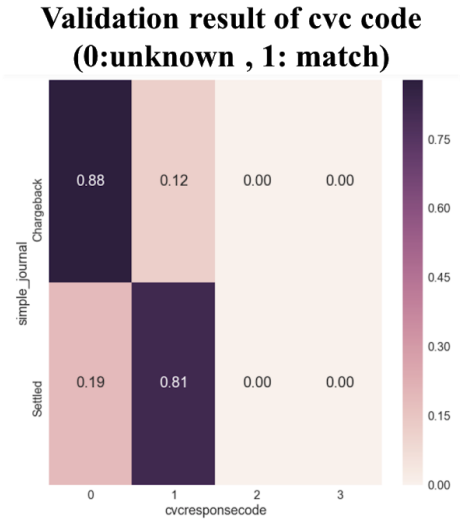
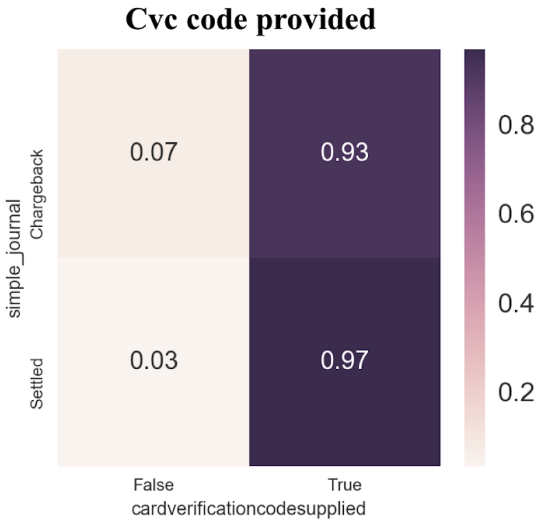
Cyber Data Analysis- assignment 1

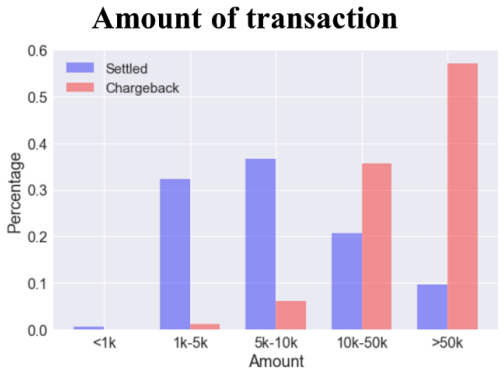
Po-Shin Chen, 4703308, P.Chen-2@student.tudelft.nl
Xin Li, 4721101, X.Li-25@student.tudelft.nl

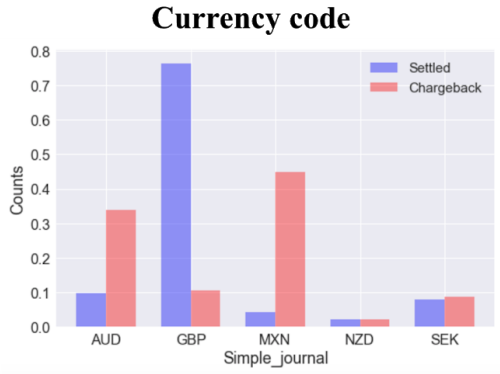
1. Visualization task

For comparing the difference between benign and malicious transactions, we tend to focus on some factors considered to be important: cvc response code provided or not, cvc matching or not, currency code, and amount of transaction. The results are presented by heatmaps and bar charts.





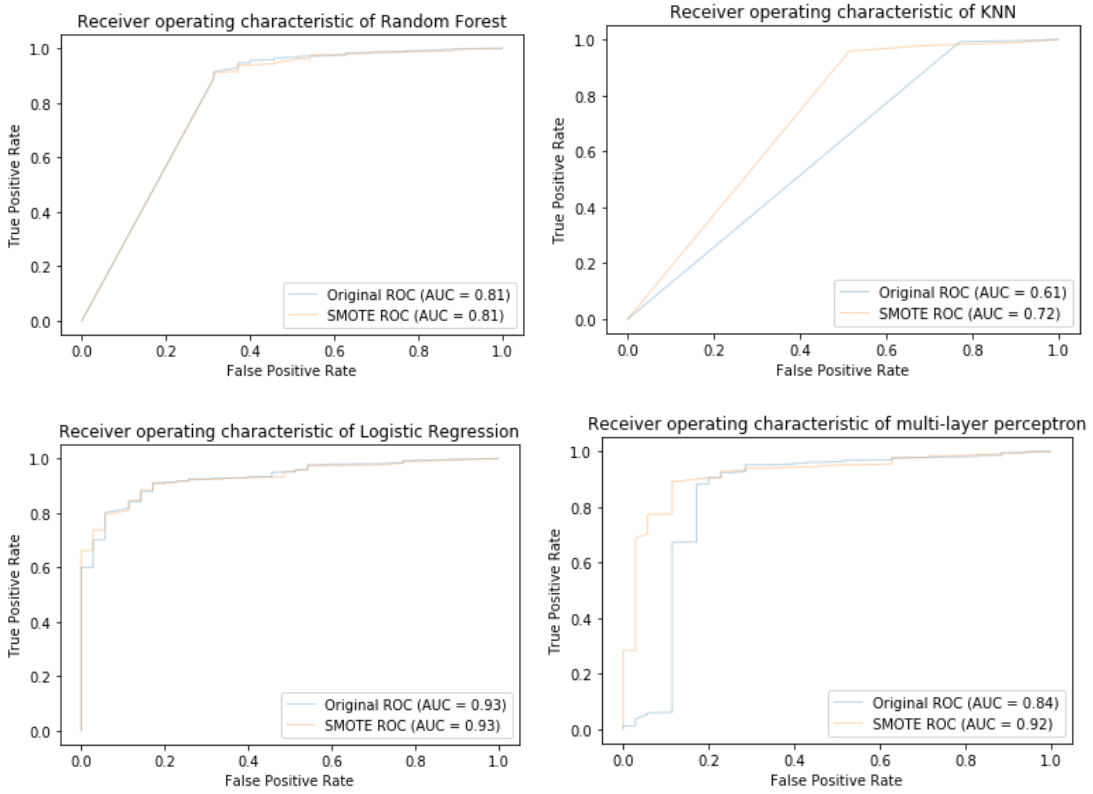




According to the figures above, there are some interesting finds:

1. From the validation result, we can notice that most of the malicious transaction could not match the cvc response code.
2. Based on the heat maps at the top, we suggest that **most of the malicious do not care whether the process needs the cvc codes or not, they would try to match in some ways.**
3. The amount of transaction in most of the malicious transactions is high (from the figure at the bottom left)
4. Most of the malicious transactions are in pounds (GBP)

2. Imbalance task



By using **random forest**, **k-nearest-neighbour**, **logistic regression** and **multi-layer perceptron classifiers**, we can see from the figure that the classifier with SMOTE outperforms the classifier without SMOTE when using the k-nearest-neighbour classifier and multi-layer perceptron. And in other classifiers like logistic regression and random forest, the differences are not so obvious. For each classifier, the AUC(Area under the curve) with SMOTE is larger than the one without SMOTE, which means the models with SMOTE have stronger prediction capability.

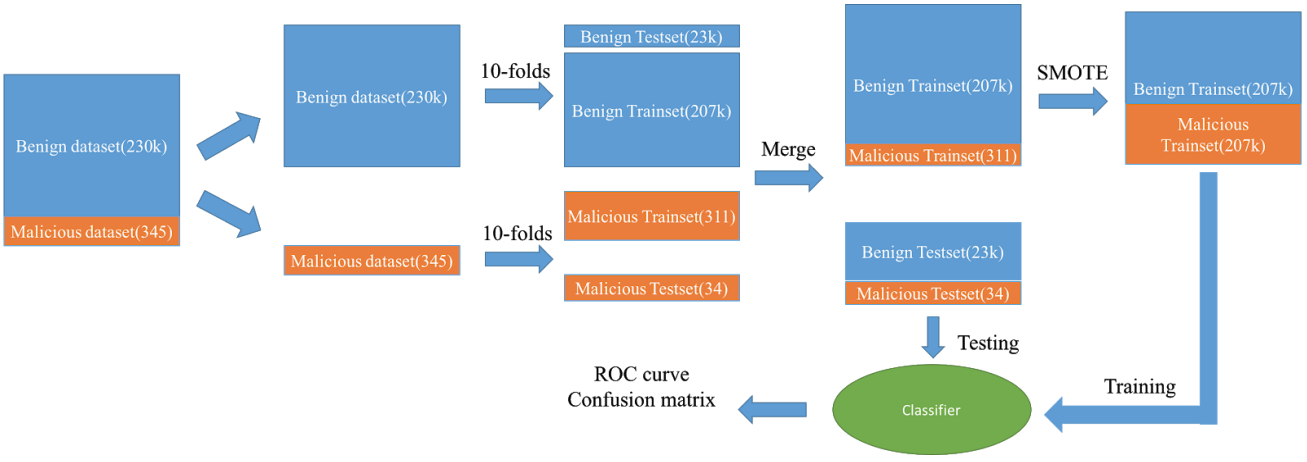
Thus, using SMOTE is a good idea. Because it can help fix the problem of imbalanced data and improve the final classification performance by dilating the minority data in this case.

3. Classification task

Owing to the fact that there are lots of non-numeric features in the dataset, we should quantify the features and simplify the dataset before training the classifiers for fraudulent transaction detection. In this preprocessing step, we did several actions:

1. Simplify the feature "cvcresponsecode" from 0-6 into 0-3
2. Delete features regarding to dates, such as bookingdate and creationdate.
3. Quantify the features by dummy variables or factorization.
4. Redefine the feature "amount", instead of taking it as numeric values, we assign the amount into clusters which has five boundings: <1k , 1k-5k, 5k-10k, 10k-50k, and >50k.
5. Delete the "Refused" dataset because we don't know these are benign or malicious transaction and we don't want these dataset affects our classification.

Our algorithm is expressed in the figure below. First, we split our dataset into benign and malicious dataset because over 99% of dataset are benign, if we randomly split the whole dataset into trainset and testset, we couldn't assure that the malicious dataset are in the trainset, which will make the performance worse. Thus by first splitting them and then merge them after creating 10-folds sets, we can make sure that the amount of malicious dataset in each trainset are equally distributed. As for the TP >100/ FP < 1000 criteria, we add up all the confusion matrices obtained from 10-folds testing results to see if we reach the requirement.



White Box Classifier

We implement K-Nearest neighbors classifier for detection. The dataset containing selected information as features is projected into a high-dimensional space. By using the KNN classifier, fraudulent transactions with similar behaviors will be distinguished from other benign transactions. We are able to explain to the customer that a transaction is halted because our system finds out that there are some similarities between the behaviors of this transaction and the behaviors of fraudulent

transactions. The confusion matrix and the performance of this classifier are shown in Table 1 and 2.

Table 1: KNN-confusion matrix

Actual \ Predicted	Benign	Malicious
Benign	86	268
Malicious	2739	233961

Table 2: KNN performance

Precision	0.03
Recall	0.242
Accuracy	0.987

Black Box Classifier

As for the black box classifier, we tried many classifiers such as random forest, decision tree, Bagging, and so on. However, it is quite hard to reach the 100/1000 criteria. We tuned the parameter "k_neighbors" in SOMTE(), which is used to prevent overfitting issue. The amount of TP and FP will be decreased by increasing k_neighbors factor. Our best performance closest to the criteria is Extra tree classifier, also known as an "Extremely randomized trees" classifier. The confusion matrix and the performance are shown in Table 3, 4.

Table 3: Confusion matrix of extra trees classifier

Actual \ Predicted	Benign	Malicious
Benign	80	274
Malicious	985	235715

Table 4: Performance of extra trees classifier

Precision	0.075
Recall	0.226
Accuracy	0.995

Analysis

As we learned from the lecture, since over 99% of the transactions are benign and a few transactions are malicious, it is easy to reach the high accuracy (0.98~0.99). However, the cost of the malicious transaction is high, we should consider the recall and precision instead of the accuracy. In addition, most of the features that we got are non-numeric variables, we should realize that factorization may result in poor performance because the order of each category should be equal. Therefore, using one-hot encoding method is important. And based on the findings in task 1, we suggest that features of the amount of transaction and cvcresponsecode are good features for detecting the benign and malicious transactions.

The codes can be retrieved on : https://github.com/Po-Shin/Cyber_data.git