Linear Regression

Updated: July 8, 2015

Creates a linear regression model

Category: Machine Learning / Initialize Model / Regression (https://msdn.microsoft.com/en-us/library/azure/dn905922.aspx)

Module Overview

You can use the **Linear Regression** module to create a linear regression model. Regression is used to predict a numeric outcome.

After you have configured the model, you must train the model using a tagged dataset and the Train Model (https://msdn.microsoft.com/en-us/library/azure/dn906044.aspx) module. The trained model can then be used to make predictions. Alternatively, the untrained model can be passed to Cross-Validate Model (https://msdn.microsoft.com/en-us/library/azure/dn905852.aspx) for cross-validation against a labeled data set.

Understanding Linear Regression

There are many different types of regression. In the most basic sense, regression means fitting a model to a target expressed as a numeric vector. However, statisticians have been developing increasingly advanced methods for regression. Use linear regression when you want a very simple model for a basic predictive task. Linear regression tends to work well on high-dimensional, sparse data sets lacking complexity.

In Azure Machine Learning Studio, linear regression is used to solve multiple linear regression problems, where there is a single dependent variable that has more than one predictor. The process of using multiple inputs to predict a single numeric outcome is also called multivariate linear regression.

The task of predicting multiple dependent variables within a single model is called multi-label regression. For example, in multi-label logistic regression, a sample can be assigned to multiple different labels. This type of regression is not supported in Azure Machine Learning; to do this, you would need to create a separate learner for each output that you wish to predict.

How to Configure a Linear Regression Model

1. Add the **Linear Regression Model** module to your experiment.

2. In the **Properties** pane, specify the computation method used to find the regression line, least squares or online gradient.

3. If you choose the **Online Gradient Descent** option, you can specify that you want to train the model with a parameter sweep, by setting the **Create trainer mode** option.

    - **Single Parameter**

      If you know how you want to configure the linear regression network, you can provide a specific set of values as arguments.

      When you set **Create trainer mode** option to **Single Parameter**, you train the model by using a tagged dataset and the Train Model (https://msdn.microsoft.com/en-us/library/azure/dn906044.aspx) module.

    - **Parameter Range**

      If you want the algorithm to find the best parameters, set **Create trainer mode** option to **Parameter Range**, specifying multiple values

      When you train the model using Sweep Parameters (https://msdn.microsoft.com/en-us/library/azure/dn905810.aspx) and a tagged dataset, the algorithm will determine the optimal parameters.

      You can use the model trained using those parameters, or you can make a note of the parameter settings to use when configuring a learner.

4. Set other options required by the regression model solution method. For more information, see the Options section.

5. Run the experiment.

   After the model has been trained, you can connect it to the Score Model (https://msdn.microsoft.com/en-us/library/azure/dn905995.aspx) module to make predictions.

   Alternatively, the untrained model can be passed to Cross-Validate Model (https://msdn.microsoft.com/en-us/library/azure/dn905852.aspx) for cross-validation against a labeled data set.

Options

When creating a linear regression model, you can customize the model using the following options.

Solution method

The selection of options depends on the method that you use for fitting the regression line: **Online gradient descent**, or **Ordinary least squares**.

- **Online gradient descent**

Gradient descent is a method that minimizes the amount of error at each step of the model training process. There are many variations on gradient descent and its optimization for various learning problems has been extensively studied.

If you choose this option, you can set a variety of parameters to control the step size, learning rate, and so forth.

This option also supports use of an integrated parameter sweep.

- **Ordinary least squares**.

  Least squares linear regression is one of the most commonly used techniques in predictive analytics. This method assumes that there is a fairly strong linear relationship between the inputs and the dependent variable. Ordinary least squares refers to the loss function, which computes error as the sum of the square of distance from the actual value to the predicted line, and fits the model by minimizing the squared error.

  Least squares is also the method that is used in the Analysis Toolpak for Microsoft Excel.

Ordinary least squares options

L2 regularization weight

Specify the weight for L2 regularization.

Use a non-zero value to avoid overfitting.

Random number seed

Specify a value to seed the random number generator used by the model.

Using a seed value is useful if you want to maintain the same results across different runs of the same experiment.

Include intercept term

Indicate whether an additional term should be added for the intercept.

Allow unknown categorical levels

Indicate whether an additional level should be created for each categorical column.

If true, any levels in the test dataset not available in the training dataset are mapped to this additional level.

Online gradient descent options

Create trainer mode

Choose the method used for configuring and training the model:

- ***Single Parameter***

  Select this option to configure and train the model with a single set of parameter values that you supply.

  If you choose this option, you should train the model by using the Train Model (https://msdn.microsoft.com/en-us/library/azure/dn906044.aspx) module.

- ***Parameter Range***

  Select this option to use the Range Builder and specify a range of possible values. You then train the model using a parameter sweep, to find the optimum configuration.

Warning

If you configure the model with specific values using the **Single Parameter** option and then switch to the **Parameter Range** option, the model will be trained using the minimum value in the range for each parameter.

Conversely, if you configure specific settings when you create the model but select the **Parameter Range** option and use a Sweep Parameters (https://msdn.microsoft.com/en-us/library/azure/dn905810.aspx), the model will be trained using the default values for the learner as the range of values to sweep over.

Learning rate

Specify the initial learning rate for the stochastic gradient descent optimizer.

Number of training epochs

Specify how many times the algorithm should iterate through examples.

For datasets with a small number of examples, this number should be large to reach convergence.

L2 regularization weight

Specify the weight for L2 regularization. Use a non-zero value to avoid overfitting.

Normalize features

Select this option to specify that instances should be normalized.

Average final hypothesis

Select this option to average the final hypothesis.

Decrease learning rate

Select this option to decrease the learning rate as iterations progress.

Random number seed

Specify a value to seed the random number generator used by the model.

Using a seed value is useful if you want to maintain the same results across different runs of the same experiment.

Allow unknown categorical levels

Select this option to create an additional level for each categorical column.

Any levels in the test dataset not available in the training dataset are mapped to this additional level.

Recommendations

For small datasets, it is best to select Ordinary Least Squares. This should give very similar results to Excel.

Gradient descent is a better loss function for models that are more complex, or that have too little training data given the number of variables.

Examples

For examples of regression models, see these sample experiments in the Model Gallery (http://gallery.azureml.net/):

- The Compare Regressors (http://go.microsoft.com/fwlink/?LinkId=525731) sample contrasts several different kinds of regression models.
- The Cross Validation for Regression (http://go.microsoft.com/fwlink/?LinkId=525735) sample demonstrates linear regression using ordinary least squares.
- The Twitter sentiment analysis (http://go.microsoft.com/fwlink/?LinkId=525274) sample uses several different regression models to generate predicted ratings.

Technical Notes

Many tools support creation of linear regression, ranging from the simple to complex. For example, you can easily perform linear regression in Excel, using the Solver Toolpak.

Many other software package support linear regression, including MatLab and the R langauge, to name just a few. However, there are some differences in nomenclature you should be aware of. One is that regression methods are often categorized by the number of response variables. For example, multiple linear regression means a model that has multiple variables to predict. In Matlab, multivariate regression refers to a model that has multiple response variables. In Azure Machine Learning, regression models support a single response variable.

In the R language, the features provided for linear regression depend on the package you are using. For example, the glm package will give you the ability to run a logistic regression using multiple independent variables.

In general, Azure Machine Learning Studio provides the same functionality as the R glm package. You can use this module, **Linear Regression**, for typical regression problems, and use Two-Class Logistic Regression (https://msdn.microsoft.com/en-us/library/azure/dn905994.aspx) or Multiclass Logistic Regression (https://msdn.microsoft.com/en-us/library/azure/dn905853.aspx) when using multiple variables to predict a class value.

If you need to use other enhancements for linear regression that are available as packages for the R language, including statistics and summaries, you can use the Execute R Script (https://msdn.microsoft.com/en-us/library/azure/dn905952.aspx) module and call the lm or glm packages, which are included in the runtime environment of Azure Machine Learning Studio.

Module Parameters

Name	**Range**	**Type**	**Default**	**Description**
Normalize features	any	Boolean	true	Indicate whether instances should be normalized
Average final hypothesis	any	Boolean	true	Indicate whether the final hypothesis should be averaged
Learning rate	>=double.Epsilon	Float	0.1	Specify the initial learning rate for the stochastic gradient descent optimizer
Number of training epochs	>=0	Integer	10	Specify how many times the algorithm should iterate through examples. For datasets with a small number of examples, this number should be large to reach convergence.
Decrease learning rate	Any	Boolean	true	Indicate whether the learning rate should decrease as iterations progress
L2 regularization weight	>=0.0	Float	0.001	Specify the weight for L2 regularization. Use a non-zero value to avoid overfitting.
Random number seed	any	Integer		

				Specify a value to seed the random number generator used by the model. Leave blank for default.
Allow unknown categorical levels	any	Boolean	true	Indicate whether an additional level should be created for each categorical column. Any levels in the test dataset not available in the training dataset are mapped to this additional level.
Include intercept term	Any	Boolean	True	Indicate whether an additional term should be added for the intercept

Outputs

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

See Also

A-Z List of Machine Learning Studio Modules (https://msdn.microsoft.com/en-us/library/azure/dn906033.aspx)
Machine Learning / Initialize Model / Regression (https://msdn.microsoft.com/en-us/library/azure/dn905922.aspx)