Cyber Data Analytics (201500037)
Assignment 1

Antonis Papadopoulos, Luigi Coniglio (Group 42)

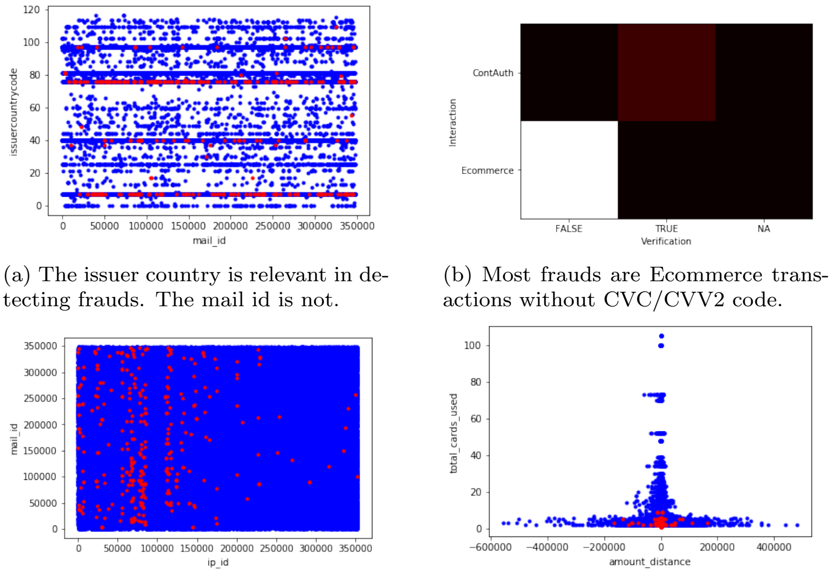
May 13, 2018

1 Visualization task

Due to the large amount of data it is often necessary to filter the samples in order to speed up the learning and being able to build a model.

For this part of the assignment we analyzed the data provided in order to understand which features characterize the most the type of a transaction (fraudulent or not) and which features are irrelevant for identifying the class to which a transaction belongs.

For this assignment we used Python and *Matplotlib*, a Python plotting library, to visualize the data.



(a) The issuer country is relevant in detecting frauds. The mail id is not.

(b) Most frauds are Ecommerce transactions without CVC/CVV2 code.

(c) Some ranges of IP addresses are more fraudulent than others.

(d) The total amount of cards used and the frequency of usage are good discriminants.

Figure 1: Visualizations of some interesting features (fraudulent transaction are marked in red)

Figure 1 shows the plots of some interesting features. Figure 1.a shows that fraudulent transactions are more likely to be issued from some particular countries, figure 1.b is an heat-map showing that most of the fraudulent transactions are non-verified online transactions and very few are monthly subscriptions, figure 1.c shows that fraudulent transactions are more likely to come from some particular ranges of IPs. Finally, figure 1.d shows two additional features that we created, those features are discussed in section 3.1 when we will discuss about the preprocessing of the data.

2 Imbalance task

In the provided dataset the large majority (over 99%) of the data is benign. To understand why this could be a problem for the learning let's imagine a classifier which flags all transactions as non fraudulent. While such classifier would not be able to distinguish between benign and fraudulent transactions, it will nonetheless score more than 99%. For this reason we need a way to balance the number of elements of each class. This is could be done by increasing the number of fraudulent transactions (oversampling) or decreasing the number of benign transactions (under sampling).

For this part of the assignment we used Synthetic Minority Over-sampling Technique (SMOTE) an oversampling algorithm used for when we wish to build classifiers from imbalanced datasets.We applied SMOTE on three classifiers and observed their performances on the SMOTEd and UNSMOTEd data.

Table 1 shows the score of such classifiers on the original as well as the oversampled data. As we can see all classifier score very high on the UNSMOTEd data however, as we discussed before, this score is biased by the enormous amount of benign transactions if compared to the amount of fraudulent transactions.

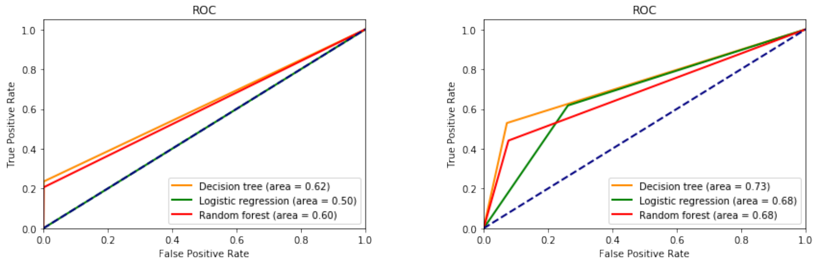
Table 1: Scores on SMOTEs and UNSMOTEd data

	Score (Original)	**Score (SMOTE)**
Decision tree	0.9980566117448246	0.9271229404309252
Logistic regression	0.9985635825940008	0.7377693282636248
Random forest	0.9983100971694128	0.9221377270806929

The ROC curves plotted in figure 2 confirm our hypothesis, indeed even if the classifier trained on the SMOTEd data seem to score less than those trained on the original dataset they perform much better in discriminating benign transactions to fraudulent ones.

3 Classification task

In this section we will start by analyzing the way the credit card transactions were pre-processed. Then, we will examine two learning algorithms, namely a Support Vector Machine(SVM) classifier and a Decision Tree regressor, since both these algorithms have been used effectively for this scenario[1]. Last but



(a) ROC curve after training on UN-SMOTEd data.

(b) ROC curve after training on SMOTEd data.

Figure 2: ROC curves of the classifiers

not least, we will measure the performance of each algorithm based on the post-processing method used.

3.1 Preprocessing

The credit card transaction data given were parsed into a Pandas dataframe, due to its convenience for certain operations. Transactions labeled as 'Refused' were excluded from the analysis, as well as transactions with missing values for important features, like mail id and bin. From the existing features, only the label, the ip id and the issuer country of the transaction(labeled as a numeric value) were used. Two additional features were also created, namely the total cards used for the mail id of each transaction, and the distance between the amount of the transaction and the average amount for the customer corresponding to a specific mail id. The transactions are also shuffled before proceeding.

3.2 Using a Support Vector Machine(SVM)

Support Vector Machines(SVM) are non-probabilistic binary linear classifiers used for supervised learning. We used the SVM classifier provided by the scikit-learn library as our black-box learning algorithm to measure performance on identifying fraudalent credit card transactions on the filtered features. To do that, we used 10-fold cross validation, so we split the dataframe containing the transactions into 10 sets of same size. We trained 10 different SVM classifiers on the 10 different sets consisting of 9 of those 10 sets and, then, tested them on the remaining set. SVMs are lightly affected by unbalanced data, which is why the classifiers were trained on un-SMOTED training sets. The average accuracy was 0.93245400500780.

Fold 1	Fold 2	Fold 3	Fold 4
0.934136037178	0.930798479087	0.930756231517	0.934854245881
Fold 5	Fold 6	Fold 7	Fold 8
0.929488804394	0.930967469371	0.936037177862	0.934516265315

Fold 9	Fold 10
0.929612573408	0.933372766065

Black-box learning algorithms are not that good in practice, however, as there is always the possibility for false positives and we need to be able to identify and understand why the algorithm flagged a transaction as fraudulent. For that reason, we shifted our focus on a white-box learning algorithm which could reach comparable performance.

3.3 Using a Decision Tree

As a white-box solution, we used the Decision Tree regressor from the scikit-learn library. Decision trees differ from SVMs since we can identify easier why a certain transaction got labeled as fraudalent/bening by following the tree structure which can be exported. Figure 3 shows an example of such structure. In that structure, leaves represent the class labels(fraudalent or benign) and branches represent conjuctions of the features and are based on each feature's gini index which is a measure of statistical dispersion.

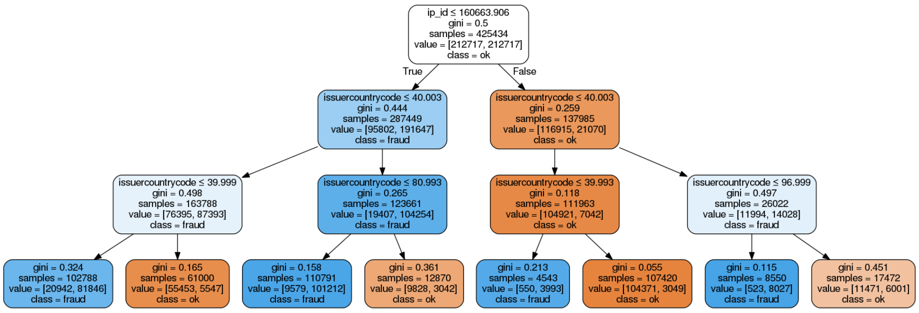


Figure 3: With a decision tree can easily understand why a transaction is flagged as fraudulent. For this reason we consider decision trees as whitebox models.

As with the previous classifier, 10-fold cross validation was used to train and measure the performance of the Decision Tree. This time around, the training set from each fold was SMOTed before training each regressor. In order to measure performance, every transaction from every test set was assigned a numeric value based on the probability of it being fraudulent by the corresponding decision tree regressor from the fold, and was appended into a new pandas dataframe. After 10 folds, the dataframe has every transaction from the original set and an extra prediction column with the assigned prediction from one of the decision trees that was trained during one of the folds.

3.4 Postprocessing

In order to minimize the False Positives, we ranked the predictions based on their regression value. The higher the value, the higher the probability the transaction is fraudulent. Based on that assumption, we sorted the dataframe based on that prediction value on descending order. Then, we iterated through the transactions and counted until 100 True Positives were identified. The rest of the transactions, up until that point, were labeled as False Positives. Due to limited computational resources, we were only able to do that for the Decision Tree regressor(Each SVM classifier took 24hours to train for each fold, so training a regressor would take a similar amount of time). The final performance was then measured.

Subsets	True Positives(TP)	False Positives(FP)
Until 100TP	100	9220

So, the Decision Tree regressor had a precision of 0.01072961373. In the credit card fraud scenario, precision is the most important measure since we want to minimize the number of benign transactions labeled as fraudulent(False Positives).

References

[1] Sahin, Yusuf & Duman, Ekrem. (2011). *Detecting Credit Card Fraud by Decision Trees and Support Vector Machines.* IMECS 2011 - International MultiConference of Engineers and Computer Scientists 2011. 442-447.

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