Assignment 1 CDA

Tim van Rossum, 4246306
Michiel Doesburg, 4343875

May 13, 2018

All code can be found on `https://github.com/Michieldoesburg/cyber_data_analytics`

1 A visualization of the data

For the visualization part of this assignment, we first started out by making bar plots of different kinds, but eventually settled for the scatterplot as seen in figure 1. This scatterplot uses the amount of Eurocent spent per transaction, to allow for better comparisons to be made, as there were five different currencies in the dataset. Exchange rates of April 25, 2018 were used. As can be seen from the scatterplot, there are no fraudulent transactions where more than 800 Euro was spent, while there are many benign transactions where more than 800 Euro was spent. Other visualizations we made included the ratio of fraudulent versus non-fraudulent transactions per country and per card issuer. Notable observations were that nearly all fraudulent transactions come from Mexico, Australia, the United Kingdom and Sweden (over 90% percent of all fraudulent transactions happened in these four countries).

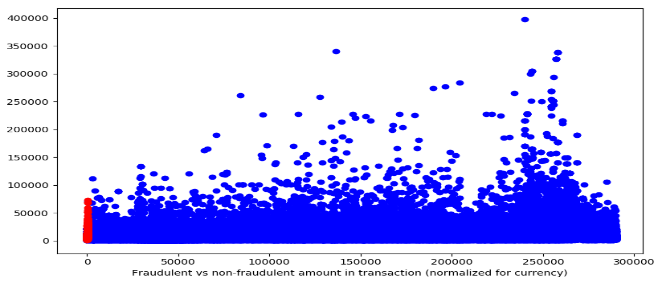

Figure 1: A scatterplot of the amount of money spent in the sampled transactions. A red dot indicates a fraudulent transaction (these are only at the very left of the plot due to very little fraudulent transactions existing), while a blue dot indicates a benign transaction. Refused transactions are treated as benign in this case, as we needed to show all data points and we wanted to show that all confirmed fraudulent cases comply with our observation.

2 Applying SMOTE to the data

Because we used Python for this assignment with `scikit-learn`, we used SMOTE as implemented in the package `imblearn`. The `fraud_detection.py` script, available on Brightspace, already preprocessed the data in such a way that applying SMOTE to it was very easy, as the implementation only needed the data and the class labels, and both were already generated by the script. The general steps of preprocessing are:

- Remove data with the "Refused" label (as that is data where we cannot be certain whether or not it is fraudulent)
- Remove the ID and booking date of the transaction, as these are not necessary.
- Transform the mail identifiers, IP identifiers etc. to simply the number that they use.

The classifiers that we used were the random forest classifier, the 5-NN classifier, and the logistic classifier. These were chosen to represent both linear and non-linear classifiers, as well as parametric and non-parametric classifiers (as NN is deemed a non-parametric classifier). The ROC curves are shown in figure 2. As can be seen, the random forest classifier can get fairly high true positive rates without having high false positive rates (up until a certain true positive rate, where it increases a lot), making it a good choice for fraud detection.

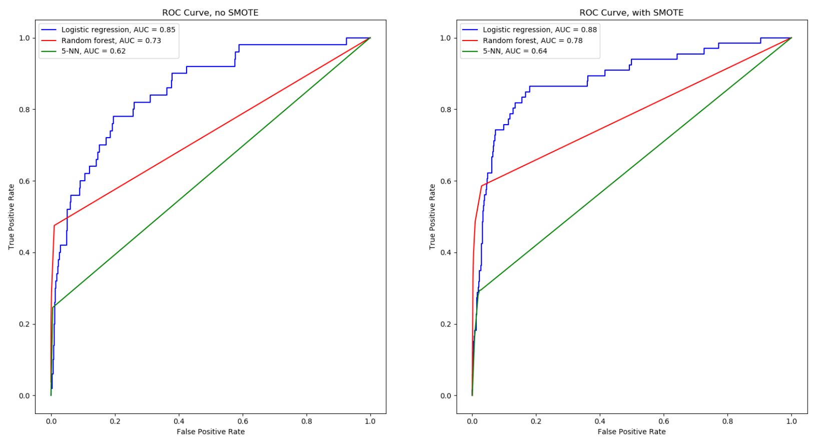

Figure 2: The ROC curves, the one on the left does not use SMOTE to oversample training data, and the one on the left does. As can be seen, the area under the curve (AuC) is slightly higher while using SMOTE on training data. In general, this means that classification is more accurate when using SMOTE.

3 The black box and white box classifier

In this section, we will compare a very simple white box classifier that we wrote with a black box classifier for detecting fraudulent cases in the data. We will also discuss the applicability of both methods in practice.

3.1 The white box classifier

For the white box classifier we took a look at several features of the data: issuer country code, transaction variant code, shopper interaction, card verification, CVC response, account code and card issuer identifier. We wrote a small script which, for the fraudulent transactions only, gathered the ranges of values that these features take. In the case of the issuer country code, only Mexico, Australia and the United Kingdom were included in the range as these three countries make up almost 90% of all the fraudulent transactions. A transaction *t* is labelled as fraudulent if for every feature in the set of features that we use (so, issuer country code, transaction variant code, shopper interaction, card verification, CVC response, account code and card issuer identifier), it has a value that matches the value in the set of values that we computed, corresponding to that attribute, and if the amount of money spent in *t* is less than 800 Euros. The implementation of the white box classifier can be seen in `white_box.py`, in the `white_box` folder. This folder also contains the other functions that the classifier uses. As this is a white box classifier with simple decision rules, no training will be performed (although one could argue that determining the values of the features selected of the fraudulent cases could technically be seen as learning), and the classifier will classify data directly. Although this is a very simple approach, it seems to work quite well on the test data. This approach produces the following results:

TP: 155
FP: 5
TN: 236686
FN: 190

While these results look very strong, the algorithm is severely overtrained. Making it likely to perform poorly on entirely new data.

3.2 The black box classifier

For the black box classifier, we will be using the random forest classifier, as this classifier tended to perform better as seen in the ROC curves of figure 2, especially on SMOTEd data. It is also a classifier that has been shown to perform better in a study done by Bhattacharyya, Jha, Tharakunnel and Westland [1]. In order to train and test this classifier on the data, we will be using ten-fold cross validation, to get a clear picture of how this classifier would perform in practice. We will also be using SMOTE to balance the training data for every round. The performance for this approach is as follows:

TP: 23
FP: 157
TN: 236196
FN: 322

3.2.1 Performance and evaluation

The performance of the black box algorithm is slightly worse than the performance of the white box algorithm, but the white box algorithm is also very simple, and an adversary only has to commit fraud using a different card (that has not been used in a fraudulent transaction before) to prevent being detected. Another way to bypass detection would be to commit fraud outside of Mexico, Australia and the United Kingdom as the white box classifier does not check if transactions from other countries are fraudulent. As such, we predict that the white box algorithm will perform very poorly in practice. Contrasting this, the black box algorithm will most likely also perform well in practice, as we do not assume anything about the fraudulent transactions.

4 Bonus task

As a way to improve our classifier even further, we tried to convert all the money spent in the transactions to Euros, using the exchange rates, once again, from April 25, 2018. This resulted in the following performance:

TP: 26
FP: 171
TN: 236182
FN: 319

As can be seen, using converted currency amounts resulted in slightly more cases correctly being labelled as fraudulent, but the false positive rate is higher as well, indicating that the performance is actually worse when using converted currency amounts.

Another thing what we did is not use SMOTE to balance training data while testing the performance of the random forest classifier using ten-fold cross-validation. This was mainly done to test the impact of SMOTE. This resulted in the following performance:

TP: 3
FP: 5
TN: 236348
FN: 342

Interestingly, not using SMOTE resulted in a far lower false positive rate, which is the main performance metric that we are interested in for developing fraud detection systems. But it should be noted that not using SMOTE causes the random forest classifier to miss a lot more cases as well, making is practically unusable.

References

[1] Siddhartha Bhattacharyya, Sanjeev Jha, Kurian Tharakunnel, and J Christopher Westland. Data mining for credit card fraud: A comparative study. *Decision Support Systems*, 50(3):602–613, 2011.

All code can be found on `https://github.com/Michieldoesburg/cyber_data_analytics`