**CREDIT CARD FRAUD DETECTION**

**Introduction:**

Credit card fraud is when someone uses another person's credit card or account information to make unauthorized purchases or access funds through cash advances. According to the recent Nilson report data, credit card fraud losses reached $28.65 billion worldwide in 2019. The coronavirus pandemic is also fuelling explosive growth in card fraud activity. It impacts consumers, merchants, and issuers alike. Its economic cost goes far beyond the cost of illegally purchased merchandise. Businesses often spend millions on protecting themselves from fraud. This problem can be tackled by using Machine Learning algorithms. Our project intends to accomplish this task. The Credit Card Fraud Detection problem includes building a model which uses data from past credit card transactions and then recognizes whether a new transaction is fraudulent or not. Our objective here is to detect as many of the fraudulent transactions while minimizing incorrect fraud classifications. Credit Card Fraud Detection is a classification problem. In this process, we have focused on analyzing the dataset and comparing the performance of different machine learning models, namely, Support Vector Machine, Logistic Regression, Neural Network, Random Forest Classifier, and Adaboost.

**Literature review:**

Numerous literature on anomaly or fraud detection in Credit Card Fraud Detection has been published already and is available in the public domain. A survey conducted by Clifton Phua and his associates has revealed that techniques employed in this domain include data mining applications, automated fraud detection, adversarial detection. In another research paper, Suman, Research Scholar, GJUS&T at Hisar HCE, presented supervised and Unsupervised Learning for credit card fraud detection techniques. Even though these methods and algorithms achieved success in some areas, they failed to provide a permanent solution to fraud detection. People have used many ML models to solve this problem, but there are many challenges like:

* Most of the transactions *(99.9%)* are not fraudulent, making it hard for the model to detect fraudulent transactions. The machine learning model tends to be biased towards non-fraudulent transactions.
* Unavailability of data in the public domain due to privacy concerns.
* Fraudsters try their best to conceal their activities and constantly change their methods and behavior.

**Methodology:**

First of all, we did exploratory data analysis on our dataset to find out the differences between fraud and everyday transactions and any other important characteristic of the dataset. Then, we used different ML algorithms like Logistic Regression, Support Vector, Neural Network, Random Forest Classifier, and Adaboost. To evaluate the performance of these models, we split the dataset into train, validation, and test sets. Accuracy would not have been an excellent metric to evaluate the models due to the high data imbalance. So, we evaluated the performance of these models based on the F1 scores achieved on the validation set. Further, we have optimized various parameters in each algorithm to achieve the best results. Finally, we compared the performance of these models based on their respective F1 scores on the test dataset.

**About the dataset:**

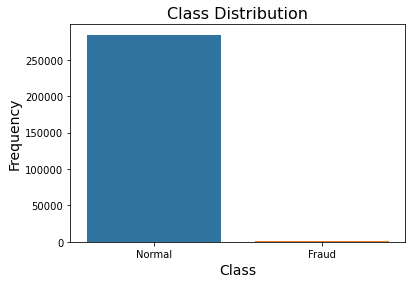
## The dataset used for this project has been taken from Kaggle, and its link is provided in the references section.

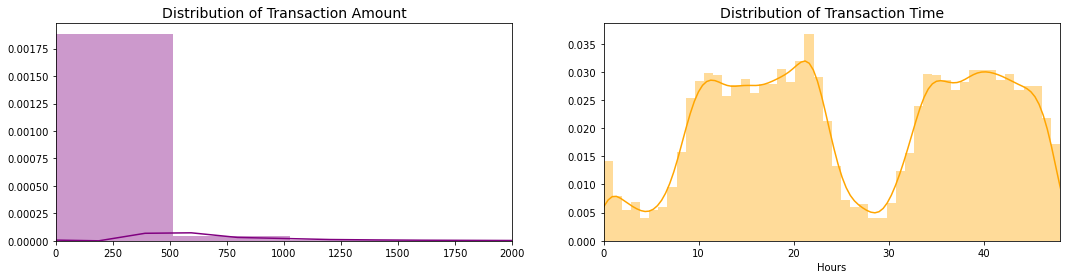
## The data contained **284,807** European credit card transactions with **492** fraudulent transactions over **two days** in September 2013.

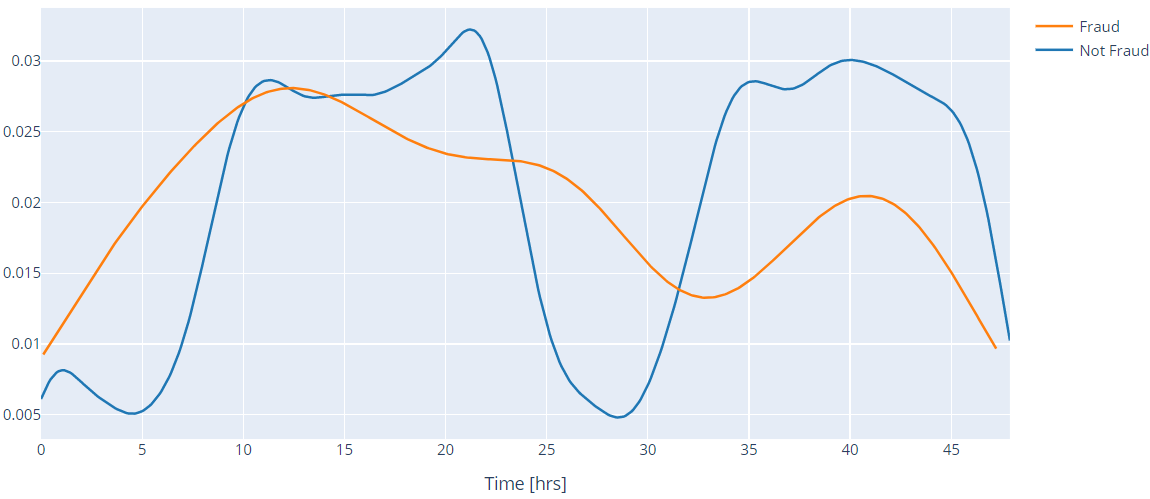
* Except for time and amount, everything has been reduced by a Principal Component Analysis (PCA) for privacy concerns. Features V1, V2, ... V28 are the principal components obtained with PCA and have been scaled already.
* Since there is little information about the features, we cannot perform much feature engineering and data exploration.
* Feature **'Time'** contains the seconds elapsed between each transaction and the first transaction in the dataset.
* The feature **'Amount'-** is the transaction Amount; this feature can be used, for example-dependent cost-sensitive learning.
* Feature **'Class'** is the response variable. It takes value 1 in case of fraud and 0 otherwise.

**Data Exploration:**

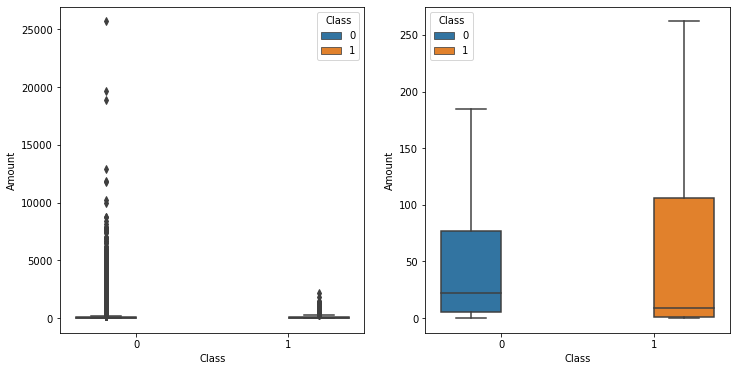
1. **Data imbalance:** There are **284315** normal transactions **(99.828%)** as opposed to **492** fraud transactions (**0.172%)**. So, clearly there is an imbalance in the dataset which may adversely affect the performance of our ML models. Thus, we will have to use some techniques to counter this data imbalance.



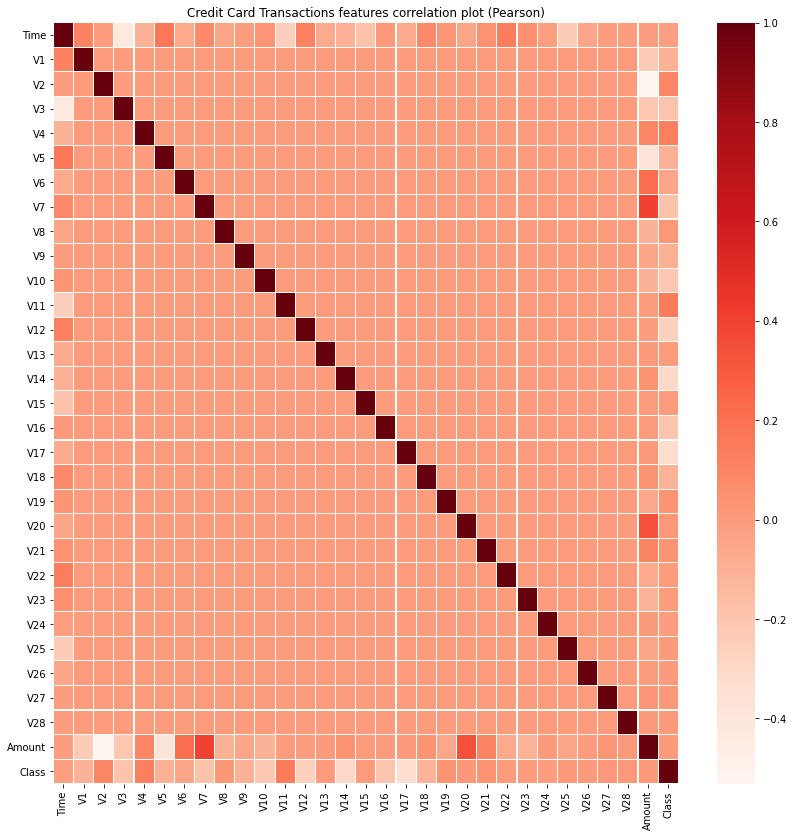
1. **Distribution of transactions with time and amount:** Going by the drop in the frequency of transactions at 0 hrs and then again after 24 hrs, we may infer that it corresponds to night time when people are mostly asleep. We can say that most of the transactions occurred during the day. The mean of the transaction amount is 88.35 USD, standard deviation is 250.1 USD and 75%ile is 77.05 USD.
2. **Difference in the distribution of Fraudulent and Normal Transactions across time:** Fraudulent transactions are more equally distributed in time than everyday transactions. This is evident from the following graph:



1. **Difference between the average amount transacted in both cases:** The average amount transacted in ordinary cases is **88.29 USD,** while in fraud cases, it is **122.21 USD**. Also, from the plot below, we can see that the outliers in fraudulent cases are less than everyday transactions. Class 1 represents fraud cases, and Class 0 represents normal cases.



1. **Correlations between features:** There is no significant correlation between features **V1**-**V28**. There are specific correlations between some of these features and **Time** (negative correlation with **V3**) and **Amount** (positive correlation with **V7** and **V20**, negative correlation with **V2** and **V5**). Also, there is some negative correlation between **V17** and **Class.**



**PERFORMANCE OF DIFFERENT MODELS**

**Support Vector Machine:**

FINAL RESULTS ON TEST DATA

* Accuracy: 0.999386
* Precision: 0.94828
* Recall: 0.63218
* F1 score: 0.75862
* ROC-AUC score: 0.81606

**Logistic Regression:**

**CASE 1: When whole data is used as it is for training**

We tried to optimize some parameters of this model, and the optimal values which we got are:

Split ratio between train and validation dataset = 0.15

FINAL RESULTS ON TEST DATA:

* F1 score : 0.7052
* ROC-AUC: 0.7960
* Precision : 0.8714
* Recall: 0.5922
* Accuracy : 0.9991

**CASE 2: When we used random undersampling for training**

Random UnderSampling: Due to imbalanced dataset it causes many learning algorithms to ignore the minority class entirely. One way of overcoming this challenge is to do random sampling from the majority class till both the labels (fraudulent and non fraudulent) are comparable in size.

We tried to optimize some parameters of this model, and the optimal values which we got are:

Split ratio= 0.25

FINAL RESULTS ON TEST DATA

* F1 score : 0.94416
* ROC-AUC: 0.94536
* Precision : 0.97895
* Recall: 0.91176
* Accuracy : 0.94416

**Random Forest Classifier:**

We tried to optimize some parameters of this model, and the optimal values which we got are:

Split ratio= 0.3

Number of estimators (trees)= 130

Maximum Depth= 10

Number of Sample leaf= 2

FINAL RESULTS ON TEST DATA

* Accuracy: 0.9995770
* Precision: 0.939759
* Recall: 0.80412
* F1 Score: 0.86667
* AUC Score: 0.90202

**AdaBoost:**

We tried to optimize some parameters of this model, and the optimal values which we got are:

Split ratio= 0.35

Number of estimators= 200

FINAL RESULTS ON TEST DATA

* Accuracy= 0.999473
* Precision = 0.91463
* Recall= 0.76531
* F1 score= 0.83333
* ROC-AUC= 0.88259

## 

## **Neural Network:**

We tried to optimize some parameters of this model and the optimal values which we got are:

Split ratio between train and validation dataset = 0.35

Size of hidden layers (number of neurons) = 250

Number of iterations = 400

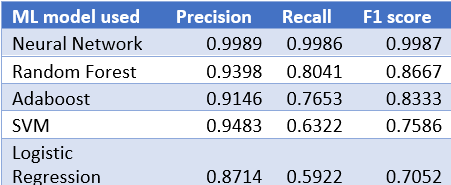
Activation function = ReLU

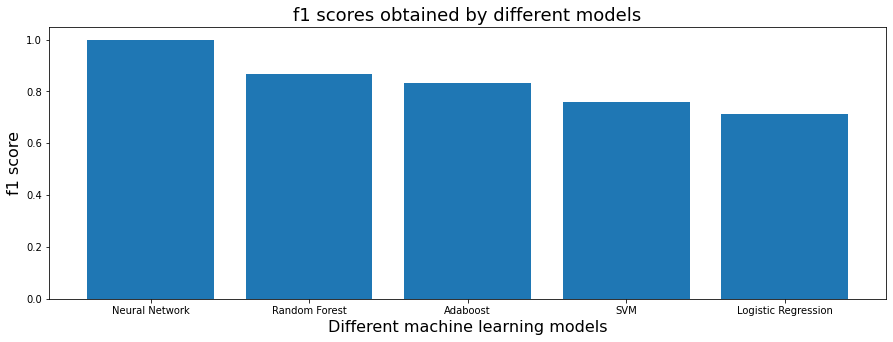
Solver= Stochastic gradient descent

FINAL RESULTS ON TEST DATA

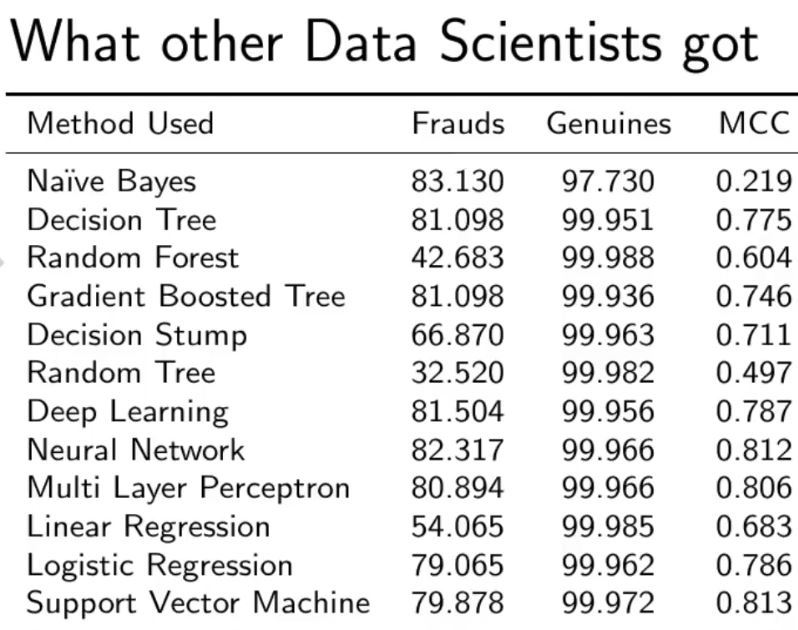
* Accuracy = 0.9986
* Precision = 0.9989
* Recall = 0.9986
* f1 score = 0.9987
* ROC\_AUC = 0.9576

**Comparing models based on f1 score on test data**





**PREVIOUS RESULTS:**



**REFERENCES:**

* <https://www.youtube.com/c/joshstarmer/videos>
* <https://www.geeksforgeeks.org/ml-credit-card-fraud-detection/>
* <https://www.kaggle.com/gpreda/credit-card-fraud-detection-predictive-models>
* <https://www.kaggle.com/anirudhg15/credit-fraud-neural-network-101>
* <https://machinelearningmastery.com/random-oversampling-and-undersampling-for-imbalanced-classification>
* <https://scikit-learn.org/stable/modules/generated/sklearn.ensemble.RandomForestClassifier.html>
* <https://www.upgrad.com/blog/bagging-vs-boosting/#:~:text=Bagging%20and%20Boosting%3A%20Differences&text=Bagging%20is%20a%20method%20of,Boosting%20decreases%20bias%2C%20not%20variance>.
* <https://medium.com/swlh/recall-precision-f1-roc-auc-and-everything-542aedf322b9>
* CLIFTON PHUA1, VINCENT LEE1, KATE SMITH1 & ROSS GAYLER2 “ A Comprehensive Survey of Data Mining-based Fraud Detection Research” published by School of Business Systems, Faculty of Information Technology, Monash University, Wellington Road, Clayton, Victoria 3800, Australia
* “Survey Paper on Credit Card Fraud Detection by Suman”, Research Scholar, GJUS&T Hisar HCE, Sonepat published by International Journal of Advanced Research in Computer Engineering & Technology (IJARCET) Volume 3 Issue 3, March 2014

**Team Members:**

* Maheesh Baijal
* Anudeep Verma
* Tanal Patel
* Abhishek N Bhavsar