Assignment 1 - Cyber Data Analytics

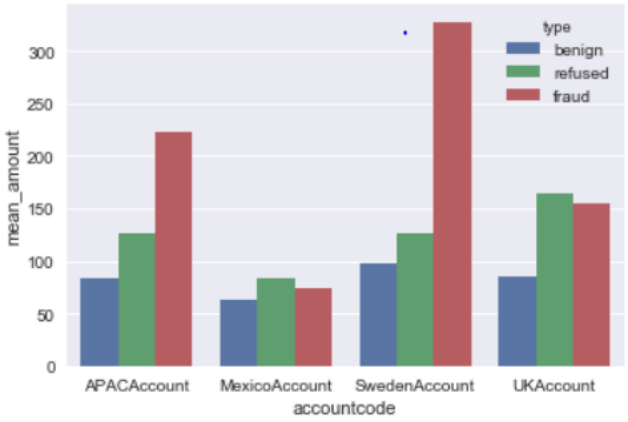
Ruben Groot Roessink (s1468642),
Christiaan van den Bogaard (s1594273)

2017, University of Twente

May 13, 2018

1 Visualisation Task

After playing around a bit with the data we concluded that there is quite an interesting relationship between the average amount (in euros) of the transactions belonging to the country where the transaction was made (accountcode or Merchant's code in the data set). We can see that the average amount of a fraudulent transaction for something bought in Sweden is much higher than average amount of a benign item bought in Sweden, as shown in the figure. A similar relation can be concluded for payments made in Asia-Pacific (APAC) (+>50%). It can also be concluded that the average fraudulent transaction is higher (in euros) than the average benign transaction overall.



The data points that contain the value 'Refused' in the end need to be classified by a (Machine Learning) algorithm as either benign or fraud. Meaning that this will be the so-called test data. The hypothesis is that the 'Refused'-dataset in APAC transactions contains relatively more benign values than fraud values, as the value of the green bar is closer to the value of the blue bar than to the red bar.

With the same reasoning the following hypothesis is set out:

Country	Hypothesis
APAC	Refused data set will contain relatively more (actual) benign transactions
Mexico	Refused data set will contain relatively more (actual) fraudulent transactions
Sweden	Refused data set will contain relatively more (actual) benign transactions
UK	Refused data set will contain relatively more (actual) fraudulent transactions

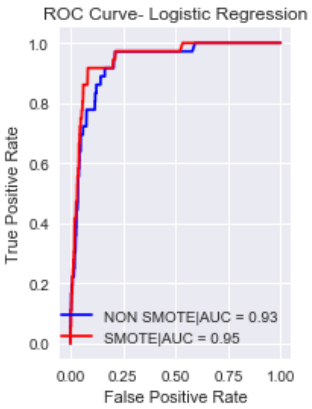
2 Imbalance Task

Three different machine learning algorithms were used for this assignment. First we have a Logistic Regression (LR) based learner, a learner based on a Random Forest (RF) and a learner which uses k-nearest neighbors (kNN) to conduct its business. Using SMOTE [NVC02], or Synthetic Minority Over-sampling Technique will probably be better for the AUC value (Area under the (ROC)[Faw05] curve) as this would mean that the specific algorithm (with SMOTE applied to the training data before the algorithm is trained) will have a higher amount of True Positive values as opposed to False positive values.

We changed things a bit to what might sound as the obvious way. We did this as it was easier to implement and showed better differences between SMOTEd and UNSMOTEd data. In our graphs, we defined a True Positive in the sense that the algorithm determines a fraudulent transaction to be fraudulent, a False Positive is a non-fraudulent transaction classified as fraudulent, and so on. So we switched the values in the confusion matrix.

All code and methods can be found in the Jupyter Notebook on Github [the].

2.1 Logistic Regression

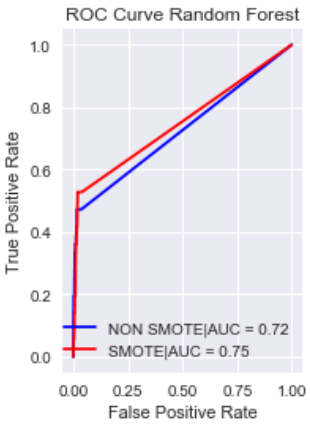


When running the Logistic Regression algorithm on the UNSMOTEd/SMOTEd data we got the following results: (The UNSMOTEd results are represented by the first value in the table, the SMOTEd results by the second).

	Predict Fraud.	**Predict Benign**
Actual Fraud.	0 / 11	36 / 5
Actual Benign	1 / 1828	23667 / 21840

What we can conclude from this is that the algorithm is better in finding true fraudulent transactions (0/11), when using SMOTEd data in the training phase and the amount of fraudulent transactions that were flagged as benign transactions is greatly decreased. However, the amount of False Positives (actual benign, but flagged as fraudulent) is spiked, meaning that when using the SMOTEd data, our algorithm flags a transaction as fraudulent much quicker. The 'SMOTEd' run performs better as the 'UNSMOTEd' run as the AUC (Area Under the Curve) is higher with the SMOTEd run, although margins are really small. The Logistic Regression algorithm functions better than all the other algorithms, both when run SMOTEd and UNSMOTEd. The LR algorithm itself functions better when used SMOTEd.

2.2 Random Forest

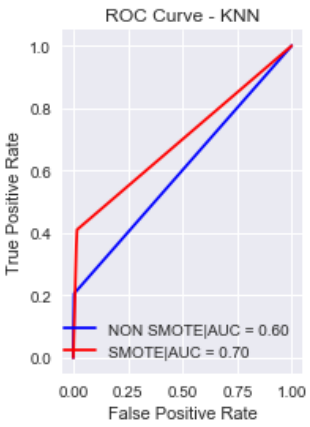


When running the Random Forest algorithm UNSMOTEd/SMOTEd data we got the following results: (The UNSMOTEd results are represented by the first value in the table, the SMOTEd results by the second).

	Predict Fraud.	**Predict Benign**
Actual Fraud.	0 / 7	36 / 29
Actual Benign	4 / 200	23664 / 23468

What we can conclude from this is that the algorithm is better in finding true fraudulent transactions in the SMOTEd run (again), although the increase is less. The amount of fraudulent transactions that were flagged as benign transactions is decreased not that much (as opposed to the LR case). Benign traffic flagged as fraudulent is spiked, although much less than the LR algorithm. The SMOTEd run performs better as the UNSMOTEd run as the AUC (Area Under the Curve) is higher with the SMOTEd run, although margins are again really small, and the Random Forest algorithm performs worse than LR algorithm when SMOTEd.

2.3 K-Nearest Neighbors



When running the Random Forest algorithm UNSMOTEd/SMOTEd we got the following results: (The UNSMOTEd results are represented by the first value in the table, the SMOTEd results by the second).

	Predict Fraud.	**Predict Benign**
Actual Fraud.	1 / 14	38 / 25
Actual Benign	2 / 322	23663 / 23343

What we can conclude from this is that the algorithm is again better in finding true fraudulent transactions in the SMOTEd data as opposed to the UNSMOTEd data. The amount of fraudulent transactions that were flagged as benign transactions again decreases and the amount of false positives greatly increases, meaning that overall more transactions were flagged as benign in the UNSMOTEd run and more were flagged as fraudulent overall in the SMOTEd run. The SMOTEd run performs better as the UNSMOTEd run, but the AUC (Area Under the Curve) is still (a lot) lower than when using the other algorithms.

3 Classification Task

In this section we will try to improve 2 aforementioned classifier to approach our goal of TP/FP of (100/1000). We will use an Logistic Regression and a Random Forest classifier. As our previous tests have proven that using SMOTE increases the classifiers performance SMOTEd data will be used in the training phase.

3.1 Preprocessing

Before we can train the data we need to make some changes to the dataset. As there are many categorical variables these must be changed so the classifier can use them. We use One hot encoding for this. There is a slight problem however, as we cannot include all features. Take for example the mail_id feature. This is a categorical variable, and has x distinct values in the dataset. This means that if we were to use one hot encoding for this variable the dataset would grow with x-1 columns, which significantly increases the computation time and memory and this reaches states where classification is impossible for a normal computer. Therefore we only include amount (calculated into euros), issuercountrycode, txvariantcode, currencycode, shoppercountrycode, shopperinteraction, cardverificationcodesupplied, cvcresponsecode and accountcode. Instructions on how this data is processed can be found in the Jupyter Notebook file which can be found on our Github[the].

3.2 Logistic Regression

We will use Logistic Regression and try to approach our goal. We will use principal component analysis to further optimize our classifier, furthermore we can set the cutoff value which can make a stricter distinction between True Positive and False Positives, We want to approach our goal rate of (100/1000) so finding an optimum for this value will help us. The classifier is ran 10 times with Stratified K-fold. We try three methods to optimize our algorithm:

- Use Principal Component Analysis and find best n_component value
- Use Hyperparameter tuning to find best parameters for Logistic Regression
- Find optimal cutoff value

First we want to determine the optimal value for the n_component value, so we run this for several values, where the results can be found in Table 1 Then we use Hyperparameter tuning, which can be found in the code.

Table 1: Rates for value for N

	n=2	n=3	n=4	n=5	n=6	n=7
TP/FP	0.007	0.015	0.147	0.0148	0.0138	0.133

This is quite a time and memory consuming operation, but it resulted in a value of $C = 0.46415$. Then running the algorithm with multiple cutoff values we determine the best value for the cutoff rate, where the results can be found Figure 1. Looking at Figure 1 we can conclude that 0.9 is the optimal value for the cutoff.

Figure 1: Precision and Recall against Cutoff for Logistic Regression

With all these optimizations in place we obtain the following results:

Table 2: Results for Logistic Regression

	Predict Fraud.	**Predict Benign**
Actual Fraud.	201	144
Actual Benign	13557	223134

We can observe that while some very good values can be obtained for true positive rates and low false negative rates we also observe a lot of false positives which makes this algorithm in practice useless. If we were to decrease the amount of false positives we also obtain more false negatives. Therefore we can conclude that this algorithm is not successful with this dataset. We can obtain the coefficients used in Logistic Regression, to get a grasp on the relation between fraud and the features used. There is a slight problem however, as there are

291 different coefficients due to the One Hot encoding. The largest coefficient is the coefficient for the amount spent (37.22031), which is significantly larger than all other values. Thus, we can conclude that the amount spent is the feature which has the most impact on the decision whether a transaction is fraudulent or not. Two more significant values which define fraudulent credit card transactions are, in decreasing order:

1. 11.95: issuercountrycode = JO
2. 8.83: shoppercountrycode = BG
3. 5.96: cardverificationcodesupplied = True
4. 5.76: shoppercountrycode = MY
5. 5.49: issuercountrycode = MY

3.3 Random Forest

We will try to obtain the best results for Random Forest. There are several things we can try to optimize to get better scores while keeping time and memory within limits. We will try to:

1. Determine optimal number of trees in the forest
2. Determine maximum leaf depth
3. Determine optimal cutoff value

For points (2,3) we classify the dataset and plot the precision and recall for the different values to find the optimal value. The results are displayed in Figure 2 and Figure 3. This is not done using Hyper Parameter Tuning as this is not feasible due to memory and computation limits. The most optimal value return with the following confusion matrix:

Table 3: Results for Random Forest

	Predict Fraud.	**Predict Benign**
Actual Fraud.	197	148
Actual Benign	17020	219671

Figure 2: Precision and Recall against Leaf depth

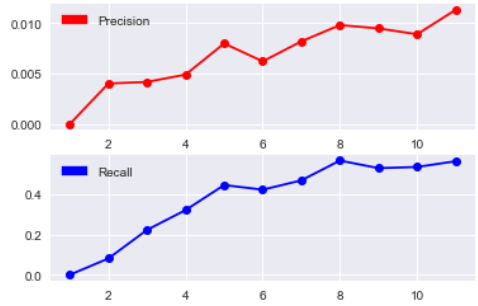
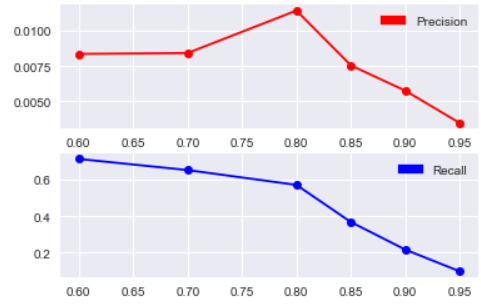


Figure 3: Precision and Recall against cutoff



3.4 Conclusion

We can safely assume that the results of aforementioned classifiers with SMOTEed data do not show promising results. While many different values for classifier operations are found to optimize the algorithm, still no perfect scores are obtained. This means that optimizations need to be done in the data, by doing either more extensive feature selection or using more data. An other method is to aggregate data features, which is attempted in the bonus assignment.

4 Bonus Task

For the Bonus task we want to investigate if we can predict a card that is used fraudulent. You can assume that a card has a higher probability that it is used in fraud cases if it is used in 6 different countries for example. For that, we will aggregate the dataset on the card_ids. We will use metrics that emphasize the aforementioned example. The dataset will be filled with the amount of unique IP's, shops, countries and mail addresses. Furthermore, the mean of all transactions that are done with that card is calculated.

We will use Logistic Regression to find whether we are more successful in predicting fraud in comparison with the previous method. For different cutoff values the following results are obtained which can be observed in Figure 4. We can observe that it functions better with a lower cutoff value but it also performs worse than Logistic Regression in Section 2. The values for the best cutoff value (0.60) are:

Table 4: Results for aggregation on Card_id

	Predict Fraud.	**Predict Benign**
Actual Fraud.	279	66
Actual Benign	42715	193976

Figure 4: Bonus: Precision and Recall against Cutoff for Logistic Regression

We can conclude that while this method generates a decent amount of True Positives, we also see a significant amount of False Positives which makes this method or algorithm in practice unusable. If a credit card company would use this algorithm a significant amount of cards will be flagged as fraudulent.

References

[Faw05] Tom Fawcett. An introduction to roc analysis. 2005.

[NVC02] Lawrence O. Hall W. Philip Kegelmeyer Nitesh V. Chawla, Kevin W. Bowyer. Smote: Synthetic minority over-sampling technique. 2002.

[the] thechib12. Github Repo for CDA lab 1. https://github.com/thechib12/cda_lab1_final.