Create R Model

Updated: July 11, 2015

Creates an R model using custom resources

Category: Data Transformation / Manipulation (https://msdn.microsoft.com/en-us/library/azure/dn905863.aspx)

Module Overview

You can use the **Create R Model** module to create an untrained model from R script that you provide. You can base your model on any learner that is included in an R package in the Azure Machine Learning environment.

After you create the model, you can use Train Model (https://msdn.microsoft.com/en-us/library/azure/dn906044.aspx) to train the model on a dataset, like any other learner in Azure Machine Learning. The trained model can be passed to Score Model (https://msdn.microsoft.com/en-us/library/azure/dn905995.aspx) to use the model to make predictions. The trained model can then be saved, and the scoring workflow can be published as a web service.

Warning

Currently it is not possible to pass the scored results of an R model to Evaluate Model (https://msdn.microsoft.com/en-us/library/azure/dn905915.aspx) or Cross-Validate Model (https://msdn.microsoft.com/en-us/library/azure/dn905852.aspx). If you need to evaluate a model, you can write custom R script and run it using the Execute R Script (https://msdn.microsoft.com/en-us/library/azure/dn905952.aspx) module.

How to Configure Create R Model

The module takes two user-defined scripts as inputs:

- Training script
- Scoring script

The following example demonstrates the structure of these scripts and how to implement a model.

For this example, we implement a two-class Naïve Bayes classifier by using the e1070 package, but you can use any type of learner that is included in the many R packages supported in Azure Machine Learning.

Training Script

The part of the R script that trains the model is run within the Train Model (https://msdn.microsoft.com/en-us/library/azure/dn906044.aspx) module.

The following example demonstrates how to load an R package, create model using one of the learners in that package, and configure the feature and label columns using the predefined constants and functions provided in **Create R Model**.

```
library(e1071)
features <- get.feature.columns(dataset)
labels   <- as.factor(get.label.column(dataset))
train.data <- data.frame(features, labels)
feature.names <- get.feature.column.names(dataset)
names(train.data) <- c(feature.names, "Class")
model <- naiveBayes(Class ~ ., train.data)
```

Let's go through the example script line-by-line.

1. The first line loads the R package, **e1071**, which contain the Naïve Bayes classifier algorithm we want to use.

   ```
   library(e1071)
   ```

   Since this is one of the packages pre-installed in the Azure Machine Learning environment, you don't need to download or install the package.

   For information about how to get the full list of supported packages, see R Language Modules (https://msdn.microsoft.com/en-us/library/azure/dn905920.aspx).

2. The next three lines get the feature columns and the label column from the dataset, and combine them into a new R data frame that is named `train.data`:

   ```
   features <- get.feature.columns(dataset)
   labels   <- as.factor(get.label.column(dataset))
   train.data <- data.frame(features, labels)
   ```

   The predefined function, `get.label.columns()`, returns the column that is selected as the class label in the Train Model (https://msdn.microsoft.com/en-us/library/azure/dn906044.aspx) module.

The predefined function, `get.feature.columns()`, selects the columns that were designated as features in the metadata for dataset.

Warning

By default, all columns except the label column are considered features in Studio. If you want to select specific columns as features, use Metadata Editor (https://msdn.microsoft.com/en-us/library/azure/dn905986.aspx), or select a set of columns within the R script.

3. In the fifth and sixth lines, the predefined function, `get.feature.column.names(dataset)`, is used to get feature column names from the dataset. Those names are designated as the names for columns in `train.data`, and a temporary name `Class` is created for the label column.

```
feature.names <- get.feature.column.names(dataset)
names(train.data) <- c(feature.names, "Class")
```

4. The final line of the code trains the Naïve Bayes classifier algorithm by using the labels and features in the `train.data` data frame.

```
probabilities <- predict(model, dataset, type="raw")[,2]
```

Warning

The model must always be assigned to a variable called `model` in both the train and score scripts.

Scoring Script

The scoring script is executed within the **Score Model** module. Here is the complete script:

```
library(e1071)
probabilities <- predict(model, dataset, type="raw")[,2]
classes <- as.factor(as.numeric(probabilities >= 0.5))
scores <- data.frame(classes, probabilities)
```

Again, let’s go through the sample R script that is used for scoring.

1. The first line loads the package.

```
library(e1071)
```

2. The second line computes the predicted probabilities for the scoring dataset by using the trained model from the training script.

```
probabilities <- predict(model, dataset, type="raw")[,2]
```

3. The third line applies a threshold of 0.5 to probabilities when assigning the predicted class labels.

```
classes <- as.factor(as.numeric(probabilities >= 0.5))
```

4. The final line combines the class labels and probabilities into the output data frame, `scores`.

```
scores <- data.frame(classes, probabilities)
```

Warning

The output dataframe must have the name `scores`.

Publishing the R Model as a Web Service

After you have run the experiment by using the R model, you can publish the experiment as a web service.

To publish the experiment

1. Click **Create Scoring Experiment** in the experiment editor.

   The trained R model is saved, and the training experiment graph is transformed into a scoring experiment.

2. Run the scoring experiment at least once.

3. Click **Publish Web Service**.

Note

By default, the web service expects all input columns from the training data to be provided, including the label column. You can add an instance of Project Columns (https://msdn.microsoft.com/en-us/library/azure/dn905883.aspx) between the input data source and the Score Model (https://msdn.microsoft.com/en-us/library/azure/dn905995.aspx) module to exclude the label you’re trying to predict.

Technical Notes

- The model is cached after the first run of the module and the module is not invoked in subsequent runs until any changes in input scripts are done. Please take this behavior into consideration if your R scripts use any of the following:
    - Functions that generate random numbers
    - Functions that generate random numbers
    - Other nondeterministic functions
- R models cannot be used with these modules: Sweep Parameters (https://msdn.microsoft.com/en-us/library/azure/dn905810.aspx), Cross-Validate Model (https://msdn.microsoft.com/en-us/library/azure/dn905852.aspx), One-vs-All Multiclass (https://msdn.microsoft.com/en-us/library/azure/dn905887.aspx) or Ordinal Regression (https://msdn.microsoft.com/en-us/library/azure/dn906029.aspx).
- R models do not automatically perform feature normalization of categorical data or handle missing values. Handling of such variables should be done within the training and scoring R scripts.

Pre-Defined Functions

Usage	Description
`get.feature.columns(dataset)`	Gets all feature columns.
`get.label.column(dataset, label.type=TrueLabelType)`	Gets the label column, given the type. See the Constants section for a list of the available types.

`get.label.column.names(dataset)`	Gets the names of all label columns.
`get.label.column.name(dataset, label.type=TrueLabelType)`	Gets the name of the label column, given the type. See the Constants section for a list of the available types.
`get.label.column.types(dataset)`	Gets the types of all label columns.
`get.feature.column.names(dataset)`	Gets the names of all feature columns.
`dataset < - set.score.column(dataset, score.type, column.name)`	Sets the score column, given a type. See the Constants section for a list of the available types.
`dataset < - set.feature.channel(dataset, channel.name, column.names)`	Sets the feature channel, given a name. See the Constants section for a list of the available names.

Pre-Defined Constants

Constant	**Description**
TrueLabelType	True label column type
ScoredLabelType	Scored label column type
RawScoreType	Raw score column type
CalibratedScoreType	Calibrated score column type
ScoredProbabilitiesMulticlassColumnTypePattern	The pattern to prepare scored probabilities column type for multiclass classifier
BayesianLinearRegressionScoresFeatureChannel	The name of the feature channel with Bayesian linear regression scores
BinaryClassificationScoresFeatureChannel	The name of the feature channel with binary classification scores

MulticlassClassificationScoresFeatureChannel	The name of the feature channel with multiclass classification scores
OrdinalRegressionScoresFeatureChannel	The name of the feature channel with ordinal regression scores
RegressionScoresFeatureChannel	The name of the feature channel with regression scores

Expected Inputs

Name	Type	Description
Trainer R script	Script	An R script that takes a dataset as input and outputs an untrained model.
Scorer R script	Script	An R script that takes a model and a dataset as input and outputs the scores specified in the script.

Outputs

Name	Type	Description
Model	ILearner interface (https://msdn.microsoft.com/en-us/library/azure/dn905938.aspx)	An untrained model

See Also

Execute R Script (https://msdn.microsoft.com/en-us/library/azure/dn905952.aspx)
Machine Learning Module Descriptions (https://msdn.microsoft.com/en-us/library/azure/dn906013.aspx)
R Language Modules (https://msdn.microsoft.com/en-us/library/azure/dn905920.aspx)