Prediction Of Credit Card Fraud

Abstract

Financial fraud is a growing problem with long term consequences in the financial industry and while many techniques have been discovered to solve this problem faced by various companies, data mining has been successfully applied to finance databases to automate analysis of huge volumes of complex data. Data mining has also played a salient role in the detection of credit card fraud in online transactions.

Fraud detection in credit card is a data mining problem. It becomes challenging due to two major reasons – first, the profiles of normal and fraudulent behaviour change frequently and second, the credit card fraud data sets are highly skewed. This paper investigates and checks the performance of Decision Tree, Random Forest and Logistic Regression on highly skewed credit card fraud data. Dataset of credit card transactions is sourced from European cardholders containing 284,786 transactions. These techniques are applied on the raw and pre-processed data.

The performance of the techniques is evaluated based on accuracy, sensitivity, specificity, precision. The results indicating the optimal accuracy for Logistic Regression, Decision Tree, Random Forest, classifiers are 95%, 91%, 90% respectively.

ACKNOWLEDGEMENTS

I would like to express my heartfelt gratitude to the UpGrad community and the instructors for providing an exceptional Data Science Bootcamp course. The comprehensive materials and engaging lectures have significantly enriched my understanding of data analysis, exploration, visualization, manipulation, and cleaning. I have also learned about various supervised learning algorithms, which have been instrumental in shaping my approach to solving complex problems.

I am particularly thankful for the supportive environment fostered by the UpGrad community, where I could collaborate with peers and share insights that enhanced my learning experience. The feedback and encouragement from my fellow students helped me stay motivated throughout the course.

While I completed this project independently, I extensively referred to syntax, methods, and hyperparameters from various online resources, including Google, video tutorials, course notebooks, and the official documentation of scikit-learn (sklearn). These resources provided invaluable support in navigating challenges and refining my skills.

Finally, I would like to acknowledge the encouragement from my family and friends, who supported me during this journey. Their belief in my abilities provided me with the strength and determination to overcome obstacles and achieve my goals.

Table of Contents

List of Figures

1 Introduction

1.1 Description

Financial fraud is a growing concern with far-reaching consequences in government, corporate organizations, and the finance industry. In today's world, the increasing dependency on internet technology has led to a surge in credit card transactions. However, this has also accelerated credit card fraud in both online and offline transactions. As credit card transactions become a widespread mode of payment, computational methodologies have gained attention for addressing the credit card fraud problem. Fraud detection solutions have been developed for various industries, including credit card, retail, e-commerce, and insurance.

Data mining techniques are effective methods in solving credit card fraud detection problems. Since it is difficult to be completely certain about the legitimacy of every transaction, mathematical algorithms are key tools for identifying potential fraud. In credit card fraud detection, the objective is to classify transactions as either legitimate or fraudulent. Several models, including Logistic Regression, Decision Trees, and Random Forests, are commonly employed for this purpose.

Credit card transaction datasets are often imbalanced and skewed, making feature selection and evaluation metrics critical to assessing model performance. Fraudulent behavior is dynamic, as fraudulent transactions can resemble legitimate ones, adding to the complexity. The choice of the sampling approach, variable selection, and detection technique plays a crucial role in model performance.

In this project, the performance of the selected models—Logistic Regression, Decision Tree, and Random Forest—is evaluated based on accuracy, sensitivity, specificity, and precision. Logistic Regression demonstrates the highest accuracy at 95%, followed by Decision Tree with 91%, and Random Forest with 90%. These results highlight the effectiveness of these models in

detecting credit card fraud, with Logistic Regression emerging as the most accurate classifier.

1.2 Problem Formulation

The problem formulation consists of just one sentence and should make it clear to everyone what research problem, you aim to address and to whom and where it is relevant.

Problem Formulation for our project:

Are the existing techniques used for detecting credit card frauds correctly and accurately providing efficient results or can it be improved using Machine Learning? Thus, our project tries to predict credit-card frauds using Machine Learning as it is believed to provide better results as compared to the existing techniques used to detect these frauds.

1.3 Proposed Solution

These are the proposed techniques used in this paper for detecting the frauds in credit card system. The comparison are made for different machine learning algorithms such as Logistic Regression, Decision Trees, Random Forest, to determine which algorithm suits best and can be adapted by credit card merchants for identifying fraud transactions. The Figure shows the architectural diagram for representing the overall system framework.

Algorithm steps:	
Step 1:	Read the Dataset.
Step 2:	Random Sampling is done on the data set to make it balanced.
Step 3:	Divide the dataset into two parts i.e., Train dataset and Test dataset.
Step 4:	Accuracy and performance metrics has been calculated to know the efficiency for different algorithms.
Step 5:	Then retrieve the best algorithm based on efficiency for the given dataset.

1.4 Scope of the Project

Credit card fraud detection is a very popular but challenging problem to solve. One of the primary challenges is the limited amount of data available, making it difficult to identify consistent patterns for fraud detection. Additionally, many entries in the dataset may show patterns of legitimate behavior, even if they contain transactions made by fraudsters. This overlap complicates the detection process.

There are several constraints when working with credit card fraud detection. Firstly, credit card transaction datasets are not easily accessible to the public, and research findings are often hidden or censored, making benchmarking models difficult. Secondly, security concerns impose limitations on the exchange of ideas and methods in fraud detection, especially in the credit card domain. Finally, the profiles of legitimate and fraudulent behaviours evolve over time, meaning that what was once considered a legitimate transaction may now be flagged as fraudulent, or vice versa.

In this project, we evaluate three advanced data mining approaches: Logistic Regression, Decision Tree, and Random Forest. A comparative analysis is conducted to determine which model performs best in detecting credit card fraud.

2. Literature Survey

Credit card fraud detection has drawn a lot of research interest and a number of techniques, with special emphasis on neural networks, data mining and distributed data mining have been suggested.

There have been various studies on credit card fraud detection. Machine learning and related methods are most commonly used, which include artificial neural networks, rule-induction techniques, decision trees, logistic regression, and support vector machines

Behrouz, Emad and Sahil [2] have proposed credit card fraud detection using supervised machine learning algorithms. They have employed these algorithms to implement a super classifier using ensemble learning methods. Also, they compared and discussed the performance of various supervised machine learning algorithms that exist in literature against the super classifier that they implemented.

Navanshu Khare and Saad Yunus Sait[3] have explained the mathematics and working of every algorithm we have used in this paper.

Siddhartha Bhattacharyya and 4 others [6] in their paper in 2011 did a detailed comparative study of Support vector machine and random forest along with logistic regression. They concluded through experiments that Random Forest technique shows most accuracy followed by Logistic Regression and Support Vector Machine.

3. System Design

3.1 Functional Requirements

- The model should be able to give accurate and trustworthy predictions.
- The application must show graphical visualization of the predicted results and data in general.
- The user should be able to enter the input values for prediction.

3.2 Non Functional Requirements

Non Functional Requirements will describe how a system should behave and what limits are constrained on its functionality. It generally specifies the system's quality attributes or characteristics.

- Availability: The system should be available to any transaction verification system.
- Correctness: The accuracy of the system should be as maximum as possible for better prediction.
- Maintainability: The system should maintain correct history of records.
- Usability: The system should satisfy a maximum number of banking system needs**.**

3.3 Specific Requirements

Software Requirements

1. Languages : Python-3
2. Operating Systems : Windows, Linux, etc.
3. Back End Software : Anaconda, Jupyter notebook.

Hardware Requirements:

1. CPU : Intel® Core™ i5-7300 HQ CPU @ 2.50 GHz
2. Hard disk space : 20 GB or more
3. Memory : 8 GB RAM

4. Implementation Details

4.1 Algorithms & Methods Used

Ability of system to automatically learn and improve from experience without being explicitly programmed is called machine learning and it focuses on the development of computer programs that can access data and use it to learn by themselves. And classifier can be stated as an algorithm that is used to implement classification especially in concrete implementation, it also refers to a mathematical function implemented by algorithm that will map input data into category. It is an instance of supervised learning i.e. where training set of correctly identified observations is available.

A. Logistic Regression

Logistic Regression is a supervised classification method that returns the probability of binary dependent variable that is predicted from the independent variable of dataset i.e. logistic regression predicts the probability of an outcome which has two values, either zero or one, no or yes and false or true. Logistic regression has similarities to linear regression, but, in linear regression a straight line is obtained, logistic regression shows a curve. The use of one or several predictors or independent variable is on what prediction is based, logistic regression produces logistic curves which plots the values between zero and one.

Logistic Regression is a regression model where the dependent variable is categorical and analyzes the relationship between multiple independent variables. There are many types of logistic regression model such as binary logistic model, multiple logistic model, binomial logistic models. Binary Logistic Regression model is used to estimate the probability of a binary response based on one or more predictors.

$$p = \frac{e^{\alpha+\beta_n X}}{1 + e^{\alpha+\beta_n X}}$$

Above equation represents the logistic regression in mathematical form.

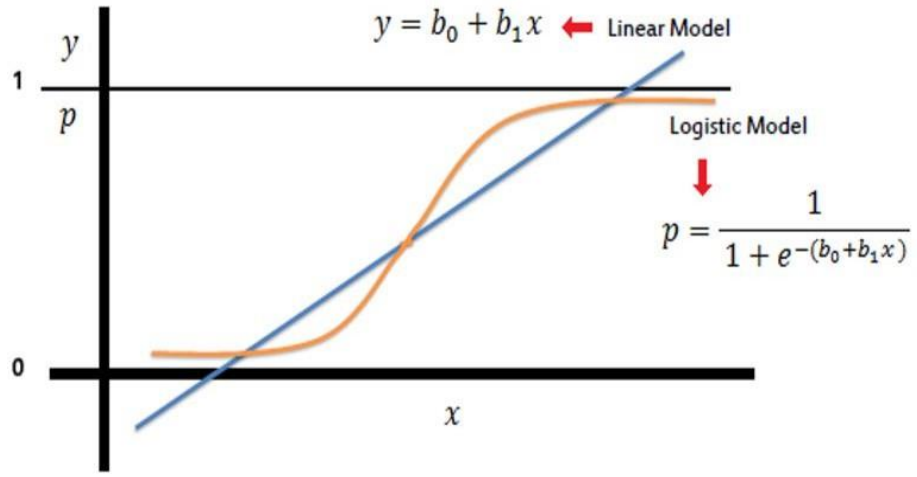


Fig.1 Logistic Curve

This graph shows the difference between linear regression and logistic regression where logistic regression shows a curve but linear regression represents a straight line.

B. Decision Tree

Decision tree is an algorithm that uses a tree like graph or model of decisions and their possible outcomes to predict the final decision, this algorithm uses conditional control statement. A Decision tree is an algorithm for approaching discrete-valued target functions, in which decision tree is denoted by a learned function. For inductive learning these types of algorithms are very famous and have been successfully applied to a broad range of tasks

We give label to a new transaction that is whether it is legit or fraudulent for which class label is unknown and then transaction value is tested against the decision tree, and after that from root node to output/class label for that transaction a path is traced.

$$Entropy(S) = \sum_{i=1} -p_i \log_2 p_i$$

$$Gain(S, A) = Entropy(S) - \sum_{v \in Values(A)} \frac{|S_v|}{|S|} Entropy(S_v)$$

Decision rules determines the outcome of the content of leaf node. In general rules have the form of 'If condition 1 and condition 2 but not condition 3 then outcome'. Decision tree helps to determine the worst, best and expected values for different scenarios, simplified to understand and interpret and allows addition of new possible scenarios.

Steps for making a decision tree are: to calculate the entropy of every attribute using the dataset in problem, then, dataset is divided into subsets using the attribute for which gain is maximum or entropy is minimum, after that, to make a decision tree node containing that attribute and lastly recursion is performed on subsets using remaining attributes to create a decision tree.

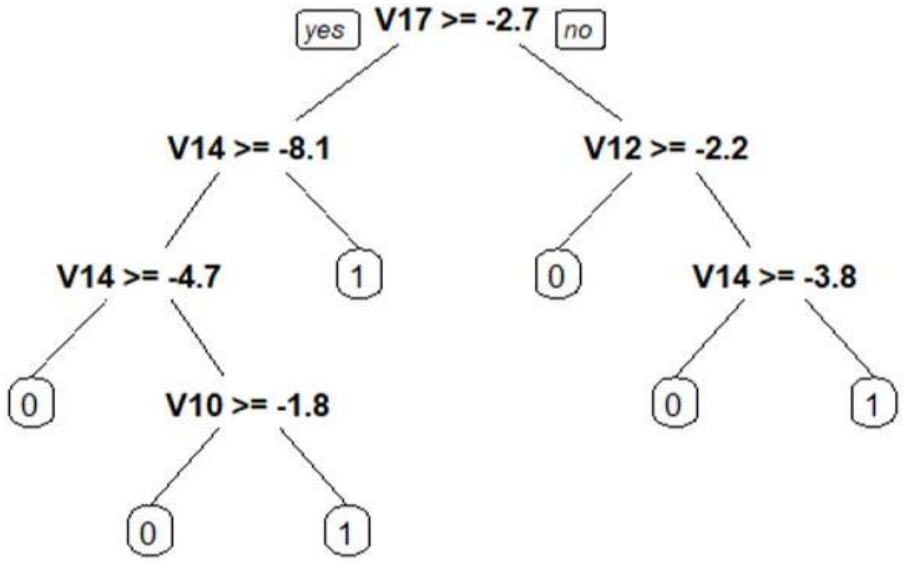


Figure 2: Decision tree

C.Random Forest

Random Forest is an algorithm for classification and regression. Summarily, it is a collection of decision tree classifiers. Random forest has advantage over decision tree as it corrects the habit of overfitting to their training set. A subset of the training set is sampled randomly so that to train each individual tree and then a decision tree is built, each node then splits on a feature selected from a

random subset of the full feature set. Even for large data sets with many features and data instances training is extremely fast in random forest and because each tree is trained independently of the others. The Random Forest algorithm has been found to provides a good estimate of the generalization error and to be resistant to overfitting. Random forest ranks the importance of variables in a regression or classification problem in a natural way can be done by Random Forest.

4.2 Working of the project

It is important that credit card companies are able to recognize fraudulent credit card transactions so that customers are not charged for items that they did not purchase.

Information about Dataset

The datasets contain transactions made by credit cards in September 2013 by European cardholders. This dataset presents transactions that occurred in two days, where we have 492 frauds out of 284,807 transactions. The dataset is highly unbalanced, the positive class (frauds) account for 0.172% of all transactions.

Load Data and Read Data:

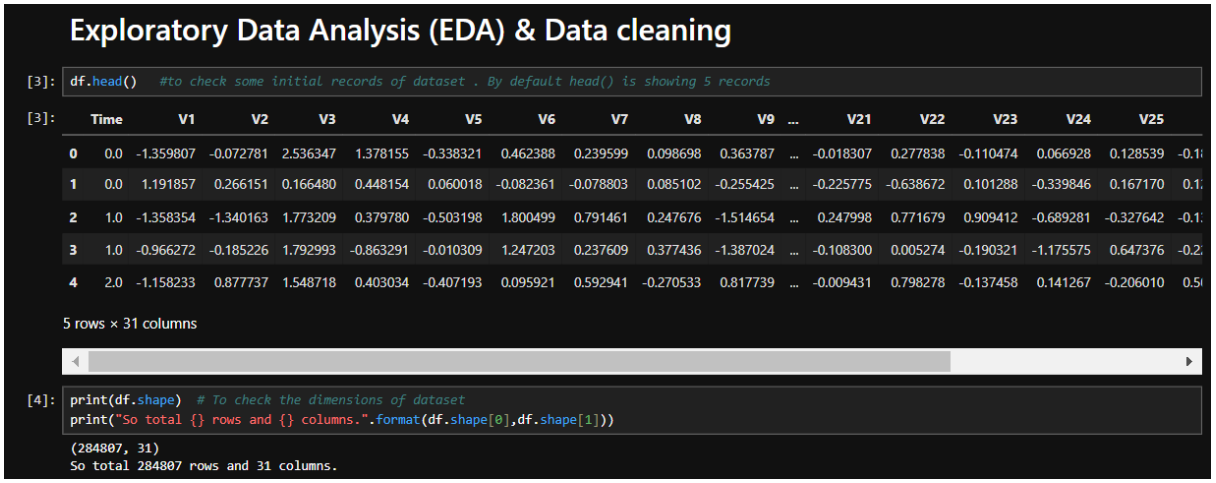
```
import pandas as pd

df = pd.read_csv('creditcard.csv')
```

```
df
```

	Time	V1	V2	V3	V4	V5	V6	V7	V8	V9	...	V21	V22	V23	V24	
0	0.0	-1.359807	-0.072781	2.536347	1.378155	-0.338321	0.462388	0.239599	0.098698	0.363787	...	-0.018307	0.277838	-0.110474	0.066928	0.1
1	0.0	1.191857	0.266151	0.166480	0.448154	0.060018	-0.082361	-0.078803	0.085102	-0.255425	...	-0.225775	-0.638672	0.101288	-0.339846	0.1
2	1.0	-1.358354	-1.340163	1.773209	0.379780	-0.503198	1.800499	0.791461	0.247676	-1.514654	...	0.247998	0.771679	0.909412	-0.689281	-0.3
3	1.0	-0.966272	-0.185226	1.792993	-0.863291	-0.010309	1.247203	0.237609	0.377436	-1.387024	...	-0.108300	0.005274	-0.190321	-1.175575	0.6
4	2.0	-1.158233	0.877737	1.548718	0.403034	-0.407193	0.095921	0.592941	-0.270533	0.817739	...	-0.009431	0.798278	-0.137458	0.141267	-0.2
...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	
284802	172786.0	-11.881118	10.071785	-9.834783	-2.066656	-5.364473	-2.606837	-4.918215	7.305334	1.914428	...	0.213454	0.111864	1.014480	-0.509348	1.4
284803	172787.0	-0.732789	-0.055080	2.035030	-0.738589	0.868229	1.058415	0.024330	0.294869	0.584800	...	0.214205	0.924384	0.012463	-1.016226	-0.6
284804	172788.0	1.919565	-0.301254	-3.249640	-0.557828	2.630515	3.031260	-0.296827	0.708417	0.432454	...	0.232045	0.578229	-0.037501	0.640134	0.2
284805	172788.0	-0.240440	0.530483	0.702510	0.689799	-0.377961	0.623708	-0.686180	0.679145	0.392087	...	0.265245	0.800049	-0.163298	0.123205	-0.5
284806	172792.0	-0.533413	-0.189733	0.703337	-0.506271	-0.012546	-0.649617	1.577006	-0.414650	0.486180	...	0.261057	0.643078	0.376777	0.008797	-0.4

284807 rows × 31 columns

Exploratory Data Analysis:



Exploratory Data Analysis (EDA) & Data cleaning

```
[3]: df.head()   #to check some initial records of dataset . By default head() is showing 5 records
```

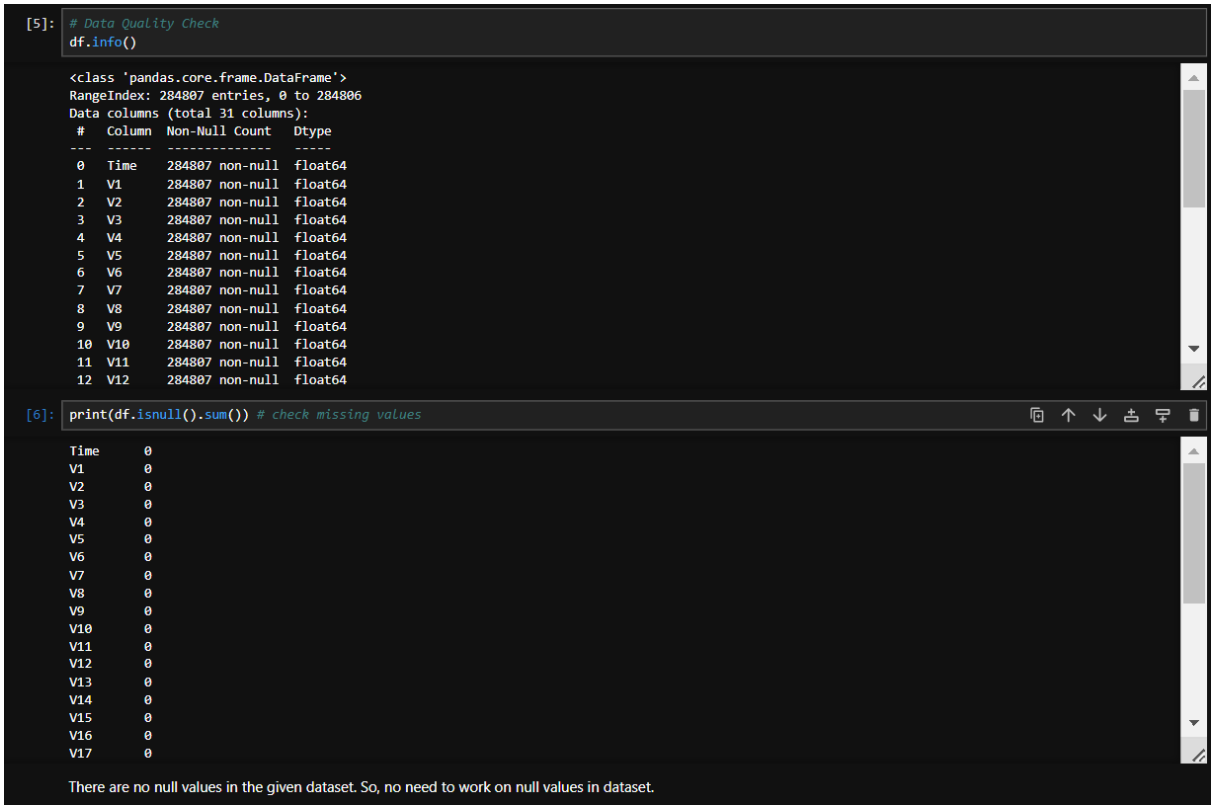
[3]:

	Time	V1	V2	V3	V4	V5	V6	V7	V8	V9	...	V21	V22	V23	V24	V25	
0	0.0	-1.359807	-0.072781	2.536347	1.378155	-0.338321	0.462388	0.239599	0.098698	0.363787	...	-0.018307	0.277838	-0.110474	0.066928	0.128539	-0.1
1	0.0	1.191857	0.266151	0.166480	0.448154	0.060018	-0.082361	-0.078803	0.085102	-0.255425	...	-0.225775	-0.638672	0.101288	-0.339846	0.167170	0.1
2	1.0	-1.358354	-1.340163	1.773209	0.379780	-0.503198	1.800499	0.791461	0.247676	-1.514654	...	0.247998	0.771679	0.909412	-0.689281	-0.327642	-0.1
3	1.0	-0.966272	-0.185226	1.792993	-0.863291	-0.010309	1.247203	0.237609	0.377436	-1.387024	...	-0.108300	0.005274	-0.190321	-1.175575	0.647376	-0.2
4	2.0	-1.158233	0.877737	1.548718	0.403034	-0.407193	0.095921	0.592941	-0.270533	0.817739	...	-0.009431	0.798278	-0.137458	0.141267	-0.206010	0.5

5 rows × 31 columns

```
[4]: print(df.shape)  # To check the dimensions of dataset
     print("So total {} rows and {} columns.".format(df.shape[0],df.shape[1]))
```

```
(284807, 31)
So total 284807 rows and 31 columns.
```



```
[5]: # Data Quality Check
     df.info()
```

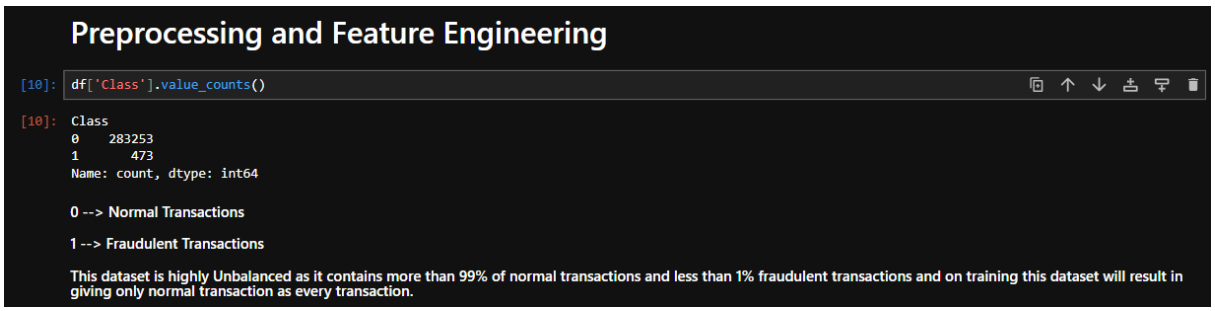
```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 284807 entries, 0 to 284806
Data columns (total 31 columns):
 #   Column  Non-Null Count   Dtype
---  ------  --------------   -----
 0   Time    284807 non-null  float64
 1   V1      284807 non-null  float64
 2   V2      284807 non-null  float64
 3   V3      284807 non-null  float64
 4   V4      284807 non-null  float64
 5   V5      284807 non-null  float64
 6   V6      284807 non-null  float64
 7   V7      284807 non-null  float64
 8   V8      284807 non-null  float64
 9   V9      284807 non-null  float64
 10  V10     284807 non-null  float64
 11  V11     284807 non-null  float64
 12  V12     284807 non-null  float64
```

```
[6]: print(df.isnull().sum()) # check missing values
```

```
Time    0
V1      0
V2      0
V3      0
V4      0
V5      0
V6      0
V7      0
V8      0
V9      0
V10     0
V11     0
V12     0
V13     0
V14     0
V15     0
V16     0
V17     0
```

There are no null values in the given dataset. So, no need to work on null values in dataset.

Preprocessing and Feature Engineering:



Preprocessing and Feature Engineering

```
[10]: df['Class'].value_counts()
```

```
[10]: Class
      0    283253
      1       473
      Name: count, dtype: int64
```

0 --> Normal Transactions

1 --> Fraudulent Transactions

This dataset is highly Unbalanced as it contains more than 99% of normal transactions and less than 1% fraudulent transactions and on training this dataset will result in giving only normal transaction as every transaction.

It contains only numeric input variables which are the result of a PCA transformation. Unfortunately, due to confidentiality issues, we cannot provide the original features and more background information about the data. Features V1, V2, ... V28 are the principal components obtained with PCA, the only features which have not been transformed with PCA are 'Time' and 'Amount'. 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-dependant cost-sensitive learning. Feature 'Class' is the response variable and it takes value 1 in case of fraud and 0 otherwise. There are no "Null" values, so we don't have to work on ways to replace values.

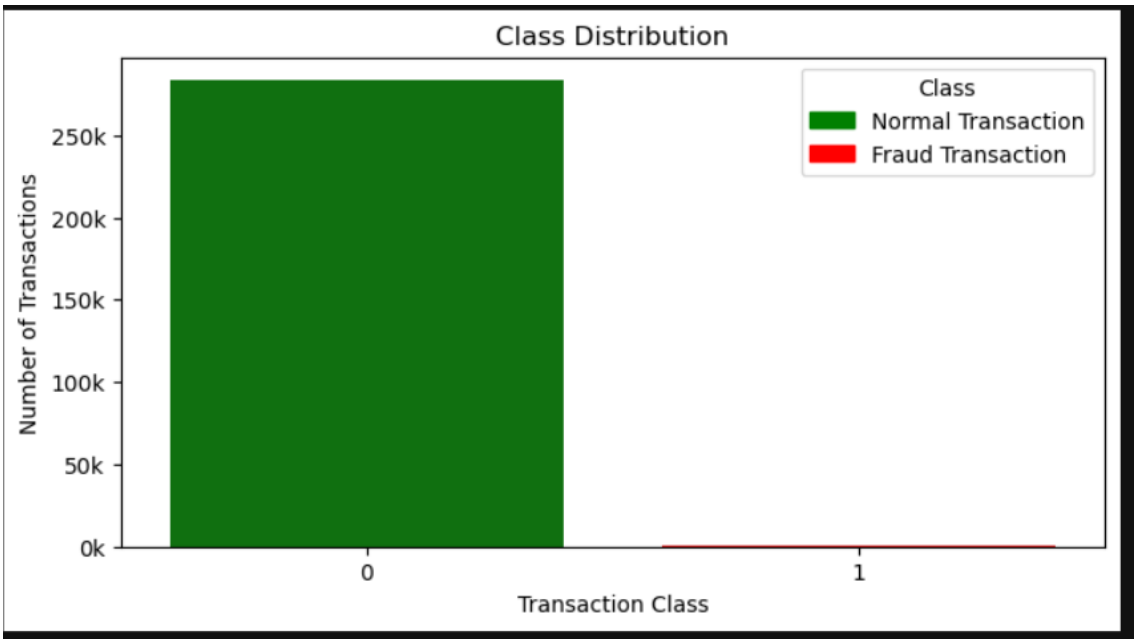
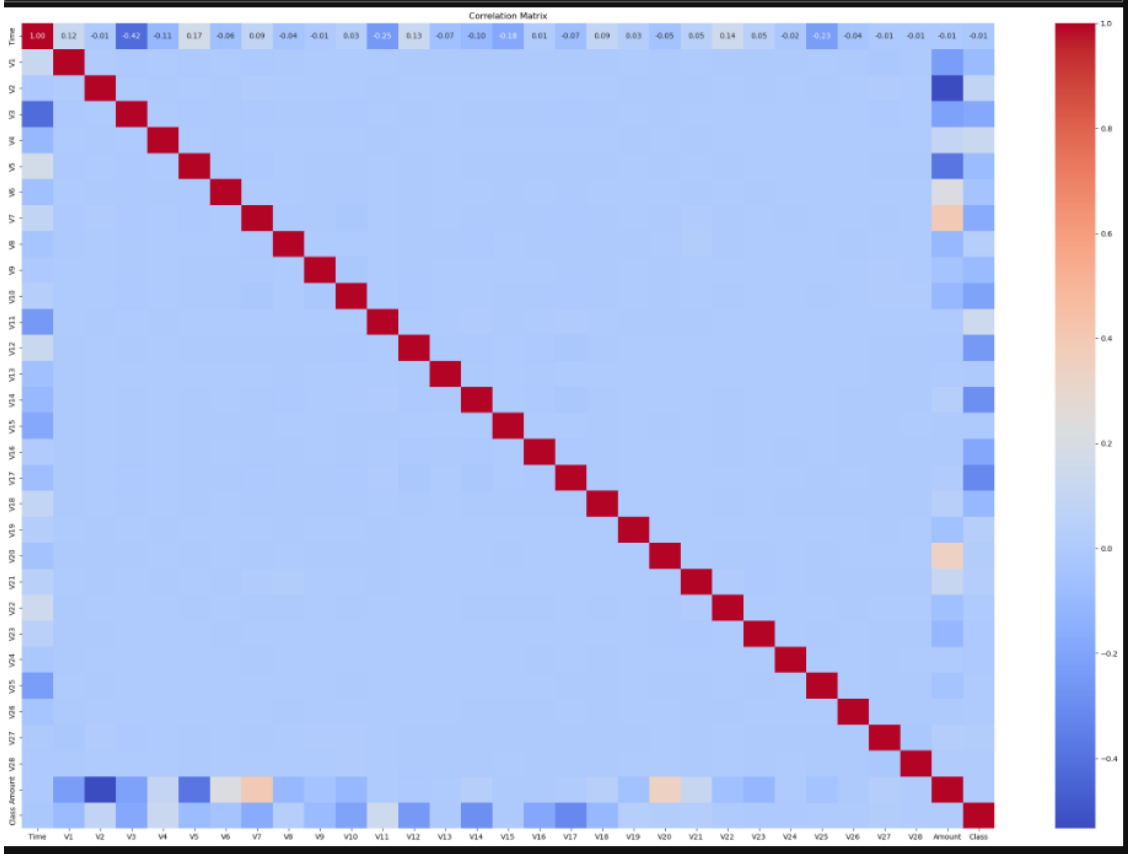


Fig.3 Class Distribution

```
# Plotting Heatmap for above Correlation
plt.figure(figsize=(30,20))
sns.heatmap(df.corr(),annot=True, fmt='.2f',cmap='coolwarm')
plt.title('Correlation Matrix')
plt.show();
```



Seprating the data for analysis

Notice how imbalanced is our original dataset is! Most of the transactions are non-fraud. If we use this data frame as the base for our predictive models and analysis, we might get a lot of errors and our algorithms will probably overfit since it will "assume" that most transactions are not fraud. But we don't want our model to assume, we want our model to detect patterns that give signs of fraud!

Credit card dataset is highly imbalanced dataset because it carries more legitimate transactions as compared to the fraudulent one. That means prediction will get very high accuracy score without detecting a fraud transaction. When we apply any classification algorithm directly on the dataset we get the following results directly applying the classification:

Dealing with Imbalanced data:

Now you can take sample or you can go with all data like 80% percent for training and 20% for sampling. But yes you will get more accuracy if your training data is more. When you dealing with imbalanced data at that moment you can segregate the sample so that you will get sample modeling. Once your model is trained you can give sample data to check the accuracy whether it is working smoothly or not.

Now for that so many methods are there. Using SMOTE you balance the class distribution in the dataset. This helps ensure that the minority class is not underrepresented when training the models. I show all this methods in my Final_pro.ipynb file. You can check there I shared the link and you will get an idea. Only the thing is in that file I comment some code. For this project, I go through small sample dataset definitely, accuracy would be less but you will know the idea about the data.

Summary:

- There are 492 cases of fraud in our dataset so we can randomly get 492 cases of non-fraud to create our new sub data frame.
- We concatenate the 492 cases of fraud and non-fraud, creating a new sub-sample.

Scaled amount and scaled time are the columns with scaled values:

Before proceeding with the Random Under Sampling technique we have to separate the original data frame for testing purposes, although we are splitting the data when implementing Random Under Sampling or Oversampling techniques, we want to test our models on the original testing set, not on the testing set created by either of these techniques. The main goal is to fit the model either with the data frames that were under sample and oversample (in order for our models to detect the patterns), and test it on the original testing set.

One of the most common and simplest strategies to handle imbalanced data is to under sample the majority class.
Oversampling the minority class can result in overfitting problems if we oversample before cross-validating.

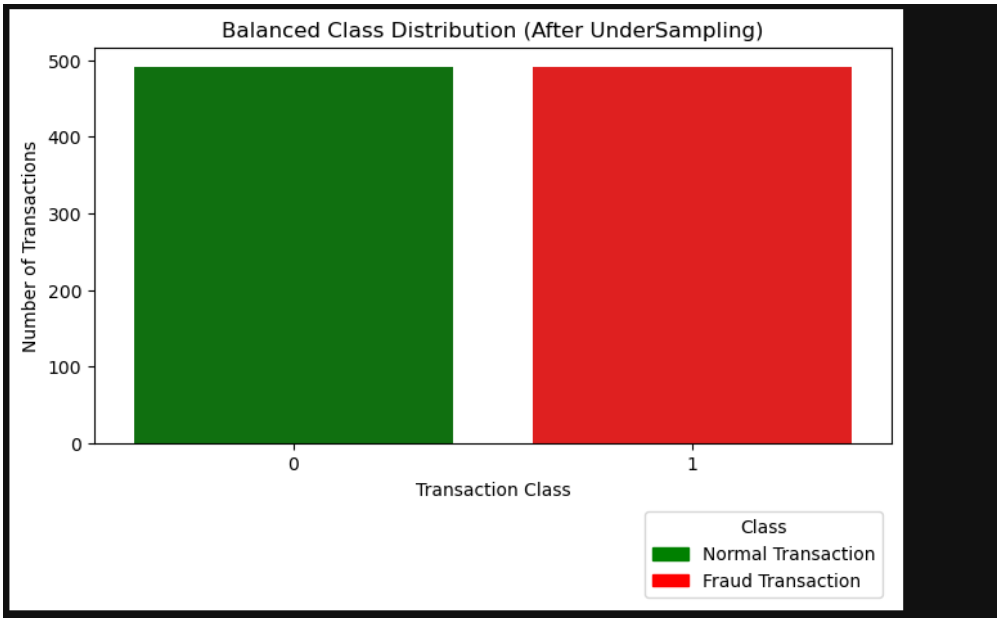
Under Sampling :

In this phase of the project we will implement *"Under Sampling"* which basically consists of removing data in order to have a more balanced dataset and thus avoiding our models to overfitting.[4]

- The first thing we have to do is determine how imbalanced is our class (use "value_counts()" on the class column to determine the amount for each label)
- Once we determine how many instances are considered fraud transactions (Fraud = "1") , we should bring the non-fraud transactions to the same amount as fraud transactions (assuming we want a 50/50 ratio), this will be equivalent to 492 cases of fraud and 492 cases of non-fraud transactions.
- After implementing this technique, we have a sub-sample of our data frame with a 50/50 ratio with regards to our classes. Then the next step we will implement is to shuffle the data to see if our models can maintain a certain accuracy every time, we run this script.

Note: The main issue with "Random Under-Sampling" is that we run the risk that our classification models will not perform as accurate as we would like to since there is a great deal of information loss (bringing 492 non-fraud transaction from 284,315 non-fraud transactio

Now that we have our dataframe correctly balanced, we can go further with our **analysis** and data preprocessing.



Now that we have sampled our data set, we apply our supervised learning models.

Model Selection:

Model	Accuracy	Balanced Accuracy	ROC AUC	F1 Score	Time Taken
DecisionTreeClassifier	1.00	1.00	1.00	1.00	25.99
XGBClassifier	1.00	1.00	1.00	1.00	2.63
ExtraTreeClassifier	1.00	1.00	1.00	1.00	0.61
ExtraTreesClassifier	1.00	1.00	1.00	1.00	25.51
RandomForestClassifier	1.00	1.00	1.00	1.00	341.13
BaggingClassifier	0.98	0.98	0.98	0.98	185.94
QuadraticDiscriminantAnalysis	0.92	0.92	0.92	0.92	0.86
SVC	0.91	0.91	0.91	0.91	321.61
KNeighborsClassifier	0.90	0.90	0.90	0.90	0.88
GaussianNB	0.90	0.90	0.90	0.90	0.50
NearestCentroid	0.88	0.88	0.88	0.88	0.42
LinearDiscriminantAnalysis	0.88	0.88	0.88	0.88	1.17
Perceptron	0.88	0.88	0.88	0.88	0.84
CalibratedClassifierCV	0.88	0.88	0.88	0.87	5.03
AdaBoostClassifier	0.85	0.85	0.85	0.85	96.73
PassiveAggressiveClassifier	0.82	0.82	0.82	0.82	0.76
BernoulliNB	0.82	0.82	0.82	0.81	0.54
LogisticRegression	0.80	0.80	0.80	0.80	0.97
LGBMClassifier	0.79	0.79	0.79	0.78	1.89
LinearSVC	0.79	0.79	0.79	0.78	1.32
SGDClassifier	0.75	0.75	0.75	0.74	0.97
RidgeClassifier	0.70	0.70	0.70	0.67	0.50
RidgeClassifierCV	0.70	0.70	0.70	0.67	1.10
DummyClassifier	0.50	0.50	0.50	0.33	0.35

As you can see in above table all these are model and you can see all f1_score and accuracy.

5. Future scope

With increasing number of bank fraudulency and cyber crime cases, need of a secure testing system is on rise. And this is a direct solution to this problem. It can be extended to a duplex verification of not only a customer(debit-ant) but also of the seller(credit-ant). It can be taken and used on a regular basis just like OTP. It can be used to even assess past transaction in database to find whether certain transactions were fraudulent or not and also would be able to produce evidence in such cases. Also optimization techniques on the proposed models and testing on new models can be done.

6. Conclusion

Although there are several fraud detection techniques available today, none is able to detect all frauds completely when they are actually happening, they usually detect it after the fraud has been committed. This happens because a very minuscule number of transactions from the total transactions are actually fraudulent in nature. So we need a technology that can detect the fraudulent transaction when it is taking place so that it can be stopped then and there and that too in a minimum cost. So the major task of today is to build an accurate, precise and fast detecting fraud detection system for credit card frauds that can detect not only frauds happening over the internet like phishing and site cloning but also tampering with the credit card itself i.e. it signals an alarm when the tampered credit card is being used. The major drawback of all the techniques is that they are not guaranteed to give the same results in all environments. They give better results with a particular type of dataset and poor or unsatisfactory results with other type. Thus, the results are purely dependent on the dataset type used.

In our undersample data, our model is unable to detect for a large number of cases, the non fraud transactions correctly and instead, mis-classifies those non fraud transactions as fraud cases. Imagine that people that were making regular purchases got their card blocked due to the reason that our model classified that transaction as a fraud transaction, this will be a huge disadvantage for the financial institution. The number of customer complaints and customer dissatisfaction will increase. The next step of this analysis will be to do an outlier removal on our undersample dataset and see if our accuracy in the test set improves.

In this paper, Machine learning technique like Logistic regression, Decision Tree, Random forest classifiers were used to detect the fraud in credit card system. Sensitivity, Specificity, accuracy and error rate are used to evaluate the performance for the proposed system. From the experiments, the result that has been concluded is that Logistic regression has an accuracy of 94.4%,Decision tree shows accuracy of 91.9% and Random forest shows accuracy of 92.9% .. Hence we conclude that Logistic regression is the best model for our system.

7. References:

- Behrouz, Emad, and Sahil. "Credit Card Fraud Detection Using Supervised Machine Learning Algorithms." *Journal of King Saud University - Computer and Information Sciences*. Link to Paper.

- Khare, Navanshu, and Saad Yunus Sait. "A Survey on Credit Card Fraud Detection Using Machine Learning Techniques." *Procedia Computer Science*. Link to Paper.

- Bhattacharyya, Siddhartha, et al. "Data Mining for Credit Card Fraud: A Review." *Journal of King Saud University - Computer and Information Sciences*. Link to Paper.