Database Privacy Using Machine Learning

(CREDIT CARD FRAUD DETECTION)

Submitted by

Ankita Kar[RA1811003010974]

Ankita Bose[RA1811003010973]

Under the guidance of

Dr.B.Arthi

(Associate Professor,

Department of Computer Science and

Engineering)

In partial fulfillment for the award of the degree

of

BACHELOR OF TECHNOLOGY

in

COMPUTER SCIENCE & ENGINEERING

of

FACULTY OF ENGINEERING AND TECHNOLOGY

S.R.M. Nagar, Kattankulathur, Chengalpattu District

NOVEMBER 2021

SRM INSTITUTE OF SCIENCE AND TECHNOLOGY

(Under Section 3 of UGC Act, 1956)

BONAFIDE CERTIFICATE

Certified that 18CSP107L minor project report titled "**Database Privacy Using Machine Learning (CREDIT CARD FRAUD DETECTION)**" is the bonafide work of "**Ankita Kar [RA1811003010974], Ankita Bose [RA1811003010973]**" who carried out the minor project work under my supervision. Certified further, that to the best of my knowledge the work reported herein does not form any other project report or dissertation on the basis of which a degree or award was conferred on an earlier occasion on this or any other candidate.

SIGNATURE

Dr.B.Arthi
Associate Professor
Dept. of Computing Technologies

SIGNATURE

Dr.M.Pushpalatha
HEAD OF THE DEPARTMENT
Professor
Dept. of Computing Technologies

Signature of the Panel Head

Dr.B.Arthi
Associate Professor

ABSTRACT

With the advancement of technology and E-Commerce, credit card transaction has gained popularity by making our day to day life so simpler. At the same time, by misusing the advanced new technologies, fraud and fallacious activities have been developed to a great extent. The main target of all these are online transactions. To deal with such issues and to detect frauds, a very powerful detection technique is required, which can alert the user not after the fraud has happened but before the fraud occurs. Different approaches to machine learning can be employed to predict suspicious and non suspicious transactions by implementing numerous classification algorithms. For detecting credit card anomaly, in this project we are planning to analyze and compare some popular classifier algorithms. More focus will be given on the performance of the classifiers. This can be helpful for the bank and financial organizations, to detect the fraud at the early stage, and then they can reduce the ongoing fraud by not accepting the suspected transactions.

TABLE OF CONTENTS

LIST OF FIGURES

ABBREVIATIONS

HMM Hidden Markov Model

SVM Service Vector Machine

RFC Random Forest Classifier

CM Correlation Matrix

CM Confusion Matrix

CV Computer Vision

DB Database

& And

Chapter 1

INTRODUCTION

1.1 Objective

- Illegal use of a credit card or its information without the knowledge of the owner is called credit card fraud. In this project we will be maintaining digital privacy (i.e. by checking whether the credit card function is legal or false and notifying users in advance.) Using a machine learning algorithm.
- The system prevents fraudulent users from misusing credit card information of real users for their own benefit. Credit card holder spending habits detect fraud. Since the fake user may not be aware of the owner's operating habits, there will be a mismatch in the spending pattern, which the system will detect. The owner is immediately notified of the fraudulent attempt and the activity is blocked. Thus, the system protects legitimate users from financial losses. The system helps to make electronic payments safer and more reliable.
- Workflow:
- Credit Card Data-> Pre-Data Processing -> Data Analysis-> Train Exam Split-> Resource Reduction Model (because it is a Double Duplication program.) -> Testing

1.2 Challenges to Address

- Huge amount of data is processed daily and model construction should be fast enough to respond to a scam early. Imbalanced Data i.e. most transactions (99.8%) are not fake which makes it really difficult to find fake ones.
- Data availability as data is very private. Incorrectly sorted data can be another major

problem, as not all fraudulent activity has been detected and reported. Practice methods used by fraudsters against the model.

- How to combine in-depth reading, Random Forests, Line Backing, K star algorithm for more accurate machine reading analysis.
- Use the most accurate binary algorithm for dividing multiple categories.
- Try to use labeled data and a more efficient algorithm to improve the performance of fraud detection.

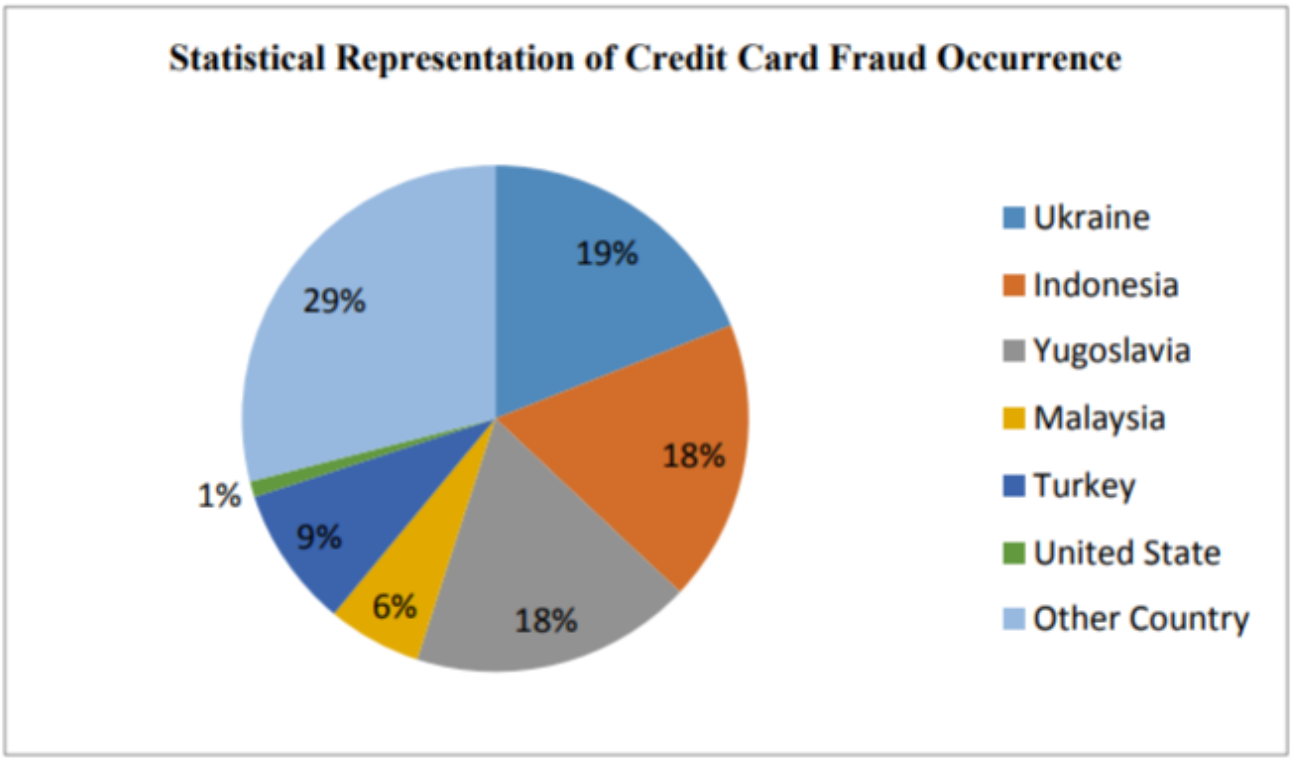


Fig.-1: Showing the Countries facing Credit Card Fraud

1.3 Scope

- Detect the fraudulent transaction
- Minimization of Credit card Fraud
- Analysis of multiple Machine Learning algorithms (In order to get better performance and accuracy.)
- Future Scope:
- The idea in the proposed system can also be adopted and implemented in other electronic payment services such as online banking facilities and payment gateways.
- In case of more complex dataset instead of linear regression, we can use an artificial neural network.

Chapter 2

LITERATURE SURVEY

2.1 The Use of Predictive Machine Learning and Decision Tree Approach to Detect Credit Card Fraud

Oversampling and Undersampling is performed in ml approach. Misclassification cost is considered in pruning step of decision tree approach. Under sampling obtained a good result. And also the study using misclassification cost has made a significant improvement in fraud detection.

2.2 Research on Credit Card Fraud Detection Model Based on Misuses (Supervised) Technique and Anomaly Detection (Unsupervised) Technique

The different models used are decision tree, neural network, rule induction. The behaviour of the user's model is extracted and accordingly classified as fraudulent or not. With the help of this dataset classification model created, we can predict whether the data is fraud or not. This has obtained a successful result.

2.3 Hidden Markov Model (HMM) Learning for Credit Card Fraud Detection

A hidden Markov Models represents a finite number of states with sufficiently high probability. The transitions between the states are handled by these probability values. If the incoming transactions are not accepted by the trained HMM with high probability it is considered as fraudulents otherwise not. This can detect fraud transactions to an extent. It is scalable in handling large amount of datum.

2.4 Detecting Credit Card Fraud by Cortical Algorithm

1. A sparse representation of the input is initially formed.
2. A representation is formed based on the previous input.
3. Finally a prediction is made based on previous step

This study provides a nice way to detect credit card fraud transactions.

2.5 Detecting Credit Card Fraud by Dempster Shafer Adder and Bayesian Learner

The incoming transaction is initially handled by the rule base using probability values.
This is flexible such that new kinds of fraud can be handled easily.

2.6 Detecting Credit Card Fraud by Supervised Learning Algorithm called Linear Discriminant/Fisher Discriminant.

Linear Perceptron Discriminant function is being used which can solve all the problems. This method can label transactions with high usable limit on the card correctly which leads to prevent losing millions of dollars in real life banking systems.

Chapter 3

SYSTEM ARCHITECTURE AND DESIGN

3.1 Details about Data & Analysis

• The data.head() function is used to have a peek look into the .csv file which shows what the dataset is.
• The data.describe() feature is used to define the class of various columns in dataset to find out the fraudulent and real cases of credit card transactions taking place.
• The places where class is 0 are the valid cases and the places where the class is 1 are the invalid cases.
• The outliner fraction is calculated thus with fraudulent and valid cases.
• Description of fraud and valid transaction in terms of mean and standard deviation at various percentage is also found out.

3.2 Correlation Matrix, Dividing the data into Training and Testing Purpose

• A correlation matrix is plotted to find out correlation between various columns of dataset in form of heat map.
• Any irregular correlation from the graph tells us about which column can give us invalid transaction result
• The data is divided into x and y value for training and testing purpose to be used in Random Forest Classifier
• Accuracy, precision, f1 score, Mathews correlation factor is also found out

3.3 Confusion matrix to find out true positive, true negative, false positive and false negative

• The confusion matrix is also implemented in this project on the basis of total fraud and valid cases
• The confusion matrix gives us the taste of the classifier used in our project.

3.4 Module Description

The system prevent fraudulent users from misusing the details of the credit-card of the genuine users for their personal gain. The spending habits of the credit-card owner detect the fraud. As the fake users might not be aware of the spending habits of the owner, there will be an irregularity in the spending pattern, which the system will detect. The owners are immediately alerted about the attempted fraud and the transaction is blocked. Thus, the system protect legitimate user from financial loss. The system help in making electronic payment safer and more reliable.

We include few basic modules:

1.Frame the problem
2. Collect the Raw data(data downloaded from kaggle)
3. Import the Libraries (Pandas, Numpy, Sklearn etc.)
4. Data Processing(using Logistic Regression)
A. Distribution of legit and fraudulent transaction
B. Statistical measures of the data
C. Split the data into training data and test data
D. Training the logistic regression model with training data
5. Evaluation(checking accuracy on training and test data)
In the dataset we are given the following information: Location, Time, Amount, Other Info

3.4.1 Workflow

Credit card Data-> Data Pre-processing -> Data Analysis->Train Test Split-> Logistic Regression model(because it's binary Classification program.)-> Evaluation

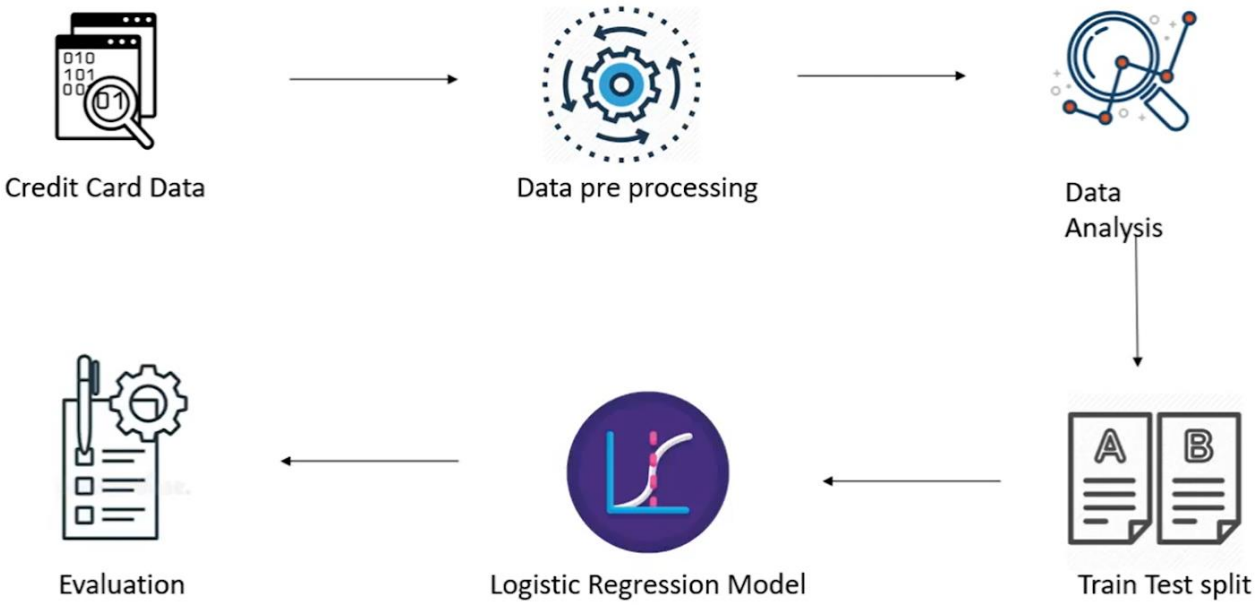


Fig: Workflow of the project

3.4.2 Block Diagram

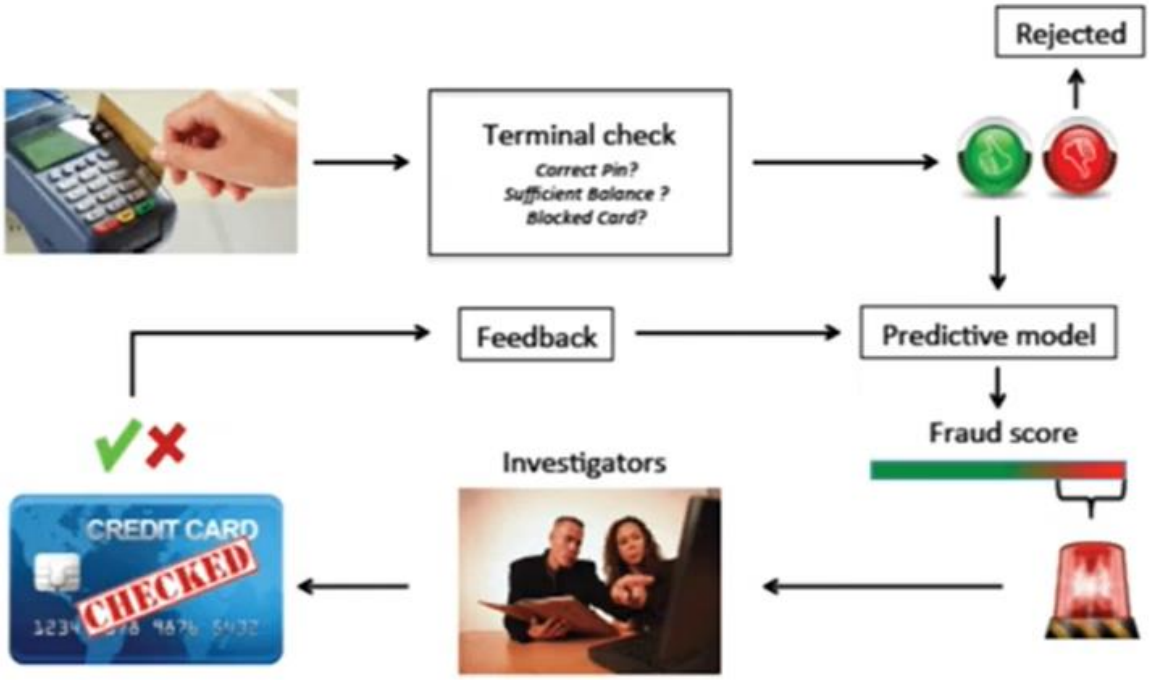


Fig. Fraud detection process

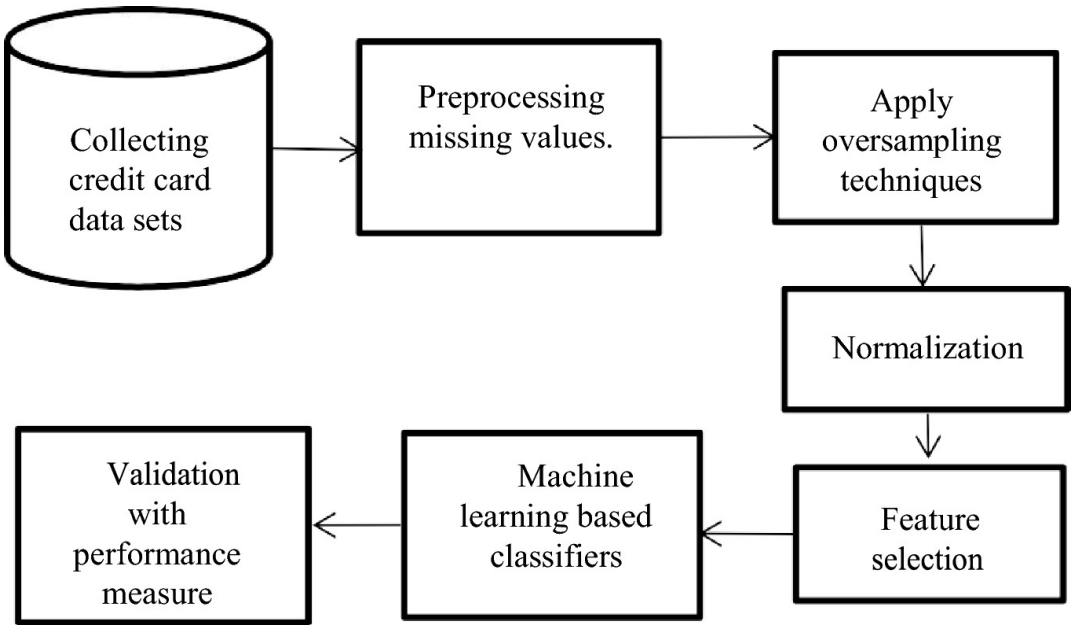


Fig: Block Diagram of the project

3.4.3 Data Flow Diagram

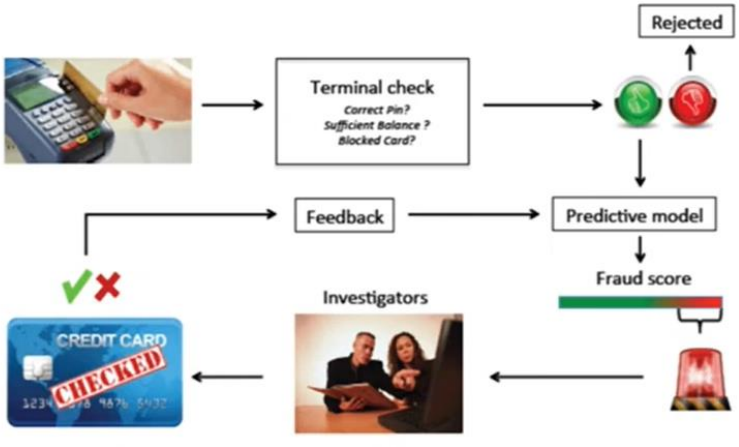


Fig. Fraud detection process

Fig: The Dataflow diagram

Chapter 4

METHODOLOGY

4.1 Random Forest Algorithm

A random forest is a type of machine-readable learning an algorithm based on integrated learning. Learning ensemble is type of learning when joining different types of algorithms or the same algorithm many times to do more a powerful predictive model. Random forest algorithm combines multiple algorithms of the same type i.e. multiple pruning trees, leading to a grove of trees, hence the name "Random Forest". A random forest algorithm can be used in both retreat and editing functions.

4.1.1 Working of Random Forest Classifier

The following are the basic steps involved in creating a random forest algorithm:

1. Select N random records in the database.
2. Build a decision tree based on these N record.
3. Select the number of tree you want in your algorithms and repeat steps 1 and 2.
4. With the problem of segregation, each tree in the forest predicts the stage a new record belongs to. Finally, a new record is allocated to the category that wins the most votes.

4.1.2 Implementation of Random Forest Algorithm

• To get a better accuracy we've implemented it using Random Forest Classifier .

• A random forest classifier builds forests by taking various decision trees together.

• Works on the principle of cross-validation.

• Takes multiple data points from the dataset and builds a forest by incorporating various decision trees together.

• Takes the mean value of all the decisions to get better accuracy.

• This algorithm is very stable. Even if new data points are introduced in the dataset the overall thing is not affected much since new data may impact one tree, but it is very difficult for it to impact all the trees.

• The random forest algorithm work good when you have both categorical and numerical features.

• The random forest algorithm also works nicely when data has missing values or it has not been scaled well.

4.1.3 Random Forest Diagrammatic Representation

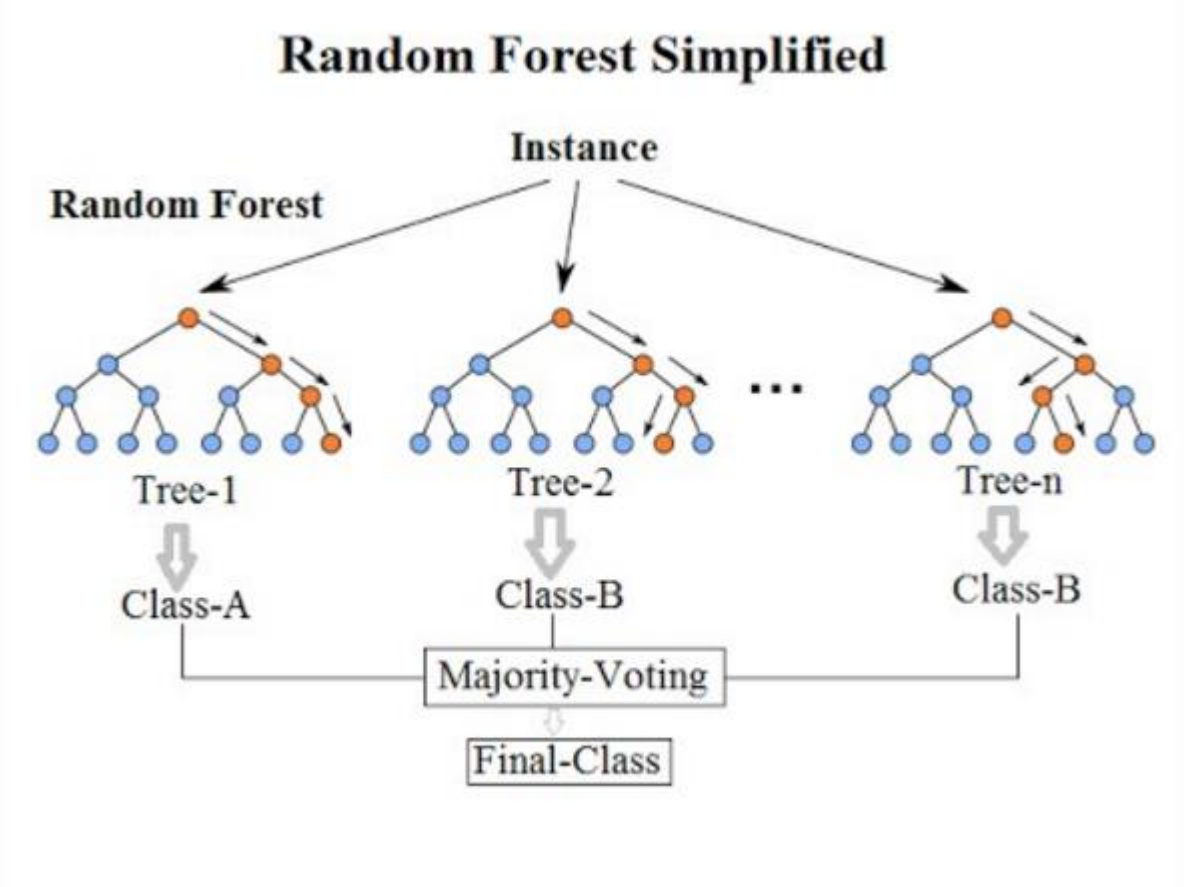


Fig: Diagrammatic representation of Random Forest algorithm

4.2 Linear Regression Algorithm

Linear Regression is a supervised machine learning algorithms where the predicted output is continuous and has a right slope. The goal of the linear regression method is to get the good values for a0 and a1 to find the best fit line. The best fit line should have the least errors, which means the error between predicted values and actual values should be minimized. It performs a regression task.

4.2.1 Working of Linear Regression Algorithm

Linear regression algorithm performs the job of predicting a dependent variable value (y) based on a given independent variable (x). So, this technique finds out linear relationships between these two, x (which is input) and y(which is output).
The following are the basic steps involved in creating a Linear Regression Algorithm:

1. Select N random records in the database.
2. Then train the machine using 80% of database.
3. Then test it against trained valued accuracy.
4. Then compare the accuracy.

4.2.2 Advantages of Linear Regression Algorithm

The benefits of using the informal forest for planning and relocation are:
1. It is simple to implement and easier to interpret the output coefficients.
2. This algorithm is susceptible to over-fitting but it can be avoided using some dimensionality reduction techniques, regularization (L1 and L2) techniques and cross-validation.
3. Linear regression fits linearly separable dataset almost perfectly and is often used to find the nature of the relationship between variables.

Chapter 5

CODING & TESTING

5.1 Coding part of Random Forest Algorithm

5.1.1 Starting the Project

- Loading the dataset using pandas.
- The dataset which is in form of a .csv file is downloaded beforehand.
- The dataset is loaded using pd.read_csv which is using to read the csv file under pandas.
- The file path is passed as an argument for pd.read_csv(file path).

```
# Data Manipulation and Linear Algebra
import pandas as pd
import numpy as np

# Plots
import seaborn as sns
sns.set_style("darkgrid")
import matplotlib.pyplot as plt

# Machine Learning
from sklearn.model_selection import StratifiedShuffleSplit, cross_val_score, cross_val_predict
from sklearn.metrics import accuracy_score, confusion_matrix, classification_report, precision_recall_curve, roc_curve
from sklearn.decomposition import PCA

from sklearn import tree, linear_model, ensemble

#ignore warning messages
import warnings
warnings.filterwarnings('ignore')
```

```
[ ] data = pd.read_csv("https://datahub.io/machine-learning/creditcard/r/creditcard.csv")
    data
```

5.1.2 Preparing the Data

Reducing the Number of Features in the Dataset using PCA. This is known as Dimensionality Reduction.

	PCA1	PCA2	Class
0	61.271382	1.319417	'0'
1	-85.661826	-1.043781	'0'
2	290.316696	0.810947	'0'
3	35.151659	0.928410	'0'
4	-18.360281	1.317441	'0'
...	...	...	...
284802	-87.586281	13.128644	'0'
284803	-63.560584	0.876877	'0'
284804	-20.470739	-1.970701	'0'
284805	-78.350638	0.408176	'0'
284806	128.652188	0.358723	'0'

284807 rows × 3 columns

5.1.3 Stratified Train Test Split

The data is divided into x and y value for training and testing purpose to be used in Random Forest Classifier in this project.

```
[ ]  split = StratifiedShuffleSplit(n_splits=1, test_size=0.2, random_state=42)
     for train_index, test_index in split.split(full_data, full_data['Class']):
         train = full_data.loc[train_index]
         test = full_data.loc[test_index]
```

```
[ ]  X_train = train.drop("Class", axis=1)
     y_train = train["Class"]

     X_test = test.drop("Class", axis=1)
     y_test = test["Class"]
```

5.1.4 Accuracy Check

```
[ ]  rf_clf = ensemble.RandomForestClassifier()

     MLA_testing(rf_clf, X_train, X_test, y_train, y_test)
```

```
K-Fold Accuracies:
 [0.9984639  0.99863946 0.99824446 0.99828835 0.99837612 0.99850772
 0.99859551 0.99859551 0.99841994 0.99837605]

Accuracy Score:
 0.9982971103542713

Confusion Matrix:
 [[56857     7]
 [   90     8]]

Classification Report:
               precision    recall  f1-score   support

          '0'       1.00      1.00      1.00     56864
          '1'       0.53      0.08      0.14        98

    accuracy                           1.00     56962
   macro avg       0.77      0.54      0.57     56962
weighted avg       1.00      1.00      1.00     56962


--------------------------------------------------------------------------------
```

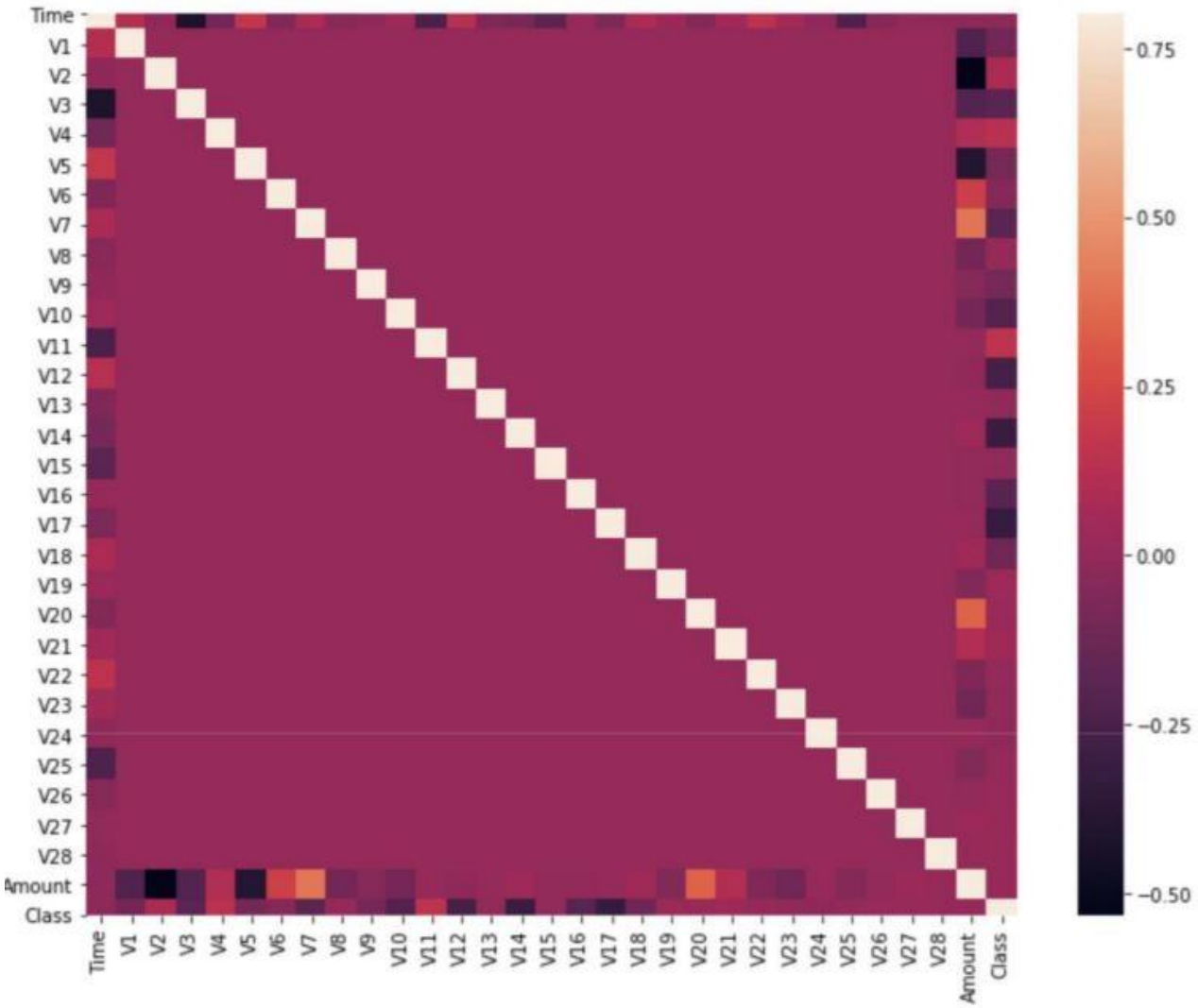
5.1.5 CODE IMPLEMENTATION

Accuracy, precision, f1 score, Mathews correlation factor is also found out for the given dataset. Here, the Correlation Matrix is found out which gives us the value of true positive, true negative, false positive, false negative.

```
K-Fold Accuracies:
 [0.9984639  0.99859557 0.99824446 0.99828835 0.99837612 0.99837605
 0.99859551 0.99846383 0.99841994 0.99837605]

Accuracy Score:
 0.9982444436641972

Confusion Matrix:
 [[56854    10]
 [   90     8]]

Classification Report:
              precision    recall  f1-score   support

         '0'       1.00      1.00      1.00     56864
         '1'       0.44      0.08      0.14        98

    accuracy                           1.00     56962
   macro avg       0.72      0.54      0.57     56962
weighted avg       1.00      1.00      1.00     56962
```

5.1.6 Plotting of Correlation Matrix

The correlation matrix graphically gives us an idea of how features match up with each other and can help us predict what are the features that are really relevant for the prediction in this project.



In the Heat Map shown in the picture, we can clearly see that most of the features do not match up to other features but there are some features that either has a +ve or a negative correlation with each other. For example, V2 and V5 are high -vely correlated with the feature called Amount. We also see some correlation between V20 and Amount. This gives us a really good understanding of the Data which are in our hands.

5.2 Coding part of Linear Regression Algorithm

5.2.1 Starting the Project

- Loading the dataset using pandas.
- The dataset which is in form of a .csv file is downloaded beforehand.
- The dataset is loaded using pd.read_csv which is using to read the csv file under pandas.
- The file path is passed as an argument for pd.read_csv(file path).

```
[ ] #Print first 5 rows of the dataset
    credit_card_data.head()
```

	Time	V1	V2	V3	V4	V5	V6	V7	V8	V9	V10	V11	V12	V13	V14	V15	V16	V17	
0	0	-1.359807	-0.072781	2.536347	1.378155	-0.338321	0.462388	0.239599	0.098698	0.363787	0.090794	-0.551600	-0.617801	-0.991390	-0.311169	1.468177	-0.470401	0.207971	0.02
1	0	1.191857	0.266151	0.166480	0.448154	0.060018	-0.082361	-0.078803	0.085102	-0.255425	-0.166974	1.612727	1.065235	0.489095	-0.143772	0.635558	0.463917	-0.114805	-0.18
2	1	-1.358354	-1.340163	1.773209	0.379780	-0.503198	1.800499	0.791461	0.247676	-1.514654	0.207643	0.624501	0.066084	0.717293	-0.165946	2.345865	-2.890083	1.109969	-0.12
3	1	-0.966272	-0.185226	1.792993	-0.863291	-0.010309	1.247203	0.237609	0.377436	-1.387024	-0.054952	-0.226487	0.178228	0.507757	-0.287924	-0.631418	-1.059647	-0.684093	1.96
4	2	-1.158233	0.877737	1.548718	0.403034	-0.407193	0.095921	0.592941	-0.270533	0.817739	0.753074	-0.822843	0.538196	1.345852	-1.119670	0.175121	-0.451449	-0.237033	-0.03

```
[ ] #print last 5 rows of the dataset
    credit_card_data.tail()
```

	Time	V1	V2	V3	V4	V5	V6	V7	V8	V9	V10	V11	V12	V13	V14	V15	V16	V17
19893	30631	-0.377215	0.973528	1.647077	0.732439	0.024728	-0.541379	0.828488	-0.060740	-0.725148	-0.450153	0.242824	0.488841	0.559073	0.137418	0.863233	-0.415339	-0.029005
19894	30631	1.209281	0.078793	0.061820	0.593730	-0.235772	-0.448524	-0.141196	0.089236	0.411825	-0.263041	-0.572076	-1.062719	-2.106307	0.195707	1.634883	0.404446	0.278583
19895	30632	1.286596	-1.450336	0.814530	-1.308949	-2.055209	-0.592064	-1.317286	0.032386	-1.720017	1.589335	1.187759	-0.705883	-0.567504	-0.062561	0.250012	0.047145	0.259016
19896	30633	-0.488090	1.018448	0.670593	-0.245462	0.828347	-0.233102	0.662586	-0.040028	-0.279439	-0.402822	-1.387400	-0.332092	0.764095	-0.630524	0.970633	0.765937	-0.533608
19897	30633	-2.609841	2.479357	0.763844	0.044509	-0.645716	0.762867	-1.626415	-7.617854	1.399746	1.000000	NaN	NaN	NaN	NaN	NaN	NaN	NaN

5.2.2 Dataset Information

```
#dataset information
credit_card_data.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 19898 entries, 0 to 19897
Data columns (total 31 columns):
 #   Column  Non-Null Count  Dtype  
---  ------  --------------  -----  
 0   Time    19898 non-null  int64  
 1   V1      19898 non-null  float64
 2   V2      19898 non-null  float64
 3   V3      19898 non-null  float64
 4   V4      19898 non-null  float64
 5   V5      19898 non-null  float64
 6   V6      19898 non-null  float64
 7   V7      19898 non-null  float64
 8   V8      19898 non-null  float64
 9   V9      19898 non-null  float64
 10  V10     19898 non-null  float64
 11  V11     19897 non-null  float64
 12  V12     19897 non-null  float64
 13  V13     19897 non-null  float64
 14  V14     19897 non-null  float64
 15  V15     19897 non-null  float64
 16  V16     19897 non-null  float64
 17  V17     19897 non-null  float64
 18  V18     19897 non-null  float64
 19  V19     19897 non-null  float64
 20  V20     19897 non-null  float64
 21  V21     19897 non-null  float64
 22  V22     19897 non-null  float64
 23  V23     19897 non-null  float64
 24  V24     19897 non-null  float64
 25  V25     19897 non-null  float64
 26  V26     19897 non-null  float64
```

5.2.3 Distribution of Transactions

```
[ ]  #distribution of legit and fraudulent transaction
     credit_card_data['Class'].value_counts()
```

```
0.0    19812
1.0       85
Name: Class, dtype: int64
```

```
[ ]
```

5.2.4 Preparing the Data

Performing Under sampling, to produce a balanced dataset.

Number of fraudulent transactione->85

```
[ ]  legit_sample = legit.sample(n=492)
```

```
[ ]
```

Concatenating two data frames

```
[ ]  new_dataset=pd.concat([legit_sample, fraud], axis=0)
```

```
[ ]  new_dataset.head()
```

	Time	V1	V2	V3	V4	V5	V6	V7	V8
4304	3758	-0.693825	-0.158285	1.621470	-1.644706	-0.647281	-0.398048	0.548695	-0.066878
6086	6942	1.475676	-0.082584	-0.176180	-0.551063	-0.255257	-1.112522	-0.028757	-0.549704
4246	3754	1.239582	-0.421236	0.792134	-0.469097	-0.762769	-0.213225	-0.667352	-0.094037
13550	24036	1.017670	-0.719840	0.285711	0.886092	1.162706	4.830132	-1.608480	1.222503
16748	28109	-0.496303	0.948806	1.625647	0.382160	0.143376	-0.099795	0.539903	0.161991

5.2.5 Stratified Train Test Spilt

The data is divided into x and y value for training and testing purpose to be used in this Linear regression.

Split the data into training data and test data

```
[ ]  X_train, X_test, Y_train, Y_test = train_test_split(X,Y,test_size=0.2,stratify=Y,random_state=2 )
```

```
[ ]  print(X.shape, X_train.shape, X_test.shape)
```

```
(577, 30) (461, 30) (116, 30)
```

```
[ ]
```

5.2.6 Model Evaluation

```
#accuracy on training data
X_train_prediction = model.predict(X_train)
training_data_accuracy = accuracy_score (X_train_prediction, Y_train)
```

```
[ ]  print ('Accuracy on training data:',training_data_accuracy)
```

```
Accuracy on training data: 0.9891540130151844
```

```
[ ]  from google.colab import drive
     drive.mount('/content/drive')
```

```
[ ]
```

```
[ ]  #accuracy on test data
     X_test_prediction= model.predict(X_test)
     test_data_accuracy= accuracy_score(X_test_prediction, Y_test)
```

```
[ ]  print ('Accuracy on test data:',test_data_accuracy)
```

```
Accuracy on test data: 0.9827586206896551
```

Chapter 6

RESULTS

In the evaluation phase, we will be comparing the results of the above-mentioned algorithms.

Results and Discussion:

●Linear regression gives a test accuracy of 98%

●Random Forest gives a test accuracy of 99%

●Algorithm with the highest accuracy will be chosen for this classification problem

●We can convert this multiclass classification into binary classification and test which model works the best in that case.

●For future work, an algorithm that can automatically inform the user even before the fraud occurs can be adopted by not only credit card companies, but also small and big businesses

Chapter 7

CONCLUSION AND FUTURE ENHANCEMENT

This study deals with techniques that help to find out the credit card fraud. Various techniques like decision tree, Computational Intelligence, Cortical Learning Algorithm, Modified Fisher Discriminant approach and a fusion approach using Dumpster Shafer and Bayesian Learning also can be used in future scope. The random forest algorithm will work best with a large number of training data in comparison to Linear Regression, but the speed during the test as well the application might suffer.

REFERENCES

- Aswathy M S, Liji Sameul “Survey on Credit Card Fraud Detection”. Y. Sahin, S. Bulkan, and E. Duman, ``A cost-sensitive decision tree approach for fraud detection," Expert Syst. Appl., vol. 40, no. 15, pp. 5916_5923, 2013.
- A. Srivastava, A. Kundu, S. Sural, and A. Majumdar,``Credit card fraud detection using hidden Markov model," IEEE Trans. Depend. Sec. Comput., vol. 5, no. 1, pp. 37, Jan. 2008.
- J. T. Quah and M. Sriganesh, ``Real-time credit card fraud detection using computational intelligence," Expert Syst. Appl., vol. 35, no. 4, pp..
- S. Panigrahi, A. Kundu, S. Sural, and A. K. Majumdar,``Credit card fraud detection: A fusion approach using Dempster Shafer theory and Bayesian learning," Inf. Fusion, vol. 10, no. 4.
- N. Mahmoudi and E. Duman, ``Detecting credit card fraud by modified fisher discriminant analysis," Expert Syst. Appl., vol. 42, no. 5.

PLAGIARISM REPORT