Two-Class Boosted Decision Tree

Updated: July 12, 2015

Creates a binary classifier using a boosted decision tree algorithm

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

Module Overview

You can use the **Two-Class Boosted Decision Tree** module to create a machine learning model that is based on the boosted decision trees algorithm.

Generally, when properly configured, boosted decision trees are the easiest methods with which to get top performance on a wide variety of machine learning tasks. However, they are also one of the more memory-intensive learners, and the current implementation holds everything in memory; therefore, a boosted decision tree model might not be able to process the very large datasets that some linear learners can handle.

For more information about how to choose an algorithm, see these resources:

- Machine learning algorithm cheat sheet for Microsoft Azure Machine Learning Studio (https://azure.microsoft.com/documentation/articles/machine-learning-algorithm-cheat-sheet/)
- How to choose Azure Machine Learning algorithms for clustering, classification, or regression (https://azure.microsoft.com/documentation/articles/machine-learning-algorithm-choice/)

Classification is a supervised learning method, and therefore requires a *tagged dataset*, which includes a label column.

You can train the model by providing the boosted decision tree model and the tagged dataset as an input to 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.

How to Configure a Boosted Tree Model

1. 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 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.

2. Continue to set other parameters that affect the behavior of the decision tree model, such as the number of trees and learning rate.

   See the Options section for details.

3. If you set the **Create trainer mode** option to **Single Parameter**, add a tagged 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 tagged 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 boosted decision trees classifier by using these parameters:

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.

Maximum number of leaves per tree

Type a number to constrain the number of terminal nodes (leaves) in any tree.

By increasing this value, you potentially increase the size of the tree and get better precision, at the risk of overfitting and longer training time.

Maximum number of samples per leaf node

Type a number to specify the minimum number of cases required to create any terminal node (leaf) in a tree.

By increasing this value, you increase the threshold for creating new rules. For example, with the default value of 1, even a single case can cause a new rule to be created. If you increase the value to 5, the training data would have to contain at least 5 cases that meet the same conditions.

Learning rate

Type a value to use as the step size while learning.

This parameter determines how fast or slow the learner converges on the optimal solution. If the learning rate (or step size) is too big you might overshoot the optimal solution. If the learning rate is too small, training takes longer to converge on the best solution.

Number of trees constructed

Configure the maximum number of decision trees that can be created in the ensemble.

By creating more decision trees, you can potentially get better coverage, but training time will increase.

Random number seed

Optionally, type a non-negative integer to use as the random seed value.

Specifying a seed ensures reproducibility across runs that have the same data and parameters.

The random seed is set by default to 0, which means the initial seed value is obtained from the system clock.

Allow unknown categorical levels

Select this option to create a group for unknown values in the training and validation sets.

If you deselect it, the model can accept only the values that are contained in the training data. In the former case, the model might be less precise for known values, but it can provide better predictions for new (unknown) values.

Recommendations

In general, boosted decision trees yield better results when features are somewhat related. If features have a large degree of entropy (that is, they are not related), they share little or no mutual information, and ordering them in a tree will not yield a lot of predictive significance.

Generally, boosting works well when you have many more examples than features because the model is prone to overfitting.

If you have missing values in your data, add indicators for appropriate features.

Because the treatment of features is a simple, non-parametric, less-than/greater-than comparison, normalization or any form of non-monotonic transformation function will have little effect. (It can possibly lead to the choice of different bins, which is effectively random.)

Examples

For examples of how boosted decision trees are used in machine learning, see these sample experiments in the Model Gallery (http://gallery.azureml.net/):

- The Direct marketing (http://go.microsoft.com/fwlink/?LinkId=525168) sample uses the **Two-Class Boosted Decision Tree** algorithm to predict customer appetency.
- The Flight delay prediction (https://gallery.azureml.net/Experiment/837e2095ce784f1ba5ac623a60232027) sample uses the **Two-Class Boosted Decision Tree** algorithm to determine whether a flight is likely to be delayed.
- The Credit card risk (http://go.microsoft.com/fwlink/?LinkId=525270) sample uses the **Two-Class Boosted Decision Tree** algorithm to predict risk.

Technical Notes

To train a boosted decision tree model, you must provide multiple data instances—that is, more than one row.

For detailed information about the boosted decision tree algorithm, see Greedy Function Approximation: A Gradient Boosting Machines (http://www-stat.stanford.edu/~jhf/ftp/trebst.pdf).

The boosted decision tree algorithm in Azure Machine Learning uses the following boosting method:

1. Start with an empty ensemble of weak learners.
2. For each training example, get the current output of the ensemble. (This is the sum of the outputs of all weak learners in the ensemble.)
3. Calculate the gradient of the loss function (problem dependent) for each example.
4. Use the example to fit a weak learner by using that gradient as the target function.
5. Add that weak learner to the ensemble with a strength indicated by the learning rate, and if desired, go to Step 2.

The implementations are identical up to the gradient of the loss, which depends on whether the task is a binary classification problem or a regression problem.

- In a binary classification model, the log-loss is used, much like in logistic regression.
- In a regression (https://msdn.microsoft.com/en-us/library/azure/dn905801.aspx) model, the squared loss is used, and the gradient is the current output, minus the target).

The weak learners in this implementation are the least-squares regression trees, which use the gradients calculated in Step 3 as the target. The trees are subject to the following restrictions:

- They are trained up to a maximum number of leaves.
- Each leaf has a minimum number of examples to guard against overfitting.
- Each decision node is a single feature that is compared against some threshold. If that feature is less than or equal to the threshold, it goes down one path, and if it is greater than the threshold, it goes down the other path.
- Each leaf node is a constant value.

The tree-building algorithm greedily selects the feature and threshold for which a split will most decrease the squared loss with regard to the gradient calculated in Step 3. It is subject to a minimum number of training examples per leaf. It repeatedly splits until it reaches the maximum number of leaves, or until no valid split is available. Features are discretized and binned prior to training, so only a relatively small set of threshold candidates are considered, even for continuous features.

Module Parameters

Name	Range	Type	Default	Description
Maximum number of leaves per tree	>=1	Integer	20	Specify the maximum number of leaves allowed per tree
Minimum number of samples per leaf node	>=1	Integer	10	Specify the minimum number of cases required to form a leaf
Learning rate	[double.Epsilon;1.0]	Float	0.2	Specify the initial learning rate
Number of trees constructed	>=1	Integer	100	Specify the maximum number of trees that can be created during training
Random number seed	Any	Integer		Type a value to seed the random number generator that is used by the model. Leave it blank for the default.
Allow unknown categorical levels	Any	Boolean	True	If True, an additional level is 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 binary classification model

See Also

Machine Learning / Initialize Model / Classification (https://msdn.microsoft.com/en-us/library/azure/dn905808.aspx)
Boosted Decision Tree Regression (https://msdn.microsoft.com/en-

us/library/azure/dn905801.aspx)
A-Z List of Machine Learning Studio Modules (https://msdn.microsoft.com/en-us/library/azure/dn906033.aspx)