Random Forest for Unbalanced Multiple Class Classification

Random Forest is a very powerful tool not only for classification but also regression. This summary gives an overview of how to overcome it when working with Random Forest.

***Random Forest and other machine learning algorithms do not perform well with unbalanced data.***

**The Imbalance Problem**

Most algorithms work best when the numbers of observations of each class are roughly equal. When the number of observations of one class exceeds the other problems arise.

Imagine for example a binary classification problem with 999 observations of class *negative* and 1 observation of class *positive*. The classifier would then probably assign all observations to class *negative* because only one out of 1000 observations will be classified incorrectly. This misclassification rate of 0.1% is very low and for the algorithm almost negligible. But looking only at the class positive there are 100% misclassified.

So the great difficulties when dealing with unbalanced data are mainly that the classifier will focus more on the larger group while the smaller group will be neglected and that the overall error rate is not representative for all the individual class error rates any longer. One way amongst multiple others to deal with unbalanced data is to use resampling methods to balance the data. Three different sampling methods are evaluated and compared based on the so-called *comecs* data set.

**The Comecs Data**

The *comecs* data set contains spectral measurements from ten different meteorites and the *substrate* class. The data set is highly unbalanced, since none of the 11 classes are of equal size. There are 1035 observations in total, which seems quite few for 297 variables. The eleven classes are of different sizes, the smallest has 27 observations, the largest class has 240 observations.

The following balancing methods were investigated and analyzed:

* oversampling
* undersampling
* same-size sampling (ntp)

**Oversampling**

In the case of oversampling each class will be blown up by sampling with replacement to the size of the biggest class. This means that every class will have 240 observations.

**Undersampling**

In case of undersampling each class will be sampled down to the size of the smallest class. For the comecs data set this means that every class will have 27 observations.

**Same-Size Sampling**

The so called ntp-method (*n times percentage*) makes sure that the overall size of the dataset will stay the same and each class will be blown up or sampled down to the same proportion of the overall size. So if there are eleven classes, each class will afterwards have the size of 1/11 times the overall number of observations (n). For the comecs data set this means that every class is blown up or sampled down to a size of 1035/11 ≈ 94 observations.

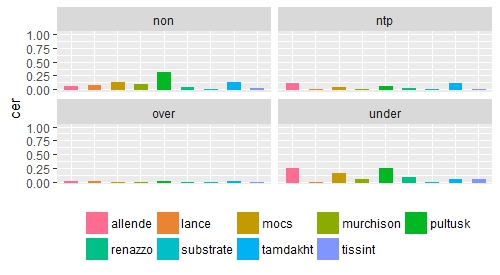
Results

All Random Forests will contain of 600 trees and 58 out of 297 variables are available at each node. Those numbers were carefully determined via a repeated simulation.



Applying oversampling improves the performance based on the overall error rate (oob). Because the numbers of observations are artificially increased by sampling with replacement the observations that were already classified correctly are almost surely again correctly classified and therefore, the error rate will decrease. The effects of applying same-size sampling (ntp) on the performance of the forest are considered to be quite similar to those of applying oversampling, since more classes have to be blown up and only a few have to be down sized.

So oversampling and same-size sampling cause a decrease of the overall error rate. Undersampling, however, leads to an increase. Since all classes, except the smallest one, have to be down sized one risks the loss of important information, hence the bad performance.



Regarding the class error rates (cer), it's conspicuous how balancing affects those especially when applying oversampling and same-size sampling (ntp). The class error rates are more evenly distributed and are related to the overall error rate (oob, which is a very desirable characteristic when it comes to multiple-class classification. Undersampling performs not that well.

It seems that oversampling and in this case also same-size sampling might lead to overfitting, but overall spoken it improves the performance of the classification. Undersampling, however, performs very poorly.

Conclusion

**Regarding the *comecs* data set it can be concluded that oversampling leads to the best results.** The class error rates are much lower. The main problem of imbalance, being that the class error rates are so to say also unbalanced and not reflected in the overall error rate, can be overcome very effectively by oversampling and also same-size sampling. Same-size sampling is almost as good as oversampling sometimes even better in terms of the effects of balancing. This and the tendency of oversampling to cause overfitting lead to the conclusion that same-size sampling may be the most appropriate balancing method for this classification problem. Also, it takes less computing time than oversampling.

Dealing with unbalanced multiple-class classification in general is a hard task. The purpose of the classification, the importance of the multiple classes and the effects of different balancing methods are different for every data set. Balancing methods that are based on a method of random subsampling seem to be more reasonable for multiple-class problems than cost-sensitive learning, which seems to be more suitable for binary classifications.