

Two-Class Logistic Regression

Published: August 18, 2015

Updated: October 13, 2015

Creates a two-class logistic regression model

Category: Machine Learning / Initialize Model / Classification (<https://msdn.microsoft.com/en-us/library/azure/dn905808.aspx>)

Module Overview

You can use the **Two-Class Logistic Regression** module to create a logistic regression model that can be used to predict one of two states of the target variable. Logistic regression is a well-known statistical technique that is used for modeling many kinds of outcomes.

Classification using logistic regression is a supervised learning method, and therefore requires a labeled dataset. For this model, the label column can contain only two values.

You can train the model by providing the model and the tagged dataset to a module such as Train Model (<https://msdn.microsoft.com/en-us/library/azure/dn906044.aspx>) or Sweep Parameters (<https://msdn.microsoft.com/en-us/library/azure/dn905810.aspx>). The trained model can then be used to predict values for the new input examples.

Understanding Logistic Regression

Logistic regression is a well-known method in statistics that is used to predict the probability of an outcome, and is especially popular for classification tasks. The algorithm predicts the probability of occurrence of an event by fitting data to a logistic function. For details about this implementation, see the Technical Notes section.

In **Two-Class Logistic Regression**, the classifier is optimized for the classification of dichotomous or binary variables. If you need to classify multiple outcomes, use the Multiclass Logistic Regression (<https://msdn.microsoft.com/en-us/library/azure/dn905853.aspx>) module.

How to Configure a Two-Class Logistic Regression Model

1. Add the **Two-Class Logistic Regression** module to the experiment.

2. Specify how you want the model to be trained, by setting the **Create trainer mode** option.

- **Single Parameter**

If you know how you want to configure the logistic regression model, you can provide a specific set of values as arguments. You might have learned these values by experimentation or received them as guidance.

- **Parameter Range**

If you are not sure of the best parameters, you can find the optimal parameters by specifying multiple values and using a parameter sweep to find the optimal configuration.

Sweep Parameters (<https://msdn.microsoft.com/en-us/library/azure/dn905810.aspx>) will iterate over all possible combinations of the settings you provided and determine the combination of settings that produces the optimal results.

3. Continue to set other parameters that affect the behavior of the logistic regression model, such as regularization terms and memory.

For more information, see the Options section.

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

If you set the **Create trainer mode** option to **Parameter Range**, add a labeled dataset and train the model using Sweep Parameters (<https://msdn.microsoft.com/en-us/library/azure/dn905810.aspx>). 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.

Options

You can customize the behavior of the logistic regression model by using these settings:

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 pass a parameter range to Train Model (<https://msdn.microsoft.com/en-us/library/azure/dn906044.aspx>), it will use only the first value in the parameter range list.
- If you pass a single set of parameter values to the Sweep Parameters (<https://msdn.microsoft.com/en-us/library/azure/dn905810.aspx>) module, when it expects a range of settings for each parameter, it ignores the values and using the default values for the learner.
- If you select the **Parameter Range** option and enter a single value for any parameter, that single value you specified will be used throughout the sweep, even if other parameters change across a range of values.

Optimization tolerance

Specify the threshold value for optimizer convergence. If you use a parameter sweep, you can either set a single value, or use the Range Builder to set multiple values to use when building each model.

If the improvement between iterations is less than the threshold, the algorithm stops and returns the current model.

L1 regularization weight, L2 regularization weight

Type a value to use for the regularization parameters L1 and L2. A non-zero value is recommended for both. If you use a parameter sweep, you can either set a single value, or use the Range Builder to set multiple values to use when building each model.

Regularization is a method for preventing overfitting by penalizing models with extreme coefficient values. Regularization works by adding the penalty that is associated with coefficient values to the error of the hypothesis. An accurate model with extreme coefficient values would be penalized more, but a less accurate model with more conservative values would be penalized less.

L1 and L2 regularization have different effects and uses.

- L1 can be applied to sparse models, which is useful when working with high-dimensional data.
- In contrast, L2 regularization is preferable for data that is not sparse.

This algorithm supports a linear combination of L1 and L2 regularization values: that is, if $x = L1$ and $y = L2$, $ax + by = c$ defines the linear span of the regularization terms. Different linear combinations of L1 and L2 terms have been devised for logistic

regression models, such as elastic net regularization (https://wikipedia.org/wiki/Elastic_net_regularization).

Memory size for L-BFGS

Specify the amount of memory to use for *L-BFGS* optimization. If you use a parameter sweep, you can either set a single value, or use the Range Builder to set multiple values to use when building each model.

L-BFGS stands for limited memory Broyden-Fletcher-Goldfarb-Shanno, and it is an optimization algorithm that is popular for parameter estimation. This parameter indicates the number of past positions and gradients to store for the computation of the next step.

This optimization parameter limits the amount of memory that is used to compute the next step and direction. When you specify less memory, training is faster but less accurate.

Random number seed

Specify a seed for the algorithm if you want the results to be repeatable over runs.

Allow unknown categorical levels

Select this option to create an additional "unknown" level in each categorical column.

Any values (levels) in the test dataset that are not available in the training dataset are mapped to this "unknown" level.

Examples

For examples of how this learning algorithm is used, see these sample experiments in the Model Gallery (<http://gallery.azureml.net/>):

- The Network intrusion detection (<http://go.microsoft.com/fwlink/?LinkId=525724>) sample uses binary logistic regression to determine whether a case represents an intrusion or not.
- The Cross-Validation for Binary Classifier (<http://go.microsoft.com/fwlink/?LinkId=525735>) sample demonstrates the use of logistic regression in a typical experimental workflow, including model evaluation.

Technical Notes

Logistic regression assumes a *logistic distribution* of the data, where the probability that an example belongs to class 1 is the formula:

$$p(x; \beta_0, \dots, \beta_{D-1})$$

Where:

- x is a D -dimensional vector containing the values of all the features of the instance.
- p is the logistic distribution function.
- $\beta_{\{0\}}, \dots, \beta_{\{D-1\}}$ are the unknown parameters of the logistic distribution.

The algorithm tries to find the optimal values for $\beta_{\{0\}}, \dots, \beta_{\{D-1\}}$ by maximizing the log probability of the parameters given the inputs. Maximization is performed by using a popular method for parameter estimation, called Limited Memory BFGS (http://en.wikipedia.org/wiki/Limited-memory_BFGS).

For more information on the implementation of this algorithm, see Scalable Training of L-1 Regularized Log-Linear Models (<http://research.microsoft.com/apps/pubs/default.aspx?id=78900>), by Andrew and Gao.

Module Parameters

Name	Range	Type	Default	Description
Optimization tolerance	$\geq \text{double.Epsilon}$	Float	0.0000001	Specify a tolerance value for the L-BFGS optimizer
L1 regularization weight	≥ 0.0	Float	1.0	Specify the L1 regularization weight
L2 regularization weight	≥ 0.0	Float	1.0	Specify the L2 regularization weight
Memory size for L-BFGS	≥ 1	Integer	20	Specify the amount of memory (in MB) to use for the L-BFGS optimizer
Random number seed	Any	Integer		Type a value to seed the random number generator used by the model. Leave it blank for the 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 that

				are not available in the training dataset are mapped to this additional level.
--	--	--	--	--

Output

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

See Also

Machine Learning / Initialize Model / Classification (<https://msdn.microsoft.com/en-us/library/azure/dn905808.aspx>)
Multiclass Logistic Regression (<https://msdn.microsoft.com/en-us/library/azure/dn905853.aspx>)
A-Z List of Machine Learning Studio Modules (<https://msdn.microsoft.com/en-us/library/azure/dn906033.aspx>)

