Two-Class Decision Forest

Updated: October 13, 2015

Creates a two-class classification model using the decision forest 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 Decision Forest** module to create a machine learning model based on the random decision forests algorithm. Decision forests are fast, supervised ensemble models. This module can be used to predict a target that has two values.

If you are not sure of the best parameters, we recommend that you use the Sweep Parameters (https://msdn.microsoft.com/en-us/library/azure/dn905810.aspx) module to train and test multiple models and find the optimal parameters.

Understanding Decision Forests

The decision forest algorithm is an ensemble learning method for classification. The algorithm works by building multiple decision trees and then voting on the most popular output class. Voting is a form of aggregation, in which each tree in a classification decision forest outputs a non-normalized frequency histogram of labels. The aggregation process sums these histograms and normalizes the result to get the "probabilities" for each label. The trees that have high prediction confidence will have a greater weight in the final decision of the ensemble.

Decision trees are non-parametric models, and they support data with varied distributions. In each tree, a sequence of simple tests is run for each class, increasing the levels of a tree structure until a leaf node (decision) is reached.

Decision trees have many advantages:

- They can represent non-linear decision boundaries.
- They are efficient in computation and memory usage during training and prediction.
- They perform integrated feature selection and classification.
- They are resilient in the presence of noisy features.

This decision forest classifier consists of an ensemble of decision trees. Generally, ensemble models provide better coverage and accuracy than single decision trees. For more information, see Decision Forests (http://go.microsoft.com/fwlink/?LinkId=403677).

This article by Microsoft Research also provides useful information about ensemble methods that use decision trees. From Stumps to Trees to Forests (http://blogs.technet.com/b/machinelearning/archive/2014/09/10/from-stumps-to-trees-to-forests.aspx).

How to Configure Two-Class Decision Forest

1. Drag the **Two-Class Decision Forest** module into your experiment.

2. For **Resampling method**, specify the method to use for creating multiple trees.

   See the Recommendations section for guidance.

3. 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 decision forest 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.

4. Set other required model parameters. For more information, see the section.5a7d5466-9928-40c8-a19c-d5de4882c77e#Options

5. Connect a labeled dataset and train the model. For this model type, the label column in the dataset can contain no more than two outcomes.

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

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

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

6. 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.

Options

This module supports extensive customization of the decision forest algorithm using these parameters:

Resampling method

Specify which method should be used to create the individual trees.

You can choose from **Bagging** or **Replicate**.

- ***Bagging***. Bagging is also called *bootstrap aggregating*. In this method, each tree is grown on a new sample, created by randomly sampling the original dataset with replacement until you have a dataset the size of the original.

  The outputs of the models are combined by *voting*, which is a form of aggregation. Each tree in a classification decision forest outputs an un-normalised frequency histogram of labels. The aggregation is to sum these histograms and normalise to get the "probabilities" for each label. In this manner, the trees that have high prediction confidence will have a greater weight in the final decision of the ensemble.

  For more information, see the Wikipedia entry for Bootstrap aggregating.

- ***Replicate***. In replication, each tree is trained on exactly the same input data. The determination of which split predicate is used for each tree node remains random and the trees will be diverse.

  For more information about the training process with the **Replicate** option, see Decision Forests for Computer Vision and Medical Image Analysis. Criminisi and J. Shotton. Springer 2013. (http://research.microsoft.com/en-us/projects/decisionforests/)

Number of decision trees

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.

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 ensemble.

Maximum depth of the decision trees

Type a number to constrain the maximum depth of any decision tree.

Increasing the depth of the tree might increase precision, at the risk of some overfitting and increased training time.

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 ensemble.

Number of random splits per node

Specify the number of splits to use when building each node of the tree. Features in each level of the tree (node) are randomly divided.

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 ensemble.

Minimum number of samples per leaf node

Type a number to specify the minimum number of samples 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.

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 ensemble.

Allow unknown values for categorical features

Select this option to create a group for unknown values in the training or 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

If you have limited data or you want to minimize the time spent training the model, try these settings:

Limited training set: If the training set contains a limited number of instances, you can:

- Create the decision forest by using a large number of decision trees (for example, more than 20).
- Use the **Bagging** option for resampling.
- Specify a large number of random splits per node (for example, more than 1,000).

Limited training time: If the training set contains a large number of instances and training time is limited, you can:

- Create the decision forest by using fewer decision trees (for example, 5-10).
- Use the **Replicate** option for resampling.

- Specify a smaller number of random splits per node (for example, fewer than 100).

Example

For examples of how decision forests are used in machine learning, see this sample experiment in the Model Gallery (http://gallery.azureml.net/):

- The News categorization (https://gallery.azureml.net/Experiment/fcb1bf27ee26443fb19bd07852a620c4) sample compares a multiclass classifier to a model built using the **Two-Class Decision Forest** algorithm with the One-vs-All Multiclass (https://msdn.microsoft.com/en-us/library/azure/dn905887.aspx).

- The Predictive maintenance (https://gallery.azureml.net/Experiment/6835de908e6b479e92f2e221e7d18195) sample is an extended walkthrough that uses the **Two-Class Decision Forest** algorithm to predict if an asset will fail within certain time frame.

Technical Notes

Module Parameters

Name	Range	Type	Default	Description
Resampling method	Any	ResamplingMethod	Bagging	Choose a resampling method
Number of decision trees	>=1	Integer	8	Specify the number of decision trees to create in the ensemble
Maximum depth of the decision trees	>=1	Integer	32	Specify the maximum depth of any decision tree that can be created
Number of random splits per node	>=1	Integer	128	Specify the number of splits generated per node, from which the optimal split is selected

Minimum number of samples per leaf node	>=1	Integer	1	Specify the minimum number of training samples that are required to produce a leaf node
Allow unknown values for categorical features	Any	Boolean	True	Indicate whether unknown values of existing categorical features can be mapped to a new, additional feature

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)
Decision Forest Regression (https://msdn.microsoft.com/en-us/library/azure/dn905862.aspx)
Multiclass Decision Forest (https://msdn.microsoft.com/en-us/library/azure/dn906015.aspx)
A-Z List of Machine Learning Studio Modules (https://msdn.microsoft.com/en-us/library/azure/dn906033.aspx)

© 2015 Microsoft