One-vs-All Multiclass

Updated: July 23, 2015

Creates a multiclass classification model from an ensemble of binary classification models

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

Module Overview

You can use the **One-Vs-All Multiclass** module to create a classification model that predicts multiple classes.

This module is useful for creating models that predict three or more possible outcomes, when the outcome depends on continuous or categorical predictor variables. This method also lets you use binary classification methods for issues that require multiple output classes.

To use this classifier, you connect it to an existing untrained two-class classification algorithm, and configure that model. You then train the model as usual by using Train Model (https://msdn.microsoft.com/en-us/library/azure/dn906044.aspx) with a labeled training dataset. Although the training dataset might have multiple class values, the **One-Vs-All Multiclass** creates multiple binary classification models, optimizes the algorithm for each class, and then merges the models.

Understanding the One-vs-All Classifier

While some classification algorithms naturally permit the use of more than two classes, others are naturally binary (two-class) algorithms. However, binary classification algorithms can be turned into multi-class classification algorithms by a variety of strategies. This module implements the *one vs. all method*, in which a binary model is created for each of the multiple output classes. Each of these binary models for the individual classes is assessed against its complement (all other classes in the model) as though it were a binary classification issue. Prediction is then performed by running these binary classifiers, and choosing the prediction with the highest confidence score.

In essence, an ensemble of individual models is created and the results are then merged, to create a single model that predicts all classes. Thus, any binary classifier can be used as the basis for a one-vs-all model.

For example, let's say you configure a Two-Class Support Vector Machine (https://msdn.microsoft.com/en-us/library/azure/dn905835.aspx) model and provide that as input to the **One-Vs-All Multiclass** module. The module would create two-class support vector machine models for all members of the output class and then apply the one-vs-all method to combine the results for all classes.

How to Configure the One-vs-All Classifier

1. Add the **One-Vs-All Multiclass** to your experiment.

2. Add one of the two-class classification models to the experiment, and configure that model.

   For example, you might use a Two-Class Support Vector Machine (https://msdn.microsoft.com/en-us/library/azure/dn905835.aspx) or Two-Class Boosted Decision Tree (https://msdn.microsoft.com/en-us/library/azure/dn906025.aspx) tree model.

   ⚠ **Warning**

   Need help choosing the right algorithm? See these resources:

   - Machine learning algorithm cheat sheet for Azure ML (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/)

   Note that the **One-Vs-All Multiclass** classifier has no configurable parameters of its own. Any customizations must be done in the model that is provided as input.

3. Connect the untrained classifier that is the output of **One-Vs-All Multiclass** to Train Model (https://msdn.microsoft.com/en-us/library/azure/dn906044.aspx).

   On the other input of Train Model (https://msdn.microsoft.com/en-us/library/azure/dn906044.aspx), connect a labeled training data set that has multiple class values.

4. After it has been trained, the model can be used to make multiclass predictions.

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

Examples

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

- The News Categorization (http://go.microsoft.com/fwlink/?LinkId=525167) sample uses **One-Vs-All Multiclass** with a Two-Class Decision Forest (https://msdn.microsoft.com/en-us/library/azure/dn906008.aspx) model.
- In the Compare Multiclass Classifier sample (http://go.microsoft.com/fwlink/?LinkId=525730), binary classifiers are used for each digit and the results combined.

Expected Input

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

Output

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

Exception

For a list of all error messages, see Machine Learning Module Error Codes (https://msdn.microsoft.com/en-us/library/azure/dn905910.aspx).

Exception	**Description**
Error 0013 (https://msdn.microsoft.com/en-us/library/azure/dn906041.aspx)	An exception occurs if the learner that was passed to the module is the wrong type.

See Also

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