chart Editor, from where the chart can be saved to clipboard.

### Frequency Chart

Displays the frequency of each value of an independent variable selected in the **Statistics Table**. The chart has two display modes:

- Click **View whole data** to display the overall predictive relationship between values of an independent variable selected in the **Statistics Table** the specified **Dependent Variable**.
- Click **View breakdown data** to display the frequency of each value of an independent variable, and its relationship to the outcomes for the specified **Dependent Variable**.

The chart can be edited and saved. Click **Edit chart** to open a the Chart Editor from where the chart can be saved to clipboard.

# Predictive power criteria

Predictive power is a way of measuring how well a particular input variable can predict the target variable.

## Pearson's Chi-Squared statistic

Pearson's Chi-squared statistic is a measure of the likelihood that the value of the target variable is related to the value of the predictor variable.

Each observation in the dataset is allocated to a cell in a contingency table, according to the values of the predictor and target variables. Pearson's Chi-squared statistic is calculated as the normalised sum of the squared deviations between the actual number of observations in each cell, and the expected number of observations in each cell if there were no relationship between the predictor and target variables.

If a predictor variable has a high Pearson's Chi-squared statistic, it means that the variable is a good predictor of the target variable, and is likely to be a good candidate to use to split the data in a binning or tree-building algorithm.

Pearson's Chi-squared statistic for a discrete target variable is calculated as

$$\chi^2 = \sum_{i=1}^{r}\sum_{j=1}^{c}\frac{\left(n_{ij}-\mu_{ij}\right)^2}{\mu_{ij}}$$

where:

- $N$ is the total number of observations in the dataset
- $r$ is the number of distinct values of the predictor variable $X$ (these are the rows in the contingency table)
- $c$ is the number of distinct, discrete values of the target variable $Y$ (these are the columns in the contingency table)
- $n_{ij}$ is the number of observations for which the predictor variable $X$ has the $i$ th value, $X_i$, and the target variable $Y$ has the $j$ th value, $Y_j$ (these are the values in the cells of the contingency table)
- $n_{i*}$ is the total number of observations for which the predictor variable $X$ has the $i$ th value, $X_i$

- $n_{*j}$ is the total number of observations for which the target variable $Y$ has the $j$ th value, $Y_j$
- $\mu_{ij}$ is the expected value of $n_{ij}$, calculated as

$$\mu_{ij} = \frac{n_{i*} n_{*j}}{N}$$

## Entropy Variance

Entropy variance is a measure of how well the value of a predictor variable can predict the value of the target variable.

If a variable in a dataset has a high entropy variance, it means that the variable is a good predictor of the target variable, and is likely to be a good candidate to use to split the data in a binning or tree-building algorithm.

Entropy variance for a discrete target variable is calculated as

$$E_r = 1 - \frac{\sum_{i=1}^{r} \left( \frac{n_{i*} E_i}{N} \right)}{E}$$

where:

- $N$ is the total number of observations in the dataset
- $r$ is the number of distinct values of the predictor variable, $X$
- $c$ is the number of distinct, discrete values of the target variable, $Y$
- $n_{ij}$ is the number of observations for which the predictor variable $X$ has the $i$ th value, $X_i$, and the target variable $Y$ has the $j$ th value, $Y_j$
- $n_{i*}$ is the total number of observations for which the predictor variable $X$ has the $i$ th value, $X_i$
- $n_{*j}$ is the total number of observations for which the target variable $Y$ has the $j$ th value, $Y_j$
- $E_i$ is the entropy calculated for just the observations where the predictor variable is $X_i$, calculated as

$$E_i = -\frac{1}{\log(c)} \sum_{j=1}^{c} \frac{n_{ij}}{n_{i*}} \log\left( \frac{n_{ij}}{n_{i*}} \right)$$

- $E$ is the entropy calculated for all the observations, calculated as

$$E = -\frac{1}{\log(c)} \sum_{j=1}^{c} \frac{n_{*j}}{N} \log\left( \frac{n_{*j}}{N} \right)$$

## Gini Variance

Gini variance is a measure of how well the value of a predictor variable can predict the target variable.

If a variable in a dataset has a high Gini variance, it means that the variable is a good predictor of the target variable, and is likely to be a good candidate to use to split the data in a binning or tree-building algorithm.

Gini variance for a discrete target variable is calculated as

$$G_r = 1 - \frac{\sum_{i=1}^{r}\left(\frac{n_{i*}G_i}{N}\right)}{G}$$

where

- $N$ is the total number of observations in the dataset
- $r$ is the number of distinct values of the predictor variable, $X$
- $c$ is the number of distinct, discrete values of the target variable, $Y$
- $n_{ij}$ is the number of observations for which the predictor variable $X$ has the $i$ th value, $X_i$, and the target variable $Y$ has the $j$ th value, $Y_j$
- $n_{i*}$ is the total number of observations for which the predictor variable $X$ has the $i$ th value, $X_i$
- $n_{*j}$ is the total number of observations for which the target variable $Y$ has the $j$ th value, $Y_j$
- $G_i$ is the Gini impurity calculated for just the observations where the predictor variable is $X_i$, calculated as

$$G_i = 1 - \frac{\sum_{j=1}^{c} n_{ij}^2}{n_{i*}^2}$$

- $G$ is the Gini impurity calculated for all the observations, calculated as

$$G = 1 - \frac{\sum_{j=1}^{c} n_{*j}^2}{N^2}$$

## Information value

Information value is a measure of the likelihood that the value of the target variable is related to the value of the predictor variable. The information value measure is only applicable for binary target variables (that is, target variables that can take one of exactly two values).

If a predictor variable has a high information value, it means that the variable is a good predictor of the target variable, and is likely to be a good candidate to use to split the data in a binning or tree-building algorithm.

The information value statistic is calculated as

$$IV = \sum_{i=1}^{r}\left(\frac{n_{i0}}{n_{*0}} - \frac{n_{i1}}{n_{*1}}\right)WOE_i$$

where:

- $r$ is the number of distinct, discrete values of the predictor variable $X$ (these are the rows in the contingency table)
- $Y_0$ and $Y_1$ are the two possible values of the binary target variable $Y$
- $n_{i0}$ is the number of observations for which the predictor variable $X$ has the $i$ th value, $X_i$, and the target variable $Y$ has the value, $Y_0$ (these are the values in the cells of the $Y_0$ column in the contingency table)
- $n_{i1}$ is the number of observations for which the predictor variable $X$ has the $i$ th value, $X_i$, and the target variable $Y$ has the value, $Y_1$ (these are the values in the cells of the $Y_1$ column in the contingency table)
- $n_{*0}$ is the total number of observations for which the target variable $Y$ has the value, $Y_0$
- $n_{*1}$ is the total number of observations for which the target variable $Y$ has the value, $Y_1$
- $\alpha \ll 1$ is the weight of evidence (WOE) adjustment, a small positive number to avoid infinite values when $n_{i0} = 0$ or $n_{i1} = 0$
- $WOE_i$ is the WOE value for observations where the predictor variable is $X_i$, calculated as

$$WOE_i = \ln\left(\frac{n_{i0}}{n_{*0}} + \alpha\right) - \ln\left(\frac{n_{i1}}{n_{*1}} + \alpha\right)$$

# Dataset File Viewer

Enables you to view, filter, sort or modify observations in a dataset.

The contents of a dataset are shown in a grid. The rows of the grid represent dataset observations, and the columns represent dataset variables.

You can display labels in the column headers of a dataset, re-organise the view by moving columns, and hide datagrid columns that are not relevant.

By default, a dataset is opened in *browse* mode. If required, you can change to *edit* mode if you want to make changes to the data. You cannot, however, edit a working dataset created by a block.

To open the **Dataset File Viewer**, select the required working dataset, right-click **Open With** and then click **Data File Viewer** in the shortcut menu.

## Modifying the dataset view

You can modify the dataset view to change the column order and hide variables that are not required.

Changes made to the layout of the datagrid in the Dataset Viewer do not change the underlying data.

### Show labels

You can specify whether to use labels as column headings rather than column names. To use labels:

1. Right-click the column header of the variable for which you want labels, and select **Preferences**.

   The **Preferences** dialog box is displayed.
2. Expand the **WPS** group and click **Dataset Viewer**.
3. Click **Show labels for column names**.
4. Click **OK**.

The setting is saved, and the **Preferences** dialog closed.

### Hide variables

To hide a datagrid column:

1. Right-click the column header for the variable you want to hide.
2. Click **Hide column**.

If you hide a column for which filtering is currently active, you need to confirm that you want to remove the filter and hide the column.

### Show hidden variables

To show previously hidden dataset variables:

1. Right-click the dataset header row.
2. Click **Show** / **Hide Columns** to display the **Show** / **Hide Columns** dialog box.
3. Select the required columns and click **OK**.

### Move columns

To move a column in the dataset view, click the column and drag to the new location in the view. You can only move one column at a time.

## Editing a dataset

An imported dataset can be opened in *edit* mode enabling you to add new observations, delete observations, or edit existing values.

To edit an imported dataset, right-click in the grid and click **Toggle edit mode**. Any changes you make in the Dataset File Viewer are not written to the dataset until you save the changes. If saving your changes fails, then any changes that were not written remain visible and details of the failure are written to the Workbench log

# Modify values

The Dataset Viewer enables you to edit values in the dataset.

1. Double click the value that you want to edit. The variable type affects how the value is edited:
   - Numeric and character types. The value is displayed in the cell where you can enter a new value. If the variable has formatting applied, to see the current value when formatted as a tool tip press and hold **Shift**+**F8**.
   - Date, datetime or time formatted types. Click on an individual element of a date or time value, and either enter the required value or use the up or down arrows to increase or decrease them in steps.
2. When you have completed your edits, press **Enter**.

   The entered value is displayed in bold type, and an asterisk (*) is appended to the observation number in the left margin of the grid.
3. To save the changes to the original dataset, click the **File** menu, and click **Save**.

# Add observations

The **Dataset Viewer** enables you to add new observations to the end of an existing dataset.

To add a new observation to the dataset:

1. Right-click the datagrid and click **Add Observation**.
2. The new row appears at the end of the dataset. Double-click each cell in the observation and modify the content as required.
3. To save the changes to the original dataset, click the **File** menu, and click **Save**.

# Delete observations

The **Dataset Viewer** enables you to remove observations from an existing dataset.

To delete an observation from a dataset:

1. Select the observation that you want to delete by clicking on its observation number in the left hand column.
2. Right-click an observation and click **Delete Observation(s)**.
3. To save the changes to the original dataset, click the **File** menu, and click **Save**.

## Missing values

Enables you to set the value of a variable as missing.

1. Select the variable value you want to set as missing.
2. Right-click the selected cell, and click **Set Missing**.

   If the cell is a:

   - Numeric variable type, the **Set Missing Value** dialog is displayed. The default . (full stop) is used as the value. Alternatively, you can change the missing value to one of: . (full stop), ._ (full stop followed by an underscore), or .A to .Z.
   - Character variable type, the cell is set to a single space (' ') character.
3. To save the changes to the original dataset, click the **File** menu, and then click **Save**.

# Filter a dataset

You can apply one or more filters to restrict the view to show only the data you require.

You can only apply a filter expression to a dataset open in browse mode. Filter criteria can be set on multiple columns and, as you apply filters, the dataset contents are automatically updated.

To filter your dataset view:

1. Click the filter button (); this sits below the variable heading for the column you want to filter.
2. Select the criteria by which you want to filter the view and complete the criteria for the filter as required.

### Dataset filter expressions

You can modify a generated filter expression applied to numeric and character variable types. The expressions available depend on the variable type. A generated filter expression applied to a variable with a date, datetime or time format cannot be modified, only be cleared and re-entered.

To clear the filter for a variable, click the filter button and then click **Clear Filter**.

The following table shows the supported expression syntax for numeric values.

| Criteria | Expression | Example |
|---|---|---|
| Equal to | `EQ` *X* | Is equal to 100<br>`EQ 100` |
| Not equal to | `NE` *X* | Is not equal to 100<br>`NE 100` |

| Criteria | Expression | Example |
|---|---|---|
| Less than | `LT X` | Is less than 100<br>`LT 100` |
| Greater than | `GT X` | Is greater than 100<br>`GT 100` |
| Less than or equal to | `LE X` | Is less than or equal to 100<br>`LE 100` |
| Greater than or equal to | `GE X` | Is greater than or equal to 100<br>`GE 100` |
| Between (inclusive) | `BETWEEN X AND y` | Is between 100 and 200<br>`BETWEEN 100 AND 200` |
| Not between (inclusive) | `NOT BETWEEN X AND y` | Is not between 100 and 200<br>`NOT BETWEEN 100 AND 200` |
| Is missing | `IS MISSING` | `IS MISSING` |
| Is not missing | `IS NOT MISSING` | `IS NOT MISSING` |
| In | `IN (x,y)` | Is one of the values 100, 200 or 300 (numeric)<br>`IN (100,200,300)` |

The following table shows the supported expression syntax for character values.

| Criteria | Expression | Example |
|---|---|---|
| Equal to | `EQ X` | Is equal to "Blanco" (string)<br>`EQ "Blanco"` |
| Not equal to | `NE X` | Is not equal to "Blanco" (string)<br>`NE "Blanco"` |
| In | `IN (x,y)` | Is one of the values "Blanco", "Jones" or "Smith" (character)<br>`IN ("Blanco","Jones","Smith")` |
| Starts with | `LIKE " s%"` | Starts with the string "Bla"<br>`LIKE "Bla%"` |
| Ends with | `LIKE " %S"` | Ends with the string "nco"<br>`LIKE "%nco"` |

| Criteria | Expression | Example |
|---|---|---|
| Contains | `LIKE " %s%"` | Contains the string "an"<br>`LIKE "%an%"` |
| Is missing | `IS MISSING` | `IS MISSING` |
| Is not missing | `IS NOT MISSING` | `IS NOT MISSING` |

## Sort a dataset

The dataset displayed in the **Dataset Viewer** can be sorted using the values in one or more variables.

You can only sort a dataset that is open in Browse mode. Variables that are part of an active sort have an icon representing the direction of the sort in the header.

To sort a dataset:

1. Right-click the column header for the variable that you want to use as the primary key.
2. Click either **Ascending Sort** or **Descending Sort** on the shortcut menu.
3. You can refine the sorting by selecting a required column and clicking either **Ascending Sort** or **Descending Sort**.

To remove any column from the sort, right-click the required column and click **Clear Sort**. The dataset is resorted using the remaining selected variables.

# Bookmarks view and Tasks view

The **Bookmarks** view lists bookmarks added to a workflow; the **Tasks** view lists tasks added to a workflow. Both bookmarks and tasks can be used to identify parts of the workflow that require further investigation.

To add a bookmark:

1. In the **Project Explorer** view, select the required workflow.
2. Click the **Edit** menu and then click **Add Bookmark**.
3. Enter a **Description** in the **Bookmark Properties** dialog box.
4. Click **OK**.

   The new bookmark is displayed in the **Bookmarks** view.

To add a task:

1. In the **Project Explorer** view, select the required workflow.
2. Click the **Edit** menu and then click **Add Task**.
3. Enter a **Description** in the **Properties** dialog box.
4. Click **OK**.

   The new task is displayed in the **Tasks** view.

You can double click on a bookmark in the **Bookmarks** view, or a task in the **Tasks** view to open the workflow in the **Workflow Editor** view.

# WPS Hub

WPS Hub is an Enterprise Management Tool that enables the centralised management of access to data sources.

WPS Hub uses authentication domains to store user credentials for database server access, which can then be referenced in a workflow, removing the need for user credentials to be stored as part of the workflow.

A WPS Hub authentication domain is a central definition of access credentials for a database. An authentication domain contains one or more credentials, each of which defines a username and password that can be used to access a database server. Each credential is associated with a WPS Hub user or group, and workflows you create can use the authentication domain instead of hard-coding the access credentials.

## Using WPS Hub with workflows

WPS Hub provides access to centrally-managed library references and authorisation domain credentials to connect to database servers.

Before connecting to WPS Hub, you require access credentials and remote host details. These details are provided by your WPS Hub administrator.

1. On the **WPS Hub** menu click **Log in**.
2. In the **Hub Login** dialog box complete the required information:
   a. Select the required **Protocol**, either `HTTP` or `HTTPS`.
   b. Enter the **Host** name provided by your WPS Hub administrator.
   c. Enter the **Port** for WPS Hub. By default, Hub configured for HTTPS will use port `8443`, and when configured for HTTP, port `8181`.
   d. Enter the WPS Hub **Username** provided by your WPS Hub administrator.
   e. Enter the WPS Hub **Password** provided by your WPS Hub administrator.

3. Click **OK** to connect.

If the connection is successful, a message stating `Hub: Logged in as` *username* is displayed in the Workbench status bar. If the connection fails, an error message is displayed in the **Hub Login** dialog box.

# Create a new workflow

A workflow is a program sequence visually represented by connected, colour-coded graphical elements (blocks) through which the background code runs from start to end.

A workflow enables collaboration between the project stakeholders; it provides a project framework so the projects can be easily replicated or audited and can be used as a framework to carry out projects in standardised manner.

Any datasets you need to use with your workflow must be located on the same host as the Workflow Engine you will use to run the workflow. If your workflow requires a connection to a database, you must have the appropriate database client software installed on the same host as the Workflow Engine.

You can create a new workflow in either the **Project Explorer** view or the **File Explorer** view.

- If you select **Project Explorer** view, the workflow can be created in any folder in the current workspace, and the file is saved on your local workstation.
- If you select **File Explorer** view, the workflow can be created in any folder on the local workstation, any remote hosts in which you have permission to create files.

1. Click the **File** menu, click **New** and click **Workflow**.
2. In the **Workflow** pane, select the required folder for the new workflow, specify a **File name** and click **Finish**.

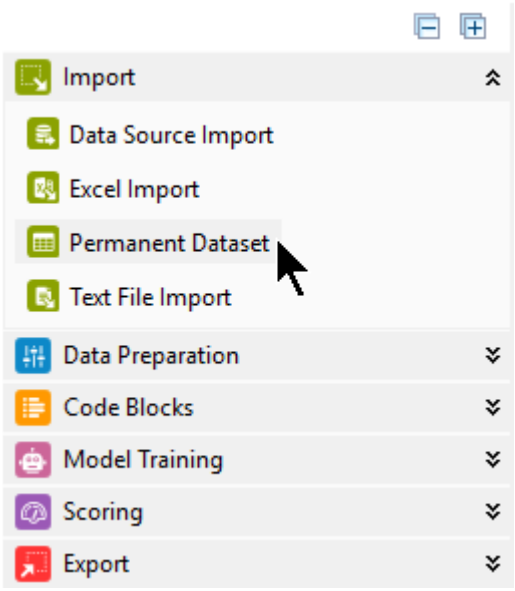
# Add blocks to a workflow

A workflow visually represents a program using blocks. The blocks include colours, icons, and labels to identify the block type. Blocks can be connected together to represent program steps and specific operations within each step.

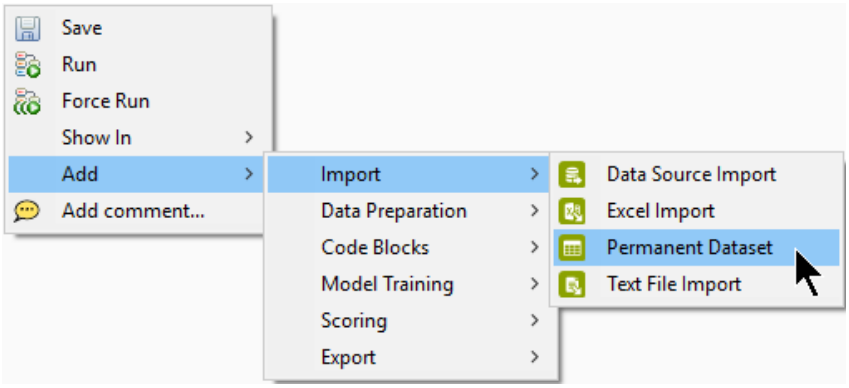
Most workflows begin by importing a dataset, for example by using a **Permanent Dataset** block.

To add a **Permanent Dataset** block to a workflow:

1. In the **Workflow Editor** view, expand **Import** in the group palette.
2. Click **Permanent Dataset** in the palette and drag to the working area of the editor.

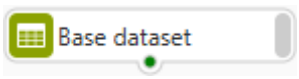



Alternatively, right-click the **Workflow Editor** view, in the shortcut menu click **Add**, click **Import**, and then click **Permanent Dataset**:




3. Right-click the **Permanent Dataset** block and click **Rename Block** in the shortcut menu.
4. In the **Rename** dialog box, enter a **Block label** to describe the dataset, for example *Base dataset* and click **OK**. The name entered is displayed as the label of the block on the **Workflow Editor** view.
5. Right-click the **Permanent Dataset** block and click **Configure** in the shortcut menu.
6. In the **Configure Permanent Dataset** dialog box, select a file location, either **Workspace** or **External**, and click **Browse** to locate an existing WPD-format dataset.
7. Click **OK** to save the changes.

If the dataset is successfully loaded, the Execution status in the **Output** port of the block is green:




You can then add other blocks to the editor and connect them together to create a workflow.

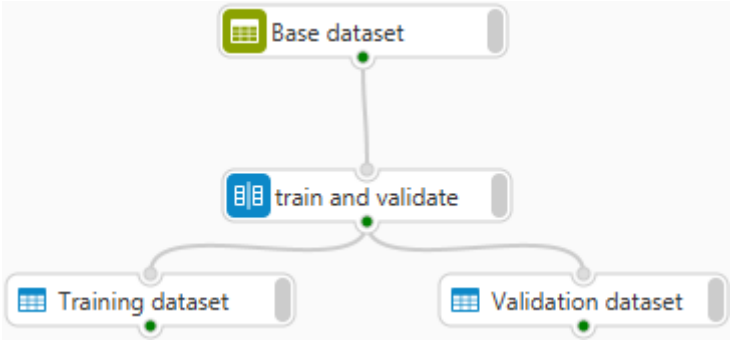
# Connect blocks in a workflow

Blocks can be connected together using connectors to create a workflow. Connectors are linked to the block at either the **Input** port located at the top of a block, or an **Output** port located at the bottom of the block. You can connect the output from the **Output** port of one block to the **Input** port of one or more other blocks.

For example, you can randomly divide a dataset into two or more datasets using a **Partition** block. To partition the *Base dataset* created in the section *Add blocks to a workflow*:

1. In the **Workflow Editor** view, click the **Data Preparation** group in the group palette.
2. Select the **Partition** block, and drag onto the working area. By default, two partitions are created and the **Partition** block has two working datasets:


3. Right-click the **Partition** block and click **Rename Block** in the shortcut menu. In the **Rename** dialog box, enter a **Block label** to describe the partition, for example *Train and validate* and click **OK**.
4. Click the **Base dataset** block **Output** port and drag towards the **Input** port of the **Train and validate** block. A connector appears as you drag. Drag this until it connects to the **Input** port.
5. When the **Base dataset** block is connected to the **Train and validate** block:
   a. Select the **Partition1** dataset and change the **Block label** to *Training dataset*.
   b. Select the **Partition2** dataset and change the **Block label** to *Validation dataset*.




More blocks can be connected to the output datasets following the same approach to create a workflow that produces, for example, a credit risk scorecard.

# Remove blocks from a workflow

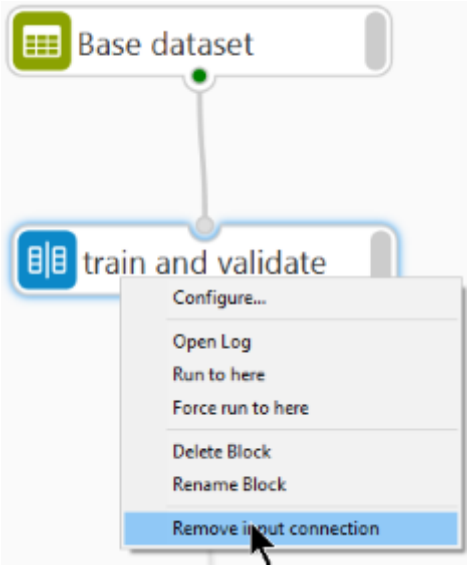
Blocks can be excluded from a workflow path by deleting the connectors leading to the **Input** port of the block.

Connections between blocks can be removed from the block at either end of the connection using the shortcut menu for the block. Where multiple connectors are linked to the **Input** port of a block, the shortcut menu displays all connectors identified by the block label.

Connectors leading to the **Input** port of a working dataset cannot be removed.

To exclude the **train and validate** block from the workflow:

1. In the **Workflow Editor** view, right-click the **train and validate** block.




2. Click **Remove input connection** in the shortcut menu.

Removing blocks enables you to alter the workflow, for example to attach a different dataset to the **train and validate** partition block to create new training and validation datasets for use in the rest of the workflow.

# Delete workflow block

Blocks and any associated working datasets can be deleted from a workflow. To delete a block, select the required block, right-click and click **Delete Block** in the shortcut menu.

Deleting a block deletes any associated working datasets created by the block; any connectors leading to either the **Input** port; any connectors leading from the **Output** port of the block; or if the block automatically creates working datasets, any connectors leading from the **Output** port for every working dataset created by the block.

Working datasets cannot be deleted; to delete a working dataset you must either delete the block that created the dataset, or change the number of working datasets created by the block. For example, to remove a partition working dataset created using the **Partition** block:

1. Right-click the **Partition** block and in the shortcut menu click **Configure**.
2. In the **Configure** dialog box, select the partition to be removed and click **Remove Partition**.

   You cannot delete the default partitions created when you drag a **Partition** block onto the **Workflow Editor** view.

# Copy and paste blocks

Blocks can be copied and pasted, either within the same workflow or to another workflow.

If a copied block has options or variables configured which are specific to a source dataset, then when the block is pasted and connected to another dataset, if the datasets are similar then the Workflow Editor will attempt to retain those configured options or variables. It is always recommended that you check the configuration of a pasted block.

Comments attached to blocks will be copied and pasted along with the block.

To copy and paste one or more Workflow blocks with any relevant connections:

1. Select one or many blocks.
   - To select a single block, left click on the block. This will also select the block's output if applicable.
   - To select multiple blocks, hold down **Ctrl** and then either use the left mouse button to drag a box around the blocks you want to select, or click individually on each block. Keep **Ctrl** held down.
2. Right click on a selected block and click **Copy Block**. You may also click the **Edit** drop down window and click **Copy**, or use the keyboard shortcut **Ctrl-C**.
3. Right click where you want to paste the block and select **Paste**. You may also click the **Edit** drop down window and click **Paste**, or use the keyboard shortcut **Ctrl-V**.

# Workflow execution

A workflow can run automatically or manually as specified in the Workflow preferences in the **Workflow** panel of the **Preferences** dialog box.

If the workflow runs automatically, additions to the workflow are evaluated when you connect new blocks to a workflow. If the workflow is manually run, select one of the following in the block shortcut menu:

- **Run to here** – evaluates the additions to the workflow, but uses the output from any previously-run blocks in the workflow.
- **Force run to here** – reruns the whole workflow, evaluating all blocks in the workflow and recreating all working datasets.

# Workflow block reference

This section provides a guide to the blocks currently supported by the **Workflow Editor** view.

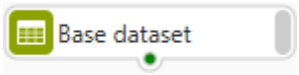
## Blocks

Blocks are the individual items that make up a workflow.

Blocks are used to represent:

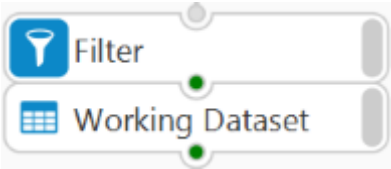
- Datasets, either imported into the workflow or created by the workflow.
- Programming code such as Python, R or the SAS language.
- Functionality that manipulates data to prepare a dataset for analysis.
- Modelling operations.

Blocks can be connected together to create a workflow using connectors. Blocks typically contain an **Output** port and an **Input** port. These ports enable you to link the output from one block to become the input to one or more other blocks. For example, **Permanent Dataset** blocks have an **Output** port located at the bottom of the block:




### Block input and output ports

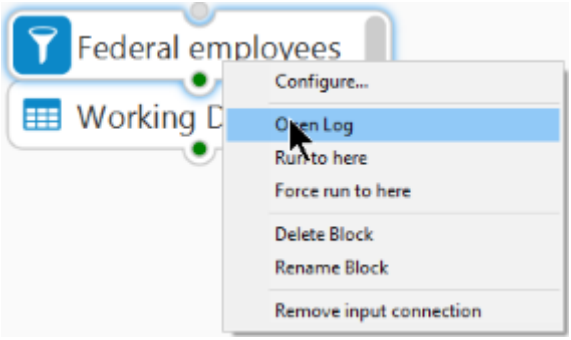
Blocks that manipulate a dataset have an **Input** port located at the top of a block, and an **Output** port located at the bottom of either the block, or the automatically-created output dataset of the block:




The Execution status in the **Output** port of the block indicates the state of the block when the workflow is run.

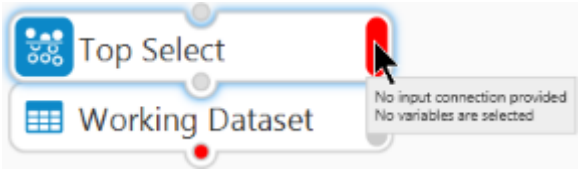
- If there are no errors, the Execution status is green.
- If the Execution status is grey, there is either an error in a connected upstream block, or the block does not yet have an input.

- If the Execution status is red there are errors in the block. You can review the error by reading the log output. Each block contains its own log file; to view the log, right click the block and click **Open Log** on the shortcut menu:




## Block configuration status

The Configuration status – the right-hand bar of a block – indicates whether all the required details have been specified for the block. If the Configuration status is grey, the block is complete and can be used in a workflow. If the Configuration status is red, some details are missing; hover over the bar to see what information is required to complete the block:




## Block shortcut menu

You can Configure, modify and delete blocks functionality using the block's shortcut menu. To do this, right-click the required block and choose the appropriate option:

- **Copy Block** - Copy the block to the clipboard so it can be subsequently pasted elsewhere, either within the current workflow or within another workflow.
- **Delete Block** – remove the block from the workflow. All connectors leading to and from the deleted block are also removed.
- **Rename Block** – enter a new display label for the block.
- **Remove output connection** – delete a connector linked to the **Output** port of the block. If there are multiple connections, you will see the option **Remove connection to**, which when highlighted opens a further menu to select the connection required.
- **Remove input connection** – delete a connector linked to the **Input** port of the block. If there are multiple connections, you will see the option **Remove connection from**, which when highlighted opens a further menu to select the connection required.
- **Configure** – specify the details of the block. For example, dataset location for **Import** blocks

Some blocks have additions to the shortcut menu. For example, the **Permanent Dataset** block and working datasets have **Open** and **Open with** options that enable you to view the dataset content with either the **Dataset File viewer** or in the **Data Profiler** view. You specify which viewer opens when you click **Open** on the shortcut menu using the **File Associations** panel in the **Preferences** dialog box.

Where the block runs some SAS language code, for example the **Partition** block, the menu contains an **Open Log** option that enables you to view the log created by the Workflow Engine.

# Block group palette

The palette provides tools to import and prepare datasets for analysis, and to then use those datasets to create and train models for use in, for example, scoring applications.



## Import group

Contains blocks that enable you to import data into your workflow.

# Database Import block

Enables you to use a data source reference to connect to a relational database management system (RDBMS) to import datasets from tables and views into a workflow.

You can use the **Database Import** block to import multiple tables and views into a workflow in a single step. The library reference used, and available library connections stored with the workflow are managed using the **Configure Database Import** dialog box.

To open the **Configure Database Import** dialog box, double-click the **Database Import** block. If no database is currently configured, a wizard will start to guide you through connecting to a database.

## Database Import wizard

**Step One: Database Type**

For step one of the wizard, select the database type from the list: DB2, MySQL, Oracle, ODBC, PostgreSQL, or SQL Server; then click **Next**.

**Step Two: Database Settings**

Step two of the wizard varies according to the database you are connecting to:

**MySQL Settings, PostgreSQL Settings, and SQL Server Settings**

Enter the **Host name**, **Port**, and choose the **Authentication** method as either **Credentials** or **Auth Domain**.

If you have chosen **Credentials**, then enter a **Username** and **Password** and click **Connect**. If you have chosen **Auth domain**, then choose an authorisation domain from the **Auth domain** drop down list.

**DB2 Settings**

Enter the **Database** name and choose the **Authentication** method as either **Credentials** or **Auth Domain**.

If you have chosen **Credentials**, then enter a **Username** and **Password** and click **Connect**. If you have chosen **Auth domain**, then choose an authorisation domain from the **Auth domain** drop down list.

**Oracle Settings**

Enter the **Net service** name (for example, the path to the Oracle database) and choose the **Authentication** method as either **Credentials** or **Auth Domain**.

If you have chosen **Credentials**, then enter a **Username** and **Password** and click **Connect**. If you have chosen **Auth domain**, then choose an authorisation domain from the **Auth domain** drop down list.

**ODBC Settings**

Enter the **Datasource Name** and choose the **Authentication** method as **Credentials**, **Auth Domain**, or **Use ODBC credentials**.

If you have chosen **Credentials**, then enter a **Username** and **Password** and click **Connect**. If you have chosen **Auth domain**, then choose an authorisation domain from the **Auth domain** drop down list.

If the connection is successful, a **Connection successful** window is displayed. If the connection fails, a **Connection failed** window is displayed; clicking **Details** on this window will display a connection log showing what went wrong.

**Step Three: Select Database and/or Schema**

Step three depends on what database you are connecting to:

**DB2 and Oracle**

Choose from the **Schema** drop down list and click **Next**.

**MySQL**

Choose from the **Database** drop down list and click **Next**.

**SQL Server and PostgreSQL**

Choose the **Database** first, then a **Schema** from the filtered drop down list, and finally click **Next**.

**ODBC**

Optionally and if supported, choose from the **Schema** drop down list and click **Next**.

**Step Four: Enter a name for the database**

In the **Name** box, enter a name to use to refer to the database.

Once the wizard is completed, a **Configure Database Import** window is displayed.

## Configure Database Import

**Database**

Specifies the database used as the source of data. The list contains libraries defined using the Database Import wizard. To add another database, click the **Create a new database** button (yellow cylinder with a green plus sign) to open the wizard.

**Tables**

Displays the tables defined in the selected database.

**Views**

Displays the views defined in the selected database. Views are a table-like object displaying the data from one or more tables that meet the selection requirements of the view.

# Excel Import block

Enables you to import a worksheet from a Microsoft Excel workbook.

You can import data from a single sheet or a named range in a Workbook. Headers can be imported as the names of observations and the data types modified during import.

**File Location**

Specifies where the dataset is located.

**Workspace**

Specifies the location of file containing the dataset is the current workspace.

**Workspace** datasets are only accessible when using the *Local Engine*.

**External**

Specifies the location of the file containing the dataset is on the file system accessible from the device running the Workflow Engine.

**External** files are accessible when using either a *Local Engine* or a *Remote Engine*.

**Path**

Specifies the path and file name for the file containing the required dataset. If you enter the **Path**:

- When importing a dataset from the Workspace, the root of the path is the Workspace. For example, to import a file from a project named `datasets`, the path is `/datasets/`*`filename`*.
- When importing a dataset from an external location, the path is the absolute (full) path to the file location.

  The **Path** to the file is only valid with the Workflow Engine in use when the workflow is created. This path will need to be re-entered if the workflow is used with a different Workflow Engine.

If you do not know the path to the file, click **Browse** and navigate to the required dataset in the **Choose file** dialog box.

**Data Format**

Specifies the import range and formats of the variables in the working dataset.

**Sheets**

Import data from a sheet in the workbook. Available sheets are listed in the **Sheets** box.

**Named Ranges**

Import data from a named range on a sheet in the workbook. Available named-ranges are listed in the **Name** box.

**Use first row as headers**

Specifies the data contains column labels in the first row that are variable labels and not part of the imported data. The headers become the variable name and label displayed in the **Data Profiler** view.

**Column Types**

Specifies the type of variables imported. If required, select the column and modify the format. You might, for example, change a numeric column to a date in the working dataset.

**Character**

Specifies the column as a character type in the working dataset.

**Numeric**

Specifies the column as a numeric type in the working dataset.

**Date**

Specifies the column as a date type in the working dataset. The **Format** specified determines how the value is displayed in the working dataset:

- `MMDDYY` specifies the display format is `MM/DD/YYYY`.
- `DDMMYY` specifies the display format is `DD/MM/YYYY`.
- `YYMMDD` specifies the display format is `YYYY-MM-DD`.

# Permanent Dataset block

Enables you to specify a WPD-format dataset to import into the workflow.

The dataset can be imported from a project in the current workspace, or from a location on the file system. Importing a dataset is configured using the **Configure Permanent Dataset** dialog box.

To open the **Configure Permanent Dataset** dialog box, double-click the **Permanent Dataset** block.

**File Location**

Specifies where the dataset is stored on disk.

**Workspace**

Specifies the location of the dataset is the current workspace.

**Workspace** datasets can only be imported when using the *Local Engine*.

**External**

Specifies that the location of the dataset is on the file system accessible from the device running the Workflow Engine.

**External** files can be imported when using either a *Local Engine* or a *Remote Engine*.

**Path**

Specifies the path and file name for file containing the WPD-format dataset. If you enter the **Path**:

- When importing a dataset from the Workspace, the root of the path is the Workspace. For example, to import a file named `mydata.wpd` from a project named `datasets`, the path is `/datasets/mydata.wpd`.
- When importing a dataset from an external location, the path is the absolute (full) path to the file location.

  The **Path** to the file is only valid for the Workflow Engine enabled when the workflow is created. This path will need to be re-entered if the workflow is used with a different Workflow Engine.

If you do not know the path to the file, click **Browse** and navigate to the required dataset in the **Choose file** dialog box.

# Text File Import block

Enables you to import a text file into the workflow. The file may have any file extension.

Importing a dataset is configured using the **Configure Text File Import** dialog box. To open the **Configure Text File Import** dialog box, double-click the **Text File Import** block

**File Location**

Specifies where the text file is stored on disk.

**Workspace**

Specifies the location of the dataset is the current workspace.

**Workspace** datasets can only be imported when using the *Local Engine*.

**External**

Specifies that the location of the dataset is on the file system accessible from the device running the Workflow Engine.

**External** files can be imported when using either a *Local Engine* or a *Remote Engine*.

**URL**

Specifies that the location of the dataset is hosted on a web site accessible from the device running the Workflow Engine. Only files specified using the HTTP or HTTPS protocols can be imported.

**External** files can be imported when using either a *Local Engine* or a *Remote Engine*.

**Path**

Specifies the path to file containing the dataset.

- When importing a dataset from the Workspace, the root of the path is the Workspace. For example, to import a file named `mydata.csv` from a project named `datasets`, the path is `/datasets/mydata.csv`.
- When importing a dataset from an external location, the path is the absolute (full) path to the file location.

The **Path** to the file is only valid with the Workflow Engine in use when the workflow is created. This path will need to be re-entered if the workflow is used with a different Workflow Engine.

Alternatively, click **Browse** and navigate to the required dataset in the selection dialog box.

**Data Format**

Specifies the file delimiter and formats of the variables in the working dataset.

**Delimiter**

Specifies the character used to mark the boundary between variables in an observation of the imported dataset.

If the required delimiter is not listed, select `Other` and enter the character to use as the delimiter for the imported dataset.

**Use first row as headers**

Specifies the data contains column labels in the first row that are variable labels and not part of the imported data. The headers become the variable name and label displayed in the **Data Profiler** view.

**Column Types**

Specifies the type of variables imported. If required, select the column and modify the format to, for example, change a numeric column to a date in the working dataset.

**Character**

Specifies the column as a character type in the working dataset. The **Width** specified determines the maximum length of the character string in the working dataset.

**Numeric**

Specifies the column as a numeric type in the working dataset.

**Date**

Specifies the column as a date type in the working dataset. The **Format** specified determines how the value is displayed in the working dataset:

- `MMDDYY` specifies the display format is `MM/DD/YYYY`.
- `DDMMYY` specifies the display format is `DD/MM/YYYY`.
- `YYMMDD` specifies the display format is `YYYY-MM-DD`.

# Data Preparation group

Contains blocks that enable you to modify the datasets in your workflow.



Enables you to apply a function to create a single value from a set of variable values grouped together using other variables in the input dataset.

# Aggregate block

Enables you to apply a function to create a single value from a set of variable values grouped together using other variables in the input dataset.

The aggregation of values is configured using the **Configure Aggregate** dialog box. To open the **Configure Aggregate** dialog box, double-click the **Aggregate** block.

**Grouping**

Specifies the variables to use when grouping observations in the dataset.

**Variable**

Specifies the variable to which the aggregation function is applied.

**Function**

Specifies the aggregation function to use. Before the function is applied, the dataset is collected into groups, and the function is applied to the values in the specified variable in the required grouping. Only functions that apply to the specified aggregation variable are displayed.

- **Average**: Returns the arithmetic mean of the specified variable. This function can only be used with numeric values.
- **Count**: Returns the number of occurrences of a numeric value or string in the specified variable.
- **Count(*)**: Returns the number of observations in a dataset.
- **Count (Distinct)**: Returns the number of unique occurrences of a numeric value or string in the specified variable.
- **Minimum**: Returns the minimum value in the input dataset for the specified variable. For character values, the returned value is the lowest value when the values are put in lexicographical order.
- **Maximum**: Returns the maximum value in the input dataset for the specified variable. For character values, the returned value is the highest value when the values are put in lexicographical order.
- **Sum**: Returns the total of all values in the specified variable. This function can only be used with numeric values.
- **Number of missings**: Returns the number of missing values in the specified variable.
- **Range (maximum - minimum)**: Returns the difference between Maximum and Minimum.
- **Standard deviation**: Returns the standard deviation for values in the specified variable.
- **Standard error**: Returns the standard error for values in the specified variable.
- **Variance**: Returns the variance for values in the specified variable.
- **Skewness**: Returns the skewness for values in the specified variable.
- **Kurtosis**: Returns the kurtosis for values in the specified variable.
- **1st Percentile**: Returns a value equivalent to the 1st percentile of the specified variable.
- **5th Percentile**: Returns a value equivalent to the 5th percentile of the specified variable.
- **10th Percentile**: Returns a value equivalent to the 10th percentile of the specified variable.
- **25th Percentile** / **Q1**: Returns a value equivalent to the 25th percentile of the specified variable.
- **75th Percentile** / **Q3**: Returns a value equivalent to the 75th percentile of the specified variable.
- **90th Percentile**: Returns a value equivalent to the 90th percentile of the specified variable.

- **95th Percentile**: Returns a value equivalent to the 95th percentile of the specified variable.
- **99th Percentile**: Returns a value equivalent to the 99th percentile of the specified variable.
- **Median**: Returns the median for values in the specified variable.
- **Mode**: Returns the mode for values in the specified variable.
- **Corrected sum of squares**: Returns the corrected sum of squares for values in the specified variable.
- **Coefficient of variation**: Returns the coefficient of variation for values in the specified variable.
- **Lower confidence limit**: Returns the lower confidence limit for values in the specified variable.
- **Upper confidence limit**: Returns the upper confidence limit for values in the specified variable.
- **Two tailed p-value for Student's t statistic**: Returns the two tailed p-value for the Student's t statistic for values in the specified variable.
- **Probability of a greater absolute value of Student's t test**: Returns the probability that a randomly drawn value from the specified variable will be greater than the absolute value given by the student's t statistic when applied to that variable.
- **Quartile range (Q3 - Q1)**: Returns the difference between the values equivalent to the Q3/75th and Q1/25th percentiles for the specified variable.
- **Student's t value**: Returns the value of the student's t statistic for the specified variable.
- **Uncorrected sum of squares**: Returns the value of the uncorrected sum of squares for the specified variable.

**New Variable**
Specifies the name for the aggregated variable in the output dataset.

# Binning block

Enables you to bin a variable in a dataset and the binning type used.

If the binning type is optimal, then you can set values that effect optimal binning. You can view the result of the binning, and adjust the settings if necessary. The block creates a working dataset in which a new variable is created for each observation. This variable contains the value for the bin into which the observation is placed.

The aggregation of values is configured using the **Configure Binning** dialog box. To open the **Configure Binning** dialog box, double-click the **Binning** block.

## Configure Binning dialog

Use the **Configure Binning** dialog to specify the details required by block.

**Binning variables**

Specifies the variable or variables to be binned. The upper list of **Unselected Variables** shows variables that can be specified for binning and the lower list of **Selected Variables** shows variables that have been specified for binning. To move a variable from one list to the other, double click on it. Alternatively, left click on a variable to select it and use the arrow keys; a selection can also be a contiguous list chosen using **Shift** and left-click, or a non-contiguous list chosen using **Ctrl** and left click.

**Binning type**

Specifies the type of binning. The available types depend on the format of the variable. Click on the box to display a list of types, and then select a type. There might only be one type listed. The types of binning that can be selected are:

**Optimal.**

Bins are created using optimal binning.

**Equal Height**

Bins are created using equal height binning.

**Equal Width**

Bins are created using equal width binning.

**Winsorised**

Bins are created after the data has been Winsorised.

**Optimal Binning**

The controls in this group are only displayed if you select Optimal as the binning type for a variable.

**Dependent variable**

Specifies the dependent variable used for optimal binning. Click the box to display a list variables, and then select the required variable.

**Variable treatment**

Specifies how the variable should be treated by the binning process. The supported treatments are:

**Binary**

Specifies a dependent variable that can take one of two values.

**Nominal**

Specifies a discrete dependent variable with no implicit ordering.

**Ordinal**

Specifies a discrete dependent variable with an implicit category ordering.

### Creating the bins

When you have selected the variable to be binned and specified the parameters for optimal binning, if required, you can create the bins. To do this, click **Bin Variables**. The bins are then created. The bins are created using values specified in the Preferences dialog. See the section **Binning** panel (page 130) for details.

The list box underneath the control lists the bins created by the binning process. These bins might be adequate, in which case you can click **OK**. The **Configure Binning** dialog is then closed, and a working dataset containing the binning variable is created.

If you decide the bins are not adequate, you can change the binning preferences. To do this, click **Binning Preferences** (  ). This opens the **Binning** panel in **Workflow** in **Preferences**. See the section **Binning** panel (page 130) for details.

## Filter block

Enables you to reduce the total number of observations in a large dataset by selecting observations based on the value of one or more variables.

When you filter a dataset using the **Filter** block, the input dataset remains unchanged and a working dataset is created containing only the observations selected with the filter.

### Filtering datasets

To filter datasets:

1. Drag the **Filter** block onto the canvas, and connect the required dataset to the **Input** port of the block.
2. Double-click the **Filter** block.

   The **Filter Editor** view displays and contains two tabs:

   - The **Basic** tab that enables you to create a filter that is either a logical conjunction (logical AND) or a logical disjunction (logical OR) of expressions.
   - The **Advanced** tab that enables you to create a complex filter statements that combines both the logical conjunction and logical disjunction of expressions .
3. Using either the **Basic** or **Advanced** filter tabs, specify a filter for the input dataset.

## Basic filter

Create a single filter that is either a logical OR expression or a logical AND expression.

The **Basic** view cannot be used to create a filter containing a combination of logical OR and logical ANDexpressions.

## Create a filter expression

To create a filter expression, click **Add expression** (  ) to create a child node, and complete the **Expression Properties** for the node. When a filter contains two or more expressions, select **AND** for a logical AND filter, or select **OR** for a logical OR filter.

## Expression settings

Expressions define individual nodes as part of a larger filter.

**Variable**

The variable in the input dataset to be filtered.

**Operator**

Specifies the operation used to select observations. The available operators are determined by the variable type.

The following operators are available for character data variables.

| | |
|---|---|
| `Equal to` | Includes observations in the working dataset if the value of the variable is equal to the specified **Value** specified. |
| `Not equal to` | Includes observations in the working dataset if the value of the variable is different to the specified **Value**.. |
| `In` | Includes observations in the working dataset if the value of the variable is contained in the defined list of values. The list is specified in the **Edit** dialog box accessed from the **Configure Filter** dialog box. |
| `Starts with` | Includes observations in the working dataset if the value of the variable starts with the specified **Value**. |
| `Ends with` | Includes observations in the working dataset if the value of the variable ends with the specified **Value**. |

The following filter operators are available for numeric data variables.

| | |
|---|---|
| `=` | Includes observations in the working dataset if the value of the variable is equal to the specified **Value** specified. |
| `!=` | Includes observations in the working dataset if the value of the variable is different to the specified **Value**.. |
| `In` | Includes observations in the working dataset if the value of the variable is contained in the defined list of values. The list is specified in the **Edit** dialog box accessed from the **Configure Filter** dialog box. |
| `<` | Includes observations in the working dataset if the value of the variable is less than the specified **Value**. |
| `<=` | Includes observations in the working dataset if the value of the variable is less than, or the same as, the specified **Value**. |
| `>` | Includes observations in the working dataset if the value of the variable is greater than the specified **Value**. |

| | |
|---|---|
| >= | Includes observations in the working dataset if the value of the variable is greater than, or the same as, the specified **Value**. |

**Value**

The variable value used as the test in the filter. All character data entered in the **Value** box is case-sensitive.

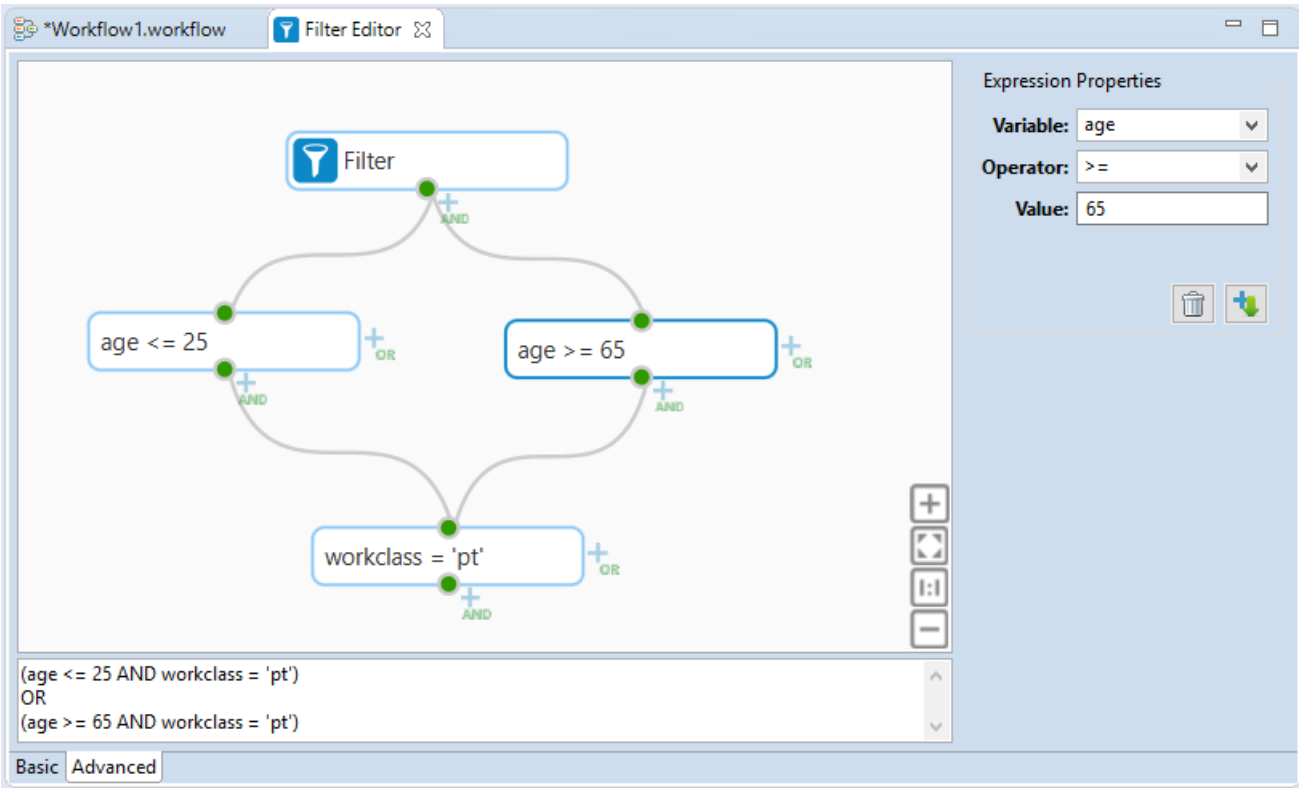
## Advanced filter

Create a complex filter statement that combines both logical OR and logical AND filters.

A filter is an expression sequence visually represented by connected expressions in a similar manner to a workflow. The filter path for each branch represents a logical AND of expression sequence. Each branch from dataset root are joined together as a logical OR sequence in the filter.

The **Advanced** filter contains the following:

- The filter canvas, which contains a visual representation of the filter.
- A box on the lower left, which contains the version of the filter that can be copied to a SAS language program for testing or production use.
- A box on the right, which enables you specify the properties for a node in the filter.



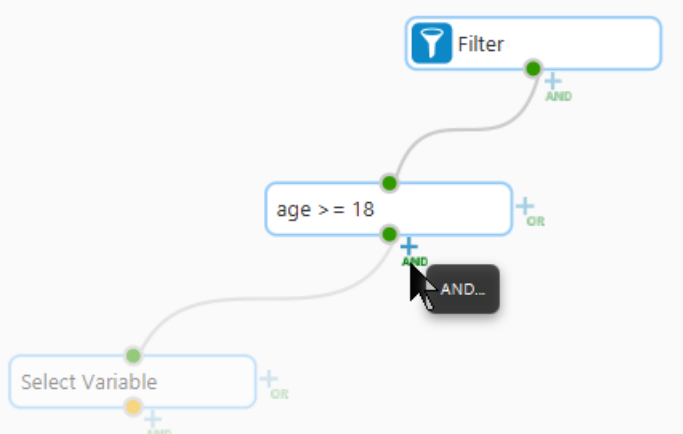
In the above example, each path is a different filter for the dataset. The alternate paths are joined at the point closest to the dataset root. Expressions can be shared between filter paths:

```
age <= 25 AND workclass='pt'
OR
age >= 25 AND workclass='pt'
```

## Create a filter expression

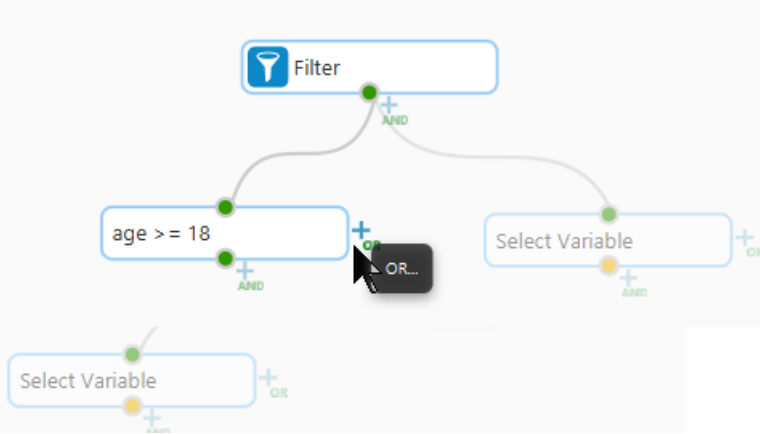
You can create complex join expressions by connecting the ports on the nodes in the filter canvas.

To add a logical AND expression, hover over the **AND** icon (AND) next to the expression node. The location of the new expression is shown as a child of the current expression node on the filter canvas:




If the expression is being added in the correct location, click the **AND** icon and complete the **Expression Properties** for the node.

To add a logical OR expression, hover over the **OR** icon (OR) next to the expression node. The location of the new expression is shown as a child of the current expression's parent node on the filter canvas:




If the expression is being added in the correct location, click the **OR** icon and complete the **Expression Properties** for the node.

## Editing the filter expression

To edit the filter properties, click the expression node and modify the properties for that expression in **Expression Properties**.

To delete the expression either click the  icon in the **Expression Properties** box, or right-click the node in the filter canvas and click **Delete expression**.

## Expression Properties

Expression properties define individual nodes as part of a larger filter.

**Variable**

The variable in the input dataset to be filtered.

**Operator**

Specifies the operation used to select observations. The available operators are determined by the variable type.

The following operators are available for character data variables.

| | |
|---|---|
| `Equal to` | Includes observations in the working dataset if the value of the variable is equal to the specified **Value** specified. |
| `Not equal to` | Includes observations in the working dataset if the value of the variable is different to the specified **Value**.. |
| `In` | Includes observations in the working dataset if the value of the variable is contained in the defined list of values. The list is specified in the **Edit** dialog box accessed from the **Configure Filter** dialog box. |
| `Starts with` | Includes observations in the working dataset if the value of the variable starts with the specified **Value**. |
| `Ends with` | Includes observations in the working dataset if the value of the variable ends with the specified **Value**. |

The following filter operators are available for numeric data variables.

| | |
|---|---|
| `=` | Includes observations in the working dataset if the value of the variable is equal to the specified **Value** specified. |
| `!=` | Includes observations in the working dataset if the value of the variable is different to the specified **Value**.. |
| `In` | Includes observations in the working dataset if the value of the variable is contained in the defined list of values. The list is specified in the **Edit** dialog box accessed from the **Configure Filter** dialog box. |
| `<` | Includes observations in the working dataset if the value of the variable is less than the specified **Value**. |
| `<=` | Includes observations in the working dataset if the value of the variable is less than, or the same as, the specified **Value**. |

| | |
|---|---|
| > | Includes observations in the working dataset if the value of the variable is greater than the specified **Value**. |
| >= | Includes observations in the working dataset if the value of the variable is greater than, or the same as, the specified **Value**. |

**Value**

The variable value used as the test in the filter. All character data entered in the **Value** box is case-sensitive.

# Impute block

Enables you to assign or calculate values to fill in any missing values in a variable.

The available methods for replacing missing values are determined by variable type and are specified in the **Configure Impute** dialog box. You can define expressions to replace missing values in all variables in a single **Impute** block. To add expressions to the block click **Add Expression** and complete the information for the expression.

Imputation of missing values is configured using the **Configure Impute** dialog box. To open the **Configure Impute** dialog box, double-click the **Impute** block.

**Variable**

Specifies a variable in the dataset with missing values. **Variable** is pre-populated with variables in the dataset

**Method**

Specifies the impute method used for missing values in the specified variable.

**Character variables**

The following impute methods are available for character variables.

**`Constant`**

Replaces missing values for the variable with the specified value. Specify the replacement in **Value**.

**`Distribution`**

Replaces missing values in the specified variable with randomly selected values present elsewhere in the variable. The replacement values are selected based on their probability of occurrence in the input dataset.

**`Mode`**

Replaces missing values in the specified variable with the mode of the specified variable.

### Numeric variables

The following impute methods are available for numeric variables. In the SAS language, numeric variables include all date-, datetime- and time-formatted variables.

**`Gaussian simulation`**
Replaces missing values in the specified variable with a random number from a Normal distribution based on the mean and standard deviation of that distribution. The mean and standard deviation are calculated from the input dataset and displayed in the **Mean** and **Standard deviation** fields respectively.

You can specify different mean and standard deviation values in the fields, but altering the values too far from those calculated might increase the amount of noise in the data.

**`Max`**
Replaces missing values in the specified variable with the maximum value of the specified variable.

**`Mean`**
Replaces missing values in the specified variable with the mean of the specified variable.

**`Median`**
Replaces missing values in the specified variable with the median of the specified variable.

**`Min`**
Replaces missing values in the specified variable with the minimum value of the specified variable.

**`Constant`**
Replaces missing values for the variable with a specified value. Specify the replacement in **Value**.

**`Distribution`**
Replaces missing values in the specified variable with randomly-selected values present in the variable. Replacement values are selected based on the frequency with which they occur in the input dataset.

**`Mode`**
Replaces missing values in the specified variable with the mode of the specified variable.

**`Trimmed mean`**
Replaces missing values in the specified variable with the mean value calculated, after removing the specified percentage of largest and smallest values. For example, if you specify a percentage of 25, the smallest 25% of values and largest 25% of values are removed from the calculation. The interquartile mean value replaces all missing values in the variable.

The percentage of values removed from the calculation is specified in the **Percentage** field.

**`Uniform simulation`**
Replaces missing values in the specified variable with a random number from a Uniform distribution. By default, the distribution range is based on the minimum and maximum values for the specified variable, displayed in the **Min** field and **Max** field.

Only values within the specified boundaries are used to replace missing values in the variable, not the values specified in **Min** and **Max**.

**`Winsorized mean`**
Replaces missing values in the specified variable with the calculated Winsorized mean value, after replacing the specified percentage of largest and smallest values. For example, if you specify 5, the largest 5% of values are replaced with the value of the 95th percentile; the lowest 5% of values are replaced with the value of the fifth percentile; and all non-missing values in the specified variable are used to calculate the Winsorized mean.

The percentage of values replaced before the calculation occurs is specified in the **Percentage** field.

**Seed**
Specifies whether the same set of imputed values is generated each time the **Impute** block is executed.

- To generate the same sequence of observations, set **Seed** to a positive value greater than or equal to 1.
- To generate a different sequence of observations, set **Seed** to 0 (zero).

# Join block

Enables you to combine variables from two datasets into a single working dataset.

Observations can be joined using one or more variables, for example if all datasets have an *ID* variable, the variables for the same value of *ID* in each dataset can be joined together in the same observation. Where variables have the same name in both datasets, the variables in the second dataset are dropped.

## Joining datasets

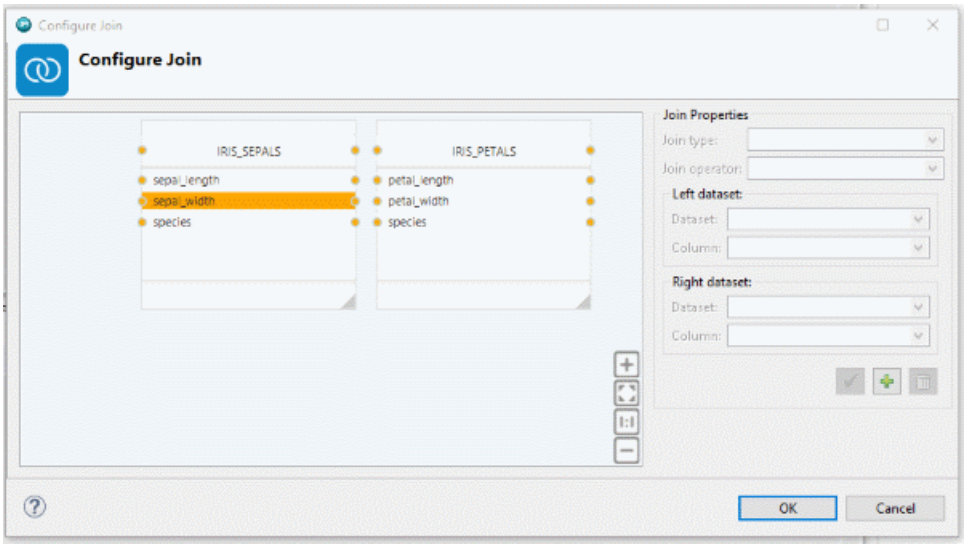
To join datasets:

1. Drag the **Join** block onto the canvas, and connect it to the datasets you want to join.
2. Double-click the **Join** block.

   The **Configure Join** dialog box is displayed.

**3.** Specify the type of join, and the variables to be joined, in the dialog box.

The **Configure Join** dialog contains two boxes:




- The left box contains a representation of the datasets, including the variables they contain.
- The right box contains controls that enable you specify the type of join, and the variables to be joined.

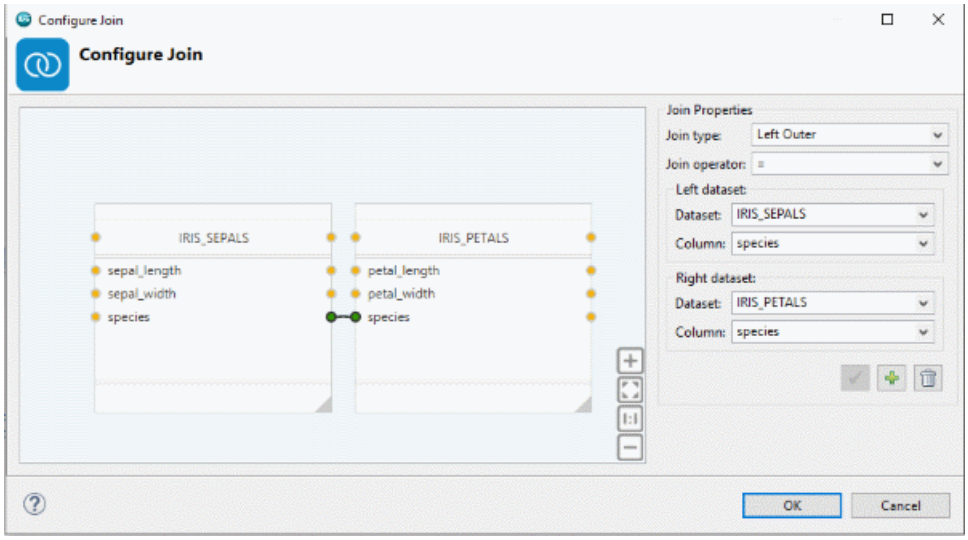
You can create a join in two ways:

- By joining the ports in the dataset representation.
- By setting properties in Join Properties.

## Joining datasets using the dataset representations

You can join datasets by connecting the ports on the dataset representations. To do this, click on the connector port ( ) on one dataset representation, and then drag to draw a line to a connector port on the other dataset representation.

When you do this, the **Join type** in the Join Properties group defaults to **Left Outer**. All the other controls in the Join Properties group are set to the variables and datasets selected.




The dataset defined as the left dataset is the dataset representation from which you drew the connecting line.

You can edit Join Type and Join Operator if you want to specify your own settings.

If you change the settings from their defaults, you must click ✓ to save the changes.

You can add another variable by connecting the appropriate ports.

You can create a natural join between datasets by joining the ports corresponding to the dataset titles in the dataset representation.

## Joining datasets using Join Properties

You can create a join by clicking **+** in the Join Properties, and then editing the other controls in Join Properties. When you have finished, click ✓. If you want to add another variable, click **+** again and set the properties for that join.

When you create a join in this way, the connection between the variables is shown in the dataset representation.

## Editing the join properties

To edit the join properties, click the connection between two variables in the dataset representation. The properties for that join are then set using the controls in Join Properties.

You can delete the join, using the icon.

## Join Properties

The properties of the join can be specified using the controls in the Join Properties group. If you join variables using the dataset representation in the left-hand box of the **Configure Join** dialog box, default properties are applied; these can be edited as required. The following properties can be specified

**Join type**

Specifies the type of join used to combine the values in the left and right datasets. Inner and Outer joins use a variable condition for selecting observations that enables you to select the variables you want to compare. The variable condition is defined using the **Join operator** dialog box.

**Inner**

Creates a working dataset that only includes observations where the variable in the left dataset has a value that matches the variable in the right dataset, as defined in the variable condition.

**Left Outer**

Creates a working dataset that contains all observations in the left dataset and appends variables to those observations from the right dataset.

Variables from the right dataset are only added to observations in the working dataset if the values of the variables match in the defined variable condition.

**Right Outer**

Creates a working dataset that contains all observations in the right dataset and appends variables to those observations from the right dataset.

Variables from the left dataset are only added to observations in the working dataset if the values of the variables match in the defined variable condition.

**Natural**

Creates a working dataset using commonly-named variables in the Left and right datasets. The dataset only includes observations where the variable in the left dataset has a value that matches the variable in the right dataset, as defined in the variable condition.

**Cross**

Creates a working dataset that consists of all observations in the left dataset multiplied by the all observations in the right datasets.

**Join operator**

Specifies the type of operator for the join. The condition created with the operator defines the variables included or excluded by a join type:

| | |
|---|---|
| = | Equal to |
| != | Not equal to |
| >= | Greater than or equal to |
| < | Less than |
| <= | Less than or equal to |

**Left dataset**

Identifies the left dataset in the join, and the variables on which to join:

**Dataset**

Select the dataset to be used as the left dataset. This is set by default if you connect datasets using the dataset representation.

**Column**

Specify the variable in the left dataset to be used in the join. This is set by default if you connect datasets using the dataset representation.

**Right dataset**

Identifies the right dataset in the join and the columns:

**Dataset**

Select the dataset to be used as the right dataset. This is set by default if you connect datasets using the dataset representation.

**Column**

Specify the variable in the right dataset to be used in the join. This is set by default if you connect datasets using the dataset representation.

# Merge block

Enables you to combine two datasets into a single working dataset.

A Merge block takes two or more connections from datasets and joins those datasets together. To configure a merge block, double click to open the **Configure Merge** dialog box, which has options as described below.

**Dataset order**

Lists the order that the datasets will be combined in. To move a dataset in the order, click on that dataset and use the arrow keys to move it.

**Merge operation**

**Concatenate**

Joins one dataset onto another. Variables from each dataset will remain distinct with no merging between observations. In both cases, variables and observations in datasets will be concatenated in the order specified in **Dataset order**.

**Interleave**

Available for selection if source datasets contain common variables (dictated by variable name and type). Observations from common variables will be joined so that they belong to the same variable. Variables and observations in datasets will be interleaved in the order specified in **Dataset order**.

**One-to-one**

Joins datasets together so that observations are merged. Variables and observations in datasets will be joined in the order specified in **Dataset order**.

**Include excess rows**

If deselected (default), trims excess observations from the larger dataset such that both datasets are presented as the same size.

If selected, includes excess observations, such that the smaller dataset contains blank observations for rows beyond its original size.

# Mutate block

Enables you to add new variables to the working dataset. These variables can be independent of, or derived from variables in the existing dataset.

The new variable name, type and format are set and the content based on an SQL expression. You can, for example, use the expression to duplicate the content of a variable into the new variable, or apply a function to create a new variable based on the value of an existing variable in the dataset.

The input dataset remains unchanged, and any variables created using the **Mutate** block are appended to the output working dataset.

New variables are defined using the **Configure Mutate** dialog box. To open the **Configure Mutate** dialog box, double-click the **Mutate** block.

## Mutated Variables

The main panel in this section gives a list of new mutated variables created by this block. Each variable displays its **Variable Name**, **Type** and **Format**. The list can be sorted by clicking on any of the header fields.

**Plus icon**

Add a new mutated variable. This will enable the fields in the **Variable Definition** section and add a new entry to the Mutated Variables list.

**Delete icon**

Deletes a selected variable from the **Mutated Variables** list.

## Variable Definition

Describes the new mutatated variable to be created.

**Name**

Specifies the variable name displayed when the working dataset is viewed in the **Data Profiler** view or the Dataset File Viewer.

**Type**

Select the SAS language type for the new variable:

- `Character` for all character- and string-based variables.

- `Numeric` for all numeral-based variables including any date, time or datetime variables.

**Label**

Specifies the display name that can be used when outputting graphical interpretation of the variable.

**Length**

Specifies the maximum length of a variable.

**Format**

The display format for the new variable. The format applied does not affect the underlying data stored in the `DATA` step. For more information see the section *Formats* in the *WPS Reference for Language Elements*.

**Informat**

The format applied to data as it is read into a dataset. For more information see the section *Informats* in the *WPS Reference for Language Elements*.

**SQL Expression**

Defines the data content of the new variable. The SQL expression can either be written into this area manually, or created by double-clicking on or dragging the input variables, functions and operators given below the **SQL Expression** entry area, which will populate the SQL Expression area for you. When typing manually, press **Ctrl**-**Spacebar** to show a list of suggested terms.

The SQL expression can be used to duplicate an existing variable by entering that variable's name; conditionally create new values using the `SQL` procedure `CASE` expression, or create values that are independent from existing variables.

For example, you can create a variable containing the current date and time by manually entering the SAS language function `DATETIME`. Applying a format such as `DATETIME.` enables the content to be displayed in a readable date and time format.

**Input Variable**

Lists the existing variables in the dataset being mutated. Double clicking an **Input Variable** will add it to the **SQL Expression**. Alternatively, input variables can be dragged into the **SQL Expression** area. Variable names with spaces are automatically enclosed in quotes and suffixed with an 'n' to define them as name literals.

**Function**

Lists supported SQL functions. Double clicking an SQL function will add it to the **SQL Expression** area. Alternatively, functions can be dragged into the **SQL Expression** area. Supported functions are:

- **AVG** and **MEAN**: Both return the arithmetic mean of a list of numeric values in a specified variable. `AVG` is an alias of `MEAN`.
- **COUNT**: Can be used in three forms:
  - `Count(*)`: returns the total number of observations.
  - `Count(variable)`: returns the number of non-missing values in the specified variable.

    - `Count(variable,"string")`: returns `true` (`1`) for each observation where the specified string occurs in the specified variable, otherwise returns `false` (`0`).
- **FREQ** and **N**: Both return a count of the number of observations for a specified variable, with behaviour dependent on the variable type:
    - If a numeric variable is specified, missing values are included.
    - If a string variable is specified, missing values are excluded.

  `FREQ` is an alias of `N`.
- **CSS**: Returns the corrected sum of squares of a list of numeric values.
- **CV**: Returns the coefficent of variation (standard deviation as a percentage of the mean) for a list of numeric values.
- **MAX**:
    - If a numeric variable is specified, returns the maximum value.
    - If a string variable is specified, returns the longest string.
- **MIN**:
    - If a numeric variable is specified, returns the minimum value.
    - If a string variable is specified, returns the shortest string.
- **NMISS**: Returns the number of missing values in a list of string or numeric values.
- **PRT**: Returns the probability that a randomly drawn value from the Student T distribution is greater than the T statistic for a numeric variable.
- **RANGE**: Returns the difference between the maximum and minimum value in a list of numeric values.
- **STD**: Returns the standard deviation of a list of numeric values.
- **STDERR**: Returns the standard error of the mean of a list of numeric values.
- **SUM**: Returns the sum of values from a list of numeric values.
- **SUMWGT**: Returns the sum of weights.
- **T**: Returns the t statistic for a variable.
- **USS**: Returns the uncorrected sum of squares of a list of numeric values.
- **VAR**: Returns the variance of a list of numeric values.
- **COALESCE**: Returns the first non-missing value in a list of numeric values.
- **COALESCEC**: Returns the first string that contains characters other than all spaces or null.
- **MONOTONIC**: Generates a monotonically increasing sequence, starting at the number of observations in the dataset plus one.

**Operators**

Lists SQL operators that can be used with Input Variables and Functions. Clicking an operator will add it to the **SQL Expression** area above.

# Partition block

Enables you to divide the observations in a dataset into two or more working datasets, where each working dataset contains a random selection of observations from the input dataset.

A **Partition** block is typically used to create *training* and *validation* working datasets for model training.

By default, two working datasets are created that each contain 50% of the input dataset observations. You can add or remove output datasets in the **Configure** dialog box, alter the weighting to reflect the relative importance of the output datasets created, and introduce a seed to replicate the split of datasets.

Data partitioning is configured using the **Configure Partition** dialog box. To open the **Configure Partition** dialog box, double-click the **Partition** block.

## Defining partitions

The **Configure Partition** dialog box contains a list of all partitions that will be created.

- To create a new partition, click **Add Partition**. The new partition has a weight of 1.00.
- To delete a partition, select the partition in the **Partition** column and click **Remove Partition**.
- To rename a partition, select the partition in the **Partition** column and enter a new name.
- To change the weighting applied to a partition, select the value in **Weight** column and enter a new weighting value.

## Controlling observations in a partition

Every time the **Partition** block is executed, the partition weighting is persisted, but the sequence of observations in the working datasets may change. The **Seed** field enables you to recreate the same sequence of observations each time the **Partition** block is executed.

- To generate the same sequence of observations, set **seed** to a positive value greater than or equal to 1.
- To generate a different sequence of observations, set **seed** to 0 (zero).

# Rank block

Enables you to rank observations in an input dataset using one or more numeric variables.

The input dataset remains unchanged, and any rank variables created using the **Rank** block are appended to the output working dataset.

Rank scores are defined using the **Configure Rank** dialog box. To open the **Configure Rank** dialog box, double-click the **Rank** block.

## Variable Selection panel

The **Variable** list contains all numeric variables in the input dataset. To select a variable, click the check box corresponding to the variable label. You can include multiple variables in the working dataset.

## Parameters panel

Enables you to specify how the rank scores for observations in the dataset are calculated.

**General**

Specifies ranking order and how identical results are ranked.

**Rank order**

Specifies the order in which ranking scores are applied to observations ordered by the variables specified in the **Variable Selection** panel.

**Ascending**

Rank scores are specified in ascending order, with the lowest rank applied to the smallest observation.

**Descending**

Rank scores are specified in descending order, with the highest rank applied to the smallest observation.

**Ties**

Specifies the ranking of identical results.

The following input dataset contains the result of the *IAAF 2015 World Championships 100 Meters Men – Final*, and is used in the examples below.

| Name | Time |
|---|---|
| Bolt, Usain | 9.79 |
| Bromell, Trayvon | 9.92 |
| De Grasse, Andre | 9.92 |
| Gatlin, Justin | 9.80 |
| Gay, Tyson | 10.00 |
| Powell, Asafa | 10.00 |
| Vicaut, Jimmy | 10.00 |
| Rodgers, Mike | 9.94 |
| Su, Bingtian | 10.06 |

**Dense**

Ranks the variable using the dense tie resolution. The method assigns the same rank value to all tied variables and the next variable is assigned the immediately-following rank.

Applying `Dense` to the input dataset above, creates the following working dataset:

| Name | Time | Time_RANK |
|---|---|---|
| Bolt, Usain | 9.79 | 1 |
| Bromell, Trayvon | 9.92 | 3 |
| De Grasse, Andre | 9.92 | 3 |
| Gatlin, Justin | 9.80 | 2 |
| Gay, Tyson | 10.00 | 5 |
| Powell, Asafa | 10.00 | 5 |
| Vicaut, Jimmy | 10.00 | 5 |
| Rodgers, Mike | 9.94 | 4 |
| Su, Bingtian | 10.06 | 6 |

**High**

Ranks the variable using the high tie resolution. The method assigns the highest rank value to all tied variables.

Applying **High** to the input dataset described above creates the following working dataset:

| Name | Time | Time_RANK |
|---|---|---|
| Bolt, Usain | 9.79 | 1 |
| Bromell, Trayvon | 9.92 | 4 |
| De Grasse, Andre | 9.92 | 4 |
| Gatlin, Justin | 9.80 | 2 |
| … | … | … |

Bromell, Trayvon and De Grasse, Andre have the same time, and ranks 3 and 4. Specifying **High** gives both a rank of 4.

**Low**

Ranks the variable using low tie resolution. The method assigns the lowest rank value to all tied variables.

Applying `LOW` to the input dataset described above creates the following dataset:

| Name | Time | Time_RANK |
|---|---|---|
| Bolt, Usain | 9.79 | 1 |
| Bromell, Trayvon | 9.92 | 3 |
| De Grasse, Andre | 9.92 | 3 |
| Gatlin, Justin | 9.80 | 2 |
| … | … | … |

Bromell, Trayvon and De Grasse, Andre have the same time, and ranks 3 and 4. Specifying **Low** gives both a rank of 3.

**Mean**

Ranks the variable using mean tie resolution. The method assigns the mean rank value to all tied variables.

Applying `Mean` to the input dataset described above creates the following working dataset:

| Name | Time | Time_RANK |
|---|---|---|
| Bolt, Usain | 9.79 | 1 |
| Bromell, Trayvon | 9.92 | 3.5 |
| De Grasse, Andre | 9.92 | 3.5 |
| Gatlin, Justin | 9.80 | 2 |
| … | … | … |

Bromell, Trayvon and De Grasse, Andre have the same time, and ranks 3 and 4. Specifying **Mean** gives both a rank of 3.5.

**Ranking Method**

Specifies how the rank value for each observation is calculated.

**Ordinal**

Assigns an incrementing rank number to each observation.

**Fractional**

Each rank value is calculated as the ordinal ranking method divided by the number of observations in the dataset.

**Fractional (N+1)**

Each rank value is calculated as the ordinal ranking method divided by the one plus the number of observations in the dataset.

**Percent**

Each rank value is calculated as the ordinal ranking method divided by the number of observations in the dataset, expressed as a percentage.

**Savage**

The rank values for the observations are transformed to an exponential distribution using the Savage method. Subtracting one from the transformation result centers the rank score around zero (0):

$$t_{RN} = \left[ \sum_{r=1}^{R} \frac{1}{(N - r + 1)} \right] - 1 \text{ for} \left( R = 1, \ldots, N \right)$$

Where:

- $R$ is the rank score.
- $N$ is the number of observations.

**Groups**

Divides the observations into the specified number of groups based on the ranking score. Observations with tied ranking scores are allocated to the same group.

**Normal**

The rank values for the observations are transformed to a standard normal distribution using the selected method:

**Blom**

Creates the standard normal distribution of rank scores using a Blom transformation:

$$F_i = \frac{\left(R_i - \frac{3}{8}\right)}{\left(N + \frac{1}{4}\right)}$$

Where:

- $R$ is the rank score.
- $N$ is the number of observations.

**Tukey**

Creates the standard normal distribution of rank scores using a Tukey transformation:

$$F_i = \frac{\left(R_i - \frac{1}{3}\right)}{\left(N + \frac{1}{3}\right)}$$

Where:

- $R$ is the rank score.
- $N$ is the number of observations.

**VW**

Creates the standard normal distribution of rank scores using a Van der Waerden transformation:

$$F_i = \frac{R_i}{(N+1)}$$

Where:

- $R$ is the rank score.
- $N$ is the number of observations.

# Sampling block

Enables you to create a working dataset that contains a small representative sample of observations in the input dataset.

A sample is configured using the **Configure Sampling** dialog box. To open the **Configure Sampling** dialog box, double-click the **Sampling** block.

**Sample Type**

Specifies the type of samples, either *Random* or *Stratified*.

**Random Sampling**

A working dataset is created by selecting a random set of observations to match the required sample size.

**Sample Size**

Specifies how the sample is constructed:

- **Percentage of obs** specifies the percentage of observations in the input dataset to include in the output working dataset.
- **Fixed number of obs** specifies the absolute number of observations in the input dataset to include in the output working dataset.

**Stratified Sampling**

The dataset is divided in groups where observations contain similar characteristics. Sampling is applied to each group, ensuring that each group is represented in the output working dataset.

**Strata Variable**

Specifies the variable in the dataset to use when dividing the whole dataset into groups.

**Sample Size**

Specifies how the sample is constructed:

- **Percentage of stratum obs** specifies the percentage of each group to include in the output working dataset.
- **Custom** enables you to determine the number of observations from each group, allowing you to increase or decrease the weight of each group in the output working dataset.

**Use Probability Proportional to Size Sampling**

Specifies that samples are taken using the same probability used to divide the input dataset into groups.

**Size variable**

Specifies a variable that indicates sample sizes.

**Seed**

Specifies whether the same set of observations is included in the output working dataset each time the **Sampling** block is executed. A **Seed** can be applied to either a random sample or a stratified sample.

- To generate the same sequence of observations, set **Seed** to a positive value greater than or equal to 1.
- To generate a different sequence of observations, set **Seed** to 0 (zero).

# Select block

Enables you to create a new dataset containing specific variables from an input dataset.

Variables included in the working dataset are specified using the **Configure Select** dialog box. To open the **Configure Select** dialog box, double-click the **Select** block.

The **Variables** list displays all variables in the input dataset. To select a variable, click the check box next to the variable label. Multiple variables can be selected to be included in the working dataset:

- To select a range of variables, click the label of the first variable. Press Shift and click the label of the last required variable and then select the check box for the variable.
- To select a non-contiguous group press **Ctrl** and select the check boxes for the required variables in the list.

# Sort block

Enables you to sort a dataset.

You can sort a dataset using one or more variables. For example, if you were sorting a dataset containing book titles and author names, you could first sort by author name, to get all authors in alphabetical order, and then title, so that the titles for each author are also sorted alphabetically. You can sort in ascending or descending order, and you can specify what happens with duplicate records and keys.

## Sorting datasets

To sort datasets:

1. Drag the **Sort** block onto the canvas, and connect the required input dataset.
2. Double-click the **Sort** block.

   The **Configure Sort** dialog box is displayed.
3. Specify the variables to be sorted, and how they are sorted, in the dialog.
4. Click **OK**.

The dataset is sorted according to the variables and sort order you specify, and a working dataset containing the sorted dataset is created.

### The Configure Sort dialog box

The **Configure Sort** dialog box contains a list in which you can specify the variables to be sorted, and controls to specify what to do with duplicate records.

**Variable list**

The variables on which to sort the dataset are specified in the list box at the top of the **Configure Sort** dialog box.

Click **Add Variable** to add a variable to the list. By default, the first variable in the dataset is added to the list. You can then change this to the variable on which you want to base the sort by clicking the variable name. A drop-down menu is displayed from which you can select the variable you require.

The ascending sort order is specified by default when you add a variable. This can be changed by clicking in the **Sort Order** box corresponding to the variable, and selecting **Descending** from the menu.

The order in which variables are specified in the listed defines the order in which variables are sorted. You can change the order by selecting a variable from the list, and click **Move Up** or **Move Down**, as required, to change the position of the variable in the list.

You can remove a variable from the list by selecting it, and click **Remove Variable**.

**Remove Duplicate Keys**

Specifies that, for observations that have duplicate keys, only the first of the observations with a duplicated key is included in the sort and written to the working dataset.

**Remove Duplicate Records**

Specifies that, for observations that are exactly the same, only the first of the duplicated observations is included in the sort and written to the working dataset.

**Maintain Order of Duplicates**

Specifies that, for observations that are exactly the same, the order of the duplicate observations in the sorted dataset remain the same as in the original dataset.

## Top Select block

Enables you to create a working dataset containing a dependent variable and its most influential independent variables.

Creating a working dataset enables you to reduce the number of variables as input to model training, for example when growing a decision tree or for logistic regression.

The dataset is configured using the **Configure Top Select** dialog box. To open the **Configure Top Select** dialog box, double-click the **Top Select** block.

To create a new dataset, select a **Dependent Variable** from the list. The **Variables** list shows all independent variables in the dataset, displayed in the list by entropy variance order. To include an independent variable in the resulting dataset, select the box next to the variable name.

When selecting independent variables:

- You should avoid selecting an independent variable with an entropy variance of 1 (one) as the values will typically be too random to provide a good predictive power and may lead to overfitting a model.
- You should avoid selecting an independent variable where the entropy value is 0 (zero) as this has a very low predictive power.
- Typically, you would select a group of independent variables where the difference in the entropy variance of the variables is small.

For more information about how the entropy variance is calculated see *Predictive power criteria* (page 24).

# Transpose block

Enables you to create a new dataset with transposed variables.

Enables a dataset to be transposed, converting columns (variables) to be rows (observations) and vice versa.

**Note:**

WPS imposes a 32,000 variable (column) limit on datasets. If a transpose would result in this limit being breached, the transpose block will warn you that the dataset will be truncated.

**Configure Transpose**

Double click on the transpose block to open the Configure Transpose dialog box. It has three sections, described below.

**Variable Selection**

Chooses which variables from the source dataset to **Transpose** and which to **Keep** the same.

**By Variables**

Specifiying a source dataset variable creates a new dataset with the first variable as this specified variable. Each observation in the specified variable is repeated a number of times corresponding to the number of observations in the source dataset. Another variable is added to the end of the dataset, listing observations from all the other variables that correspond to the specified variable's repeated observation.

Setting the **Sort Order** for a specified variable will sort the specified variable's observations either **ascending** or **descending**.

Selecting multiple source dataset variables will nest this repetition.

**Variable Names**

Allows variables created by the tranpose block to be named.

- **Output Variable Names**: Defines the name of each output variable.
  - **Prefix**: Adds the specified string to the beginning of each new variable name.
  - **Suffix**: Adds the specified string to the end of each new variable name.
  - **Use ID variable**: Specifies which variable's observations from the source dataset are used to name the variables in the output dataset.

  When setting the above values, if any resulting variable name would be too long, a warning will be given.
- **Transposed Variables**: Defines
  - **Names variable**: Sets the name of the names variable in the new dataset.
  - **Labels variable**: Sets the name of the labels variable in the new dataset.

# Code Blocks group

Contains blocks that enable you to add a new program to the workflow.

Although programming is not a prerequisite for creating a workflow, if you have the required programming skills you can carry out more advanced tasks in a workflow, using one or more of the available coding blocks.



# Python Language block

Enables you to use Python language programs in a workflow.

To use the **Python Language** block, you must have a Python interpreter installed and correctly configured on the local or remote host on which the Workflow Engine is run. As a minimum, you must include the *pandas* module in the Python installation.

The **Python Language** block can be used to access either existing Python language modules such as *scikit-learn* or your own modules for your workflows.

Reference names for the input dataset are displayed in the **Inputs** list. Output working dataset names are displayed in the **Outputs** list.

Workflow interaction with the Python language, including managing input and output datasets is configured using the **Configure Python Language** dialog box. To open the **Configure Python Language** dialog box, double-click the **Python Language** block

**Python Language editor**

This editor is used to enter the code for the Python language program. Program code must follow the same indentation rules as Python language programs created outside a workflow.

All code for the program must be present in the editor, you cannot import program code from an external file. Modules located in the Python site package folders can be imported into your program.

**Inputs**

Lists the input datasets and variables in the dataset. You can drag a name label to the Python Language editor rather than typing a label.

**Outputs**

Lists the pandas DataFrames created in the Python language program that are made available to other blocks in the workflow. You can drag a name label to the Python Language editor rather than typing a label.

## Input datasets

Input datasets displayed in the **Inputs** list represent the working datasets connected to the **Python Language** block.

To create a new input dataset, on the Workflow canvas, drag a connector from the **Output** port of the working dataset to the **Input** port of the **Python Language** block

To modify the dataset name click **Edit** ( ), and enter a new name for the dataset in the **Inputs** list. If the dataset is already referenced in your Python language program, the references are not automatically updated and must be manually modified for the **Python Language** block to run successfully.

## Output datasets

Output datasets displayed in the **Outputs** list represent the mapping between a dataset created or modified in the Python language program, and the working dataset made available to other blocks in the workflow.

To create a new output working dataset, click **Add** ( ). To use the new output dataset reference in the Python Language editor, drag the label to the editor.

To delete the output dataset name click **Delete** ( ). If the dataset is already referenced in your Python language program, the references are not automatically updated and the references to the dataset must be manually modified for the **Python Language** block to run successfully.

To modify the dataset name click **Edit** ( ), and enter a new name for the dataset in the **Outputs** list. If the dataset is already referenced in your Python language program, the references are not automatically updated and must be manually modified for the **Python Language** block to run successfully.

# R Language block

Enables you to use R language programs in a workflow.

To use the **R Language** block, you must have an R interpreter installed and correctly configured on the local or remote host on which the Workflow Engine is run.

The **R Language** block can be used to access existing R language modules or create your own modules for your workflows.

Workflow interaction with the R language, including managing input and output datasets is configured using the **Configure R Language** dialog box. To open the **Configure R Language** dialog box, double-click the **R Language** block

**R language editor**
: Used to enter the code for the R language program.

  All code for the program must be present in the editor, you cannot import program code from an external file using the `source()` command. Packages located in folders referenced in the R `.libPaths` variable can be imported into your program.

**Inputs**
: Lists the input datasets and variables in the dataset. You can drag a name label to the R Language editor rather than typing a label

**Outputs**
: Lists the data frames created in the R language program that are made available to other blocks in the workflow. You can drag a name label to the R Language editor rather than typing a label.

## Input datasets

Input datasets displayed in the **Inputs** list represent the working datasets connected to the **R Language** block.

To create a new input dataset, on the Workflow canvas, drag a connector from the **Output** port of the working dataset to the **Input** port of the **R Language** block

To modify the dataset name click **Edit** (  ), and enter a new name for the dataset in the **Inputs** list. If the dataset is already referenced in your R language program, the references are not automatically updated and must be manually modified for the **R Language** block to run successfully.

### Output datasets

Output datasets displayed in the **Outputs** list represent the mapping between a dataset created or modified in the R language program, and the working dataset made available to other blocks in the workflow.

To create a new output working dataset, click **Add** (  ). To use the new output dataset reference in the R Language editor, drag the label to the editor.

To delete the output dataset name click **Delete** (  ). If the dataset is already referenced in your R language program, the references are not automatically updated and the references to the dataset must be manually modified for the **R Language** block to run successfully.

To modify the dataset name click **Edit** (  ), and enter a new name for the dataset in the **Outputs** list. If the dataset is already referenced in your R language program, the references are not automatically updated and must be manually modified for the **R Language** block to run successfully.

## SAS Language block

Enables you to use SAS language programs in a workflow.

The **SAS Language** block provides a self-contained environment that supports all of the SAS language syntax available in the WPS engine.

Input and output datasets must be referred to in your SAS language program using macro syntax label names. For example, to use an input dataset named `Input_1`, you must refer to the dataset as `&Input_1`.

Macro reference names for the input dataset are displayed in the **Inputs** list. Output working dataset names are displayed in the **Outputs** list.

The SAS language program input and output datasets are configured using the **Configure SAS Language** dialog box. To open the **Configure SAS Language** dialog box, double-click the **SAS Language** block.

**SAS Language editor**
: Used to enter the code for the SAS language program. All code for the program must be present in the editor, you cannot import program code from an external file.

**Inputs**
: Lists the input datasets and variables in the dataset. You can drag a name label to the SAS Language editor rather than typing a label.

**Outputs**

Lists the datasets created in the SAS language program to be made available to other block in the workflow. You can drag a name label to the SAS Language editor rather than typing a label.

## Input datasets

Input datasets displayed in the **Inputs** list represent the working datasets connected to the **SAS Language** block.

To create a new input dataset, on the Workflow canvas, drag a connector from the **Output** port of the working dataset to the **Input** port of the **SAS Language** block

To modify the dataset name click **Edit** ( ), and enter a new name for the dataset in the **Inputs** list. If the dataset is already referenced in your SAS language program, the references are not automatically updated and must be manually modified for the **SAS Language** block to run successfully.

## Output datasets

Output datasets displayed in the **Outputs** list represent the mapping between a dataset created or modified in the SAS language program, and the working dataset made available to other blocks in the workflow.

To create a new output working dataset, click **Add** ( ). To use the new output dataset reference in the SAS Language editor, drag the label to the editor.

To delete the output dataset name click **Delete** ( ). If the dataset is already referenced in your SAS language program, the references are not automatically updated and the references to the dataset must be manually modified for the **SAS Language** block to run successfully.

To modify the dataset name click **Edit** ( ), and enter a new name for the dataset in the **Outputs** list. If the dataset is already referenced in your SAS language program, the references are not automatically updated and must be manually modified for the **SAS Language** block to run successfully.

# SQL Language block

Enables you to create a dataset using the SQL language `SELECT` statements in a workflow.

Input datasets must be referred to in your SQL language code using SAS language macro syntax label names. For example, to use an input dataset named `Input_1`, you must refer to the dataset as `&Input_1`.

Input dataset macro reference names are managed using the **Manage Inputs** dialog box.

**SQL Language editor**

The editor is used to enter SQL language `SELECT` statements.

**Inputs**

Lists the input datasets and variables in the datasets. You can drag a name label to the SQL Language editor rather than typing a label.

### Input datasets

Input datasets displayed in the **Inputs** list represent the working datasets connected to the **SQL Language** block.

To create a new input dataset, on the Workflow canvas, drag a connector from the **Output** port of the working dataset to the **Input** port of the **SQL Language** block

To modify the dataset name click **Edit** ( ), and enter a new name for the dataset in the **Inputs** list. If the dataset is already referenced in your SQL language statements, the references are not automatically updated and must be manually modified for the **SQL Language** block to run successfully.

# Model Training group

Contains blocks that enable you to discover predictive relationships in your data.

# Decision Forest block

Provides an interface to create a decision forest prediction model.

A decision forest is a prediction model that uses a collection of *decision trees* to predict the value of a dependent (target) variable based on other independent (input) variables in the input dataset.

A decision forest is created and edited using the decision forest **Preferences** dialog box To open the decision forest **Preferences** dialog box, right-click the **Decision Forest** block and click **Configure**.

The **Decision Forest** block generates a report containing the summary statistics for the model and the scoring code generated by the model. To view this report, right-click the **Decision Forest Model** block and click **View Result**. The generated scoring code can be copied into a SAS language program for testing or production use.

## Variable Selection panel

Enables you to specify the dependent (target) variable and category you want to predict based on other independent (input) variables in the input dataset.

**Dependent Variable**
Specifies the dependent (target) variable for which the category is being calculated by the decision tree.

**Dependent Variable Treatment**
Specify the type of the dependent (target) variable. The supported variable types are:

**Interval**
Specifies a continuous variable with an implicit category ordering.

**Nominal**
Specifies a discrete variable with no implicit ordering.

**Ordinal**
Specifies a discrete variable with an implicit category ordering.

**Independent Variables**
Displays a list of all variables in the dataset and the measure of how well the independent (input) variable can predict the value of the selected dependent (target) variable in the *Entropy Variance* column, in the range of 0–1.

If a variable in a dataset has a high entropy variance, it means that the variable is a good predictor of the dependent variable. However where the entropy variance is very high, or 1 the variable might not be a good predictor  as using this variable could lead to overfitting the model

## Variable Treatment panel

Enables you to specify the type of each independent (input) variable selected in the **Variable Selection** panel. The supported variable types are:

**Interval**
Specifies a continuous independent (input) variable with an implicit category ordering.

**Nominal**
Specifies a discrete independent (input) variable with no implicit ordering. When partitioning this variable into nodes in a tree, any category can be merged with any other category.

**Ordinal**
Specifies a discrete independent (input) variable with an implicit category ordering. When partitioning this variable into nodes in a tree, only adjacent categories can be merged together.

## Preferences panel

Specifies the options for the creation of the decision forest.

**Number of trees**
Specifies the number of decision trees in the forest.

**Minimum improvement**
Specifies the minimum improvement (decrease) in Gini Impurity required for the node to be split.

**Exclude Missings**
When selected, excludes missing values from the source dataset.

**Classifier combination**
Specifies the way the probabilities predicted by each tree in the forest are combined to obtain the overall probabilities scored by the decision forest. Choose either **Vote**, to use the proportion of trees that voted for each classification; or **Mean Probability**, to use the means of the probabilities calculated by each tree.

This option is unavailable if the dependent variable is an **Interval** variable (a regression forest is used).

**Criterion**
Specifies the variable from the subset of specified dependent variables that gives the largest overall decrease in *Gini Impurity* is used to split nodes in all trees.

**Bootstrap**
Specifies the dataset is resampled using random sampling with replacement to attempt to quantify uncertainty in the model.

**Inputs at split**
Defines the number of input variables to randomly select each time a node is split.

**Use default**
Species the default number of input variables is selected.

For $N$ selected independent variables, the default value is $\lfloor\sqrt{N}\rfloor$ for classification trees and $N/3$ for regression trees.

**Number of inputs at each split**
Specifies the number of input variables

**Split size**
Enables you to specify the minimum number of observations for the node to be split either as a proportion of the input dataset or an absolute number.

**Minimum split size**
Specifies the absolute minimum number of observations in a node.

**Minimum split size as ratio**
Specifies the minimum number of observations as a proportion of the input dataset.

**Frequency and weight variables**

**Frequency variable**
Specifies a variable in the input dataset containing the frequency associated with an observation.

**Weight variable**
Specifies a variable in the input dataset giving the prior weight associated with each observation.

# Decision Forest Results

Displays a summary of the Decision Forest model and its results.

## Model Information

Provides a summary of the model.

## Target Summary

Shows the target variable that the decision forest is required to predict.

## Input Summary

Summarises the input variables, their types, categories and the other properties.

## Out-of-Bag Classification Table

Shows the out-of-bag (OOB) error estimate from the decision forest. This is the proportion of observations in the training dataset that were incorrectly classified using OOB classification.

To calculate the predicted OOB classification for an observation, the observation is classified using each tree for which the observation was out-of-bag (that is, each tree for which the observation wasn't selected to train that tree). The most popular value is chosen as the predicted OOB classification for that observation. The OOB error estimate is the proportion of observations which are incorrectly classified using this algorithm.

# Decision Tree block

Provides an interface to create a decision tree for classification purposes.

Decision trees are created and edited using the Decision Tree editor. Required information to build a tree – such as dependent and independent variables and tree growth preferences – is set using the decision tree **Preferences** dialog box.

You can use the **Decision Tree Editor** view **Tree** tab to grow the tree manually based on the optimal binning measure of the selected independent variables, or automatically grow one level of the tree at a time or a full decision tree using the growth preference algorithm specified in the decision tree **Preferences** dialog box.

Growing a decision tree requires a connected source dataset, although without a connected dataset, trees may still be viewed and pruned (Unsplit). If a dataset is reconnected after a period of disconnection, the decision tree will be recalculated.

To open the **Decision Tree Editor** view, right-click the **Decision Tree** block and click **Edit Decision Tree**.

## Decision Tree Editor

Displays graphical, tabular and coded representations of the decision tree and information about specific nodes or tree levels as the tree is grown.

## Decision Tree preferences

Enables you to specify the independent and dependent variables and set options for creating the decision tree.

**Opening the Decision Tree preferences dialog box**

The decision tree **Preferences** dialog box is opened in three ways:

- When first creating a Decision Tree, the decision tree **Preferences** dialog box will open automatically.
- By clicking the Preferences (  ) button in the Decision Tree editor **Tree** tab.

- By right-clicking on the Decision Tree block in the workflow and clicking **Edit Decision Tree**.

The contents of the decision tree **Preferences** dialog box are described below.

### Variable Selection

Enables you to specify the dependent (target) variable and category you want to predict based on other independent (input) variables in the input dataset.

#### Dependent Variable

Specifies the dependent (target) variable for which the category is calculated by the decision tree. If a continuous variable is selected, Target Category selection is disabled and the Growth Algorithm (specified in the Growth Preferences tab) is limited to CART.

#### Target Category

Specifies the classification of the dependent variable the decision tree will predict using the selected independent (input) variables selected in the **Independent Variables** list.

#### Independent Variables

Displays a list of all variables in the dataset and the measure of how well the independent (input) variable can predict the value of the selected dependent (target) variable in the *Entropy Variance* column, in the range of 0–1.

If a variable in a dataset has a high entropy variance, it means that the variable is a good predictor of the dependent variable. However where the entropy variance is very high, or 1 the variable might not be a good predictor as using this variable could lead to overfitting the model.

### Variable Treatment

Enables you to specify the type of each independent (input) variable selected in the **Variable Selection** panel. The supported variable types are:

#### Interval

Specifies a continuous independent (input) variable with an implicit category ordering.

#### Nominal

Specifies a discrete independent (input) variable with no implicit ordering. When partitioning this variable into nodes in a tree, any category can be merged with any other category.

#### Ordinal

Specifies a discrete independent (input) variable with an implicit category ordering. When partitioning this variable into nodes in a tree, only adjacent categories can be merged together.

### Growth Preferences

Specifies the decision tree growth algorithm and associated growth preferences. The **Growth Algorithm** specifies the method used to construct the decision tree, and can be one of `C4.5`, `CART`, or `BRT`:

**C4.5**

Specifies the C4.5 algorithm should be used to build the decision tree. You can then specify options for the algorithm.

**Maximum depth**

Specifies the maximum depth of the decision tree.

**Minimum node size (%)**

Specifies the minimum number of observations in a decision tree node, as a proportion of the input dataset.

**Merge categories**

Specifies that discrete independent (input) variables are grouped together to optimize the dependent (target) variable value used for splitting.

**Prune**

Select to specify that nodes which do not significantly improve the predictive accuracy are removed from the decision tree to reduce tree complexity.

**Prune confidence level**

Specifies the confidence level for pruning, expressed as a percentage.

**Exclude missings**

Specifies that observations containing missing values are excluded when determining the best split at a node.

**CART**

Specifies the CART algorithm should be used to build the decision tree. You can then specify options for the algorithm.

**Criterion**

Specifies the criterion used to split nodes in the decision tree. The supported criteria are:

**Gini**

Specifies that *Gini Impurity* is used to measure the predictive power of variables.

**Ordered twoing**

Specifies that the *Ordered Twoing Index* is used to measure the predictive power of variables.

**Twoing**

Specifies that the *Twoing Index* is used to measure the predictive power of variables.

**Prune**

Select to specify that nodes which do not significantly improve the predictive accuracy are removed from the decision tree to reduce tree complexity.

**Pruning method**

Specifies the method to be used when pruning a decision tree. The supported methods are:

**Holdout**

Specifies that the input dataset is randomly divided into a test dataset containing one third of the input dataset and a training dataset containing the balance.

The output decision tree is created using the training dataset, then pruned using risk estimation calculations on the test dataset.

**Cross validation**

Specifies that the input dataset is divided as evenly as possible into ten randomly-selected groups, and the analysis repeated ten times. The analysis uses a different group as test dataset each time, with the remaining groups used as training data.

Risk estimates are calculated for each group and then averaged across all groups. The averaged risk values are used to prune the final tree built from the entire dataset.

**Maximum depth**

Specifies the maximum depth of the decision tree.

**Minimum node size (%)**

Specifies the minimum number of observations in a decision tree node, as a proportion of the input dataset.

**Minimum improvement**

Specifies the minimum improvement in impurity required to split decision tree node.

**Exclude missings**

Specifies that observations containing missing values are excluded when determining the best split at a node.

**BRT**

Specifies the algorithm options for binary response trees.

Specifies the BRT algorithm should be used to build a binary decision tree. You can then specify options for the algorithm.

**Criterion**

Specifies the criterion used to split nodes in the decision tree. The supported criteria are:

**Chi-squared**

Specifies that *Pearson's Chi-Squared* statistic is used to measure the predictive power of variables.

**Entropy variance**
Specifies that *Entropy Variance* is used to measure the predictive power of variables.

**Gini variance**
Specifies that *Gini Variance* is used to measure the predictive power of variables by measuring the strength of association between variables.

**Information value**
Specifies that *Information Value* is used to measure the predictive power of variables.

**Maximum depth**
Specifies the maximum depth of the decision tree.

**Select minimum node size by ratio**
When selected enables you to specify the minimum number of observations in a node as a proportion of the input dataset. When cleared, enables you to specify the absolute minimum number of observations in a node.

**Minimum node size (%)**
Specifies the minimum number of observations in a decision tree node, as a proportion of the input dataset.

**Minimum node size**
When selected enables you to specify the minimum split size of a decision tree node as a proportion of the input dataset. When cleared, enables you to specify the absolute minimum split size of a decision tree node .

**Select minimum split size by ratio**
When selected enables you to specify the minimum number of observations for the node to be split as a proportion of the input dataset. When cleared, enables you to specify the absolute minimum number of observations in a node.

**Minimum split size (%)**
Specifies the minimum number of observations a decision tree node must contain for the node to be split, as a proportion of the input dataset.

**Minimum split size**
Specifies the absolute minimum number of observations a decision tree node must contain for the node to be split.

**Allow same variable split**
Specifies that a variable can be used more than once to split a decision tree node.

**Open left**
For a continuous variable, when selected, specifies that the minimum value in the node containing the very lowest values is $-\infty$ (minus infinity). When clear, the minimum value is the lowest value for the variable in the input dataset.

**Open right**
For a continuous variable, when selected, specifies that the maximum value in the node containing the very largest values is ∞ (infinity). When clear, the maximum value is the largest value for the variable in the input dataset.

**Merge missing bin**
Specifies that missing values are considered a separate valid category when binning data.

**Monotonic Weight of Evidence**
Specifies that the Weight of Evidence value for ordered input variables is either monotonically increasing or monotonically decreasing.

**Exclude missings**
Specifies that observations containing missing values are excluded when determining the best split at a node.

**Initial number of bins**
Specifies the number of bins available when binning variables.

**Maximum number of optimal bins**
Specifies the maximum number of bins to use when optimally binning variables.

**Weight of Evidence adjustment**
Specifies the adjustment applied to weight of evidence calculations to avoid invalid results for pure inputs.

**Maximum change in predictive power**
Specifies the maximum change allowed in the predictive power when optimally merging bins.

**Minimum predictive power for split**
Specifies the minimum predictive power required for a decision tree node to be split.

## Tree tab

The Tree tab displays the decision tree and associated information.

**Main tree view**

The main part of the Tree tab displays a graphical representation of the decision tree as it grows. Each node is numbered, with numbering starting at zero at the root, with each subsequent level then numbered from left to right following on from the previous level. For discrete dependent variables, each node shows a frequency breakdown of the target categories. For continuous variables, each node shows the mean and LSD (least squares deviation) of the dependent variable for that node.

The growth of the tree is determined by options specified in the **Node properties** panel. Other Decision Tree settings are configured using the Preferences window, accessed by clicking the **Preferences** button (two cogs symbol).

The Node Information panel shows information about the selected node. If a continuous dependent variable is being used, two other panels are available: **Node Frequencies** and **Peer Comparison**. The **Node Frequencies** panel shows a frequency breakdown of the target categories for the selected node. The **Peer Comparison** panel, which is collapsed by default, displays information about a selected node compared to other nodes on the same tree level.

To prune a tree (remove child branches), right click on a parent branch and click **Unsplit**.

### Node Information panel

Displays information about the node currently-selected.

**Node**
Displays the label of the node. If the node is the root of the tree, the label *<root>* is displayed, otherwise the **Node label** specified in **Node Properties** is displayed.

**Size**
Displays the total number of observations in the current node.

**% of Population**
Displays the size of the node as a percentage of the input dataset.

### Node Frequencies panel (continuous dependent variables only)

Displays the frequency of the dependent variable categories in the selected node. The frequency can be displayed as either a histogram showing the frequency percentage for each category, or a table showing the frequency and associated observation count.

The histogram chart can be edited and saved. Click **Edit chart** to open a the Chart Editor from where the chart can be saved to clipboard. Frequency data for the selected node can be saved, click **Copy data to clipboard** to save the table of data.

### Peer comparison panel (continuous dependent variables only)

Displays the frequency of the dependent variable categories in the selected node and other nodes at the same tree level (peer nodes). The frequency for each category in all peer nodes can be displayed as either a histogram showing the frequency percentage for each category, or a table showing the frequency and associated observation count in each node.

The peer comparison chart can be edited and saved. Click **Edit chart** to open a the Chart Editor from where the chart can be saved to clipboard. Frequency data for the selected node and peer nodes can be saved, click **Copy data to clipboard** to save the table of data.

### Node Properties panel

Enables you to grow a tree and specify how new tree level nodes are split.

**Split**

Displays the independent variable from the **Split Variable** list used to split the parent node to create the tree level containing the selected node.

**Node Label**

Specifies the label displayed in the graphical decision tree and the **Node Information** panel for the node selected in the graphical tree. The default label is the value or values of the displayed split variable used to create the node. If you modify the label, click **Default** to return the label to the Workbench-generated value.

**Grow 1 Level**

Grows the next level of the tree using the **Growth Algorithm** specified in the **Growth Preferences** panel of the Decision tree preferences.

**Grow**

Grows a complete decision tree using the **Growth Algorithm** specified in the **Growth Preferences** panel of the Decision tree preferences.

**Optimal split**

Will grow one level of the tree using the Optimal Binning **Measure** specified in the **Binning** panel of the **WPS** section of the **Preferences** dialog box.

**Split Variable**

Displays the variable used to split the selected node and the categories used to determine the split points. You can change the created nodes using one of the available binning methods or by selecting a different **Split variable**. The available binning methods, depending on variable type, are optimal binning  , equal width binning  , equal height binning  , or Winsorized binning  .

**Map panel**

The map panel at the bottom right shows how the main tree view, represented by a grey rectangle, relates to the tree as a whole, shown with blue and red blocks. The main tree view can be moved by clicking and dragging the grey rectangle.

## Table tab

The Table tab gives a tabular representation of the decision tree, with every leaf node as a row.

**Sorting the table**

The table can be sorted by clicking on the variable you want to sort by.

**Exporting to Excel**

The **Export to excel** button will export the table's contents to an MS Excel `xlsx` file with a specified location and filename. The sort order of the table view has no influence on the sorted order of the generated Excel file.

**Dependent variable value**

Sets the value of the dependent variable. The dependent variable is specified in **Decision Tree preferences**.

### Scoring Code tab

The **Scoring Code** tab displays code for the decision tree in a specified language with specified variable names. SAS language code can be used in a `DATA` step in the **SAS Language** block.

**Choosing the language**

From the **Select Language** list box, select the required language as either **SAS** or **SQL**.

**Choosing score variable names**

Click **Options**, then in **Source Generation Options**, click the score variable you want to change and type the new name. When you are happy with all the score variable names, click **OK**.

## Hierarchical Clustering block

Provides an interface to create a hierarchical cluster model.

A hierarchical cluster model is created and edited using **Hierarchical Clustering** view. Required information to build the model is set using the hierarchical clustering **Preferences** dialog box. The hierarchical clustering **Preferences** dialog box is displayed the first time you open a new **Hierarchical Clustering** view.

To open the **Hierarchical Clustering** view, right-click the **Hierarchical Clustering** block and click **Edit Clusters**.

### Hierarchical clustering preferences

Enables you to specify the variables and set options for creating the cluster model.

#### Variable Selection

Enables you to specify the variables to be used in the clustering model.

#### Configuration

Enables you to specify settings for the hierarchical model created using variables selected in the **Variable Selection** panel.

**Standardize**
Standardises the variables to normal form, and uses these values when calculating clusters.

**Impute**
Replaces any missing values with a calculated variable in the input variables specified in the **Variable Selection** panel.

**Perform pre-clustering**
Specifies that input observations are allocated to a smaller number of clusters to reduce the problem complexity. Observations are assigned to the clusters using k-means clustering.

**Method**
Specifies how cluster distances and cluster centres are calculated.

**Average linkage**
Specifies the group average method to create data clusters.

**Centroid method**
Specifies the centroid method to create data clusters.

**Complete linkage**
Specifies the complete-linkage method to create data clusters.

**Density linkage**
Specifies the density-linkage method to create data clusters.

**Flexible-beta method**
Specifies the flexible-beta method to create data clusters, as described in Lance, G.N., and Williams, W.T., 1967. A general theory of classificatory sorting strategies, 1. Hierarchical systems in *Computer Journal*, Volume 9, pp 373–380.

**McQuitty similarity analysis**
Specifies that McQuitty-similarity analysis is used to create, as described in McQuitty, L.L., 1966. Similarity Analysis by Reciprocal Pairs for Discrete and Continuous Data, in *Educational and Psychological Measurement*, Volume 26, pp 825–831.

**Median Method**
Specifies the median method of centroids to create data clusters.

**Single linkage**
Specifies a single linkage or nearest neighbour calculation is used to create data clusters.

**Ward minimum variance model**
Specifies that the Ward minimum variance model is used to create data clusters, as described in Ward, H.J., 1963. Hierarchical Grouping to Optimize an Objective Function, in *Journal of the American Statistical Association*, Volume 58, pp. 236–244

**Suppress squaring distances**
Suppresses the squaring of distances in distances calculations. This option can be specified for the *Average linkage*, *Centroid*, *Median*, and *Ward minimum variance* models.

**Mode**

Specifies the minimum number of observations required for a cluster to be distinguished as a *modal cluster*. A modal cluster is a region of high density separated from another region of high density by a region of low density.

**Beta**

Specifies the beta value required to implement the flexible-beta method.

**Dimensionality**

Specifies the number of dimensions to use when calculating density estimates in the *density linkage* model.

**Frequency variable**

Specifies a variable selected in the **Value Selection** panel that defines the frequency of each observation.

**K Nearest Neighbours**

Specifies the number of nearest neighbours to each observation in cluster calculations when using the *Density linkage* method.

**Radius of Sphere for uniform-kernel**

Specifies the radius of the a sphere around each observation used to compute the nearest neighbours in cluster calculations when using the *Density linkage* method.

# Hierarchical Clustering view

Displays a graphical representation of the cluster model and information about specific clusters in the model.

The **Hierarchical Clustering** view displays information using the following tabs.

## Clustering

Displays a dendrogram graph, which is a tree displaying the calculated clusters of observations and the distances between clusters. The graphs show the Cubic Clustering Criteria (CCC), Pseudo F and Pseudo T-Square statistics for the generated clusters. On all graphs, the vertical dotted line shows the number of clusters currently selected.

The view contains two slider controls:

- **Dendrogram details** enables you to adjust the level of detail in the tree.
- **Dendogram Height Variable** enables you to choose the height variable used.
- **Number of clusters** enables you to select the number of clusters in the final model. The number selected determines the number of clusters listed in both the **Univariate View** and **Distribution Comparison**.

### Univariate View

Displays the summary statistics for the input dataset and the same summary statistics for each cluster in the model. These statistics can be based on either the data values in the dataset – the **Input Space** – or a standardised normal distribution of the variable values – the **Standardised Space**.

The view displays a frequency distribution for a specified variable in the overall or cluster lists. The frequency distribution of the specified variable can be displayed as a histogram, line chart or pie chart. In all cases, the charts display the frequency of values as the percentage of the total number of observations for the variable.

The chart can be edited and saved. Click **Edit chart**  to open a the Chart Editor from where the chart can be saved to clipboard.

### Distribution Comparison

Enables you to select a variable and view the frequency for a specified variable in one or more of the specified clusters. The displayed frequency can be based on either the data values in the dataset – the **Input Space** – or a standardised normal distribution of the variable values – the **Standardised Space**.

The frequency distribution of the specified variable and clusters can be displayed as a histogram, line chart or pie chart. In all cases, the charts display the frequency of values as the percentage of the total number of observations for the variable.

The chart can be edited and saved. Click **Edit chart**  to open a the Chart Editor from where the chart can be saved to clipboard.

## K-Means Clustering block

Enables you to cluster data using *k*-means clustering.

Clustering, or cluster analysis, is an exploratory data analysis technique that divides data into groups. The groups contain items that are more similar to each other than they are to items in other groups. *k*-means clustering is a method for creating such groups.

To open the **Configure K-Means Clustering** dialog box, right-click the **K-Means Clustering** block and click **Configure**.

The **K-Means Clustering** block generates a report displaying summary and graphical information about of the cluster model. To view this report, right-click the **K-Means Cluster Model** block and click **View Result**.

### Configure K-Means Clustering

Use the **Configure K-Means Clustering** dialog box to specify the details required by the block. Only numeric data can be clustered.

**Variables**

Specifies which variables are to be clustered. Only numeric data can be clustered, therefore only numeric variables are listed and can be selected.

**Standardize**

Specifies that input variables are standardised; that is, the scales of variables are adjusted so that they are in a similar range.

**Impute**

Specifies that the value of missing values is imputed, based on the mean value for the variable.

**Number of Clusters**

Specifies the number of clusters into which the data is to be partitioned.

**Max iterations**

Specifies the maximum number of iterations to be performed by the clustering algorithm before terminating.

**Convergence**

Specifies a convergence threshold at which the clustering algorithm terminates.

**Frequency Variable**

Specifies a variable that contains the frequency associated with another variable.

**Weight Variable**

Specifies a variable that contains the prior weight associated with each variable.

# K-Means Clustering Report view

Displays a summary and graphical representation of the cluster model and information about specific clusters in the model.

The **K-Means Clustering Report** view is accessed by double clicking on the K-Means Cluster Model that is output by the K-Means Clustering block and displays information using the following tabs.

## Clustering Results

Displays summary statistics for the dataset and each specified cluster. The **Summary** section shows the Cubic Clustering Criterion (CCC), Pseudo F and Pseudo T-Square for the input dataset. The **Clusters** section shows summary statistics for each cluster, and can be based on either the data values in the dataset – the **Input Space** –or a standardised normal distribution of the variable values – the **Standardised Space**.

### Univariate View

Displays the summary statistics for the input dataset and the same summary statistics for each cluster in the model. These statistics can be based on either the data values in the dataset – the **Input Space** – or a standardised normal distribution of the variable values – the **Standardised Space**.

The view displays a frequency distribution for a specified variable in the overall or cluster lists. The frequency distribution of the specified variable can be displayed as a histogram, line chart or pie chart. In all cases, the charts display the frequency of values as the percentage of the total number of observations for the variable.

The chart can be edited and saved. Click **Edit chart**  to open a the Chart Editor from where the chart can be saved to clipboard.

### Distribution Comparison

Enables you to select a variable and view the frequency for a specified variable in one or more of the specified clusters. The displayed frequency can be based on either the data values in the dataset – the **Input Space** – or a standardised normal distribution of the variable values – the **Standardised Space**.

The frequency distribution of the specified variable and clusters can be displayed as a histogram, line chart or pie chart. In all cases, the charts display the frequency of values as the percentage of the total number of observations for the variable.

The chart can be edited and saved. Click **Edit chart**  to open a the Chart Editor from where the chart can be saved to clipboard.

### Scoring Code

Displays the K-means model code. The code is available as SAS language code to be used in a **SAS Language** block.

## Linear Regression block

Enables you to create a linear regression.

Linear regression attempts to model the linear relationship between multiple independent (regressor) variables and a dependent (response) variable. When you have found the linear relationship, you can use this to estimate the value of other response variables.

The **Linear Regression** block generates a report, diagnostic information, and scoring code. The report contains details of the variance in the dataset and parameter estimates. The diagnostic charts can be used to identify outlier points in the dataset. The generated scoring code can be copied into a SAS language program for testing or production use.

The model details are specified in the Variables included in the working dataset, using the **Configure Linear Regression** dialog box. To open the **Configure Linear Regression** dialog box, double-click the **Linear Regression** block.

## Configure Linear Regression dialog

Enables you to specify the variables used to create the linear regression model. Only numeric variables can be specified. You can also specify parameters that define the relationship between the dependent and linear regressors in the model.

**Dependent Variable**

Specifies the dependent variable.

**Regressors**

Specifies the regressor (independent) variable. The box contains a list or variables from which you can select.

**Method**

Specifies the method used to build the linear regression. The method can be:

**Forward**

Specifies the use of forward model selection.

The model initially contains intercept, if specified, and the selected regressor (independent) variables added one at a time at each iteration of the model. The most significant variable (the effect variable with the smallest probability value and below the **Entry Significance Level**) is added at each iteration, and iterations continue until no specified effect variable is below the **Entry Significance Level**.

**Backward**

Specifies the use of backward model selection.

All selected regressor (independent) variables are included in the model, the variables are tested for significance, and the least significant dropped at each iteration of the model. The model is recalculated and the least-significant variable is dropped again.

A variable will not be dropped, however, if it is below the value specified for this method in **Exit Significance Level**. When all variables are below this level, the model stops.

**Stepwise**

Specifies that the model uses a combination of forward model and backward model selection.

The model initially contains an intercept, if specified, and one or more forward steps are taken to add effect variables to the model. Backward and forward steps are used to remove effect variables from and add effect variables to the model until no further improvement can be made to the model.

**Maxr**

At each iteration of the model selection process, this option chooses the variable that gives the largest increase in the R-square statistic.

**Minr**

At each iteration of the model selection process, this option chooses the variable that gives the smallest increase in the R-square statistic.

**Rsquare**

Specifies that the linear regression should be defined using the coefficient of determination (adjusted $R^2$).

**Adjrsq**

Specifies that the linear regression should be defined using the adjusted coefficient of determination ($R^2$).

**CP**

Specifies that the linear regression should be defined using Mallows' $C_p$.

**Intercept**

Specifies that an intercept should be used.

**Frequency Variable**

Specifies a variable that defines the relative frequency of each observation.

**Weight Variable**

Specifies a variable that defines the prior weight associated with each observation.

**Start**

Specifies the number of regressor (independent) variables to be included in the initial model for forward and stepwise model selection.

**Stop**

Specifies the maximum number of regressor (independent) variables the model can contain.

**Entry Significance Level**

Specifies the value below which a variable is included in the `Forward` or `Stepwise` methods.

**Exit Significance Level**

Specifies the value above which a variable is removed in the `Backward` or `Stepwise` methods.

**Single Value**

Specifies the tolerance value used to test for linear dependency among the effect (independent) variables.

## Linear Regression Report view

Displays a summary and graphical information to help evaluate the Linear Regression model, along with SAS language scoring code.

The **Linear Regression Report** view is accessed by double clicking on the Linear Regression Model that is output by the Linear Regression block. The report view displays information across two tabs and code in another.

### Linear Regression Report tab

Displays a summary of the linear regression applied to the input dataset.

The **Model Information** section shows the Selection Method and Dependent Variable chosen in the Linear Regression block configuration, as well as the number of observations read and used by the model.

The **Summary Statistics** section gives statistics summarising the model used, including Root MSE (Means Squared Error), Dependent mean, R-square, Adjusted R-square (adjusted for the number of predictors in the model), and Coeff var (the coefficient of variation).

### Diagnostics Panel tab

Displays plots of statistics relating to the regression model. Plots displayed are: predicted against residual, predicted against R-Student, leverage against R-Student, quantile-quantile, prediction against actual value, observation against Cook's Distance, and a residual histogram. Click on a plot to view it enlarged in the right hand pane.

### Scoring code tab

Displays the linear regression model code. The code is available as SAS language code to be used in a **SAS Language** block.

# Logistic Regression block

Fits a logistic model between a set of independent variables and a binary dependent variable.

The **Logistic Regression** block generates a report and deployment code. The report contains details of the Model fit statistics, Maximum likelihood estimate analysis, odds ratio estimates, and predicted probability and observed responses. The generated deployment code is a `DATA` step that can be copied into a SAS language program for testing or production use.

The model details are specified in the Variables included in the working dataset are specified using the **Configure Logistic Regression** dialog box. To open the **Configure Logistic Regression** dialog box, double-click the **Logistic Regression** block.

**Dependent variable**

Specifies the binary dependent variable for the calculation.

**Event**

Specifies the target category for which the probability of occurrence is to be calculated.

**Effect Variables**

Displays a list of effect (independent) variables in the input dataset which you can select and specify which variables are used for classification.

**Method**

Specifies the model effect variable selection method.

**None**

Specifies that the fitted model includes every selected effect (independent) variable.

**Forward**

Specifies the use of forward model selection.

The model initially contains intercept, if specified, and the selected effect (independent) variables added one at a time at each iteration of the model. The most significant variable (the effect variable with the smallest probability value and below the **Entry Significance Level**) is added at each iteration, and iterations continue until no specified effect variable is below the **Entry  Significance Level**.

**Backward**

Specifies the use of backward model selection.

All selected effect (independent) variables are included in the model, the variables are tested for significance, and the least significant dropped at each iteration of the model. The model is recalculated and the least-significant variable is dropped again.

A variable will not be dropped, however, if it is below the value specified for this method in **Stay Significance Level**. When all  variables are below this level, the model stops.

**Stepwise**

Specifies that the model uses a combination of forward model and backward model selection.

The model initially contains an intercept, if specified, and one or more forward steps are taken to add effect variables to the model. Backward and forward steps are used to remove effect variables from and add effect variables to the model until no further improvement can be made to the model.

**Score**

Specifies that the model returned has the best likelihood score statistic.

The model is created using a branch-and-bound method. Initially a model is created using the specified effect (independent) variable with the best individual likelihood score statistic. A second model is created using the two specified effect variables with the best combined likelihood score statistic. The model with the best likelihood score statistic is considered the best current solution and becomes the bound for the next iteration.

At each subsequent iteration, an effect variable is added to the model where the combination of all effect variables has the best likelihood score statistic. The model is compared to the best current solution and if the new model likelihood score statistic is better than the current solution, the new model becomes the bound for the next iteration

When the best likelihood score statistic cannot be improved, the model is returned.

**Link Function**

Specifies how effect (independent) variables are linked to the dependent variable.

**CLOGLOG**

Specifies the *cloglog* (complementary log-log) function is used:

$$f(p) = \log(-\log(1-p))$$

Where p is the probability of the **Event** occurring.

**LOGIT**

Specifies the *logit* function (the inverse of the logistic function) is used:

$$f(p) = \log\left(\frac{p}{1-p}\right)$$

Where p is the probability of the **Event** occurring.

**PROBIT**

Specifies the *probit* function is used.

$$f(p) = \Phi^{-1}(p)$$

Where $\Phi^{-1}$ is the inverse cumulate distribution function of the standard normal distribution where the mean = 0 and variance = 1.

**Frequency Variable**

Specifies a variable that defines the frequency of each observation.

**Weight Variable**

Specifies a variable that defines the prior weight associated with each observation.

**Entry Significance Level**

Specifies the value below which a variable is included in the `Forward` or `Stepwise` methods.

**Stay Significance Level**

Specifies the value above which a variable is removed in the `Backward` or `Stepwise` methods.

**Singular Tolerance**

Specifies the tolerance value used to test for linear dependency among the effect (independent) variables.

# Logistic Model Report view

Displays summary and graphical information to help evaluate the Logistic Regression model.

The **Logistic Model Report** view is accessed by double clicking on the Linear Regression Model that is output by the Linear Regression block. The report view displays information in one tab and code in another.

## Logistic Regression Results tab

Displays a summary of the logistic regression applied to the input dataset.

The **Model Information** section confirms the dataset and response variable used along with the number of response levels. Also shown are the model and optimisation technique used and finally the number of observations read and used.

The **Model Fit Statistics** section shows the output of three tests of model fit for relative comparisons: AIC (Akaike Information Criterion), SC (Schwarz Criterion) and -2 Log L.

The **Testing Global Null Hypothesis: BETA=0** section shows the output of three tests: Likelihood ratio, score and Wald.

The **Analysis of Maximum Likelihood Estimates** section shows an Analysis of Maximum Likelihood Estimates for each observation in the dataset.

The **Odds Ratio Estimates** section shows Odds Ratio Estimates for each observation in the dataset.

The **Association of Predicted Probabilities and Observed Responses** section shows a comparison of observations in the dataset and the predicted values from the logistic regression, showing information on the concordant, discordant and tied pairs.

## Scoring code tab

Displays the linear regression model code. The code is available as SAS language code to be used in a **SAS Language** block.

# MLP block

Enables you to build a multilayer perceptron (MLP) neural network from an input dataset and use the trained network to analyse other datasets.

Multilayer perceptron neural networks are a class of non-linear machine learning algorithms that can be used for classification and regression.

Networks are usually conceptualised as layers of non-linear functions analogous to layers of neurons in biological neural networks connected by layers of weights analogous to synapses.

Training algorithms are used to find the weights that enable an MLP to approximate relationships between the effect variables and a response variable, making it possible to create a network that can predict unknown responses from known effects.

For more information about MLP neural networks, see *MLP procedure* in the *WPS Machine Learning* section in *WPS Reference for Language Elements*.

## Multilayer perceptron preferences

Enables you to specify the preferences used to create a trained network.

### Dataset Roles

Defines how the input datasets are used in the trained network.

**Train**
: Specifies the dataset used to train the network.

**Validate**
: Specifies a validation dataset. If specified, the result of training is the network that achieved the minimum error against this dataset.

**Test**
: Specifies a dataset used to test the network at the end of training.

### Variable Selection

Enables you to specify the dependent (target) variable and independent (input) variables in the input dataset to use the trained network.

**Dependent variable**
: Specifies the dependent (target) variable for the trained network.

**Independent variables**
: Displays a list of effect (independent) variables in the input dataset which you can select and specify which variables are used for classification.

## Optimizer

Specifies the optimisation algorithm to be used to train the network.

The following optimizers run using minibatches from the training dataset. A minibatch is a subset of observations, where all subsets are used in one iteration (epoch) when training the network:

- *ADADELTA* (page 108)
- *ADAM* (page 109)
- *NADAM* (page 109)
- *RMSPROP* (page 110)
- *SGD* (page 111)
- *SMORMS3* (page 112)

The following optimizers run using the whole training dataset in each iteration (epoch) when training the network:

- *RPROP* (page 111)

**ADADELTA**

Specifies an adaptive step size minibatch learning algorithm that reduces sensitivity to the choice of learning rate.

**Epsilon**

Specifies a value for the epsilon parameter.

**Smoothing factor**

Specifies a value for the smoothing factor of the network.

**Minibatch type**

Specifies how observations are selected for minibatches.

**DYNAMIC**

Specifies that observations are selected randomly without replacement.

**STATIC**

Specifies that observations are selected in the order they appear in the training dataset.

**BOOTSTRAP**

Specifies that observations are selected randomly with replacement.

**Minibatch size**

Specifies the number of observations in each minibatch.

**Learning rates**

Specifies a learning rate schedule as a series of minibatch number-learning rate pairs. To define learning rates, enter a **minibatch** number and the required **Learning rate** for that minibatch and click **Add**. Only one learning rate can be specified per minibatch number. If you specify a single learning rate, that learning rate is used for all iterations.

To change a learning rate, select the  row in the **Learning rates** box, change the rate and click **Update**.

To remove a specified minibatch number-learning rate pair, select the required row in the **Learning rates** box and click **Delete**.

**ADAM**

An algorithm that attempts to maximize speed of convergence by using estimates of lower-order moments.

**Epsilon**

Specifies a value for the epsilon parameter.

**Beta1**

Specifies a value for the beta1 parameter.

**Beta2**

Specifies a value for the beta2 parameter.

**Minibatch type**

Specifies how observations are selected for minibatches.

**DYNAMIC**

Specifies that observations are selected randomly without replacement.

**STATIC**

Specifies that observations are selected in the order they appear in the training dataset.

**BOOTSTRAP**

Specifies that observations are selected randomly with replacement.

**Minibatch size**

Specifies the number of observations in each minibatch.

**Learning rates**

Specifies a learning rate schedule as a series of minibatch number-learning rate pairs. To define learning rates, enter a **minibatch** number and the required **Learning rate** for that minibatch and click **Add**. Only one learning rate can be specified per minibatch number. If you specify a single learning rate, that learning rate is used for all iterations.

To change a learning rate, select the  row in the **Learning rates** box, change the rate and click **Update**.

To remove a specified minibatch number-learning rate pair, select the required row in the **Learning rates** box and click **Delete**.

**NADAM**

A version of the `ADAM` optimiser method that incorporates *Nesterov* momentum.

**Epsilon**

Specifies a value for the epsilon parameter.

**Beta1**

Specifies a value for the beta1 parameter.

**Beta2**

Specifies a value for the beta2 parameter.

**Minibatch type**

Specifies how observations are selected for minibatches.

**DYNAMIC**

Specifies that observations are selected randomly without replacement.

**STATIC**

Specifies that observations are selected in the order they appear in the training dataset.

**BOOTSTRAP**

Specifies that observations are selected randomly with replacement.

**Minibatch size**

Specifies the number of observations in each minibatch.

**Learning rates**

Specifies a learning rate schedule as a series of minibatch number-learning rate pairs. To define learning rates, enter a **minibatch** number and the required **Learning rate** for that minibatch and click **Add**. Only one learning rate can be specified per minibatch number. If you specify a single learning rate, that learning rate is used for all iterations.

To change a learning rate, select the  row in the **Learning rates** box, change the rate and click **Update**.

To remove a specified minibatch number-learning rate pair, select the required row in the **Learning rates** box and click **Delete**.

**RMSPROP**

An adaptive step size minibatch learning algorithm.

**Epsilon**

Specifies a value for the epsilon parameter.

**Smoothing factor**

Specifies a value for the smoothing factor.

**Minibatch type**

Specifies how observations are selected for minibatches.

**DYNAMIC**

Specifies that observations are selected randomly without replacement.

**STATIC**
Specifies that observations are selected in the order they appear in the training dataset.

**BOOTSTRAP**
Specifies that observations are selected randomly with replacement.

**Minibatch size**
Specifies the number of observations in each minibatch.

**Learning rates**
Specifies a learning rate schedule as a series of minibatch number-learning rate pairs. To define learning rates, enter a **minibatch** number and the required **Learning rate** for that minibatch and click **Add**. Only one learning rate can be specified per minibatch number. If you specify a single learning rate, that learning rate is used for all iterations.

To change a learning rate, select the  row in the **Learning rates** box, change the rate and click **Update**.

To remove a specified minibatch number-learning rate pair, select the required row in the **Learning rates** box and click **Delete**.

**RPROP**
An adaptive step size full batch learning algorithm.

**Initial delta**
Specifies the initial step size.

**Min delta**
Specifies the minimum step size.

**Max delta**
Specifies the maximum step size.

**Step size increment (ETA+)**
Specifies the factor by which the step size is increased.

**Step size decrement (ETA-)**
Specifies the factor by which the step size is reduced.

**SGD**
A fixed step size minibatch learning algorithm that supports *Classical* and *Nesterov* momentum.

**Momentum**
Specifies the type of momentum, either `NESTEROV` or `CLASSICAL`.

**Value**
Specifies the value of the selected momentum.

**Minibatch type**
Specifies how observations are selected for minibatches.

**DYNAMIC**
Specifies that observations are selected randomly without replacement.

**STATIC**
Specifies that observations are selected in the order they appear in the training dataset.

**BOOTSTRAP**
Specifies that observations are selected randomly with replacement.

**Minibatch size**
Specifies the number of observations in each minibatch.

**Learning rates**
Specifies a learning rate schedule as a series of minibatch number-learning rate pairs. To define learning rates, enter a **minibatch** number and the required **Learning rate** for that minibatch and click **Add**. Only one learning rate can be specified per minibatch number. If you specify a single learning rate, that learning rate is used for all iterations.

To change a learning rate, select the  row in the **Learning rates** box, change the rate and click **Update**.

To remove a specified minibatch number-learning rate pair, select the required row in the **Learning rates** box and click **Delete**.

**SMORMS3**
An adaptive step size minibatch learning algorithm that automatically adjusts the amount of smoothing to improve the trade-off between rate of convergence and stability.

**Epsilon**
Specifies a value for the epsilon parameter.

**Minibatch type**
Specifies how observations are selected for minibatches.

**DYNAMIC**
Specifies that observations are selected randomly without replacement.

**STATIC**
Specifies that observations are selected in the order they appear in the training dataset.

**BOOTSTRAP**
Specifies that observations are selected randomly with replacement.

**Minibatch size**
Specifies the number of observations in each minibatch.

**Learning rates**

Specifies a learning rate schedule as a series of minibatch number-learning rate pairs. To define learning rates, enter a **minibatch** number and the required **Learning rate** for that minibatch and click **Add**. Only one learning rate can be specified per minibatch number. If you specify a single learning rate, that learning rate is used for all iterations.

To change a learning rate, select the  row in the **Learning rates** box, change the rate and click **Update**.

To remove a specified minibatch number-learning rate pair, select the required row in the **Learning rates** box and click **Delete**.

## Regularisers

Multiple regularisers can be specified, so that more than one type of regularisation can be applied simultaneously. To add a regulariser, click **Add Regulariser** and fill in the required details for the regulariser type.

**Regulariser**

Specifies the type of regularisation to be used during training.

**LNNORM**

Specifies that a norm-based regulariser be applied to all non-bias weights.

**DROPOUT**

Specifies that dropout regularisation should be used on all hidden layers and that neurons will be dropped out during training with the specified probability.

**Probability**

Specifies the probability of neurons being dropped out during training using the `DROPOUT` regulariser.

**Strength**

Specifies the strength of the `LNNORM` regulariser.

**Power**

Specifies the power of the `LNNORM` regulariser.

## Stopping Criteria

Defines when model training stops.

**Max time (s)**

Specifies a target maximum training time in seconds.

**Number of epochs**

Specifies that training should terminate if the likelihood cannot be computed for the specified number of successive epochs.

**Max unimproved validation assessment**

Specifies the maximum number of validation assessments to that can be run without termination where the validation error does not improve.

**Validation interval**

Specifies the interval between validation set assessments. If you specify the `RPROP` optimizer, the interval is the number of epochs. For all other optimizers, the interval is the number of minibatches.

## Multilayer Perceptron view

Enables you to create a Multilayer Perceptron (MLP), export the resulting trained network code and view the tabular output of the trained network.

The **Multilayer Perceptron** view is used to create a trained network based on the settings specified in the MLP **Preferences** dialog box. The **Multilayer Perceptron** tab enables you to create a trained network and view the progress as the network is trained. The **Scoring Code** tab enables you to export the trained network SAS language code. The **Report** tab displays tabular output information from the **MLP** block.

## Multilayer Perceptron Tab

The **Multilayer Perceptron** tab enables you define the layers of the trained network. The **Layers** panel contains the detail of the trained network, and the **Error** panel displays the current status of the network as it is created.

To create a trained network, click **Train**.

**Configure Multilayer Perceptron**

**Layers**

Displays the layers in the trained network. Every network requires an Input, an Output and at least one Hidden layer. To add a new hidden layer to the trained network, click **Add Layer** and complete the information for the new row in the list. To remove a Hidden layer, click **Delete Layer**.

You can specify the **Number of Neurons** for each hidden later and the **Activation Function** to specify how the activity $y$ of a neuron is calculated from its post-synaptic potential $x$. The available Activation functions are:

| | |
|---|---|
| `ARCTAN` | $y = \arctan(x)$ |
| `ELLIOTT` | $y = \frac{x}{1+\lvert x+1\rvert}$ |

| | |
|---|---|
| LEAKYRECTIFIEDLINEAR | $y = x \ \ if \ x > 0$<br>$y = 0.01x \ otherwise$ |
| LINEAR | $y = x$ |
| LOGISTIC | $y = \frac{1}{1+\exp(-x)}$ |
| RECTIFIEDLINEAR | $y = x \ \ if \ x > 0$<br>$y = 0 \ otherwise$ |
| SOFTMAX | $y_i = \frac{\exp(x_i)}{\sum_{n=1}^{N}\exp(x_n)}$<br>where $N$ is the number of network outputs. |
| SOFTPLUS | $y = \log\lvert 1+e^x \rvert$ |
| SOFTSIGN | $y = \frac{x}{1+\lvert x \rvert}$ |
| TANH | $y = \tanh(x)$ |

**Error**

Displays the performance of the **MLP** block as it runs and automatically updates the history of any training error and validation error at each iteration.

## Scoring Code tab

The **Scoring Code** tab displays the trained network code. The code is available as SAS language code to be used in a DATA step in the **SAS Language** block.

## Report tab

The **Report** tab displays the tabular output of the trained network.

**Configuration**

The configuration table records the specified configuration used during this execution of the **MLP** block.

**Input Encoding**
Shows the mapping between the independent variables and the input neuron in the MLP. Numeric variables are mapped to a single input node. Categorical variables are mapped to multiple neurons, and the range is displayed in this table.

**Input Lengths**
Shows the weights of each input neuron.

**Input Mapping**
Shows the mapping between independent variables and input neuron for each neuron in the input layer of the MLP. Numerical variables have a single row in the table showing the neuron to which the variable is mapped. Categorical variables have multiple rows in the table showing which neuron each category is mapped to.

**Input Scaling**
Shows the original range in values of numerical values before being scaled to a consistent range, typically -1 to 1.

**Network Architecture**
The network architecture table summarises the MLP generated during the execution of the **MLP** block.

**Results**
The results table displays the results of training.

**Stopping reasons**
The stopping reason table lists all the reasons why training stopped.

**Training History**
The training history table shows how training, validation and regularisation errors have changed during training.

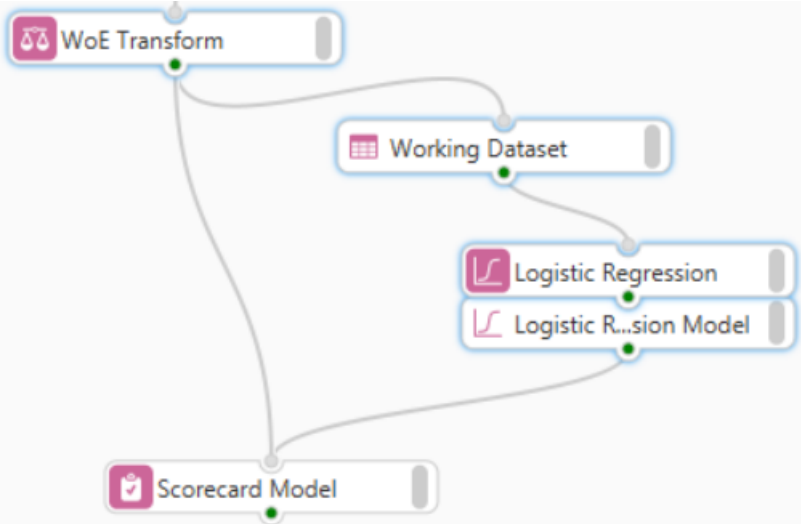
# Scorecard Model block

Enables the generation of a standard scorecard (credit scorecard) and its deployment code that can be copied into a SAS language program for testing or production use.

The **Scorecard Model** block creates a scorecard model consisting of a set of attributes each with an assigned weighted score (either positive or negative). The sum of those scores equals the final credit score representing the lending risk.

The **Scorecard Model** block requires two inputs:

- The transformation model from a **WoE Transform** block.
- A logistic regression model created from the working dataset of the same **WoE Transform** block.




The input models and settings specified in the **Scorecard Editor** view are used to create the deployable scorecard model.

## Point Allocation

**Scaling Parameters**

**Base Points**
Specifies the number of points to represent the **Base Odds**.

**Base Odds**
Specifies the number of *Bad* category entries for each *Good* category entry.

**Points to Double the Odds**
Specifies the number of points decrease in score at which the probability of default is doubled.

**Higher Score Category**
Specifies which category of borrower the higher scores are allocated to.

- `Good` indicates borrowers that have a lower credit risk
- `Bad` indicates borrowers that have a higher credit risk.

**Scorecard**
Displays the variables and associated scores making up the credit scorecard.

## Scoring Code

Displays the scorecard model code. The code is available as either SQL or as SAS language code to be used in a **SAS Language** block.

The final generated code can be copied using **Copy generated text to clipboard** .

# WoE Transform block

Enables you to measure the influence of an independent variable on a specified dependent variable.

You can use the **WoE Transform** block to measure risk, for example to test hypothesis such as the risk of loan default based on area of residence if your dataset is grouped by geographic areas.

A WoE Transform is edited through the **WoE Transform Editor** view. To open the editor, double-click the **WoE Transform** block.

The **WoE Transform Editor** view is used to select dependent variable and its target category, and the most influential independent variables and their treatment.

## Variable Selection view

**Dependent Variable**
Specifies the target variable for the calculation.

**Target Category**
Specifies the target category for which the probably of occurrence is to be calculated.

**Independent Variables**
Enables you to select the most influential independent variables for the WoE transform.

**Variable**
Displays the name of all independent variables in the input dataset.

**Entropy Variance**
Displays the *Entropy Variance* value for each variable in relation to the specified **Dependent Variable**. For more information, see *Predictive power criteria* (page 24)

**Treatment**
Enables you to specify the type of each variable.

**Excluded**
Specifies the variable is excluded from Weight of Evidence calculations.

**Interval**
Specifies a continuous independent (input) variable with an implicit category ordering.

**Nominal**
Specifies a discrete independent (input) variable with no implicit ordering.

**Ordinal**
Specifies a discrete independent (input) variable with an implicit category ordering.

**Monotonic WoE**
Specifies that the Weight of Evidence value for the variable is either monotonically increasing or monotonically decreasing.

**Frequency Chart**

shows the effect the selected independent variable has on each potential target category for the specified dependent variable.

## Optimisation view

Displays the output from transforming the specified independent variables using optimal binning.

**Independent variable**

Specifies the variables to use in the weight of evidence transformation. The list contains the variables selected in the **Variable Selector** view.

**Binning Definition**

Displays the effects of binning the specified independent variable. Click **Calculate optimal binning**  to optimally bin the variable. The table then displays the classes used for each bin and the Workbench-generated label for the bin, which can be modified if required.

**W0E Transform**

Displays a table with information about each optimal bin created including the number of observations in the bin; the number of observations in the bin matching the **Target Category** specified for the dependent variable; the WoE for each bin; the information value for each bin.

The panel contains graphs showing the proportion of specified target category to the non-target category for all bins, the Weight of Evidence for each bin, and the total number of observations in each bin.

The charts can be edited and saved. Click **Edit chart**  at the top of the required chart to open the Chart Editor from where the chart can be saved to clipboard. The frequency data used to create each node can be saved, click **Copy data to clipboard**  at the top of the required chart to save the table of data.

## Transformation Code

Displays the transformation code for the WoE calculations. The code is available as either SQL, for use in the **SQL Language** block, or as SAS language code to be used in a `DATA` step in the **SAS Language** block.

Click **Options** to open the **Source Generation Options** dialog box to modify the names of the transformation variables in the generated code.

The final generated code can be copied using **Copy generated text to clipboard**  .

# Scoring group

Contains blocks that enable you build a predictive model.



## Analyse Models block

Enables you to test the same scored model against multiple datasets to ensure the model has not been over-fitted to the training data.

To test the same model with multiple datasets, create one **Analyse Models** block for each dataset, and connect the same model output to each block.

The **Analyse Models** block generates a report showing the Gains chart, K-S Chart, ROC chart and Lift chart for the model. To display the report, select the required block in the **Workflow Editor** view and in the **Configure** dialog box, click **Open Report**.

## Score block

Enables you to test multiple models against the same dataset, or to test the validity of a model on holdout (test) samples.

To test multiple models, create one **Score** block for each model you want to test, and connect the same dataset to each **Score** block. The output from the multiple blocks will enable you to identify the preferred model to deploy.

# Export group

Contains blocks that enable you to export data from a workflow.

Data exported can be saved in either the current workspace or to the file system accessible through the active workflow link.

# Chart Builder block

Enables you to export a working dataset to a variety of different charts.

The **Chart Builder** block requires a single dataset input and can generate one or many charts from variables contained within that dataset.

Charts are created and edited using the Chart Builder view, which is opened by double clicking on the **Chart Builder** block. The Chart Builder view is divided into five areas:

**Note:**

Changes to a chart must be saved for them to appear in the preview area or Chart Viewer output.

**Chart Setup**

Allows you to configure labels and size for the chart.

**Title**

A title, displayed above the chart.

**Footnote**

A footnote, displayed below the chart.

**Width**

The width of the chart, in pixels.

**Height**

Specifies the height of the chart, in pixels.

**Plots**

Allows you to add, delete and configure one or many plots to appear on the chart. If there are multiple plots, the order of the plots can be changed, which determines their display order on the chart. Each plot type has its own set of definable parameters.

| Plot Type | Parameters |
|---|---|
| **Band**: Plots a series plot (line graph) together with a shaded band. | **X**: the main series to be plotted.<br>**Upper**: the upper limit of the band.<br>**Lower**: the lower limit of the band. |

| Plot Type | Parameters |
|---|---|
| **Bubble**: Plots isolated points from two variables, one assigned to the x axis and one to the y axis. Each point is represented by a bubble. | **X**: the variable to be plotted on the x axis.<br>**Y**: the variable to be plotted on the y axis.<br>**Size**: the variable used to determine the size of each bubble. |
| **Density** Plots a density plot for a specified variable, with the x axis showing variable values, and the y axis showing probability density. | **Variable**: the variable to be plotted. |
| **Ellipse**: When selected in addition to a scatter plot, draws an ellipse around values that fall within the 95th percentile of the total observations. | **X**: the variable to be plotted on the x axis.<br>**Y**: the variable to be plotted on the y axis. |
| **High Low**: Plots three series plots (line graphs) on one axes: a main series and an upper and lower series. | **X**: the main series to be plotted.<br>**Upper**: the upper series.<br>**Lower**: the lower series. |
| **Histogram**: Plots a histogram for a specified variable, with the x axis showing automatically binned variable values, and the y axis showing percentages of the dataset's observations. | **Variable**: the variable to be plotted. |
| **Horizontal Bar**: Plots a horizontal bar graph for a specified variable, with the y axis showing variable values and the x axis showing frequency of occurrence for those values. | **Response variable**: the variable to be plotted. |
| **Horizontal Bar (parameterized)**: Plots a horizontal bar graph for a specified variable, categorised by another variable. The x axis shows variable values and the y axis shows the categories. | **Category**: the variable to be used for categorisation.<br>**Response**: the variable to be plotted. |
| **Horizontal Box**: Plots a single variable on the x axis, drawing a box centered on the variable's median value, with sides set at the 25th and 75th percentile. Also shown are whiskers on the left and right sides of the box. The lower (left side) whisker is the lowest value still within 1.5 times the interquartile range of the lower quartile. The upper (right side) whisker is the highest value still within 1.5 times the interquartile range of the upper quartile. | **Response variable**: the variable to be plotted. |

| Plot Type | Parameters |
|---|---|
| **Horizontal Line**: Plots a line graph for a specified variable, with the y axis showing variable values, and the x axis showing frequency of occurrence. | **Response variable**: the variable to be plotted. |
| **LOESS**: Plots isolated points from two variables, one assigned to the x axis and one to the y axis. Each point is represented by a diamond. Also performs a LOESS (Locally Weighted Scatterplot Smoothing) fit to the data and plots this as a dotted line on the same axes. | **X**: the variable to be plotted on the x axis.<br>**Y**: the variable to be plotted on the y axis. |
| **Needle**Plots isolated points from two variables, one assigned to the x axis and one to the y axis. Each point is joined to the x axis with a line. | **X**: the variable to be plotted on the x axis.<br>**Y**: the variable to be plotted on the y axis. |
| **Penalized B-Spline**: Plots isolated points from two variables, one assigned to the x axis and one to the y axis. Each point is represented by a diamond. Also performs a Penalized B-Spline fit on the dataset and plots this as a dotted red line on the same axes. | **X**: the variable to be plotted on the x axis.<br>**Y**: the variable to be plotted on the y axis. |
| **Regression**: Plots isolated points from two variables, one assigned to the x axis and one to the y axis. Each point is represented by a diamond. Also performs a linear regression on the dataset and plots the resultant model as a dotted red line on the same axes . | **X**: the variable to be plotted on the x axis.<br>**Y**: the variable to be plotted on the y axis. |
| **Scatter**: Plots isolated points from twovariables, one assigned to the x axis and oneto the y axis. Each point is represented by adiamond. | **X**: the variable to be plotted on the x axis.<br>**Y**: the variable to be plotted on the y axis. |
| **Series**: Plots points from two variables, one assigned to the x axis and one to the y axis. All points are joined directly with lines. | **X**: the variable to be plotted on the x axis.<br>**Y**: the variable to be plotted on the y axis. |
| **Step**: Plots points from two variables, one assigned to the x axis and one to the y axis. All points are joined with horizontal and vertical lines, creating steps. | **X**: the variable to be plotted on the x axis.<br>**Y**: the variable to be plotted on the y axis. |

| Plot Type | Parameters |
|---|---|
| **Vector**: Plots points from two variables, one assigned to the x axis and one to the y axis. Each point is treated as the end-point of a vector, plotted with an arrow; with the start point as the origin. | **X**: the variable to be plotted on the x axis.<br>**Y**: the variable to be plotted on the y axis. |
| **Vertical Bar**: Plots a vertical bar graph for a specified variable, with the x axis showing variable values and the y axis showing frequency of occurrence for those values. | **Response variable**: the variable to be plotted. |
| **Vertical Bar (parameterized)**: Plots a horizontal bar graph for a specified variable, categorised by another variable. The y axis shows variable values and the x axis shows the categories. | **Category**: the variable to be used for categorisation.<br>**Response**: the variable to be plotted. |
| **Vertical Box**: Plots a single variable on the y axis, drawing a box centered on the variable's median value, with sides set at the 25th and 75th percentile. Also shown are whiskers above and below the box. The lower whisker is the lowest value still within 1.5 times the interquartile range of the lower quartile. The upper whisker is the highest value still within 1.5 times the interquartile range of the upper quartile. | **Response variable**: the variable to be plotted. |
| **Vertical Line**: Plots a line graph for a specified variable, with the y axis showing variable values, and the x axis showing frequency of occurrence. | **Response variable**: the variable to be plotted. |

**Options**

Sets options for the plot selected in the **Plots** area using the check box next to the plot.

- **Group**: Choose which variable to group plotted elements by, with the grouping shown by style and colour.
- **Legend label**: A label for the plot legend.
- **Transparency**: Transparency for the plot, where 1.00 is completely transparent and 0.00 is completely opaque.

**By Variables**

Generates separate charts for each unique value in the selected variables. The charts can be viewed in Chart Builder using the left and right arrow buttons; Chart Viewer will generate thumbnails for each chart that can be selected to view the full chart. The selected variables must be sorted, so that all observations are grouped together by value.

## Chart Builder Chart Viewer

The chart builder block outputs its results as a Chart Viewer block, which can be opened by double clicking on it.

If there are multiple charts, specified by using the **By Variables** function in Chart Builder, thumbnails of these are displayed on the left hand side, with the full chart displayed on the right hand side when each thumbnail is selected.

**Note:**

Changes to a chart must be saved for them to appear in the Chart Viewer output.

### Chart Editor

Double click the **Resize Chart** button at the top right of the Chart Viewer to open the **Chart Editor** window.

To copy the chart to the clipboard, click **Copy Chart**. The size of the copied chart is determined by the size of the **Chart Editor** window when the chart is copied.

# Delimited File Export block

Enables you to export a working dataset to a text file, specifying the field delimiters.

The **Delimited File Export** block requires a single dataset input, and creates a file containing the exported dataset.

Exporting a dataset is configured using the **Configure Delimited File Export** dialog box. To open the **Configure Delimited File Export** dialog box, double-click the **Delimited File Export** block.

**Format**

**Delimiter**

Specifies the character used to mark the boundary between variables in an observation of the exported dataset.

If the required delimiter is not listed, select `Other` and enter the character to use as the delimiter for exported dataset.

**Write headers to first row**

Specifies that the names of the variables in the input dataset are written as column headings to the first row of the exported file.

**File Location**

Specifies where the file containing the output dataset is located.

**Workspace**

Specifies the location of file containing the dataset is the current workspace.

**Workspace** datasets are only accessible when using the *Local Engine*.

**External**

Specifies the location of the file containing the dataset is on the file system accessible from the device running the Workflow Engine.

**External** files are accessible when using either a *Local Engine* or a *Remote Engine*.

**Path**

Specifies the path and file name for the file containing the output dataset. If you enter the **Path**:

- When exporting a dataset to a file in the Workspace, the root of the path is the Workspace. For example, to export a file to a project named `datasets`, the path is `/datasets/filename`.
- When exporting a dataset to an external location, the path is the absolute (full) path to the file location.

  The export **Path** for the file is only valid with the Workflow Engine in use when the workflow is created. The path will need to be re-entered if the workflow is used with a different Workflow Engine.

If you do not know the path to the file, click **Browse** and navigate to the required dataset in the **Choose file** dialog box.

# Excel Export block

Enables you to export a dataset from a workflow to an Excel Workbook.

The **Excel Export** block requires a single dataset input, and creates an Excel workbook with a single worksheet.

Exporting a dataset to Microsoft Excel is configured using the **Configure Excel Export** dialog box. To open the **Configure Excel Export** dialog box, double-click the **Excel Export** block.

**Format**

Specifies the Excel Workbook version of the saved file containing the exported dataset. The format options are:

- Excel 2007-2013 Workbook (`.xlsx`)
- Excel 97-2003 Workbook .(`xls`)

**Write headers to first row**

Specifies that the names of the variables in the input dataset are written as column headings to the first row of the exported file.

**File Location**

Specifies where the file containing the output dataset is located.

**Workspace**

Specifies the location of file containing the dataset is the current workspace.

**Workspace** datasets are only accessible when using the *Local Engine*.

**External**

Specifies the location of the file containing the dataset is on the file system accessible from the device running the Workflow Engine.

**External** files are accessible when using either a *Local Engine* or a *Remote Engine*.

**Path**

Specifies the path and file name for the file containing the output dataset. If you enter the **Path**:

- When exporting a dataset to a file in the Workspace, the root of the path is the Workspace. For example, to export a file to a project named `datasets`, the path is `/datasets/`*`filename`*.
- When exporting a dataset to an external location, the path is the absolute (full) path to the file location.

  The export **Path** for the file is only valid with the Workflow Engine in use when the workflow is created. The path will need to be re-entered if the workflow is used with a different Workflow Engine.

If you do not know the path to the file, click **Browse** and navigate to the required dataset in the **Choose file** dialog box.

# Workflow Environment preferences

The Workflow Environment preferences in the **WPS** section of the **Preferences** dialog box specify defaults and preferences when using the Workflow Environment perspective.



## Data panel

Specifies the default level for classifying variables and whether a subset of the dataset is used for profiling or growing decision trees.

**Classification threshold**

Specifies the number of unique values in a variable above which the variable classification is Continuous. If the number of unique values is below the threshold, the classification is Categorical for numerical type variables or Discrete for character type variables. The classification is displayed in the Summary View tab of the **Data Profiler** view .

**Limit cells processed to**

Specifies the maximum number of cells processed when creating statistics in the **Data Profiler** view or **Decision Tree Editor** view. If left empty, all cells in the dataset are used when creating statistical information or growing a decision tree.

Reducing the number of processed cells in a dataset can help reduce the time taken to view information about a dataset or grow a decision tree.

The  restriction can be applied to one or both views enabling you to, for example, use a subset of the dataset in the **Data Profiler** view and use the whole dataset to grow a decision tree.

# Data Profiler panel

Specifies the default settings for profiling datasets in the **Data Profiler** view.

## Summary View preferences

Enable you to specify whether the frequency graph for variables is displayed in the Summary View panel. In a dataset with a large number of variables displaying the frequency graph may reduce the responsiveness of the **Data Profiler** view

To display the graph, select **Show Frequency Graph**. To hide the graph, clear **Show Frequency Graph**.

## Predictive power preferences

Enable you to specify the number of variables displayed in the Entropy chart of the Predictive Power panel. Variables displayed are those calculated to have the highest Entropy Variance. If the number of displayed variables is too small, the graph may only display variables with a value variance that is too random to enable a good selection of dependent variables.

# Workflow panel

Specifies default settings for running a workflow and where intermediate datasets are stored.

## Execution

Specifies whether the workflow is run automatically or manually. To run the workflow manually, on the **File** menu, click **Run Workflow**. This option will only run blocks that have changed. If you need to regenerate all working datasets in the workflow, on the **File** menu click **Force Run Workflow**.

**Auto Run Workflow**

Select to run a workflow when a new block is added, or an existing block updated. If left clear the workflow needs to be run manually when modified.

## Temporary Resources

Specifies whether working datasets and temporary resources used by the workflow are saved to disk, and where the resources are located.

**Persist to disk**

Select to save working datasets in a temporary location on disk. If left clear, working datasets are stored in memory while the workflow is open; this may lead to a workflow failing if the datasets produced are large.

**Location**

Specifies the folder where working datasets and other temporary resources are located. By default this folder is in your user profile. To change the location, click **Browse**, and select an alternative folder.

**Time-To-Live**

Specifies the number of minutes, hours or days the temporary resources are kept for before being deleted.

**Clear temporary resources**

Click to remove all working datasets and other temporary resources saved to disk.

# Binning panel

Specifies the default number of bins created, and the maximum number of bins that can be created.

**Default bin count**

Specifies the number of bins to be used in equal-width or equal-height binning.

**Winsorrate**

Specifies the minimum and maximum percentile value for winsorized binning. All values below or above this percentile of observations are set to the lower or upper observational values at this point.

For example, if you specify 0.05; prior to the data being split into bins, all values below the 5th percentile are set to the lower observational value at the 5th percentile, and all values above the 95th percentile are set to the upper observational value at the 95th percentile.

## Optimal binning

Specifies the default settings used to determine how a dataset is split into optimal groups for further analysis.

**Measure**

Specifies the split measure used when performing optimal binning. for more information about these measures, see Predictive power criteria (page 24).

**Chi-squared**

Specifies that Pearson's Chi-Squared Test is used to measure the predictive power of variables by measuring the likelihood that the value of the dependent variable is related to the value of the independent variable.

**Gini Variance**

Specifies that Gini Variance is used to measure the predictive power of variables by measuring the strength of association between variables.

**Entropy Variance**

Specifies that Entropy Variance is used to measure the predictive power of variables by measuring how well an independent variable value can predict the dependent variable value.

**Information Value**

Specifies that Information Value is used to measure the predictive power of variables by measuring the likelihood that the dependent variable value is related to the value of the independent variable.

**Information Value**

Specifies the default settings for the Information Value measure.

**Minimum information value**

Specifies the minimum value for the information variable. If the information value of a variable falls below this limit it is not added to the bin.

**Maximum change when merging**

Specifies the maximum change allowed in the predictive power when optimally merging bins.

**WoE adjust**

Specifies the adjustment applied in weight of evidence calculations to avoid invalid results for pure inputs.

**Entropy Variance**

Specifies the default settings for the Entropy Variance measure.

**Minimum value**

Specifies the minimum value for the Entropy Variance. If the Entropy Variance of a variable falls below this limit it is not added to the bin.

**Maximum change when merging**

Specifies the maximum change allowed in the predictive power when optimally merging bins.

**Gini Variance**

Specifies the default settings for the Gini Variance measure.

**Minimum value**

Specifies the minimum value for the Gini Variance. If the Gini Variance of a variable falls below this limit it is not added to the bin.

**Maximum change when merging**

Specifies the maximum change allowed in the predictive power when optimally merging bins.

**Chi-squared**

Specifies the default settings for the Chi-squared measure.

**Minimum value**
Specifies the minimum value for the Chi-squared measure. If the Chi-squared measure of a variable falls below this limit it is not added to the bin.

**Maximum change when merging**
Specifies the maximum change allowed in the predictive power when optimally merging bins.

**Initial number of bins**
Specifies the number of bins into which data is merged before the optimal binning begins.

For example, if you specify 50, then before starting the optimal binning process, a numerical variable is equally binned into 50 bins; if the number of categories in a variable is greater than 50, similar categories are merged to create 50 bins.

**Maximum number of bins**
Specifies the maximum number of bins to use when optimally binning variables.

**Exclude Missing**
Excludes missing values.

**Merge a bin with missing values**
Specifies that missing values are merged into the bin containing the most similar values. If left clear, a separate bin is created for missing values.

**Monotonic Weight of Evidence**
Specifies that the Weight of Evidence value for ordered input variables is either monotonically increasing or monotonically decreasing.

**Minimum bin size**
Specifies the minimum number of observations for bin as either a proportion of the input dataset or an absolute value.

**Count**
Specifies the absolute minimum number of observations a bin can contain.

**Percent**
Specifies the minimum size of each bin as a proportion of the number of input observations.

# Chart panel

Specifies preferences for the **Correlation Analysis** view of the **Data Profiler** view.

**Scatter plot observations threshold**
Specifies the limit above which a heat map of values is shown in the **Scatter Plot** panel of the **Correlation Analysis** view.

**Correlation Matrix**

Specifies how items are displayed in the **Correlation Coefficient Matrix** chart in the **Correlation Analysis** view.

**Vary matrix item size by coefficient magnitude**

Select to display the size of the items in the **Correlation Coefficient Matrix** chart relative to the coefficient of the two variables being compared. Clear to display all items in the chart at the same size.

**Matrix item shape**

Specifies the shape of the item on the **Correlation Coefficient Matrix** chart, either `Circle` or `Square`

# Decision Tree panel

Specifies the default tree growth preferences for a decision tree.

The **Decision Tree** block uses the information specified in this panel when growing a decision tree using the `DECISIONTREE` procedure in WPS Analytics. For more information see the *`DECISIONTREE` procedure* in the *WPS Reference for Language Elements*

**Default Growth**

Specifies the default growth algorithm for a new decision tree, one of:

- `C4.5`. Specifies the C4.5 algorithm is used for generating a classification decision trees.
- `CART`. Specifies the CART algorithm is used for generating classification and regression decision trees.
- `BRT`. Specifies the BRT algorithm is used for generating binary response decision trees.

## C4.5

Specifies the default preferences for a decision tree grown using the C4.5 method. A decision tree grown with this method always uses the Entropy Variance to determine when to split a node in the tree.

**Maximum depth**

Specifies the maximum depth of the decision tree.

**Minimum node size (%)**

Specifies the minimum number of observations in a decision tree node, as a proportion of the input dataset.

**Merge categories**

Specifies that discrete independent (input) variables are grouped together to optimize the dependent (target) variable value used for splitting.

**Prune**

Select to specify that nodes which do not significantly improve the predictive accuracy are removed from the decision tree to reduce tree complexity.

**Prune confidence level**

Specifies the confidence level for pruning, expressed as a percentage.

**Exclude missings**

Specifies that observations containing missing values are excluded when determining the best split at a node.

## CART

Specifies the default preferences for a decision tree grown using the CART method.

**Criterion**

Specifies the criterion used to split nodes in the decision tree. The supported criteria are:

**Gini**

Specifies that Gini Impurity is used to measure the predictive power of variables.

**Ordered twoing**

Specifies that the Ordered Twoing Index is used to measure the predictive power of variables.

**Twoing**

Specifies that the Twoing Index is used to measure the predictive power of variables.

**Prune**

Select to specify that nodes which do not significantly improve the predictive accuracy are removed from the decision tree to reduce tree complexity.

**Pruning method**

Specifies the method to be used when pruning a decision tree. The supported methods are:

**Holdout**

Specifies that the input dataset is randomly divided into a test dataset containing one third of the input dataset and a training dataset containing the balance. The output decision tree is created using the training dataset, then pruned using risk estimation calculations on the test dataset.

**Cross validation**

Specifies that the input dataset is divided as evenly as possible into ten randomly-selected groups, and the analysis repeated ten times. The analysis uses a different group as test dataset each time, with the remaining groups used as training data.

Risk estimates are calculated for each group and then averaged across all groups. The averaged risk values are used to prune the final tree built from the entire dataset.

**Maximum depth**

Specifies the maximum depth of the decision tree.

**Minimum node size (%)**
Specifies the minimum number of observations in a decision tree node, as a proportion of the input dataset.

**Minimum improvement**
Specifies the minimum improvement in impurity required to split decision tree node.

**Exclude missings**
Specifies that observations containing missing values are excluded when determining the best split at a node.

## BRT

Specifies the default preferences for a decision tree grown using the BRT method.

**Criterion**
Specifies the criterion used to split nodes in the decision tree. The supported criteria are:

**Chi-squared**
Specifies that Pearson's Chi-Squared statistic is used to measure the predictive power of variables.

**Entropy variance**
Specifies that Entropy Variance is used to measure the predictive power of variables.

**Gini variance**
Specifies that Gini Variance is used to measure the predictive power of variables by measuring the strength of association between variables.

**Information value**
Specifies that Information Value is used to measure the predictive power of variables.

**Maximum depth**
Specifies the maximum depth of the decision tree.

**Select minimum node size by ratio**
When selected enables you to specify the minimum number of observations in a node as a proportion of the input dataset. When cleared, enables you to specify the absolute minimum number of observations in a node.

**Minimum node size (%)**
Specifies the minimum number of observations in a decision tree node, as a proportion of the input dataset.

**Minimum node size**
When selected enables you to specify the minimum split size of a decision tree node as a proportion of the input dataset. When cleared, enables you to specify the absolute minimum split size of a decision tree node .

**Select minimum split size by ratio**
When selected enables you to specify the minimum number of observations for the node to be split as a proportion of the input dataset. When cleared, enables you to specify the absolute minimum number of observations in a node.

**Minimum split size (%)**
Specifies the minimum number of observations a decision tree node must contain for the node to be split, as a proportion of the input dataset.

**Minimum split size**
Specifies the absolute minimum number of observations a decision tree node must contain for the node to be split.

**Allow same variable split**
Specifies that a variable can be used more than once to split a decision tree node.

**Open left**
For a continuous variable, when selected, specifies that the minimum value in the node containing the very lowest values is $-\infty$ (minus infinity). When clear, the minimum value is the lowest value for the variable in the input dataset.

**Open right**
For a continuous variable, when selected, specifies that the maximum value in the node containing the very largest values is $\infty$ (infinity). When clear, the maximum value is the largest value for the variable in the input dataset.

**Merge missing bin**
Specifies that missing values are considered a separate valid category when binning data.

**Monotonic Weight of Evidence**
Specifies that the Weight of Evidence value for ordered input variables is either monotonically increasing or monotonically decreasing.

**Exclude missings**
Specifies that observations containing missing values are excluded when determining the best split at a node.

**Initial number of bins**
Specifies the number of bins available when binning variables.

**Maximum number of optimal bins**
Specifies the maximum number of bins to use when optimally binning variables.

**Weight of Evidence adjustment**
Specifies the adjustment applied to weight of evidence calculations to avoid invalid results for pure inputs.

**Maximum change in predictive power**
Specifies the maximum change allowed in the predictive power when optimally merging bins.

**Minimum predictive power for split**

Specifies the minimum predictive power required for a decision tree node to be split.

# Legal Notices



## Trademarks

WPS and World Programming are registered trademarks or trademarks of World Programming Limited in the European Union and other countries. (r) or ® indicates a Community trademark.

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

All other trademarks are the property of their respective owner.

## General Notices

World Programming Limited is not associated in any way with the SAS Institute.

WPS is not the SAS System.

The phrases "SAS", "SAS language", and "language of SAS" used in this document are used to refer to the computer programming language often referred to in any of these ways.

The phrases "program", "SAS program", and "SAS language program" used in this document are used to refer to programs written in the SAS language. These may also be referred to as "scripts", "SAS scripts", or "SAS language scripts".

The phrases "IML", "IML language", "IML syntax", "Interactive Matrix Language", and "language of IML" used in this document are used to refer to the computer programming language often referred to in any of these ways.

WPS includes software developed by third parties. More information can be found in the THANKS or acknowledgments.txt file included in the WPS installation.