LINGI2262 - Machine Learning: classification and evaluation

A5 - A Machine Learning Competition

Alexandre Tytgat (22 06 15 00)

Thursday 7th May, 2020

I. Introduction

The goal of this project was to build the best possible classification model on a training set to predict the (undisclosed) class labels on a given test set, and to predict the actual classification performance your model will have on the test set.

II. Design choices

The training set was split into two distinct parts : the actual training set with 80% of the data and a validation set with 20% of the data. The validation set was used to evaluate the performances of the models created using the actual training set. The metric of interest for this task is the BCR.

A. Preprocessing

First, all the NA values were transformed into zero values. This choice was made because it is generally easier to handle numerical values. Furthermore, it was decided to set them to the zero value because it is hard to predict their real expression values. An alternative choice could have been to set all unknown gene expression level as the mean of all their known expression level. However, since most of them are zeros, the mean would be close to zero anyway. In addition, after this transformation, all the genes that were not expressed in any samples (=column of zeros) of the training set were removed from the training set, the validation set and the test set (around 4000 gene features).

Secondly, in order to improve the computing time and to fight the curse of dimensionality, the PCA method was applied to the gene features using the prcomp function [1]. This method reduces the number of variables by transforming the old features into new uncorrelated features which possess all the variance from the old ones. Furthermore, to improve the generalization, only the 50 most important principal components were kept and normalized.

Thirdly, all the alphabetical categorical features were transformed into numerical factors. This was a necessary transformation as some methods are not able to handle data expressed as characters. Plus, there is no loss of information as this transformation is easily reversible.

Finally, it was observed that the class are strongly imbalanced (around 90% are classified in the first class -1 and the rest in the second class +1). This can be a huge issue because a classifier trained on such a set may find it more difficult to learn well the differences between the two classes . It might be able to classify well samples from the majority class since it is the class it knows the best. However, this will likely lead to poor prediction on the minority class. The function SMOTE [2], [3] was used in the training set. SMOTE works as follow : first it uses the k-nearest-neighbors of samples of the minority class to create new samples of this class. This is known as oversampling. Then, it undersamples the majority class to have a more balanced training set. SMOTE allows the user to choose the number of nearest neighbors but also the tuning of the oversampling and undersampling.

B. Models

The caret package [4] was used to train several models using bootstrap632 over 10 folds to find the optimal metaparameters that optimize some metric. Caret gives the possibility to use a custom metric [5] thus the metric chosen was the BCR (or in some case the sensitivity).
Several learning algorithms were tested, mostly different implementations of random forest, but only six were used in the final model.

- Gradient boosting machine (gbm) : random forests generated by stochastic gradient boosting machines - utilising the concept of regularisation to prevent

overfitting the individual decision trees to the training set [6]
- eXtreme Gradient Boosting (xgbTree) : random forest generated with gradient boosting [7]
- ranger : a fast implementation of random forests or recursive partitioning, particularly suited for high dimensional data [8]
- xgbDART : random forest generated with gradient boosting [9]
- xgblinear : gradient boosting
- ada :fit a variety stochastic boosting models for a binary response [10]

Random forest were chosen because they are great learner for imbalance data [11] Once they were all trained, the ensemble learning method was used to make the final prediction. This method collects the predictions of all the models and yields the prediction that received the most votes among the models. The results varied due to random factors but a BCR of at least 0.7 was measured on the validation set.

III. Conclusion

In summary, the data was first preprocessed then reduced to a lower number of features using PCA. Then the SMOTE method was used to bring more balance in training set. Finally several models were optimized and combined to yield the final prediction using ensemble voting.

References

[1] Principal Component Analysis (PCA) 101, using R https://towardsdatascience.com/principal-component-analysis-pca-101-using-r-361f4c53a9ff105 199–202

[2] SMOTE function from the DMwR package https://cran.r-project.org/web/packages/DMwR/index.html28 581–600

[3] Classification: Get the Balance Right https://datawookie.netlify.app/blog/2018/04/classification-get-the-balance-right/

[4] A Short Introduction to the caret Package https://cran.r-project.org/web/packages/caret/vignettes/caret.html

[5] Caret: using custom metric https://stats.stackexchange.com/questions/210361/r-caret-train-rfe-optimize-for-positive-predictive-value-instead-of-accuracy-o

[6] gbm method https://rstudio-pubs-static.s3.amazonaws.com/240020$_f$8bb6a727e1d4c99842659d34a4c62c2.$html$

[7] xgboost https://cran.r-project.org/web/packages/xgboost/vignettes/xgboost.pdf

[8] ranger method https://www.rdocumentation.org/packages/ranger/versions/0.12.1/topics/ranger

[9] xbgDart method https://xgboost.readthedocs.io/en/latest/tutorials/dart.html

[10] sgbLinear method https://www.rdocumentation.org/packages/ada/versions/2.0-5/topics/ada

[11] Using Random Forest to Learn Imbalanced Data https://statistics.berkeley.edu/sites/default/files/tech-reports/666.pdf