Data Challenge

Yves Greatti

January 26, 2019

1 Problem Statement

The assignment is to investigate and analyze the attached data, producing a model predicting the ”response”. The first column of the provided data is the binary variable ”response”. The 16,562 other columns are binary columns that can be used to predict the response.

2 Preliminary Exploration into the Data

2.1 Cleaning

There are 530 rows of data and every feature is binary. The very large amount of features does not allow to easily plot any distributions. I need to identify the most important features to do so and I will come back to more in depth analysis once I have identify these features. The data is very cleaned, there are no missing data (identified using plain numpy statements and also using the python package msno). The data is between 0 or 1 and with a quick check I do not see any other values possible for any of the binary features, so no unexpected values are found.

2.2 Visual Inspection Of Features

I identified an imbalanced data set regarding the ’response’ variable. The negative class, ’0’, outnumbers the positive class ’1’ by a factor of about 4:1.

Using Principal Components Analysis and plotting the cumulative sum of the explained variance, I saw that we will need about the first 300 components to describe close to 80% of the variance.

In addition, visualizing the data against the two first components shows that the two type of responses (0 or 1) are very close to each other indicating that a classifier might have difficulties to classify them.

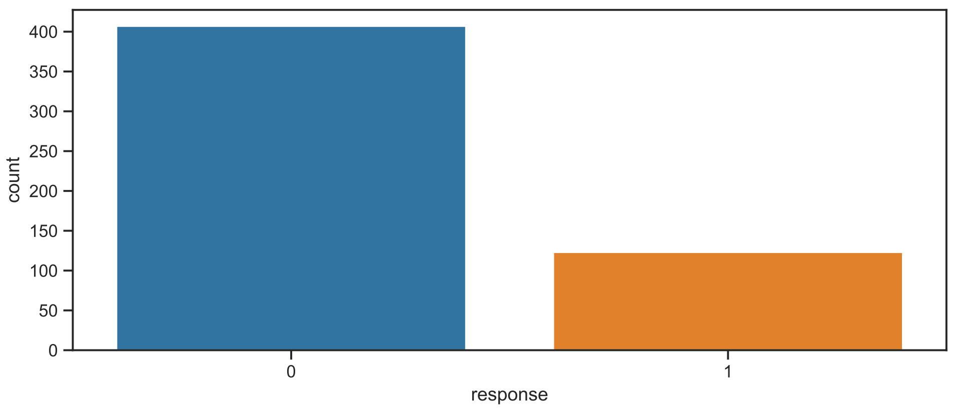


Figure 1: Imbalanced data: four times more ’0’ responses than ’1’ responses

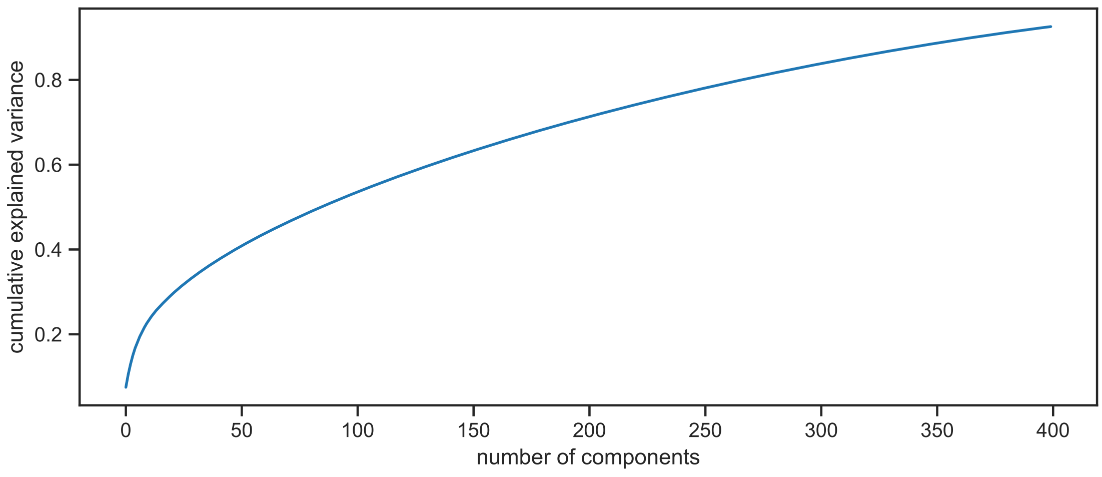


Figure 2: Cumulative explained variance

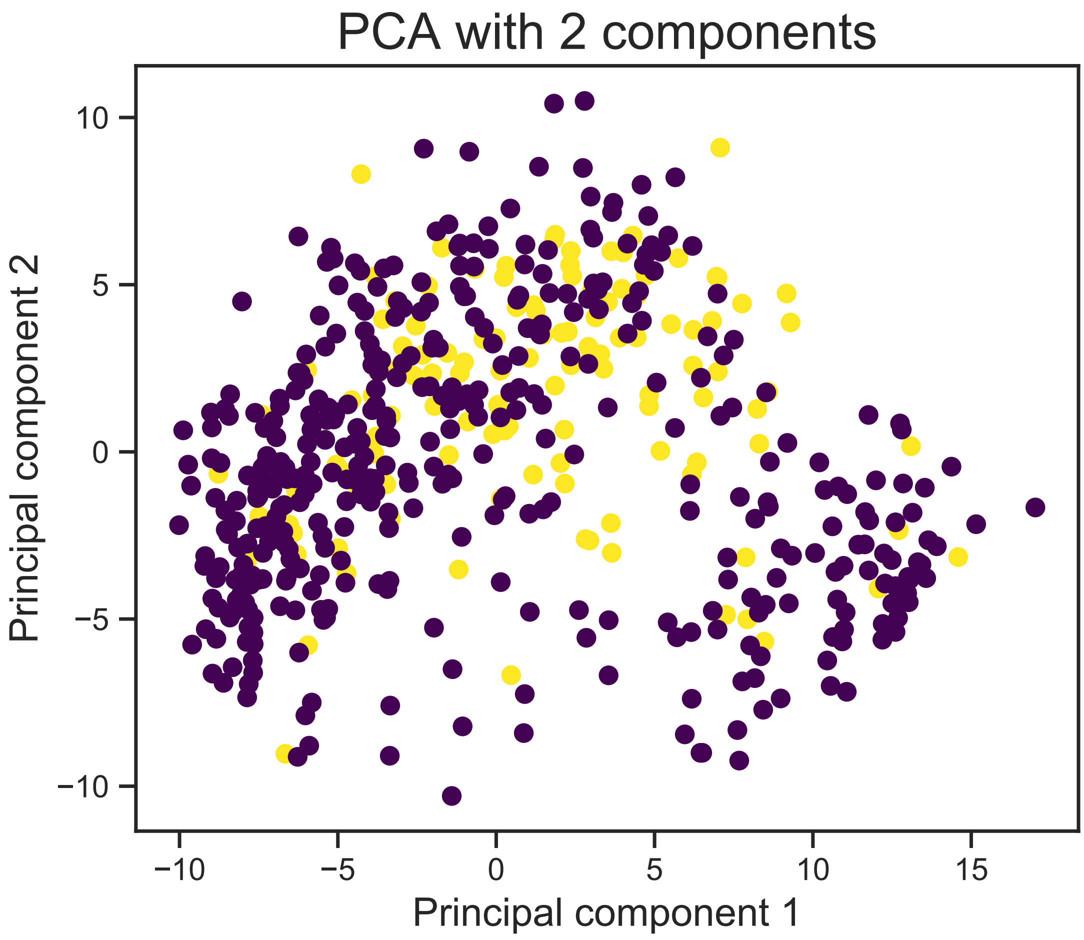


Figure 3: PCA With Two Components

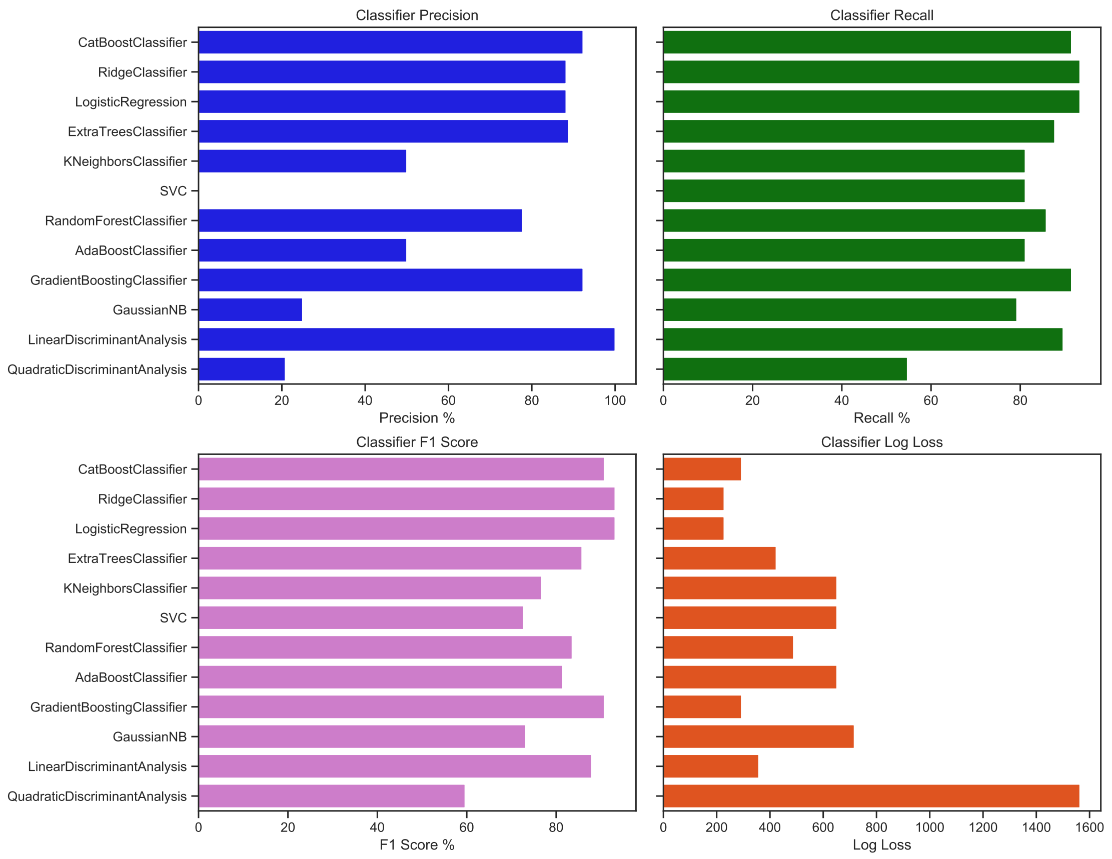


Figure 4: Classifiers macro-averaging performance comparison, overall they had similar performances with specific differences depending the metric under consideration.

3 Model Selection

First I used out of the box a random forrest classifier, and interestingly the classifier had better performances than expected. Then I evaluated different classifiers using the imbalanced set, and once I identified a performing model I addressed the imbalanced issue.

I selected a combination of linear classifiers and ensemble models. At the exception of the SVC classifier with rbf kernel, they all had, in a ballpark, similar performances. Now depending on the problem to solve, more obvious choices could have been possible. For example, if we wanted a high accuracy or recall for the positive '1' class, we would have selected a ridge or logistic classifier. I assumed that it was critical to have a good accuracy and recall for the prediction of the positive or '1' cases. The Linear Discriminant Classifier, LDA, had very good performances too and it seemed logical to use it. At the same time, the CatBoost classifier was not far behind the ridge and logistic regression classifiers, and had a better recall compared to the LDA classifier for the '1' class. Also it is specifically designed to handle categorical variables and, by association, binary variables. So my objective was by selecting it, and tuning its hyper parameters, to be in good position to have a very performing classifier.

The ROC curves showed that many classifiers had good prediction "power". But when there is a moderate to large class imbalance, ROC curves could be deceptive.

> "If the proportion of positive to negative instances changes in a test set, the ROC curves will not change. Metrics such as accuracy, precision, lift and F scores use values from both columns of the confusion matrix. As a class distribution changes these measures will change as well, even if the fundamental classifier performance does not. ROC graphs are based upon TP rate and FP rate, in which each dimension is a strict columnar ratio, so do not depend on class distributions."
>
> — ROC Graphs: Notes and Practical Considerations for Data Mining Researchers, 2003.

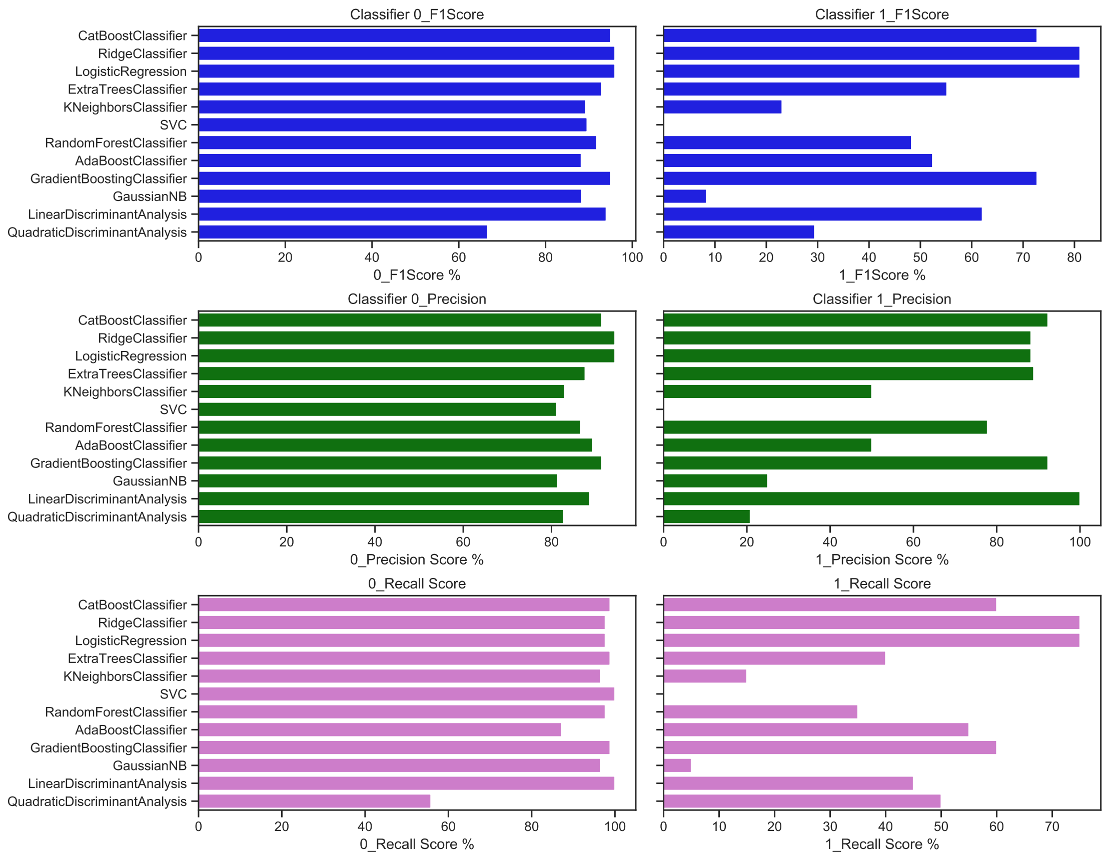


Figure 5: Classifiers per-class prediction performance comparison, case by case there were differences. For a given classifier that had overall good performances for predictions about the class '0' or '1', I decided to favor the performances of the predictions regarding class '1' and selected the CatBoost classifier which had good precision and recall for the class '1'.

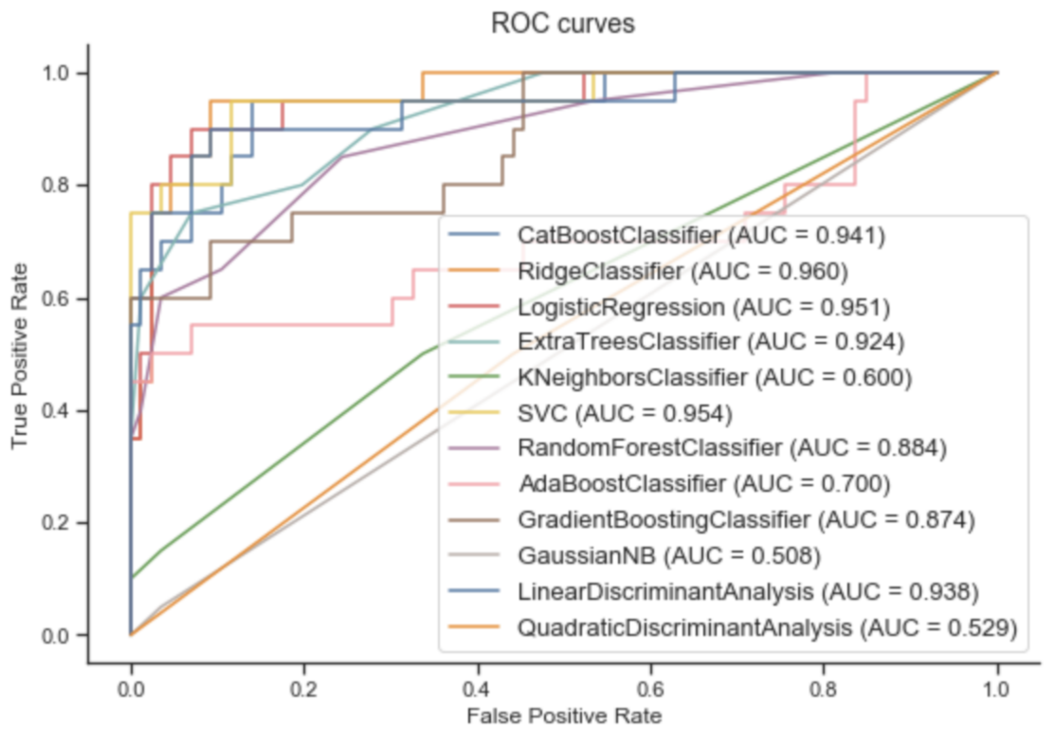


Figure 6: ROC curves for the trained classifiers show that top-performers classifiers were: CatBoostClassifier, RidgeClassifier, LogisticRegression, ExtraTreesClassifier or LDA classifier.

Since in the case of imbalanced datasets, ROC curve may be misleading, I also looked into the precision-recall curve of the CatBoost classifier. The precision-recall curve confirmed that the CatBoost classifier was a performing model.

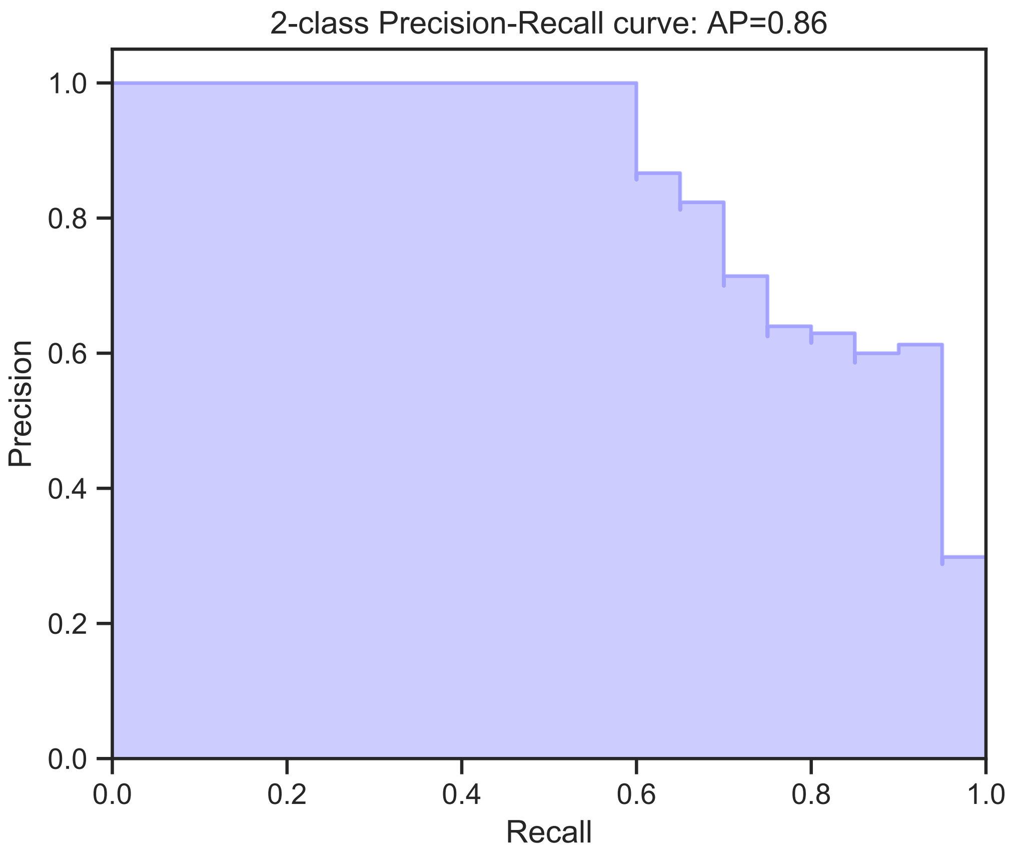


Figure 7: Inverted U shaped curve showed that CatBoost classifier had a good predictive performance.

The confusion matrix for the CatBoost classifier showed that out of 106 test sample points, 40% of the positive cases ('1') were not classified as positive (False Negatives). I later tried to address this large amount of false negatives. Looking at the feature importance rank given by the CatBoost classifier, the variable V15261 seemed to be important to make accurate predictions of the response variable.

4 Handling Imbalanced Data Set And Modeling

There are many ways to handle classification for imbalanced data sets. But to have an estimation how this imbalance affected the CatBoost classifier, I started by oversampling using the SMOTE method from the python package imblearn . SMOTE creates synthetic points between neighbors of the minority class. To avoid a data "leakage", I created two random training and test subsets, and from the training subset I used SMOTE to balance the data. Then I used the balanced data set (sampled data set) to train the CatBoost classifier. Finally, I measured the performance of the same classifier on the test subset. With the sampled balanced data set, the CatBoost classifier did not have improved performances indicated by the logloss cross-validation curves and the confusion matrix. The logloss error on the validation set increased after 15 iterations. And accuracy, precision and recall were fluctuating suggesting an overfitting. I decided to use the overfitting detector from CatBoost .

Before building each new tree, CatBoost checks the number of iterations since the iteration with the optimal loss function value. And the model is considered overfitted if the number of iterations exceeds the value specified in the training parameters (number of trees=200, learning rate=0.5).

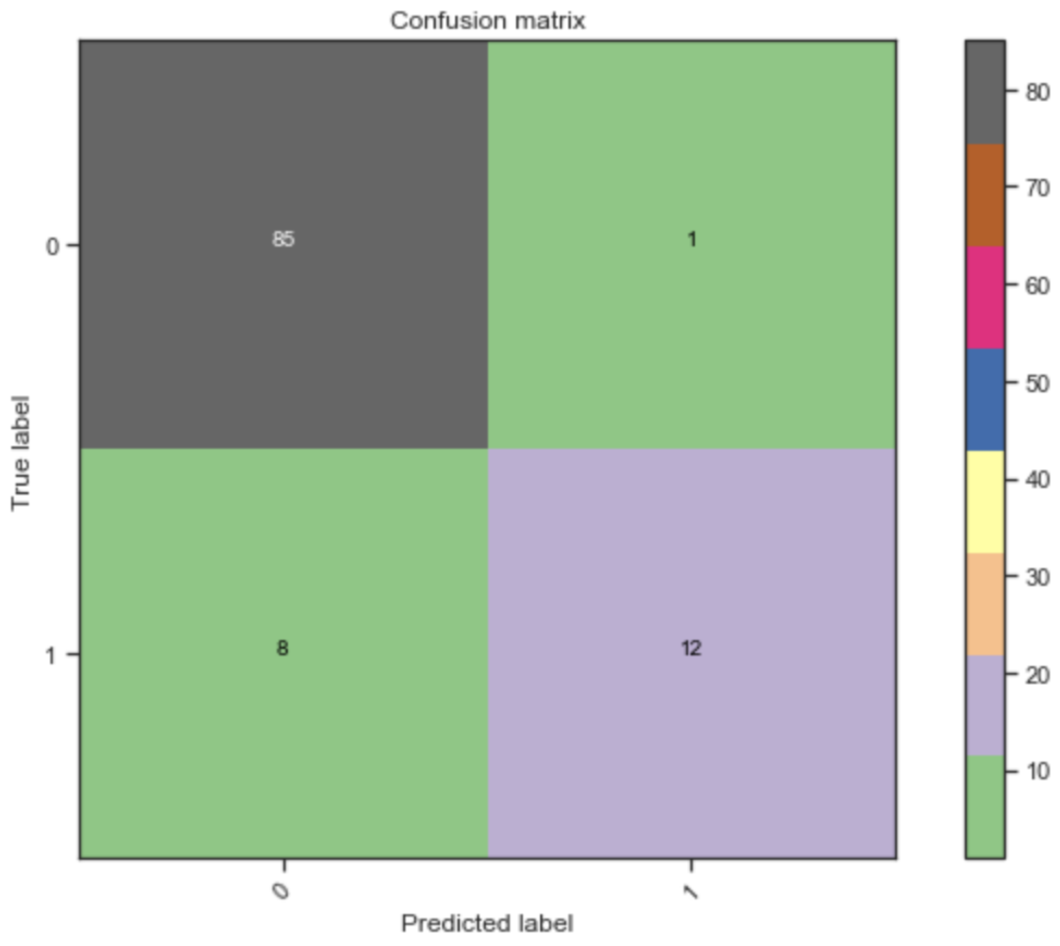


Figure 8: Non-tuned CatBoost classifier: 40 % of False negatives.

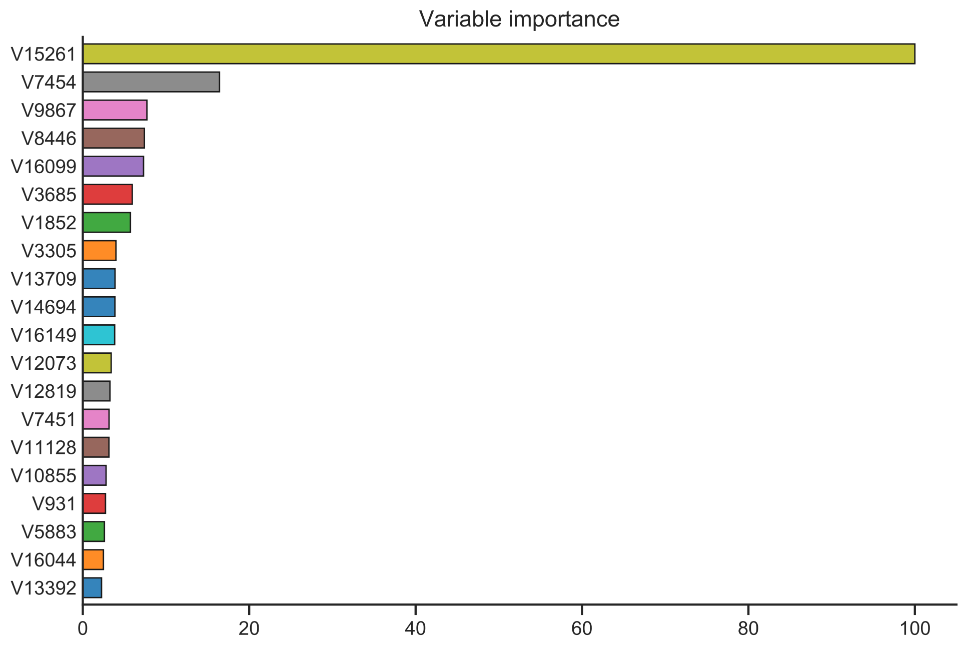
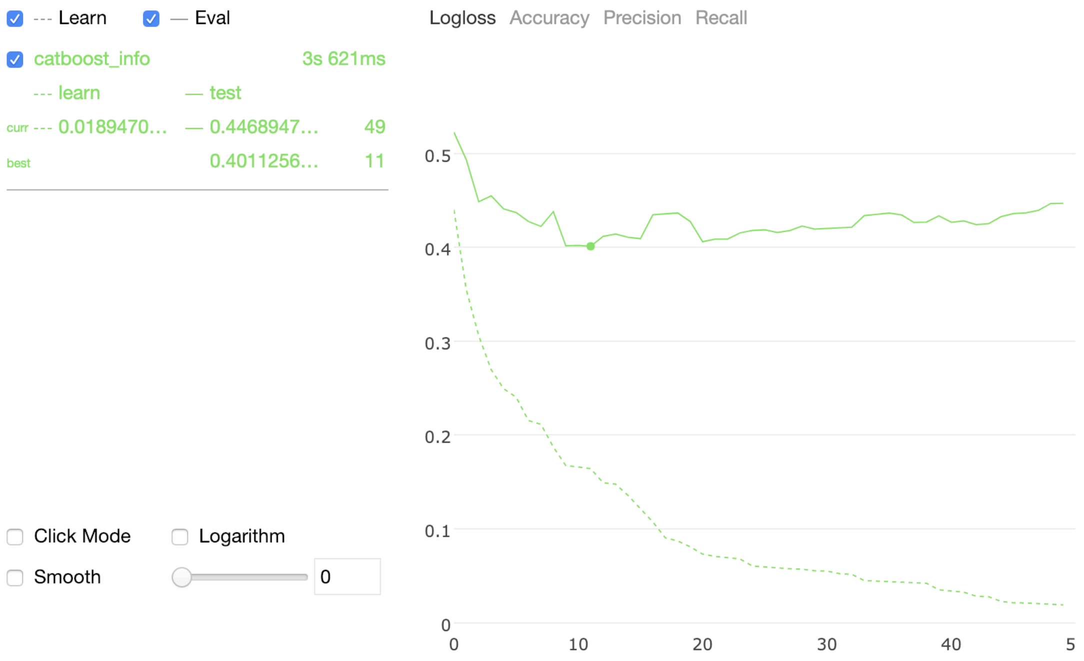


Figure 9: V15261 had an overweight importance.



```
catboost_info                 3s 621ms
     --- learn        — test
curr --- 0.0189470…   — 0.4468947…   49
best                    0.4011256…   11
```

Figure 10: After oversampling, logloss curves for both the training and validation sets decreased nicely up to, about 15, and then the validation loss started to increase.

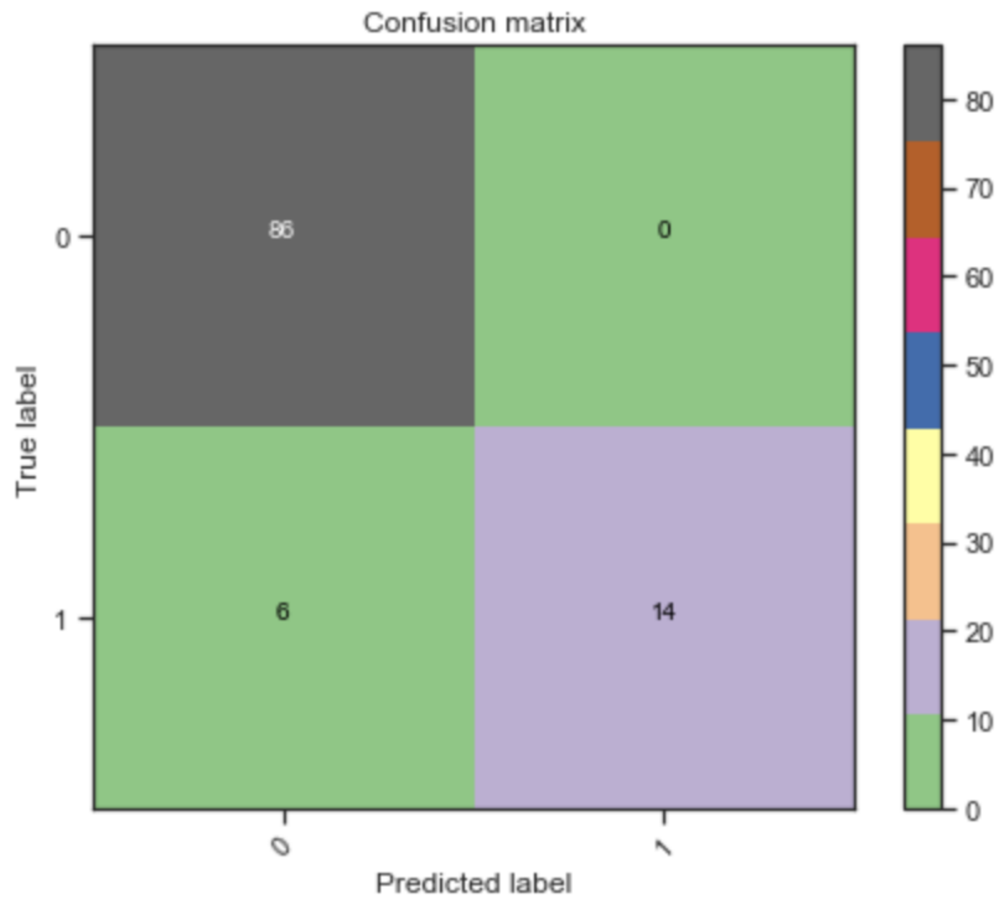


Figure 11: After tuning the hyper-parameters using a grid search: only 30% of false negatives were remaining.

5 Grid Search And Stratified KFold

Performing a grid search with the cross-validated number of trees of 12, I found that the optimal values for depth, l2 leaf regularization were:

- trees:12
- depth:4
- l2 leaf regularization:9
- learning rate:0.5

I used also a StratifiedKFold from sklearn and a grid search on the critical CatBoost classifier hyper parameters. The cross-validation object is a variation of KFold that returns stratified folds. The folds are made by preserving the percentage of samples for each class. I found that, the best parameters for the number of trees, the depth of the trees, l2 leaf regularization or the learning rate were:

- trees:12
- depth:10
- l2 leaf regularization:9
- learning rate:0.5

However the performance of the CatBoost classifier did not improve. I tried also to select a subset of the ten most important features and train a classifier with this subset (as a side note, with this approach you may introduce a bias and overfit), but the performances decreased and I abandoned the idea. So the best performance was reached with a balanced data set produced by SMOTE and a model with 50 trees and a learning rate of 0.5. The false negatives were left at 30%.

6 Conclusion and Further Exploration

In conclusion, more research could be done to handle the imbalance of the classification, for example by using different sampling methods. However, in the absence of specific requirements regarding the problem to solve, the performances reached through experimentation, can be considered good enough.

7 References

1. `http://www.blogspot.udec.ugto.saedsayad.com/docs/ROC101.pdf`
2. `https://machinelearningmastery.com/roc-curves-and-precision-recall-curves-for-classification`
3. `https://medium.com/usf-msds/choosing-the-right-metric-for-evaluating-machine-learning-models`