Spectrum™ Technology Platform

Version 12.0 SP2

Machine Learning Guide

Table of Contents

10 - Data Science Demonstration Flows

1 - Introduction

In this section

Machine Learning Module

The Spectrum™ Technology Platform Machine Learning Module provides the ability to bin numeric data and fit supervised and unsupervised machine learning models data in those models.

> **Note:** The Machine Learning Module is supported only on Windows and Linux operating systems.

Binning

Binning divides records into groups (bins) for a continuous variable without taking into account objective information. You can perform unsupervised binning in one of two ways: using equal-width bins or equal-frequency bins.

K-Means Clustering

K-Means Clustering creates models based on analytical clustering, which segments a set of records into clusters of similar records based on data values.

Linear Regression

Linear Regression performs machine learning by creating models from datasets that use continuous objectives with input variables.

Logistic Regression

Logistic Regression creates models from datasets that use binary objectives with input variables.

Principal Component Analysis

Principal Component Analysis is a statistical process that converts a set of observations of possibly correlated variables into a set of values of linearly uncorrelated variables known as principal components.

Random Forest Classification

Random Forest Classification performs machine learning by creating models from datasets that use continuous objectives with input variables.

Random Forest Regression

Random Forest Regression performs machine learning by creating models from datasets that use binary objectives with input variables.

Machine Learning Model Management

Machine Learning Model Management includes Model Assessment, which enables you to manage all machine learning models on your Spectrum™ Technology Platform server, and Binning Management, which enables you to manage all binning on your Spectrum™ Technology Platform server.

> **Note:** The Machine Learning Module uses an underlying H2O.ai library for modeling algorithms in K-Means Clustering, Linear Regression, Logistic Regression, Principal Component Analysis, Random Forest Classification, and Random Forest Regression.

A Machine Learning Workflow

A typical machine learning workflow includes the following steps that take place in one or more dataflows:

1. Access the data using other Spectrum modules, such as Data Integration.
2. Prepare the data using stages from other Spectrum modules such as those in Data Integration, Data Quality, and the Core modules.
3. Fit a machine learning model, run the dataflow, and then review the contents of the Model Output tab in the model stage. You can then tweak the model if necessary and rerun the dataflow. Following that, you need to review the full set of model assessment output in Machine Learning Model Management tool. You can review one model at a time or compare two models.
4. (Optional) If the model will be used to score data, expose the model in the Machine Learning Model Management tool, which makes the model available to the Java Model Scoring stage.

   a. Create a Spectrum™ Technology Platform data flow with steps 1-2 above, then replace step 3 with the Java Model Scoring stage. Set up this dataflow to run in batch mode to populate a file with model scores applied to refreshed data (the fields used as Xs or inputs are refreshed in step 1-2 as a natural part of doing business).

   b. Alternatively, use a web service in Spectrum™ Technology Platform to score data on demand. For example, access the website, get the customer ID and model inputs, score those and return the score to a process that customizes web content for your customer.

5. (Optional) You can also deploy model scores into a Data Hub graph database as an entity property, onto maps, or into CES applications.

2 - Binning

In this section

Introduction to Binning

The Binning stage performs what is known as unsupervised binning, which divides a continuous variable into groups (bins) without taking into account objective information. The data captured includes ranges, quantities, and percentage of values within each range.

Advantages to performing binning include the following:

- It allows records with missing data to be included in the model.
- It controls or mitigates the impact of outliers over the model.
- It solves the issue of having different scales among the characteristics, making the weights of the coefficients in the final model comparable.

In Spectrum™ Technology Platform unsupervised binning, you can use equal-width bins, where the data is divided into bins of equal size, or equal-frequency bins, where the data is divided into groups containing approximately the same number of records. In the Binning stage, equal-width bins are referred to as Equal Range bins and equal-frequency bins are referred to as Equal Population bins.

You can perform more binning functions using the Machine Learning Model Management **Binning Management** tool.

You can also view a list of binning and delete binning using command line instructions. See "Machine Learning Module" in the **Administration Utility** section of the Administration Guide.

> **Note:** If you upgrade Spectrum™ Technology Platform from version 12.0 SP1 to 12.0 SP2, you must manually unexpose any upgraded binning in the Machine Learning Model Management Binning Management tool before using them for rebinning in 12.0 SP2. This step is not necessary if you are using upgraded binning in Binning Lookup instead of traditional Binning.

Defining Binning Properties

1. Under **Primary Stages** / **Deployed Stages** / **Machine Learning**, click the **Binning** stage and drag it onto the canvas, placing it where you want on the dataflow and connecting it to other stages. Note that the input stage must be the data source that contains both the objective and input variable fields for your model; an output stage is not required unless you select the Score input data option on the Basic Options tab. You may also connect an output stage if you wish to capture your output independent of the Machine Learning Model Management tool.
2. Double-click the Binning stage to show the **Binning Options** dialog box.
3. Enter a **Binning name** if you do not want to use the default name.

4. Check the **Overwrite** box to overwrite the existing model with new data.
5. Enter a **Description** of the model.
6. Click **Include** for each field whose data you want included in binning. Note that only numeric fields will appear in this list.
7. Click **OK** to save your settings.

Configuring Basic Options

1. Select whether you want to perform an equal-range or equal-population **Binning style**.
2. Select in **Null value bin** how you want to handle empty bin fields, which represent unknown values due to missing data. Select **Highest** to assign null values to the highest bin and select **Lowest** to assign null values to the lowest bin. The lowest bin is always bin 1.
3. Click **Target internal bins** and enter the number of bins you want to fill between the end bins. If you are performing equal-range binning, you may select this type of processing or **Bin width**, but not both. If you are performing equal-population binning, you may only perform internal-bin processing.
4. If you are performing equal-range binning and want to select this type of processing rather than internal-bin processing, click **Bin width** and enter the number of units you want in each bin.
5. Click **Include** for each field whose data you want included in binning. Note that only numeric fields will appear in this list.
6. Click **OK** to save your settings.

Binning Output

The Binning stage has two output ports. The first port will output all input fields plus a binned field for each selected input field. For example, if the input contains Name, Age, and Income fields and you perform binning on Age and Income, the output from the first port will contain the following fields:

• Name
• Age
• Binned_Age
• Income
• Binned_Income

The second port outputs four types of information for each selected input field. For example, if you perform binning on Age, the output from the second port will contain the following fields:

• Age_Bins

- Age_BinValue
- Age_Count
- Age_Percentage

3 - K-Means Clustering

In this section

Introduction to K-Means Clustering

K-Means Clustering creates models based on analytical clustering, which segments a set of records into clusters of similar records based on data values.

To create your model, you must first complete the Model Properties tab. The Basic Options and Advanced Options tabs provide sufficient default settings to complete a job, but you can change those settings to meet your needs. You then run your job and a limited version of the resulting model output details appears on the Model Output tab; the model is stored on the Spectrum™ Technology Platform server and the complete output is available in the Machine Learning Model Management tool.

Defining Model Properties

1. Under **Primary Stages** / **Deployed Stages** / **Machine Learning**, click the **K-Means Clustering** stage and drag it onto the canvas, placing it where you want on the dataflow and connecting it to other stages. Note that the input stage must be the data source that contains input variable fields for your model; an output stage is not required unless you select the Score input data option on the Basic Options tab. You may also connect an output stage if you wish to capture your output independent of the Machine Learning Model Management tool.
2. Double-click the K-Means stage to show the **K-Means Clustering Options** dialog box.
3. Enter a **Model name** if you do not want to use the default name.
4. Optional: Check the **Overwrite** box to overwrite the existing model with new data.
5. Enter the **Number of clusters** you want in your model if you do not want the default number (5).
6. Optional: Enter a **Description** of the model.
7. Click **Include** for each field whose data you want added to the model.
8. Use the **Model Data Type** drop-down to specify whether the input field is to be used as a numeric, categorical, or datetime field.
9. Click **OK** to save the model and configuration or continue to the next tab.

Configuring Basic Options

1. Leave **Standardize input fields** checked to standardize the numeric columns to have zero mean and unit variance.

   If you do not use standardization, the results may include components dominated by variables appearing to have larger variances relative to other attributes as a matter of scale rather than true contribution.

2. Check **Score input data** to add a column for the model prediction (score) to the input data.

3. Check **Estimate number of clusters** to have the K-Means algorithm attempt to determine the number of clusters that your model will contain. Even though you designate the number of desired clusters on the Model Properties tab, the routine may discover in its processing that a different number of clusters is more appropriate given the data.

4. Specify a value between 1 and 100 as the **Percentage for training data** when the input data is randomly split into training and test data samples.

5. Enter the value of 100 minus the amount you entered in Step 5 as the **Percentage for test data**.

6. Enter a number as the **Seed for sampling** to ensure that when the data is split into test and train data it will occur the same way each time you run the dataflow. Uncheck this field to get a random split each time you run the flow.

7. Click **OK** to save the model and configuration or continue to the next tab.

Configuring Advanced Options

1. Leave **Ignore constant fields** checked to skip fields that have the same value for each record.

2. Leave **Seed for algorithm** checked and enter a seed number to ensure that when the data is split into test and training data it will occur the same way each time you run the dataflow. Uncheck this field to get a random split each time you run the flow.

3. Select the correct initialization mode in the **Init** drop-down.

Furthest	Initializes the first centroid randomly, but then initializes the second centroid to be the data point farthest away from it. Initializes the centroids to be well spread-out from each other.
Plus-Plus	Initializes the cluster centers before proceeding with the standard k-means optimization iterations. With the k-means++ initialization, the algorithm is guaranteed to find a solution that is O(log k) competitive to the optimal k-means solution.

Random	Default. Chooses K clusters from the set of N observations at random so that each observation has an equal chance of being chosen.

4. Leave **Seed for N fold** checked and enter a seed number to ensure that when the data is split into test and train data it will occur the same way each time you run the dataflow. Uncheck this field to get a random split each time you run the flow.

5. Check **N fold** and enter the number of folds if you are performing cross-validation.

6. Check **Fold assignment** and select from the drop-down list if you are performing cross-validation. This field is applicable only if you entered a value in **N fold**.

Auto	Default. Allows the algorithm to automatically choose an option; currently it uses Random.
Modulo	Evenly splits the dataset into the folds and does not depend on the seed.
Random	Randomly splits the data into nfolds pieces; best for large datasets.

7. Check **Maximum iterations** and enter the number of training iterations that should take place.

8. Click **OK** to save the model and configuration or continue to the next tab.

Model Output

This tab shows the metrics you are using to assess the fitted model. These fields cannot be edited. The Training column will always contain data. If you selected a train/test split on the Basic Options tab, the Test column will also be filled, unless you have selected an N Fold validation on the Advanced Options tab, in which case the N Fold column will be filled. Click the **Output** button to regenerate the output, and click **Model details** to view the entire output in the Machine Learning Model Management tool.

4 - Linear Regression

In this section

Introduction to Linear Regression

Linear Regression enables you to perform machine learning by creating models from datasets that use continuous objectives with input variables.

To create your model, you must first complete the Model Properties tab. The Basic Options and Advanced Options tabs provide sufficient default settings to complete a job, but you can change those settings to meet your needs. You then run your job and a limited version of the resulting model appears on the Model Output tab; the complete output is available in the Machine Learning Model Management tool.

Defining Model Properties

1. Under **Primary Stages** / **Deployed Stages** / **Machine Learning**, click the **Linear Regression** stage and drag it onto the canvas, placing it where you want on the dataflow and connecting it to other stages. Note that the input stage must be the data source that contains both the objective and input variable fields for your model; an output stage is not required unless you select the Score input data option on the Basic Options tab. You may also connect an output stage if you wish to capture your output independent of the Machine Learning Model Management tool.
2. Double-click the Linear Regression stage to show the **Linear Regression Options** dialog box.
3. Enter a **Model name** if you do not want to use the default name.
4. Check the **Overwrite** box to overwrite the existing model with new data.
5. Click the **Objective field** drop-down and select a numerical field.
6. Enter a **Description** of the model.
7. Click **Include** for each field whose data you want added to the model; be sure to include the field you selected as the Objective field.
8. Use the **Model Data Type** drop-down to specify whether each input field is to be used as a numeric, categorical, or datetime field.
9. Click **OK** to save the model and configuration or continue to the next tab.

Configuring Basic Options

1. Leave **Standardize input fields** checked to standardize the numeric columns to have zero mean and unit variance.

If you do not use standardization, the results may include components dominated by variables appearing to have larger variances relative to other attributes as a matter of scale rather than true contribution.

2. Check **Score input data** to add a column for the model prediction (score) to the input data.

3. Select a **Link function** from the drop-down list. This specifies the link between random and systematic components. It says how the expected value of the response relates to the linear predictor of explanatory variables.

Identity	Predicts nonsense "probabilities" less than zero or greater than one; sometimes used for binomial data to yield a linear probability model.
	$g(p) = p$
Inverse	Computes the inverse of link functions for real estimates.
	$g(\mu i)=1\mu i$
Log	Counts occurrences in a fixed amount of time and space.
	$g(\mu i)=log(\mu i)$

4. Specify how to handle missing data by checking **Skip** or **Impute means**, which will add the mean value for any missing data.

5. Specify a value between 1 and 100 as the **Percentage for training data** when the input data is randomly split into training and test data samples.

6. Enter the value of 100 minus the amount you entered in Step 5 as the **Percentage for test data**.

7. Enter a number as the **Seed for sampling** to ensure that when the data is split into test and train data it will occur the same way each time you run the dataflow. Uncheck this field to get a random split each time you run the flow.

8. Click **OK** to save the model and configuration or continue to the next tab.

Configuring Advanced Options

1. Leave **Ignore constant fields** checked to skip fields that have the same value for each record.

2. Check **Compute p values** to calculate p values for the parameter estimates.

3. Check **Remove collinear column** to automatically remove collinear columns during model building. This will result in a 0 coefficient in the returned model.

   This option must be checked if **Compute p values** is also checked.

4. Leave **Include constant term (Intercept)** checked to include a constant term (intercept) in the model.

   This field must be checked if **Remove collinear column** is also checked.

5. Select a **Solver** from the drop-down list. Note that CoordinateDescent and CoordinateDescentNaive are currently experimental.

Auto	Solver will be determined based on input data and parameters.
CoordinateDescent	IRLSM with the covariance updates version of cyclical coordinate descent in the innermost loop.
CoordinateDescentNaive	IRLSM with the naive updates version of cyclical coordinate descent in the innermost loop.
IRLSM	Ideal for problems with a small number of predictors or for Lambda searches with L1 penalty.
LBFGS	Ideal for datasets with many columns.

6. Leave **Seed for N fold** checked and enter a seed number to ensure that when the data is split into test and train data it will occur the same way each time you run the dataflow. Uncheck in this field to get a random split each time you run the flow.

7. Check **N fold** and enter the number of folds if you are performing cross-validation.

8. Click **Fold assignment** and select from the drop-down list if you are performing cross-validation. This field is applicable only if you entered a value in **N fold** and **Fold field** is not specified.

Auto	Allows the algorithm to automatically choose an option; currently it uses Random.
Modulo	Evenly splits the dataset into the folds and does not depend on the seed.
Random	Randomly splits the data into nfolds pieces; best for large datasets.

9. If you are performing cross-validation, check **Fold field** and select the field that contains the cross-validation fold index assignment from the drop-down list.

   This field is applicable only if you did not enter a value in **N fold** and **Fold assignment**.

10. Check **Maximum iterations** and enter the number of training iterations that should take place.

11. Check **Objective epsilon** and enter the threshold for convergence; this must be a value between 0 and 1. If the objective value is less than this threshold, the model will be converged.

12. Check **Beta epsilon** and enter the threshold for convergence; this must be a value between 0 and 1. If the objective value is less than this threshold, the model will be converged. If the L1 normalization of the current beta change is below this threshold, consider using convergence.

13. Click **OK** to save the model and configuration or continue to the next tab.

Model Output

This tab shows the metrics you are using to assess the fitted model. These fields cannot be edited. The Training column will always contain data. If you selected a train/test split on the Basic Options tab, the Test column will also be filled, unless you have selected an N Fold validation on the Advanced Options tab, in which case the N Fold column will be filled.

After you run your job, the resulting model is stored on the Spectrum™ Technology Platform server. Click the **Output** button to regenerate the output and click **Model details** to view the entire output in the Machine Learning Model Management tool.

5 - Logistic Regression

In this section

Introduction to Logistic Regression

Logistic Regression enables you to perform machine learning by creating models from datasets that use binary objectives with input variables.

To create your model, you must first complete the Model Properties tab. The Basic Options and Advanced Options tabs provide sufficient default settings to complete a job, but you can change those settings to meet your needs. You then run your job and a limited version of the resulting model appears on the Model Output tab; the complete output is available in the Machine Learning Model Management tool.

Defining Model Properties

1. Under **Primary Stages** / **Deployed Stages** / **Machine Learning**, click the **Logistic Regression** stage and drag it onto the canvas, placing it where you want on the dataflow and connecting it to other stages. Note that the input stage must be the data source that contains both the objective and input variable fields for your model; an output stage is not required unless you select the Score input data option on the Basic Options tab. You may also connect an output stage if you wish to capture your output independent of the Machine Learning Model Management tool.

2. Double-click the Logistic Regression stage to show the **Logistic Regression Options** dialog box.

3. Enter a **Model name** if you do not want to use the default name.

4. Check the **Overwrite** box to overwrite the existing model with new data.

5. Click the **Objective field** drop-down and select "Categorical."

6. Enter a **Description** of the model.

7. Click **Include** for each field whose data you want added to the model; be sure to include the field you selected as the Objective field.

8. Use the **Model Data Type** drop-down to specify whether each input field is to be used as a numeric, categorical, or datetime field.

9. Click **OK** to save the model and configuration or continue to the next tab.

Configuring Basic Options

1. Leave **Standardize input fields** checked to standardize the numeric columns to have zero mean and unit variance.

    If you do not use standardization, the results may include components dominated by variables appearing to have larger variances relative to other attributes as a matter of scale rather than true contribution.

2. Check **Score input data** to add a column for the model prediction (score) to the input data.

3. Check **Prior** if the data has been sampled and the mean of response does not reflect reality; then enter the prior probability for p(y==1) in the text field.

4. Specify how to handle missing data by checking **Skip** or **Impute means**, which will add the mean value for any missing data.

5. Specify a value between 1 and 100 as the **Percentage for training data** when the input data is randomly split into training and test data samples.

6. Enter the value of 100 minus the amount you entered in Step 5 as the **Percentage for test data**.

7. Enter a number as the **Seed for sampling** to ensure that when the data is split into test and train data it will occur the same way each time you run the dataflow. Uncheck this field to get a random split each time you run the flow.

8. Click **OK** to save the model and configuration or continue to the next tab.

Configuring Advanced Options

1. Leave **Ignore constant fields** checked to skip fields that have the same value for each record.

2. Leave **Compute p values** checked to calculate p values for the parameter estimates.

3. Leave **Remove collinear column** checked to automatically remove collinear columns during model building. This will result in a 0 coefficient in the returned model.

    This option must be checked if **Compute p values** is also checked.

4. Leave **Include constant term (Intercept)** checked to include a constant term (intercept) in the model.

    This field must be checked if **Remove collinear column** is also checked.

5. Select a **Solver** from the drop-down list. Note that CoordinateDescentNaive and CoordinateDescentNaive are currently experimental.

Auto	Solver will be determined based on input data and parameters.

CoordinateDescentNaive	IRLSM with the covariance updates version of cyclical coordinate descent in the innermost loop.
CoordinateDescentNaive	IRLSM with the naive updates version of cyclical coordinate descent in the innermost loop.
IRLSM	Ideal for problems with a small number of predictors or for Lambda searches with L1 penalty.
L_BFGS	Ideal for datasets with many columns.

6. Leave **Seed for N fold** checked and enter a seed number to ensure that when the data is split into test and train data it will occur the same way each time you run the dataflow. Uncheck this field to get a random split each time you run the flow.

7. Check **N fold** and enter the number of folds if you are performing cross-validation.

8. Check **Fold assignment** and select from the drop-down list if you are performing cross-validation. This field is applicable only if you entered a value in **N fold** and **Fold field** is not specified.

Auto	Allows the algorithm to automatically choose an option; currently it uses Random.
Modulo	Evenly splits the dataset into the folds and does not depend on the seed.
Random	Randomly splits the data into nfolds pieces; best for large datasets.
Stratified	Stratifies the folds based on the response variable for classification problems. Evenly distributes observations from the different classes to all sets when splitting a dataset into train and test data. This can be useful if there are many classes and the dataset is relatively small.

9. If you are performing cross-validation, check **Fold field** and select the field that contains the cross-validation fold index assignment from the drop-down list.

   This field is applicable only if you did not enter a value in **N fold** and **Fold assignment**.

10. Check **Maximum iterations** and enter the number of training iterations that should take place.

11. Check **Objective epsilon** and enter the threshold for convergence; this must be a value between 0 and 1. If the objective value is less than this threshold, the model will be converged.

12. Check **Beta epsilon** and enter the threshold for convergence; this must be a value between 0 and 1. If the objective value is less than this threshold, the model will be converged. If the L1 normalization of the current beta change is below this threshold, consider using convergence.

13. Click **OK** to save the model and configuration or continue to the next tab.

Model Output

This tab shows the metrics you are using to assess the fitted model. These fields cannot be edited. The Training column will always contain data. If you selected a train/test split on the Basic Options tab, the Test column will also be filled, unless you have selected an N Fold validation on the Advanced Options tab, in which case the N Fold column will be filled.

After you run your job, the resulting model is stored on the Spectrum™ Technology Platform server. Click the **Output** button to regenerate the output and click **Model details** to view the entire output in the Machine Learning Model Management tool.

6 - Principal Component Analysis

In this section

Introduction to Principal Component Analysis

Principal Component Analysis (PCA) is a statistical process that converts a set of observations of possibly correlated variables into a set of values of linearly uncorrelated variables known as principal components.

To create your model, you must first complete the Model Properties tab. The Basic Options and Advanced Options tabs provide sufficient default settings to complete a job, but you can change those settings to meet your needs. You then run your job and a limited version of the resulting model appears on the Model Output tab; the complete output is available in the Machine Learning Model Management tool. If you are satisfied with the output of your model, you can then expose it and use it in a scoring dataflow.

Defining Model Properties

1. Under **Primary Stages** / **Deployed Stages** / **Machine Learning**, click the **PCA Options** stage and drag it onto the canvas, placing it where you want on the dataflow and connecting it to other stages. Note that the input stage must be the data source that contains the principal components for your model. An output stage is not required but you may connect one if you wish to capture your output independent of the Machine Learning Model Management tool.
2. Double-click the PCA Options stage to show the **PCA Options** dialog box.
3. Enter a **Model name** if you do not want to use the default name.
4. Optional: Check the **Overwrite** box to overwrite the existing model with new data.
5. Enter the number of **Principal components** you want your model to contain.
6. Optional: Enter a **Description** of the model.
7. In the **Inputs** table click "Include" for each field whose data you want added to the model.
8. Use the **Model Data Type** drop-down to specify whether the input field is to be used as a categorical, datetime, numeric, string, or uniqueid field.
9. Click **OK** to save the model and configuration or continue to the next tab.

Configuring Basic Options

1. Leave **Use all factor level** unchecked to skip the first principal component, which has the largest variance in the data. Check this box to to retain the first principal component.

2. Check **Score input data** to add a column for the model prediction (score) to the input data.
3. Select the appropriate **Transform** for the training data.

Demean	Subtracts the mean of each column.
Descale	Divides by the standard deviation of each column.
None	
Normalize	Demeans and divides each column by its range (maximum minus minimum).
Standardize	Uses zero mean and unit variance. Default.

4. Specify how to handle **Missing data** by checking **Skip** or **Impute means**, which will add the mean value for any missing data.
5. Click **OK** to save the model and configuration or continue to the next tab.

Configuring Advanced Options

1. Leave **Ignore constant fields** checked to skip fields that have the same value for each record.
2. Select a **PCA method** from the drop-down list. Note that GLRM and Power are currently experimental.

GLRM	Fits a generalized low-rank model with L2 loss function and no regularization; solves for the SVD using local matrix algebra. This option is enabled only if you checked **Use all factor level** on the Basic Options tab.
GramSVD	Uses a distributed computation of the Gram matrix, followed by a local SVD using the JAMA package.
Power	Computes the SVD using the power iteration method.
Randomized	Uses the randomized subspace iteration method.

3. Leave **Maximum iterations** unchecked to have an unlimited number of training iterations (default). Check the box and enter a number to limit the amount of training iterations.
4. Click **OK** to save the model and configuration or continue to the next tab.

Model Output

This tab shows the metrics you are using to assess the fitted model. These fields cannot be edited.

After you run your job, the resulting model is stored on the Spectrum™ Technology Platform server. Click the **Output** button to regenerate the output and click **Model details** to view the entire output in the Machine Learning Model Management tool.

7 - Random Forest Classification

In this section

Introduction to Random Forest Classification

Random Forest Classification enables you to perform machine learning by creating models from datasets that use continuous objectives with input variables.

To create your model, you must first complete the Model Properties tab. The Basic Options and Advanced Options tabs provide sufficient default settings to complete a job, but you can change those settings to meet your needs. You then run your job and a limited version of the resulting model appears on the Model Output tab; the complete output is available in the Machine Learning Model Management tool.

> **Note:** Click **here** for additional information regarding Random Forest Classification and its options.

Defining Model Properties

1. Under **Primary Stages** / **Deployed Stages** / **Machine Learning**, click the **Random Forest Classification** stage and drag it onto the canvas, placing it where you want on the dataflow and connecting it to other stages. Note that the input stage must be the data source that contains both the objective and input variable fields for your model; an output stage is not required unless you select the Score input data option on the Basic Options tab. You may also connect an output stage if you wish to capture your output independent of the Machine Learning Model Management tool.

2. Double-click the Random Forest Classification stage to show the **Random Forest Classification Options** dialog box.

3. Enter a **Model name** if you do not want to use the default name.

4. Optional: Check the **Overwrite** box to overwrite the existing model with new data.

5. Click the **Objective field** drop-down and select a numeric field.

6. Click **Multinomial levels** and enter the maximum number of categories into which the objective field can be grouped.

7. Optional: Enter a **Description** of the model.

8. Click **Include** for each field whose data you want added to the model; be sure to include the field you selected as the Objective field.

9. Use the **Model Data Type** drop-down to specify whether each input field is to be used as a numeric, categorical, or datetime field.

10. Click **OK** to save the model and configuration or continue to the next tab.

Configuring Basic Options

1. Enter the maximum **Number of trees** in your model.
2. Enter the **Maximum depth**—or the maximum number of levels you want your model to contain.
3. Enter the **Minimum rows**—the minimum number of rows (or records) you want your model to contain.
4. Enter the **Number of bins numeric**—the number of bins you want the histogram to build and then split at the best point.
5. Enter the **Number of bins top level**—the minimum number of bins you want at the root level.
6. Enter the **Number of bins categorical**—the maximum number of bins you want the histogram to build and then split at the best point.
7. Check **Sample rate** and enter the percentage of the rows to be used as a sample in each tree. This can be a value from 0.0 to 1.0.
8. Check **Column sample rate per tree** and enter the column sampling rate for each tree. This can be a value from 0.0 to 1.0.
9. Check **Columns at each level** and enter the relative change of the column sampling rate for every level. Valid values range from 1.0 to the number of the selected input predictor. Default is 1.0.
10. Check **Score input data** to add a column for the model prediction (score) to the input data.
11. Specify a value between 1 and 100 as the **Percentage for training data** when the input data is randomly split into training and test data samples.
12. Enter the value of 100 minus the amount you entered in Step 5 as the **Percentage for test data**.
13. **Seed for sampling** to ensure that when the data is split into test and train data it will occur the same way each time you run the dataflow. Uncheck this field to get a random split each time you run the flow.
14. Click **OK** to save the model and configuration or continue to the next tab.

Configuring Advanced Options

1. Leave **Ignore constant fields** checked to skip fields that have the same value for each record.
2. Check **Balance classes** to balance the class distribution and either undersample the majority classes or oversample the minority classes.
3. Select a **Histogram type**.

Auto	Buckets are binned from minimum to maximum in steps of (max-min)/N. Use this option to specify the type of histogram for finding optimal split points.
QuantilesGlobal	Buckets have equal population. This computes `nbins` quantiles for each numeric (non-binary) column, then refines/pads each bucket (between two quantiles) uniformly (and randomly for remainders) into a total of `nbins_top_level` bins.
Random	The algorithm will sample N-1 points from minimum to maximum and use the sorted list of those to find the best split.
RoundRobin	The algorithm will cycle through all histogram types (one per tree).
UniformAdaptive	Each feature is binned into buckets of equal step size (not population). This is the quickest method but can lead to less accurate splits if the distribution is highly skewed.

4. Select a **Categorical encoding**.

Auto	Automatically performs `enum` encoding.
Binary	Converts categories to integers, then to binary, and assigns each digit a separate column. Encodes the data in fewer dimensions but with some distortion of the distances. **Note:** No more than 32 columns can exist per categorical feature.
Eigen	*k* columns per categorical feature, keeping projections of one-hot-encoded matrix onto *k*-dim eigen space only.
Enum	Cycles through all histogram types (one per tree).
OneHotExplicit	One column exists per category, with "1" or "0" in each cell representing whether the row contains that column's category.

5. Leave **Seed for algorithm and N fold** checked and enter a seed number to ensure that when the data is split into test and training data it will occur the same way each time you run the dataflow. Uncheck this field to get a random split each time you run the flow.

6. Check **N fold** and enter the number of folds if you are performing cross-validation.

7. Check **Fold assignment** and select from the drop-down list if you are performing cross-validation. This field is applicable only if you entered a value in **N fold** and **Fold field** is not specified.

Auto	Allows the algorithm to automatically choose an option; currently it uses Random.
Modulo	Evenly splits the dataset into the folds and does not depend on the seed.
Random	Randomly splits the data into nfolds pieces; best for large datasets.

Stratified   Stratifies the folds based on the response variable for classification problems. Evenly distributes observations from the different classes to all sets when splitting a dataset into train and test data. This can be useful if there are many classes and the dataset is relatively small.

8. If you are performing cross-validation, check **Fold field** and select the field that contains the cross-validation fold index assignment from the drop-down list.

    This field is applicable only if you did not enter a value in **N fold** and **Fold assignment**.

9. Check **Stopping rounds** to end training when the Stopping_metric option does not improve for the specified number of training rounds and enter the number of unsuccessful training rounds to occur before stopping. To disable this feature, specify 0. The metric is computed on the validation data (if provided); otherwise, training data is used.

10. Select a **Stopping metric** to determine when to quit creating new trees.

    AUC   Area under ROC curve.

                               Note: Applicable only to binomial models.

    Auto   Defaults to `deviance`.

    Lifttopgroup   Top 1%.

    Logloss   Logarithmic loss.

    Meanperclasserror   The average misclassification rate.

    Misclassification   The value of (1 - (correct predictions/total predictions)) * 100.

    MSE   Mean squared error; incorporates both the variance and the bias of the predictor.

    RMSE   Root mean square error; measures the differences between values (sample and population values) predicted by a model or an estimator and the values actually observed. Also the square root of MSE.

11. Check **Stopping tolerance** and enter a value to specify the relative tolerance for the metric-based stopping to end training if the improvement is less than this value. This field is enabled only if you checked **Stopping rounds**.

12. Check **Minimum split improvement** and enter a value to specify the minimum relative improvement in squared error reduction in order for a split to happen. When properly executed, this option can help reduce overfitting. Optimal values would be in the 1e-10...1e-3 range. This field is enabled only if you checked **Stopping rounds**

13. Click **OK** to save the model and configuration or continue to the next tab.

Model Output

This tab shows the metrics you are using to assess the fitted model. These fields cannot be edited. The Training column will always contain data. If you selected a training/test split on the Basic Options tab, the Test column will also be filled, unless you have selected an N Fold validation on the Advanced Options tab, in which case the N Fold column will be filled.

After you run your job, the resulting model is stored on the Spectrum™ Technology Platform server. Click the **Output** button to regenerate the output and click **Model details** to view the entire output in the Machine Learning Model Management tool.

8 - Random Forest Regression

In this section

Introduction to Random Forest Regression

Random Forest Regression enables you to perform machine learning by creating models from datasets that use binary objectives with input variables.

To create your model, you must first complete the Model Properties tab. The Basic Options and Advanced Options tabs provide sufficient default settings to complete a job, but you can change those settings to meet your needs. You then run your job and a limited version of the resulting model appears on the Model Output tab; the complete output is available in the Machine Learning Model Management tool.

> **Note:** Click **here** for additional information regarding Random Forest Regression and its options.

Defining Model Properties

1. Under **Primary Stages** / **Deployed Stages** / **Machine Learning**, click the **Random Forest Regression** stage and drag it onto the canvas, placing it where you want on the dataflow and connecting it to other stages. Note that the input stage must be the data source that contains both the objective and input variable fields for your model; an output stage is not required unless you select the Score input data option on the Basic Options tab. You may also connect an output stage if you wish to capture your output independent of the Machine Learning Model Management tool.

2. Double-click the Random Forest Regression stage to show the **Random Forest Regression Options** dialog box.

3. Enter a **Model name** if you do not want to use the default name.

4. Optional: Check the **Overwrite** box to overwrite the existing model with new data.

5. Click the **Objective field** drop-down and select a numeric field.

6. Optional: Enter a **Description** of the model.

7. Click **Include** for each field whose data you want added to the model; be sure to include the field you selected as the Objective field.

8. Use the **Model Data Type** drop-down to specify whether each input field is to be used as a numeric, categorical, or datetime field.

9. Click **OK** to save the model and configuration or continue to the next tab.

Configuring Basic Options

1. Enter the maximum **Number of trees** in your model. Default is 50.
2. Enter the **Maximum depth**—or the maximum number of levels you want your model to contain. Default is 5.
3. Enter the **Minimum rows**—the minimum number of rows (or records) you want your model to contain. Default is 10.
4. Enter the **Number of bins numeric**—the number of bins you want the histogram to build and then split at the best point. Default is 20.
5. Enter the **Number of bins top level**—the minimum number of bins you want at the root level. Default is 1024.
6. Enter the **Number of bins categorical**—the maximum number of bins you want the histogram to build and then split at the best point. Default is 1024.
7. Check **Sample rate** and enter the percentage of the rows to be used as a sample in each tree. This can be a value from 0.0 to 1.0.
8. Check **Column sample rate per tree** and enter the column sampling rate for each tree. This can be a value from 0.0 to 1.0.
9. Check **Columns at each level** and enter the relative change of the column sampling rate for every level. This option defaults to 1.0 and can be a value from 0.0 to 2.0.
10. Check **Score input data** to add a column for the model prediction (score) to the input data.
11. Specify a value between 1 and 100 as the **Percentage for training data** when the input data is randomly split into training and test data samples.
12. Enter the value of 100 minus the amount you entered in Step 5 as the **Percentage for test data**.
13. **Seed for sampling** to ensure that when the data is split into test and train data it will occur the same way each time you run the dataflow. Uncheck this field to get a random split each time you run the flow.
14. Click **OK** to save the model and configuration or continue to the next tab.

Configuring Advanced Options

1. Leave **Ignore constant fields** checked to skip fields that have the same value for each record.
2. Select a **Histogram type**.

Auto	Buckets are binned from minimum to maximum in steps of (max-min)/N. Use this option to specify the type of histogram for finding optimal split points.
QuantilesGlobal	Buckets have equal population. This computes `nbins` quantiles for each numeric (non-binary) column, then refines/pads each bucket (between two quantiles) uniformly (and randomly for remainders) into a total of `nbins_top_level` bins.
Random	The algorithm will sample N-1 points from minimum to maximum and use the sorted list of those to find the best split.
RoundRobin	The algorithm will cycle through all histogram types (one per tree).
UniformAdaptive	Each feature is binned into buckets of equal step size (not population). This is the quickest method but can lead to less accurate splits if the distribution is highly skewed.

3. Select a **Categorical encoding**.

Auto	Automatically performs `enum` encoding.
Binary	Converts categories to integers, then to binary, and assigns each digit a separate column. Encodes the data in fewer dimensions but with some distortion of the distances. **Note:** No more than 32 columns can exist per categorical feature.
Eigen	*k* columns per categorical feature, keeping projections of one-hot-encoded matrix onto *k*-dim eigen space only.
Enum	Cycles through all histogram types (one per tree).
OneHotExplicit	One column exists per category, with "1" or "0" in each cell representing whether the row contains that column's category.

4. Leave **Seed for algorithm and N fold** checked and enter a seed number to ensure that when the data is split into test and training data it will occur the same way each time you run the dataflow. Uncheck this field to get a random split each time you run the flow.

5. Check **N fold** and enter the number of folds if you are performing cross-validation.

6. Check **Fold assignment** and select from the drop-down list if you are performing cross-validation. This field is applicable only if you entered a value in **N fold** and **Fold field** is not specified.

Auto	Allows the algorithm to automatically choose an option; currently it uses Random.
Modulo	Evenly splits the dataset into the folds and does not depend on the seed.
Random	Randomly splits the data into nfolds pieces; best for large datasets.

7. If you are performing cross-validation, check **Fold field** and select the field that contains the cross-validation fold index assignment from the drop-down list.

    This field is applicable only if you did not enter a value in **N fold** and **Fold assignment**.

8. Check **Stopping rounds** to end training when the Stopping_metric option does not improve for the specified number of training rounds and enter the number of unsuccessful training rounds to occur before stopping. To disable this feature, specify 0. The metric is computed on the validation data (if provided); otherwise, training data is used.

9. Select a **Stopping metric** to determine when to quit creating new trees.

Auto	Defaults to `deviance`.
deviance	Mean residual deviance; identical to MSE.
MAE	Mean absolute error; the difference between two continuous variables.
MSE	Mean squared error; incorporates both the variance and the bias of the predictor.
RMSE	Root mean square error; measures the differences between values (sample and population values) predicted by a model or an estimator and the values actually observed. Also the square root of MSE.
RMSLE	Root mean squared logarithmic error; measures the ratio between predicted and actual.

10. Check **Stopping tolerance** and enter a value to specify the relative tolerance for the metric-based stopping to end training if the improvement is less than this value.

11. Check **Minimum split improvement** and enter a value to specify the minimum relative improvement in squared error reduction in order for a split to happen. When properly executed, this option can help reduce overfitting. Optimal values would be in the 1e-10...1e-3 range. This field is enabled only if you checked **Stopping rounds**.

12. Click **OK** to save the model and configuration or continue to the next tab.

Model Output

This tab shows the metrics you are using to assess the fitted model. These fields cannot be edited. The Training column will always contain data. If you selected a training/test split on the Basic Options tab, the Test column will also be filled, unless you have selected an N Fold validation on the Advanced Options tab, in which case the N Fold column will be filled.

After you run your job, the resulting model is stored on the Spectrum™ Technology Platform server. Click the **Output** button to regenerate the output and click **Model details** to view the entire output in the Machine Learning Model Management tool.

9 - Machine Learning Model Management

In this section

Accessing Machine Learning Model Management

There are three ways to access Machine Learning Model Management:

• Use the Spectrum™ Technology Platform Welcome Page:

  • Open a web browser and go to the Spectrum™ Technology Platform Welcome Page at:

    <servername>:<port>

    For example, if you installed Spectrum™ Technology Platform on a computer named "myspectrumplatform" and it is using the default port 8080, you would go to:

    myspectrumplatform:8080

  • Click **Spectrum Machine Learning**.
  • Click **Open Machine Learning Model Management**.

• Click **For model details click here** from one of the model-building stages.
• Use a a web browser:

  • Open a web browser and go to the Spectrum™ Technology Platform Machine Learning Model Management page at:

    <servername>:<port>/machinelearning

    For example, if you installed Spectrum™ Technology Platform on a computer named "myspectrumplatform" and it is using the default port 8080, you would go to:

    myspectrumplatform:8080/machinelearning

  • Enter a valid Spectrum™ Technology Platform username and password.

Model Assessment

Introduction to Model Assessment

The Model Assessment tab in Machine Learning Model Management shows a list of all machine learning models on your Spectrum™ Technology Platform server. You can filter this list by entering a string in the text box; every field in the table will be searched for that string.

Several operations can be performed on these models. You can import, export, expose, unexpose, or delete models. Exposed models are used in the Java Model Scoring stage to score new data using formulas created when you fit machine learning models. Additionally, you can view detailed information for each model; the details returned depend on the type of model whose data you are viewing. Finally, you can compare any two models of the same type. This comparison shows side-by-side the same information that is on the Model Detail tab for each of the models you are comparing.

Model Assessment Operations

Perform these operations by selecting a model and clicking the appropriate button:

	View model output detail. You can also access this information from the K-Means Clustering and Logistic Regression stages by clicking "For model details click here" on the Model Output tab.
	Compare models.
	Import a model from a specific path. Select whether to overwrite an existing model of the same name, if appropriate.
	Export a model to a specific path. Select whether to overwrite an existing model of the same name, if appropriate.
	Expose the model to make it available to the Java Model Scoring stage. If a model is not exposed, it cannot be used for scoring.
	Unexpose the model.
	Delete the model. **Note:** You cannot delete an exposed model; however, at this time there is no inherent security that prevents a user from deleting another user's models.

Model Detail Tab

The Model Detail screen shows the following information for all models:

• **Model Name**—The name of the model
• **Model Type**—The type of machine learning model
• **User**—The username of the person who created the model
• **Description**—The description of the model if one was provided when it was created

- **Status**—Whether the model is exposed or unexposed
- **Dataflow Name**—The name of the dataflow that produced the model
- **Creation Time**—The date and time the model was created

Additional details are provided based on the model type.

K-Means Clustering Details

The Model Detail screen includes the following information for K-Means Clustering models:

Model Summary

- Number of Rows
- Number of Clusters
- Number of Categorical Columns
- Number of Iterations
- Within Cluster Sum of Squares
- Total Sum of Squares
- Between Cluster Sum of Squares

Metrics

Provides training, test, and n-fold data for the following:

- Total within cluster sum of squares
- Total sum of squares
- Between cluster sum of squares

Centroid Statistics

Provides the following training, test, and n-fold data for each centroid:

- Size
- Within cluster sum of squares

Cluster Means

Provides detailed information for each centroid. Content varies based on input data. A cluster is a group of observations from a data set identified as similar according to a particular clustering algorithm

Standardized Cluster Means

Provides standardized information for each centroid. Content varies based on input data.

Logistic Regression Details

The Model Detail screen includes the following information for Logistic Regression models:

Metrics

Provides training, test, and n-fold data for the following:

- Mean squared error (MSE)

- Root mean squared error (RMSE)
- Number of observations
- R-squared (R2)
- Logarithmic loss (Logloss)
- Area under the curve (AUC)
- Gini coefficient
- Mean per class error
- Akaike information criterion (AIC)
- Residual deviance
- Null deviance
- Null degree of freedom
- Residual degree of freedom

Maximum Metrics Threshold

Provides the Training Maximum Metrics Threshold for training, test, and n-fold data using the following metrics:

- max f1
- max f2
- max f0point5
- max accuracy
- max precision
- max recall
- max specificity
- max absolute_mcc
- max min_per_class_accuracy
- max mean_per_class_accuracy

Confusion Matrix

Illustrates the performance of a model on a set of training, test, and n-fold data for which the true values are known.

Standardized Coefficient Chart

Shows the most important predictors by providing the relative value of the coefficients, which indicates how much a change in input changes the objective.

GLM Coefficients

Shows coefficients for a Generalized Linear Model, which estimates regression models for outcomes following exponential distributions.

AUC Curves

Area under the curve; determines which of the used models predicts the classes best using training, test, and n-fold data.

Lift/Gain Curves

Evaluate the prediction ability of a binary classification model using training, test, and n-fold data.

Linear Regression Details

The Model Detail screen includes the following information for Linear Regression models:

Metrics

Provides training test, and n-fold data for the following:

- Mean squared error (MSE)
- Root mean squared error (RMSE)
- Number of observations
- R-squared (R2)
- Mean residual deviance
- Mean absolute error (MAE)
- Root mean squared logarithmic error (RMSLE)
- Akaike information criterion (AIC)
- Residual deviance
- Null deviance
- Null degree of freedom
- Residual degree of freedom

Standardized Coefficient Chart

Shows the most important predictors by providing the relative value of the coefficients, which indicates how much a change in particular predictor coefficient value changes the objective value positively or negatively. Also charts the top 25 coefficients in the model.

GLM Coefficients

Shows coefficients for a Generalized Linear Model, which estimates regression models for outcomes following exponential distributions.

Random Forest Regression Details

The Model Detail screen includes the following information for Random Forest Regression models:

Metrics

Provides training, test, and n-fold data for the following:

- Mean squared error (MSE)
- Root mean squared error (RMSE)
- Number of observations
- R-squared (R2)
- Mean residual deviance
- Mean absolute error (MAE)

• Root mean squared logarithmic error (RMSLE)

Variable Importances

Provides importance values for each variable using the following metrics:

• Relative importance
• Scaled importance
• Percentage

Also charts the top 25 variables in the model.

Random Forest Classification Details—Binomial

The Model Detail screen includes the following information for **binomial** Random Forest Classification models:

Metrics

Provides training, test, and n-fold data for the following:

• Mean squared error (MSE)
• Root mean squared error (RMSE)
• Number of observations
• R-squared (R2)
• Logloss
• Area under the curve (AUC)
• Gini
• Mean per class error

Maximum Metrics Threshold

Provides the Training Maximum Metrics Threshold for training, test, and n-fold data using the following metrics:

• max f1
• max f2
• max f0point5
• max accuracy
• max precision
• max recall
• max specificity
• max absolute_mcc
• max min_per_class_accuracy
• max mean_per_class_accuracy

Confusion Matrix

Illustrates the performance of a model on a set of training, test, and n-fold data for which the true values are known.

Variable Importances

Provides importance values for each variable using the following metrics:

• Relative importance
• Scaled importance
• Percentage

Also charts the top 25 variables in the model.

AUC Curves

Area under the curve; determines which of the used models predicts the classes best using training, test, and n-fold data.

Lift/Gain Curves

Evaluates the prediction ability of a binary classification model using training, test, and n-fold data.

Random Forest Classification Details—Multinomial

The Model Detail screen includes the following information for **multinomial** Random Forest Classification models:

Metrics

Provides training, test, and n-fold data for the following:

• Mean squared error (MSE)
• Root mean squared error (RMSE)
• Number of observations
• R-squared (R2)
• Logloss
• Mean per class error

Confusion Matrix

Illustrates the performance of a model on a set of training, test, and n-fold data for which the true values are known.

Variable Importances

Provides importance values for each variable using the following metrics:

• Relative importance
• Scaled importance
• Percentage

Also charts the top 25 variables in the model.

Principal Component Analysis Details

The Model Detail screen includes the following information for PCA models:

Importance of components

Shows the principal components in order of importance based on the following metrics:

• Standard deviation
• Proportion of variance
• Cumulative proportion

Rotation

Charts the matrix of variable loadings, the weight by which each standardized original variable should be multiplied to get the component score.

Binning Management

Introduction to Binning Management

The Binning Management tab in Machine Learning Model Management shows a list of all binning on your Spectrum™ Technology Platform server. You can filter this list by entering a string in the text box; every field in the table will be searched for that string.

Several operations can be performed on binning. You can import, export, expose, unexpose, or delete binning. Exposed bins are used by the Binning Lookup stage to apply previously defined binning to new data.

Binning Management Operations

Perform these operations by selecting a binning and clicking the appropriate button:

➔]	Import a binning. Select whether to overwrite an existing binning of the same name, if appropriate.
[➔	Export a binning. Select whether to overwrite an existing binning of the same name, if appropriate.
⬆	Expose the binning to make it available to the Binning Lookup stage. If a binning is not exposed, it cannot be used for lookup.
⬇	Unexpose the binning.

	Delete the binning.

> **Note:** You cannot delete an exposed binning; however, at this time there is no inherent security that prevents a user from deleting another user's binning.

10 - Data Science Demonstration Flows

In this section

Introduction

The Machine Learning Module and Analytics Scoring Modules, along with modules to prepare data for modeling, are part of the Spectrum Data Science offer. These demonstrations show examples of data preparation, modeling, and model scoring. You can create your own dataflows using the step-by-step instructions, or you can use the provided dataflows as a reference.

Supervised Learning: Loan Default Prediction

Download the supervised learning demonstration

The Data Science supervised learning demonstration conducts loan default prediction using Lending Club data. It utilizes several files that together demonstrate the functionality of the Spectrum™ Technology Platform Data Science Solution in Enterprise Designer.

Spectrum_DataScience_Supervised_Learning.zip includes the following files:

- Spectrum_DataScience_Supervised_Learning.pdf—Documentation that walks you through how to build and use the single categorizer dataflow, the scoring dataflow, and all supporting files.
- Data.zip—The required input files, test files, and training files for each of the included dataflows.

  - loan.csv
  - LoanStats_2016Q1.csv
  - LoanStats_2016Q2.csv
  - LoanStats_2016Q3.csv
  - testData.txt
  - testDataCollege.txt
  - testDataStable.txt
  - testDataThankful.txt
  - trainData.txt
  - trainDataCollege.txt
  - trainDataStable.txt
  - trainDataThankful.txt
  - training.xml
  - trainingCollege.xml
  - trainingStable.xml
  - trainingThanks.xml

- Lending_Club_Demo_DF_(V12.1).zip—The dataflows for Spectrum™ Technology Platform 12.1.

- LendingClub_2007_2016Q12_v121_MultipleCategorizers.df
- LendingClub_2007_2016Q1Q2_v121_SingleCategorizer.df
- LendingClub_2016Q3_v121_SingleCategorizer_Scoring.df

- Lending_Club_Demo_DF_(V12.2).zip—The dataflows for Spectrum™ Technology Platform 12.2.

  - LendingClub_2007_2016Q12_v122_MultipleCategorizers.df
  - LendingClub_2007_2016Q1Q2_v122_SingleCategorizer.df
  - LendingClub_2016Q3_v122_SingleCategorizer_Scoring.df

- ReadMe.txt—High-level descriptions and instructions for the previously mentioned files.

You can create your own dataflow by following the step-by-step instructions in the documentation, or you can use the included dataflows as references to confirm what the individual completed stages and dataflows as a whole should look like.

Unsupervised Learning: Segmentation

Download the unsupervised learning demonstration

The Data Science unsupervised learning demonstration conducts segementation using Consumer Expenditure data. It utilizes several files that together demonstrate the functionality of the Spectrum™ Technology Platform Data Science Solution in Enterprise Designer.

Spectrum_DataScience_Unsupervised_Learning.zip includes the following files:

- Spectrum_DataScience_Unsupervised_Learning.pdf—Documentation that walks you through how to build and use the primary dataflow, the subflow, the scoring dataflow, and all supporting files
- Data.zip—The required input files and output files for each of the included dataflows

  - Input folder—The required input files files for each of the included dataflows
  - Output folder—The required output files files for each of the included dataflows
  - PythonBased folder—Required input and output files to use optional Python processing in lieu of Group Statistics and Transformer stages in primary dataflow

- Consumer_Expenditure_Demo_DF_(v12.1).zip—The dataflows for Spectrum™ Technology Platform 12.1

  - ConsumerExpenditure_v121_sampleandcluster.df
  - ConsumerExpenditure_v121_sampleandcluster_subflow.df
  - ConsumerExpenditure_v121_score.df
  - ConsumerExpenditure_v121_subflow.df

- PythonBased folder—Required dataflows, process flows, bat script, Python script and documentation to use optional Python processing in lieu of Group Statistics and Transformer stages in primary dataflow

- Consumer_Expenditure_Demo_DF_(v12.2).zip—The dataflows for Spectrum™ Technology Platform 12.2

  - ConsumerExpenditure_v122_sampleandcluster.df
  - ConsumerExpenditure_v122_sampleandcluster_subflow.df
  - ConsumerExpenditure_v122_score.df
  - ConsumerExpenditure_v122_subflow.df
  - PythonBased folder—Required dataflows, process flows, bat script, Python script and documentation to use optional Python processing in lieu of Group Statistics and Transformer stages in primary dataflow

- ReadMe.txt—High-level descriptions and instructions for the previously mentioned files.

You can create your own dataflow by following the step-by-step instructions in the documentation, or you can use the included dataflows as references to confirm what the individual completed stages and dataflows as a whole should look like.

Notices

USPS® Notices

Pitney Bowes Inc. holds a non-exclusive license to publish and sell ZIP + 4® databases on optical and magnetic media. The following trademarks are owned by the United States Postal Service: CASS, CASS Certified, DPV, eLOT, FASTforward, First-Class Mail, Intelligent Mail, LACS^Link, NCOA^Link, PAVE, PLANET Code, Postal Service, POSTNET, Post Office, RDI, Suite^Link , United States Postal Service, Standard Mail, United States Post Office, USPS, ZIP Code, and ZIP + 4. This list is not exhaustive of the trademarks belonging to the Postal Service.

Pitney Bowes Inc. is a non-exclusive licensee of USPS® for NCOA^Link® processing.

Prices for Pitney Bowes Software's products, options, and services are not established, controlled, or approved by USPS® or United States Government. When utilizing RDI™ data to determine parcel-shipping costs, the business decision on which parcel delivery company to use is not made by the USPS® or United States Government.

Data Provider and Related Notices

Data Products contained on this media and used within Pitney Bowes Software applications are protected by various trademarks and by one or more of the following copyrights:

© Copyright United States Postal Service. All rights reserved.

© 2014 TomTom. All rights reserved. TomTom and the TomTom logo are registered trademarks of TomTom N.V.

© 2016 HERE

Fuente: INEGI (Instituto Nacional de Estadística y Geografía)

Based upon electronic data © National Land Survey Sweden.

© Copyright United States Census Bureau

© Copyright Nova Marketing Group, Inc.

Portions of this program are © Copyright 1993-2007 by Nova Marketing Group Inc. All Rights Reserved

© Copyright Second Decimal, LLC

© Copyright Canada Post Corporation

This CD-ROM contains data from a compilation in which Canada Post Corporation is the copyright owner.

© 2007 Claritas, Inc.

The Geocode Address World data set contains data licensed from the GeoNames Project (**www.geonames.org**) provided under the Creative Commons Attribution License ("Attribution

License") located at **http://creativecommons.org/licenses/by/3.0/legalcode**. Your use of the GeoNames data (described in the Spectrum™ Technology Platform User Manual) is governed by the terms of the Attribution License, and any conflict between your agreement with Pitney Bowes Software, Inc. and the Attribution License will be resolved in favor of the Attribution License solely as it relates to your use of the GeoNames data.

pitney bowes