# Credit Card Fraud Detection


| | |
|---|---|
| Name: | **Hardik Soni** |
| Roll No.: | 19131 |
| Institute Name: | IISER Bhopal |
| Stream: | DSE |
| Problem Release date: | February 02, 2022 |
| Date of Submission: | April 24, 2022 |


## 1 Introduction

As we're transferring closer to the virtual world — cybersecurity is turning into a critical part of our existence. When we communicate approximately safety in a virtual existence, the principle venture is to discover the bizarre activity.When we make any transaction simultaneously as buying any product online — a remarkable number of human beings opt for credit cards. The credit score restriction in credit cards occasionally lets us make purchases even supposing we don't have the quantity at that time. But, on the opposite hand, those capabilities are misused with the aid of using cyber attackers.We want a method that could abort the transaction if it unearths fishy to address this trouble.Here I am creating a project that could track the sample of all of the transactions, and if any sample is odd, then the transaction needs to be separated from the rest of the transactions. As can be seen in Figure 1, the number of fraudulent transactions are much less as compared to the regular transactions.

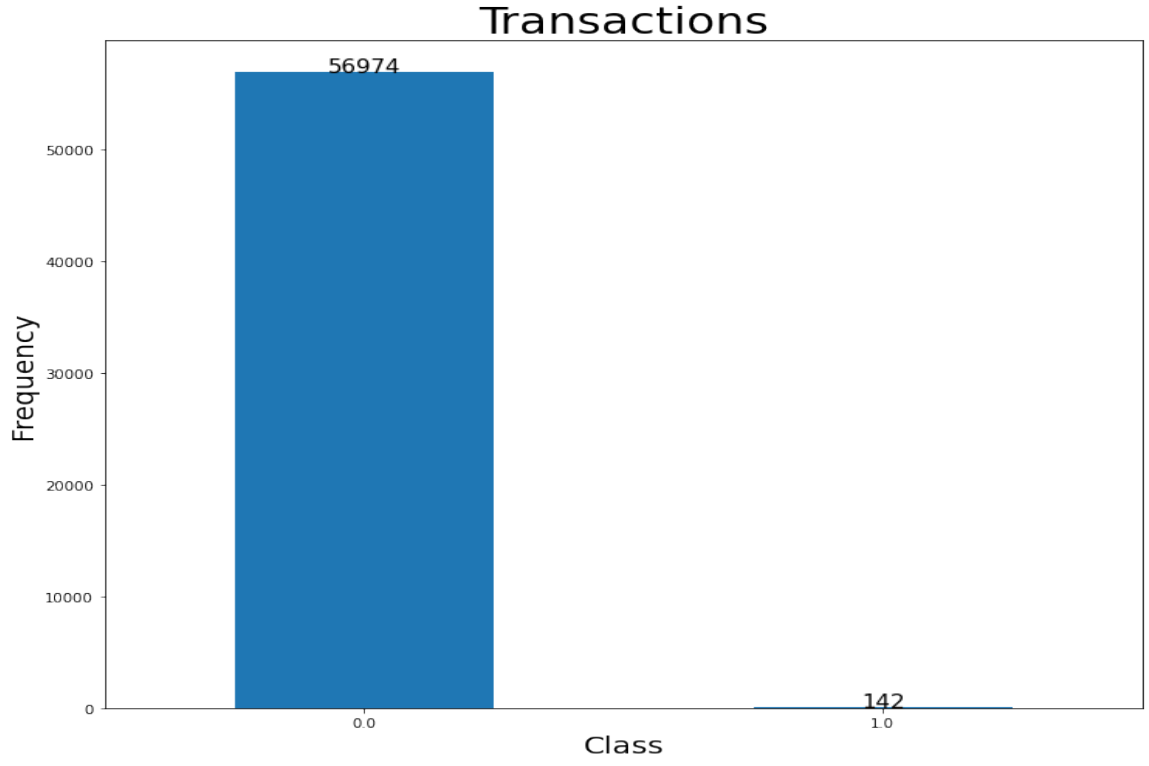


Figure 1: Overview of Data

## 2 Methods

On the dataset provided, which is highly imbalanced, i.e., 56974 compared to 142, some features positively correlate with fraudulent transactions. These are shown in the correlation heat plot. The best features are selected using GridsearchCV. For feature selection, feature importance and feature scores are used, which gave me the best ten features of the data. Now, those features are taken and used for making predictions. Gaussian Naive Bayes, KNN, SVM, Adaboost, and Random Forest algorithms have been used to predict the data points' class labels.

1. **GAUSSIAN NAIVE BAYES:** Gaussian Naive Bayes is a variant of Naive Bayes that follows Gaussian normal distribution and supports continuous data. Naive Bayes Classifiers are based on the Bayes Theorem. One assumption taken is the strong independence assumptions between the features. These classifiers assume that the value of a particular part is independent of the importance of any other attribute.

2. **KNN:** The KNN working can be explained based on the below algorithm:

   - **Step-1:** Select the number K of the neighbors
   - **Step-2:** Calculate the Euclidean distance of K number of neighbors
   - **Step-3:** Take the K nearest neighbors as per the calculated Euclidean distance.
   - **Step-4:** Among these k neighbors, count the number of the data points in each category.
   - **Step-5:** Assign the new data points to that category for which the number of the neighbor is maximum.

3. **SVM:** The objective of the SVM algorithm is to find a hyperplane in an N-dimensional space that distinctly classifies the data points. The dimension of the hyperplane depends upon the number of features. If the number of input features is two, then the hyperplane is just a line. If the number of input features is three, then the hyperplane becomes a 2-D plane. It becomes difficult to imagine when the number of elements exceeds three.

4. **ADA BOOST:** AdaBoost (Adaptive Boosting) is a popular boosting technique that combines multiple weak classifiers to build one robust classifier. A single classifier may not accurately predict the class of an object. Still, when we group multiple weak classifiers, each progressively learning from the others' wrongly classified things, we can build one such robust model.

5. **RANDOM FOREST:** Random forest randomly selects observations, builds a decision tree, and takes the average result. Random forests are created from subsets of data, and the final output is based on average, or majority ranking hence the problem of overfitting is taken care of.

The code for the project is uploaded at GitHub

## 3 Evaluation Criteria

Precision is the ratio of correctly predicted positive observations to the total predicted positive observations.

$$Precision = \frac{True\ Positive}{True\ Positive + False\ Positive}$$

Recall is the ratio of correctly predicted positive observations to the all observations in actual class - yes

$$Recall = \frac{True\ Positive}{True\ Positive + False\ Negative}$$

F1 Score is the weighted average of Precision and Recall. Therefore, this score takes both false positives and false negatives into account.

$$F1Score = \frac{2 \times Precision \times Recall}{Precision + Recall}$$

where each of the above term is defined as :-

- **True Positives:** It is the case where we predicted Yes and the real output was also yes.
- **True Negatives:** It is the case where we predicted No and the real output was also No.
- **False Positives:** It is the case where we predicted Yes but it was actually No.
- **False Negatives:** It is the case where we predicted No but it was actually Yes.

## 4 Analysis of Results

As can be observed from the table, KNN and Random Forest are pretty accurate in predicting the class labels of the transactions.

Table 1: Performance Of Different Classifiers

| Classifier | Precision | Recall | F1-Score |
|---|---|---|---|
| **GNB** | 0.55 | 0.92 | 0.59 |
| **KNN** | 0.96 | 0.91 | 0.93 |
| **SVM** | 0.90 | 0.87 | 0.89 |
| **AdaBoost** | 0.93 | 0.91 | 0.92 |
| **Random Forest** | 0.96 | 0.91 | 0.93 |

## 5 Discussions and Conclusion

For further discussions, we can work on KNN and Random Forest for some better features to work on using feature extraction. Also we can use scalars on our data and check if it affects the accuracy.