[Title]

How to Use One-vs-Rest and One-vs-One for Multi-Class Classification

Algorithms such as the Perceptron, Logistic Regression, and Support Vector Machines were designed for binary classification and do not natively support classification tasks with more than two classes.

One approach for using binary classification algorithms for multi-classification problems is to split the multi-class classification dataset into multiple binary classification datasets and fit a binary classification model on each. Two different examples of this approach are the One-vs-Rest and One-vs-One strategies.

This tutorial is divided into three parts; they are:
1. Binary Classifiers for Multi-Class Classification
2. One-Vs-Rest for Multi-Class Classification
3. One-Vs-One for Multi-Class Classification

Binary Classifiers for Multi-Class Classification

Classification is a predictive modeling problem that involves assigning a class label to an example.

Binary classification are those tasks where examples are assigned exactly one of two classes. Multi-class classification is those tasks where examples are assigned exactly one of more than two classes.

- **Binary Classification**: Classification tasks with two classes.
- **Multi-class Classification**: Classification tasks with more than two classes.

Some algorithms are designed for binary classification problems. Examples include:

- Logistic Regression
- Perceptron
- Support Vector Machines

As such, they cannot be used for multi-class classification tasks, at least not directly.

Instead, heuristic methods can be used to split a multi-class classification problem into multiple binary classification datasets and train a binary classification model each.

Two examples of these heuristic methods include:

- One-vs-Rest (OvR)
- One-vs-One (OvO)

One-Vs-Rest for Multi-Class Classification

One-vs-rest (OvR for short, also referred to as One-vs-All or OvA) is a heuristic method for using binary classification algorithms for multi-class classification.

It involves splitting the multi-class dataset into multiple binary classification problems. A binary classifier is then trained on each binary classification problem and predictions are made using the model that is the most confident.

For example, given a multi-class classification problem with examples for each class '*red*,' '*blue*,' and '*green*'. This could be divided into three binary classification datasets as follows:

- **Binary Classification Problem 1**: red vs [blue, green]
- **Binary Classification Problem 2**: blue vs [red, green]
- **Binary Classification Problem 3**: green vs [red, blue]

A possible downside of this approach is that it requires one model to be created for each class. For example, three classes requires three models. This could be an issue for large datasets (e.g. millions of rows), slow models (e.g. neural networks), or very large numbers of classes (e.g. hundreds of classes).

One-Vs-One for Multi-Class Classification

One-vs-One (OvO for short) is another heuristic method for using binary classification algorithms for multi-class classification.

Like one-vs-rest, one-vs-one splits a multi-class classification dataset into binary classification problems. Unlike one-vs-rest that splits it into one binary dataset for each class, the one-vs-one approach splits the dataset into one dataset for each class versus every other class.

For example, consider a multi-class classification problem with four classes: '*red*,' '*blue*,' and '*green*,' '*yellow*.' This could be divided into six binary classification datasets as follows:

- **Binary Classification Problem 1**: red vs. blue
- **Binary Classification Problem 2**: red vs. green
- **Binary Classification Problem 3**: red vs. yellow
- **Binary Classification Problem 4**: blue vs. green
- **Binary Classification Problem 5**: blue vs. yellow
- **Binary Classification Problem 6**: green vs. yellow

This is significantly more datasets, and in turn, models than the one-vs-rest strategy described in the previous section.

The formula for calculating the number of binary datasets, and in turn, models, is as follows:

- (NumClasses * (NumClasses – 1)) / 2

We can see that for four classes, this gives us the expected value of six binary classification problems:

- (NumClasses * (NumClasses – 1)) / 2
- (4 * (4 – 1)) / 2
- (4 * 3) / 2
- 12 / 2
- 6

Each binary classification model may predict one class label and the model with the most predictions or votes is predicted by the one-vs-one strategy.

An alternative is to introduce K(K – 1)/2 binary discriminant functions, one for every possible pair of classes. This is known as a one-versus-one classifier. Each point is then classified according to a majority vote amongst the discriminant functions.

— Page 183, Pattern Recognition and Machine Learning, 2006.

Similarly, if the binary classification models predict a numerical class membership, such as a probability, then the argmax of the sum of the scores (class with the largest sum score) is predicted as the class label.

Classically, this approach is suggested for support vector machines (SVM) and related kernel-based algorithms. This is believed because the performance of kernel methods does not scale in proportion to the size of the training dataset and using subsets of the training data may counter this effect.

The support vector machine implementation in the scikit-learn is provided by the SVC class and supports the one-vs-one method for multi-class classification problems. This can be achieved by setting the "*decision_function_shape*" argument to '*ovo*'.