A
Mini Project Report

On

"Bank Fraud Detection Using Machine Learning"

Submitted in partial fulfillment of the

Requirements for the award of the degree of

Bachelor of Technology

In

Computer Science & Information technology

By

20R21A3321 - Kasoji Kavyasri

Under the guidance of

Dr. P. Subhashini
Professor

MARRI LAXMAN REDDY GROUP OF INSTITUTIONS
MLR INSTITUTE OF TECHNOLOGY
(UGC AUTONOMOUS)
Affiliated to JNTUH, Approved by AICTE
Laxman Reddy Avenue, Dundigal, Hyderabad-500 043, Telangana, India
NBA NATIONAL BOARD OF ACCREDITATION
A NAAC

Department of Computer Science & Information Technology

2022

Department of Computer Science & Information Technology

CERTIFICATE

This is to certify that the project entitled "**BANK FRAUD DETECTION USING MACHINE LEARNING"** has been submitted by **Kasoji Kavyasri(20R21A3321)** in partial fulfillment of the requirements for the award of degree of Bachelor of Technology in Computer Science & Information technology from MLR Institute of Technology, Hyderabad. The results embodied in this project have not been submitted to any other University or Institution for the award of any degree or diploma.

Internal Guide **Head of the Department**

External Examiner

MARRI LAXMAN REDDY GROUP OF INSTITUTIONS
MLR INSTITUTE OF TECHNOLOGY
(UGC AUTONOMOUS)
Affiliated to JNTUH, Approved by AICTE
Laxman Reddy Avenue, Dundigal, Hyderabad-500 043, Telangana, India
NBA NATIONAL BOARD ACCREDITATION
A NAAC

Department of Computer Science & Information Technology

DECLARATION

I here by declare that the project entitled **"BANK FRAUD DETECTION USING MACHINE LEARNING"** is the work done during the period from **AUGUST 2022 to DECEMBER 2022** and is submitted in partial fulfillment of the requirements for the award of degree of Bachelor of Technology in Computer Science & Information technology from MLR Institute of Technology, Hyderabad. The results embodied in this project have not been submitted to any other university or Institution for the award of any degree or diploma.

Kasoji Kavyasri 20R21A3321

ACKNOWLEDGEMENT

The satisfaction and euphoria that accompany the successful completion of any task would be incomplete without the mention of people who made it possible, whose constant guidance and encouragement crowned our efforts with success. It is a pleasant aspect that I now have the opportunity to express the guidance for all of them.

First of all, I would like to express my deep gratitude towards the internal guide **Dr. SUBHASHINI P, Professor, Department of CSIT** for her support in the completion of my dissertation. I wish to express my sincere thanks to **Dr. SUBHASHINI P,** HOD, Dept. of CSIT and also principal **Dr. K. SRINIVAS RAO** for providing the facilities to complete the dissertation.

I would like to thank all my faculty and friends for their help and constructive criticism during the project period. Finally, I am very much indebted to my parents for their moral support and encouragement to achieve goals.

Kasoji Kavyasri 20R21A3321

ABSTRACT

In our modern era, there has been rapid growth in the banking sector in India. However, with the increase in bank functionality, there is a drastic increment in fraud. Today the banking sector offers many different financial services such as ATM cards, Internet banking, Debit card, and Credit card, which allows for attracting a large number of new customers. This project proposes an information system for detecting credit card fraud using a machine learning algorithm. Usually, credit cards are used by customers across the world, so the bank's server can track all transactions using machine learning algorithms. It must find or predict fraud detection. The data set contains some characteristics for each transaction and fraudulent transactions need to be classified and detected. For these purposes, the work proposes the use of the Machine Learning algorithm

LIST OF FIGURES

INDEX

1. INTRODUCTION

1.1 OVERVIEW

Credit card fraud is referred to as the illegal use of a credit card or its information without the owner's involvement. Nowadays there has been a massive use of credit cards for online transactions, due to which the frequency of transactions is increasing and at the same time, the number of fraudulent transactions is also increasing rapidly. In the period of digitization, there is a necessity to identify credit card fraud. Fraud detection involves monitoring and analyzing the behavior of various users to estimate detectors and avoid unwanted behavior. To identify credit card fraud detection successfully, we need to understand the various algorithms, technologies and types involved in detecting credit card fraud. The different machine learning algorithms can be used to differentiate transactions that are fraudulent or not. To find fraud, they need to pass the data set and knowledge of the fraudulent transaction.

1.2 PURPOSE OF THE PROJECT

We propose a Machine learning model to detect fraudulent credit card activities in online financial transactions. Analyzing fake transactions manually is impracticable due to vast amounts of data and its complexity. However, adequately given informative features, could make it is possible using Machine Learning. This thing will be explored in the project. To classify fraudulent and non-fraudulent credit card transaction by supervised learning Algorithm such as Random Forest. To help us to get awareness about the fraudulent and without loss of any financially

1.3 MOTIVATION

Machine Learning (ML) is a type of Artificial Intelligence (AI) that allows software applications to become more accurate at predicting outcomes without being explicitly programmed to do so. Machine Learning algorithms use historical data as input to predict new output values. This helps in avoiding the frauds or misuse of technology as well banking procedure gets secured for day-to-day transactions. People can react earlier before the frauds occurrence.No one gets into loss due to the frauds happening when there is an earlier reaction.

2. LITERATURE SURVEY

There are several algorithms for detecting frauds in banking sector. For that we analyze through different classifiers in different papers. The classifiers are Random Forest, Logistic Regression etc. The accuracy obtained by different algorithms are different. Here our data set is too large so we are using Random Forest Algorithm which works more efficient when the data set is too large.

2.1 **EXISTING SYSTEM**

Now a day's credit card has become the most popular mode of payment for online transactions, but cases of fraud associated with it are also rising tremendously. The techniques used before are

A. **DIGITAL FOOTPRINT ANALYSIS:**

A feature that leverages digital data. In a nutshell, it's about performing a search for personal information based on their Internet usage to get a sense of who they are.

- **DISADVANTAGES:**

- Personal data can be misused without our knowledge.
- It can be irritating and distracting to be constantly monitoring advertisements and information**.**

B. **KYC CHECKS**:

Customer ID verification is an important part of KYC (Know Your Customer) checks. A know-your-customer check is a process where you collect information on customers and verify their identities. This protects us from working with customers who may be involved in economic crimes in past like money laundering.

- **DISADVANTAGES:**

- Money and time wastage is more on false positives.
- Because of poor data, undetected risks may occur.

C. REAL-TIME MONITORING AND ALERTS:

It constantly checks our network for performance issues. It also gives your business updates and alerts on your network's performance as they happen. Many networks performance monitoring (NPM) solutions are built to analyse the network in real time, giving you valuable insights into your network's operation. Understanding how fraud happens at our bank is one step. Setting up real-time monitoring and managing notifications is the next step.

- **DISADVANTAGES:**

- Monitoring 24x7, increases stress levels in staff members.
- The staff feels insecure when they noticed that they are being monitored, and each of their activities is under the control of higher authorities. This is one of the biggest negative impacts of employee monitoring.

3. PROPOSED SYSTEM

3.1 PROPOSED SYSTEM

Credit card fraud detection can also be done using machine learning. Different Machine Learning approaches can be applied to this problem. This project proposes the use of the Random Forest algorithm to detect financial fraud in transactions of bank customers. The data on financial transactions of the bank's clients will be used as the initial data.

3.2 ADVANTAGES OF PROPOSED SYSTEM

- **Speed** - ML algorithms can evaluate enormous amounts of data in a very short amount of time. They have the ability to continuously collect and analyze new data in real-time. Speed is increasingly important as the velocity and volume of e commerce increases.

- **Efficiency** - ML algorithms can analyze thousands of payments per second, which is more work than several human analysts can do in the same amount of time. This reduces costs as well as time taken to analyze transactions, thus making the process more efficient.

- **Scalability** - As the number of transactions increases for banks, the pressure on a rules-based system and human analysis increases. This means a rise in costs and time, and a reduction in accuracy. With a machine learning algorithm, it’s just the opposite. The more data, the better. The program improves as more data comes in, enabling it to detect fraud faster and with more accuracy.

- **Accuracy** - ML algorithms can be trained to analyze and detect patterns across seemingly insignificant data. They can identify subtle or non-intuitive patterns which would be difficult, or maybe even impossible, for humans to catch. This increases the accuracy of fraud detection, meaning that there will be fewer false positives and frauds that go undetected.

3.3 SYSTEM REQUIREMENTS

Software Requirements:

Windows 8 & above for "JUPYTER NOTEBOOK"

python 3.7 and related libraries.

Hardware requirements:

Processor: i3

Hard disk:5gb

Memory:1GB

4. SYSTEM DESIGN

4.1 PROPOSED SYSTEM ARCHITECTURE

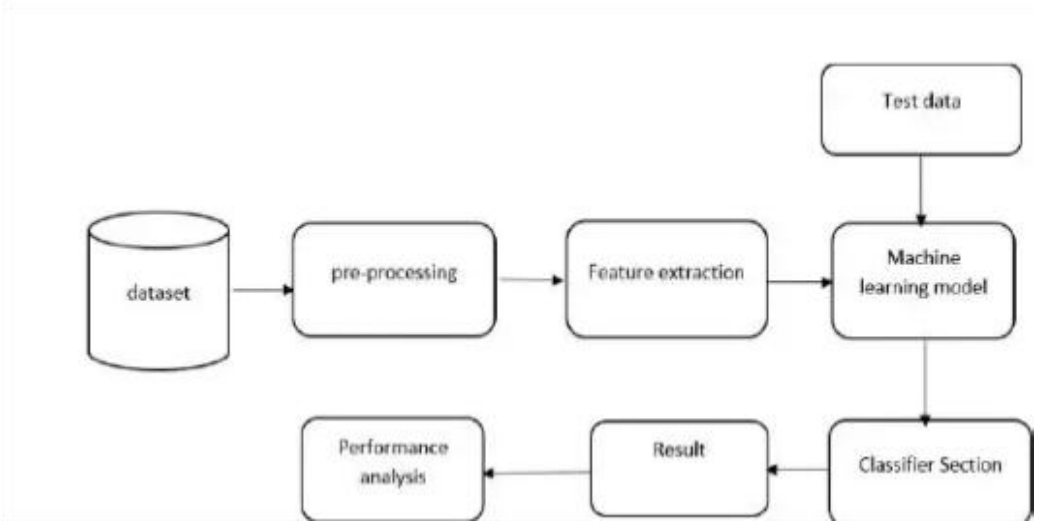


Fig 1: Proposed System Architecture

4.2 UML DIAGRAM

The UML diagrams are categorized into **structural diagrams, behavioral diagrams,** and also interaction **overview diagrams.** The diagrams are hierarchically classified in the following figure:

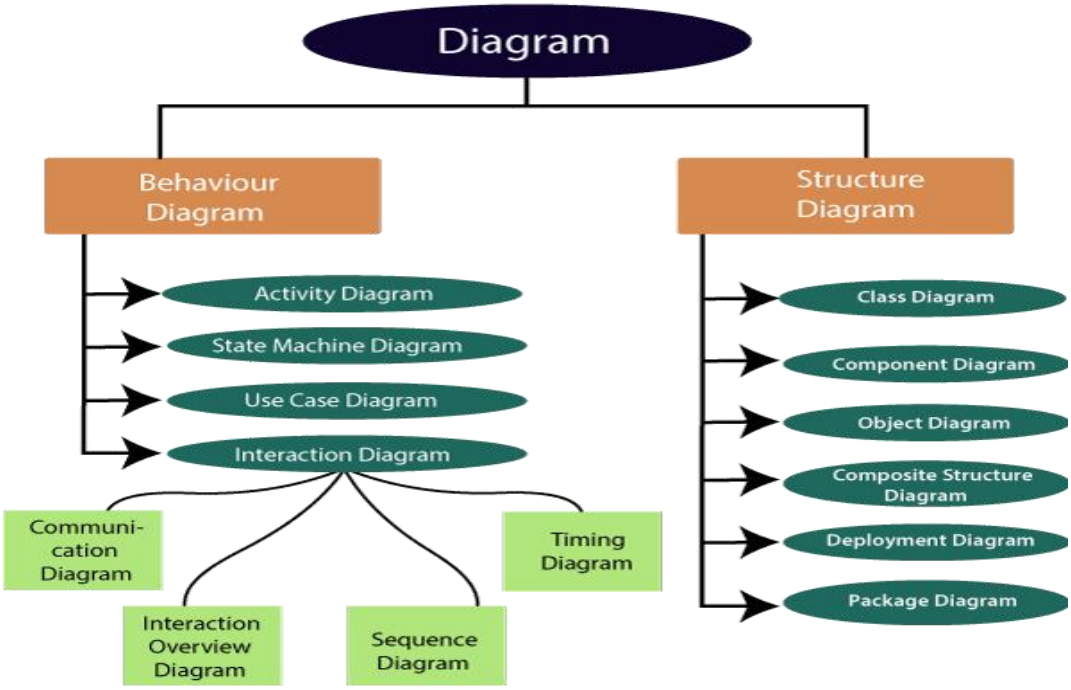


Fig 2.1: UML Diagram categories

1. Behavioral Diagrams:

Behavioral diagrams portray a dynamic view of a system or the behavior of a system, which describes the functioning of the system. It includes use case diagrams, state diagrams, and activity diagrams. It defines the interaction within the system.

- **State Machine Diagram:** It is a behavioral diagram. it portrays the system's behavior utilizing finite state transitions. It is also known as the **State-charts** diagram. It models the dynamic behavior of a class in response to external stimuli.
- **Activity Diagram:** It models the flow of control from one activity to the other. With the help of an activity diagram, we can model sequential and concurrent

activities. It visually depicts the work flow as well as what causes an event to occur.

- **Use Case Diagram:** It represents the functionality of a system by utilizing actors and use cases. It encapsulates the functional requirement of a system and its association with actors. It portrays the use case view of a system.

1.2. Interaction Diagrams

Interaction diagrams are a subclass of behavioral diagrams that give emphasis to object interactions and also depicts the flow between various use case elements of a system. In simple words, it shows how objects interact with each other and how the data flows within them. It consists of communication, interaction overview, sequence, and timing diagrams.

- **Sequence Diagram:** It shows the interactions between the objects in terms of messages exchanged over time. It delineates in what order and how the object functions are in a system.
- **Communication Diagram:** It shows the interchange of sequence messages between the objects. It focuses on objects and their relations. It describes the static and dynamic behavior of a system.
- **Timing Diagram:** It is a special kind of sequence diagram used to depict the object's behavior over a specific period of time. It governs the change in state and object behavior by showing the time and duration constraints.
- **Interaction Overview diagram:** It is a mixture of activity and sequence diagram that depicts a sequence of actions to simplify the complex interactions into simple interactions.

2. Structural Diagrams

Structural diagrams depict a static view or structure of a system. It is widely used in the documentation of software architecture. It embraces class diagrams, composite structure diagrams, component diagrams, deployment diagrams, object diagrams, and

package diagrams. It presents an outline for the system. It stresses the elements to be present that are to be modeled.

- **Class Diagram:** Class diagrams are one of the most widely used diagrams. It is the backbone of all the object-oriented software systems. It depicts the static structure of the system. It displays the system's class, attributes, and methods. It is helpful in recognizing the relation between different objects as well as classes.
- **Composite Structure Diagram:** The composite structure diagrams show parts within the class. It displays the relationship between the parts and their configuration that ascertain the behavior of the class. It makes full use of ports, parts, and connectors to portray the internal structure of a structured classifier. It is similar to class diagrams, just the fact it represents individual parts in a detailed manner when compared with class diagrams.
- **Object Diagram:** It describes the static structure of a system at a particular point in time. It can be used to test the accuracy of class diagrams. It represents distinct instances of classes and the relationship between them at a time.
- **Component Diagram:** It portrays the organization of the physical components within the system. It is used for modeling execution details. It determines whether the desired functional requirements have been considered by the planned development or not, as it depicts the structural relationships between the elements of a software system.
- **Deployment Diagram:** It presents the system's software and its hardware by telling what the existing physical components are and what software components are running on them. It produces information about system software. It is incorporated whenever software is used, distributed, or deployed across multiple machines with dissimilar configurations.
- **Package Diagram:** It is used to illustrate how the packages and their elements are organized. It shows the dependencies between distinct packages. It manages UML diagrams by making it easily understandable. It is used for organizing the class and use case diagrams.

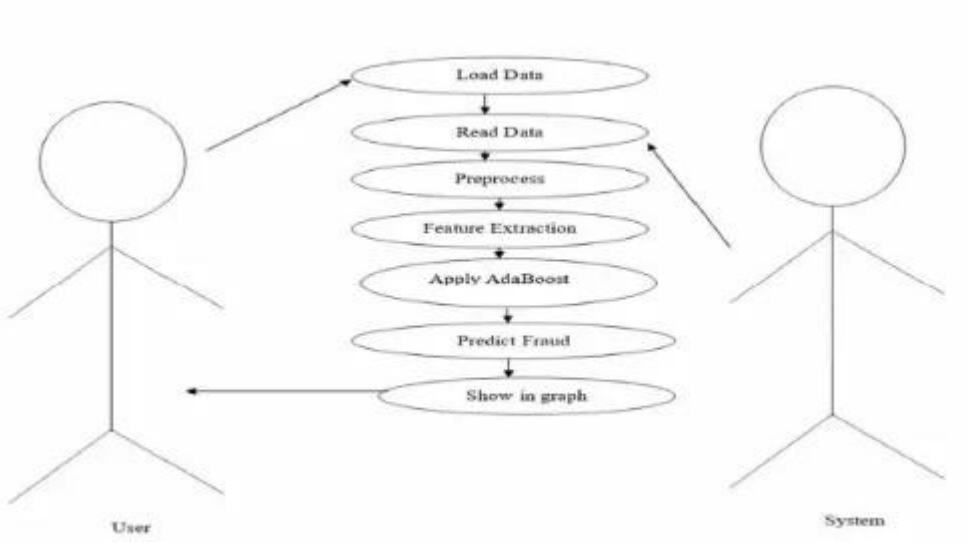


Fig 2.2: Use Case Diagram

4.3 MODULES

- Data collection
- Data exploration
- Data per-processing
- Feature extraction and Data classification
- Data visualization

Module 1: DATA COLLECTION

Machine Learning problems start with the collection of DATASETS. The data set used in this project is collected from Credit Card transaction records. This step is concerned with selecting the subset of all available data.

Our data set is a CSV (Comma Separated Values) file.

The data set contains transactions made by credit cardholders in September 2013 by European countries. This data set contains transactions that occurred in two days, which has 492 frauds out of 284,807 transactions. The data set is highly unbalanced, the fraud transactions are of 0.172%.

NON-FRAUD (0)	284315
FRAUD (1)	492
TOTAL	284,807

Fig 3.1: Data Collection

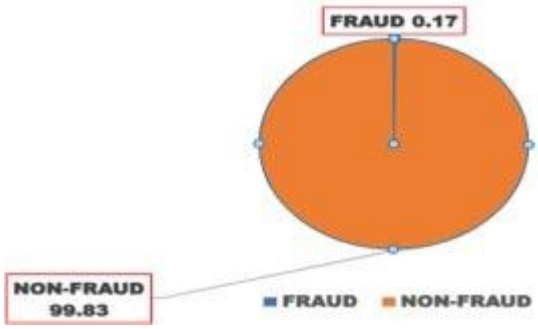


Fig 3.2: Data set percentage

- **ABOUT THE DATASET:**

- It contains only numerical input variables which are the result of a PCA transformation
- 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 data set.
- The feature 'Amount' is the transaction Amount, this feature can be used for example-dependent cost-sensitive learning.
- Feature 'Class' is the response variable and it takes value 1 in case of fraud and 0 otherwise.

- LINK: Credit Card Fraud Detection | Kaggle

Module 2: DATA EXPLORATION

- Import the "PANDAS LIBRARY"
- First, we need to load the data set. After downloading the data set, extract the data and keep the file in the data set under the project folder.
- Here, we can find all the answers to the below questions.

✧ Data set size

✧ Number of samples(rows) and features(columns)

✧ Names of the features

ABOUT THE LIBRARIES

- **PANDAS:** It has many functions for manipulating, analyzing, cleaning and exploring data. Pandas offers a time saving solution when working with data sets.
- **NUMPY:** Many of its functions are very useful for performing any mathematical or scientific calculation.
- **MATPLOTLIB: Matplotlib** is one of the most popular and oldest plotting libraries, it helps to understand the huge amount of data through different visualizations.
- **SKLEARN:** The sklearn library contains a lot of efficient tools for machine learning and statistical modeling including classification, regression, clustering and dimensionality reduction. sklearn is used to build machine learning models.

Module 3: DATA PEE-PROCESSING

- **FORMATTING:** It is the process of putting the data in a way that is most suitable to work with. Most recommended format is .csv files.
- **CLEANING:** Data cleaning is very important procedure. It includes removing missing data.

- **SAMPLING:** This is the technique of analyzing the subsets from large datasets, which could provide a better result.

Module 4: FEATURE EXTRACTION AND DATA CLASSIFICATION

Feature extraction is an attribute reduction method. Unlike feature selection, which makes the existing attributes according to their predictive significance, feature extraction transforms the attributes. The transformed features were linear combinations of the original attributes. Finally, our models are trained using Machine Learning algorithm. We use the labeled data set. The rest of our labeled data will be used to evaluate the models. Machine Learning algorithms plays vital role to classify pee-processed data. The chosen classifiers were Random Forest. This algorithm is very popular in text classification tasks.

Module 5: Data visualization

Data visualization is the method of representing the data in a graphical or pictorial way.

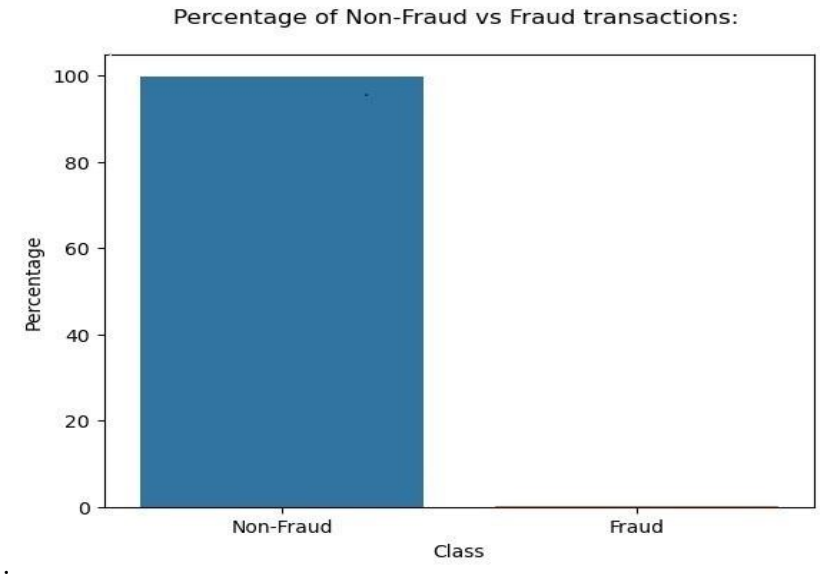


Fig -4: Percentage of transactions.

TRAIN AND TEST:

Fig -5: Train and Test data split

5.IMPLEMENTATION

5.1 ALGORITHMS

- **RANDOM FOREST**

Random forest is a supervised machine learning algorithm based on ensemble learning. Ensemble learning is an algorithm where the predictions are derived by assembling or bagging different models or similar model multiple times. The random forest algorithm works in a similar way and uses multiple algorithms i.e., multiple decision trees, resulting in a forest of trees, hence the name "Random Forest". The random forest algorithm can be used for both regression and classification tasks.

- **LOGISTIC REGRESSION**

This type of statistical model (also known as *logit model*) is often used for classification and predictive analytic. Logistic regression estimates the probability of an event occurring, such as voted or didn't vote, based on a given data set of independent variables. Since the outcome is a probability, the dependent variable is bounded between 0 and 1. In logistic regression, a logit transformation is applied on the odds—that is, the probability of success divided by the probability of failure. This is also commonly known as the log odds,

5.2 IMPLEMENTATION STEPS

- ✧ The client application sends information to the fraudulent transaction detection application.
- ✧ The fraudulent transaction detection model determines the risk score (0-100) for the input data using a machine learning model that is trained using historical data.
- ✧ A score of 0 considered to have the lowest possible risk, and a score of 1 is considered to have the highest possible risk.
- ✧ If the risk score for a particular forecast falls below a predetermined threshold, no further action is taken.
- ✧ If the risk assessment exceeds a predetermined threshold (for example, 90), the cycle starts automatically sends forecasts for review by the bank staff. Bank staff review the transaction and make a decision (approve, reject, or send for further verification).
- ✧ The result of approval or rejection is stored in the database. The data from the database can be used to retrain the fraudulent transaction detection model.

5.2 SOURCE CODE

DATA COLLECTION

In [1]:
```
import pandas as pd
import numpy as np

import matplotlib.pyplot as plt
%matplotlib inline
import seaborn as sns
from matplotlib import gridspec

import warnings
warnings.filterwarnings('ignore')

```

In [2]:
```
df = pd.read_csv("creditcard.csv")
```

DATA EXPLORATION

In [3]:
```
df.head()
```

Out[3]:

	Time	V1	V2	V3	V4	V5	V6	V7	V8	V9	...	V21	V22	V23	V24	V2
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	0.12853
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	0.16717
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	-0.32764
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	0.64737
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	-0.20601

5 rows × 31 columns

In [4]:
```
df.shape
```

Out[4]:
```
(284807, 31)
```

```
In [5]:  1 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
 13  V13     284807 non-null  float64
```

```
In [6]:  1 df.describe()
```

Out[6]:

	Time	V1	V2	V3	V4	V5	V6	V7	V8	V9	..
count	284807.000000	2.848070e+05	2.848070e+05	2.848070e+05	2.848070e+05	2.848070e+05	2.848070e+05	2.848070e+05	2.848070e+05	2.848070e+05	..
mean	94813.859575	3.918649e-15	5.682686e-16	-8.761736e-15	2.811118e-15	-1.552103e-15	2.040130e-15	-1.698953e-15	-1.893285e-16	-3.147640e-15	..
std	47488.145955	1.958696e+00	1.651309e+00	1.516255e+00	1.415869e+00	1.380247e+00	1.332271e+00	1.237094e+00	1.194353e+00	1.098632e+00	..
min	0.000000	-5.640751e+01	-7.271573e+01	-4.832559e+01	-5.683171e+00	-1.137433e+02	-2.616051e+01	-4.355724e+01	-7.321672e+01	-1.343407e+01	..
25%	54201.500000	-9.203734e-01	-5.985499e-01	-8.903648e-01	-8.486401e-01	-6.915971e-01	-7.682956e-01	-5.540759e-01	-2.086297e-01	-6.430976e-01	..
50%	84692.000000	1.810880e-02	6.548556e-02	1.798463e-01	-1.984653e-02	-5.433583e-02	-2.741871e-01	4.010308e-02	2.235804e-02	-5.142873e-02	..
75%	139320.500000	1.315642e+00	8.037239e-01	1.027196e+00	7.433413e-01	6.119264e-01	3.985649e-01	5.704361e-01	3.273459e-01	5.971390e-01	..
max	172792.000000	2.454930e+00	2.205773e+01	9.382558e+00	1.687534e+01	3.480167e+01	7.330163e+01	1.205895e+02	2.000721e+01	1.559499e+01	..

8 rows × 31 columns

```
In [7]:  1 classes = df['Class'].value_counts()
         2 print("Number of non-fraud and fraud transactions are: ")
         3 classes
```

```
Number of non-fraud and fraud transactions are:
```

```
Out[7]: 0    284315
        1       492
        Name: Class, dtype: int64
```

```
In [8]:  1 non_fraud = round((classes[0]/df['Class'].count()*100),2)
         2 print("Non_fraud : ",non_fraud)
```

```
Non_fraud :  99.83
```

```
In [9]:  1 fraud = round((classes[1]/df['Class'].count()*100),2)
         2 print("Fraud : ",fraud)
```

```
Fraud :  0.17
```

In [10]:
```
data_fraud = df[df['Class'] == 1]
data_non_fraud = df[df['Class'] == 0]
```

In [11]:
```
print("Amount details of the Fraud Transactions:")
data_fraud.Amount.describe()
```

```
Amount details of the Fraud Transactions:
```

Out[11]:
```
count     492.000000
mean      122.211321
std       256.683288
min         0.000000
25%         1.000000
50%         9.250000
75%       105.890000
max      2125.870000
Name: Amount, dtype: float64
```

In [12]:
```
print("Amount details of the Non-Fraud Transactions:")
data_non_fraud.Amount.describe()
```

```
Amount details of the Non-Fraud Transactions:
```

Out[12]:
```
count    284315.000000
mean         88.291022
std         250.105092
min           0.000000
25%           5.650000
50%          22.000000
75%          77.050000
max       25691.160000
Name: Amount, dtype: float64
```

In [13]:
```
df.groupby('Class').count()
```

Out[13]:

Class	Time	V1	V2	V3	V4	V5	V6	V7	V8	V9	...	V20	V21	V22	V23	V24	V25	V26	V2
0	284315	284315	284315	284315	284315	284315	284315	284315	284315	284315	...	284315	284315	284315	284315	284315	284315	284315	28431
1	492	492	492	492	492	492	492	492	492	492	...	492	492	492	492	492	492	492	49

2 rows × 30 columns

DATA CLEANING

In [14]:
```
df.isnull()
```

Out[14]:

	Time	V1	V2	V3	V4	V5	V6	V7	V8	V9	...	V21	V22	V23	V24	V25	V26	V27	V28	Amount	Class
0	False	False	False	False	False	False	False	False	False	False	...	False	False	False	False	False	False	False	False	False	False
1	False	False	False	False	False	False	False	False	False	False	...	False	False	False	False	False	False	False	False	False	False
2	False	False	False	False	False	False	False	False	False	False	...	False	False	False	False	False	False	False	False	False	False
3	False	False	False	False	False	False	False	False	False	False	...	False	False	False	False	False	False	False	False	False	False
4	False	False	False	False	False	False	False	False	False	False	...	False	False	False	False	False	False	False	False	False	False
...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...
284802	False	False	False	False	False	False	False	False	False	False	...	False	False	False	False	False	False	False	False	False	False
284803	False	False	False	False	False	False	False	False	False	False	...	False	False	False	False	False	False	False	False	False	False
284804	False	False	False	False	False	False	False	False	False	False	...	False	False	False	False	False	False	False	False	False	False
284805	False	False	False	False	False	False	False	False	False	False	...	False	False	False	False	False	False	False	False	False	False
284806	False	False	False	False	False	False	False	False	False	False	...	False	False	False	False	False	False	False	False	False	False

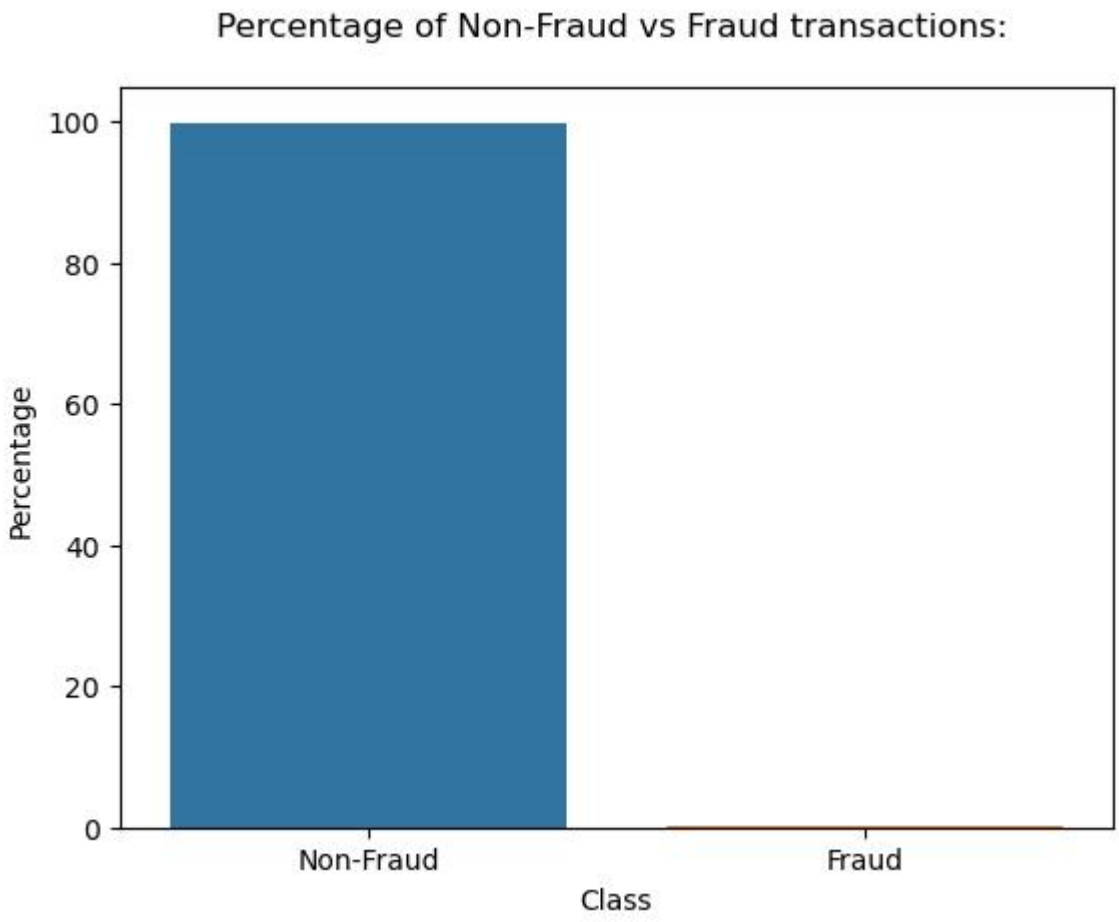
284807 rows × 31 columns

In [15]:
```
df_missing = (round(((df.isnull().sum()/len(df.index))*100),2).to_frame('null')).sort_values('null', ascending = False)
df_missing
```

Out[15]:

	null
Time	0.0
V16	0.0
Amount	0.0
V28	0.0
V27	0.0
V26	0.0
V25	0.0
V24	0.0
V23	0.0
V22	0.0
V21	0.0
V20	0.0
V19	0.0
V18	0.0
V17	0.0
V15	0.0
V1	0.0
V14	0.0
V13	0.0
V12	0.0
V11	0.0
V10	0.0
V9	0.0

In [16]:
```
fraud_percentage = {'Class':['Non-Fraud', 'Fraud'],'Percentage':[non_fraud, fraud]}
df_percentage = pd.DataFrame(fraud_percentage)
sns.barplot(x='Class', y='Percentage', data = df_percentage)
plt.title("Percentage of Non-Fraud vs Fraud transactions: \n")
plt.show()
```



TRAIN-TEST SPLIT

In [17]:
```
x = df.drop(['Class'],axis = 1)
y = df["Class"]
print(x.shape)
print(y.shape)
```

```
(284807, 30)
(284807,)
```

In [18]:
```
from sklearn.model_selection import train_test_split
```

In [19]:
```
x_train, x_test, y_train, y_test = train_test_split(x, y, train_size=0.8, test_size=0.2, random_state=100)
```

FEATURE SELECTION

We need to scale only the Amount column as all other columns are already scaled by the PCA transformation

In [20]:
```
from sklearn.preprocessing import StandardScaler
```

In [21]:
```
scaler = StandardScaler()
```

In [22]:
```
x_train['Amount'] = scaler.fit_transform(x_train[['Amount']])
```

In [23]:
```
x_train.head()
```

Out[23]:

	Time	V1	V2	V3	V4	V5	V6	V7	V8	V9	...	V20	V21	V22	V23
201788	134039.0	2.023734	-0.429219	-0.691061	-0.201461	-0.162486	0.283718	-0.674694	0.192230	1.124319	...	-0.171390	-0.195207	-0.477813	0.340513
179369	124044.0	-0.145286	0.736735	0.543226	0.892662	0.350846	0.089253	0.626708	-0.049137	-0.732566	...	0.206709	-0.124288	-0.263560	-0.110568
73138	54997.0	-3.015846	-1.920606	1.229574	0.721577	1.089918	-0.195727	-0.462586	0.919341	-0.612193	...	0.842838	0.274911	-0.319550	0.212891
208679	137226.0	1.851980	-1.007445	-1.499762	-0.220770	-0.568376	-1.232633	0.248573	-0.539483	-0.813368	...	-0.196551	-0.406722	-0.899081	0.137370
206534	136246.0	2.237844	-0.551513	-1.426515	-0.924369	-0.401734	-1.438232	-0.119942	-0.449263	-0.717258	...	-0.045417	0.050447	0.125601	0.215531

5 rows × 30 columns

In [24]:
```
x_test['Amount'] = scaler.transform(x_test[['Amount']])
x_test.head()
```

Out[24]:

	Time	V1	V2	V3	V4	V5	V6	V7	V8	V9	...	V20	V21	V22	V23
49089	43906.0	1.229452	-0.235478	-0.627166	0.419877	1.797014	4.069574	-0.896223	1.036103	0.745991	...	-0.057922	-0.170060	-0.288750	-0.130270
154704	102638.0	2.016893	-0.088751	-2.989257	-0.142575	2.675427	3.332289	-0.652336	0.752811	1.962566	...	-0.147619	-0.184153	-0.089661	0.087188
67247	52429.0	0.535093	-1.469185	0.868279	0.385462	-1.439135	0.368118	-0.499370	0.303698	1.042073	...	0.437685	0.028010	-0.384708	-0.128376
251657	155444.0	2.128486	-0.117215	-1.513910	0.166456	0.359070	-0.540072	0.116023	-0.216140	0.680314	...	-0.227278	-0.357993	-0.905085	0.223474
201903	134084.0	0.558593	1.587908	-2.368767	5.124413	2.171788	-0.500419	1.059829	-0.254233	-1.959060	...	0.249457	-0.035049	0.271455	0.381606

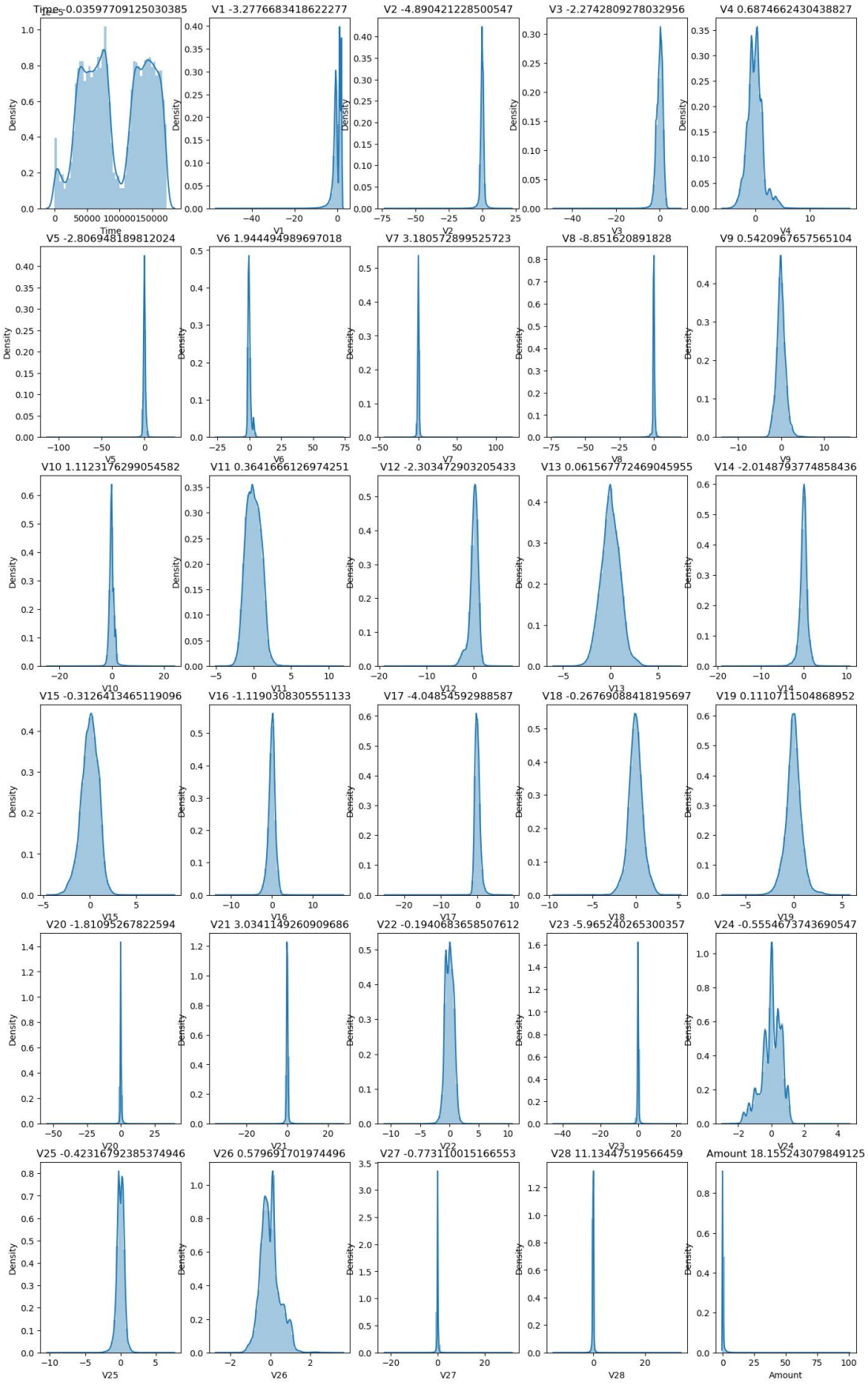
5 rows × 30 columns

CHECKING THE SKEWNESS

In [25]:
```
cols = x_train.columns
cols
```

Out[25]:
```
Index(['Time', 'V1', 'V2', 'V3', 'V4', 'V5', 'V6', 'V7', 'V8', 'V9', 'V10',
       'V11', 'V12', 'V13', 'V14', 'V15', 'V16', 'V17', 'V18', 'V19', 'V20',
       'V21', 'V22', 'V23', 'V24', 'V25', 'V26', 'V27', 'V28', 'Amount'],
      dtype='object')
```

In [26]:
```
k=0
plt.figure(figsize=(17,28))
for col in cols :
    k=k+1
    plt.subplot(6, 5,k)
    sns.distplot(x_train[col])
    plt.title(col+' '+str(x_train[col].skew()))
```


Time -0.03597709125030385
V1 -3.2776683418622277
V2 -4.890421228500547
V3 -2.2742809278032956
V4 0.6874662430438827
V5 -2.806948189812024
V6 1.944494989697018
V7 3.180572899525723
V8 -8.851620891828
V9 0.5420967657565104
V10 1.1123176299054582
V11 0.3641666126974251
V12 -2.303472903205433
V13 0.061567772469045955
V14 -2.0148793774858436
V15 -0.3126413465119096
V16 -1.1190308305551133
V17 -4.04854592988587
V18 -0.26769088418195697
V19 0.1110711504868952
V20 -1.81095267822594
V21 3.0341149260909686
V22 -0.1940683658507612
V23 -5.965240265300357
V24 -0.5554673743690547
V25 -0.42316792385374946
V26 0.579691701974496
V27 -0.773110015166553
V28 11.13447519566459
Amount 18.155243079849125
Density
Time
Amount

In [27]:
```
def draw_roc( actual, probs ):
    fpr, tpr, thresholds = metrics.roc_curve( actual, probs,drop_intermediate = False )
    auc_score = metrics.roc_auc_score( actual, probs )
    plt.figure(figsize=(5, 5))
    plt.plot( fpr, tpr, label='ROC curve (area = %0.2f)' % auc_score )
    plt.plot([0, 1], [0, 1], 'k--')
    plt.xlim([0.0, 1.0])
    plt.ylim([0.0, 1.05])
    plt.xlabel('False Positive Rate or [1 - True Negative Rate]')
    plt.ylabel('True Positive Rate')
    plt.title('Receiver operating characteristic example')
    plt.legend(loc="lower right")
    plt.show()

    return None
```

RANDOM FOREST

In [28]:
```
from sklearn.ensemble import RandomForestClassifier
```

In [29]:
```
from sklearn import metrics
from sklearn.metrics import precision_score, recall_score
from sklearn.metrics import confusion_matrix
from sklearn.metrics import f1_score, matthews_corrcoef
from sklearn.metrics import classification_report
```

In [30]:
```
rfc = RandomForestClassifier(n_estimators=50)
```

In [31]:
```
rfc.fit(x_train, y_train)
```

Out[31]:
```
RandomForestClassifier(n_estimators=50)
```

In [32]:
```
y_pred = rfc.predict(x_test)
```

In [33]:
```
labels = ['Non-Fraud', 'Fraud']
conf_mat = metrics.confusion_matrix(y_test, y_pred)
plt.figure(figsize = (7,5))
sns.heatmap(conf_mat, xticklabels = labels, yticklabels = labels, annot = True, fmt ="d")
plt.title("Confusion matrix")
plt.ylabel('True class')
plt.xlabel('Predicted class')
plt.show()
```


```
In [34]:  1  rfc.score(x_test,y_test)
```

```
Out[34]: 0.999403110845827
```

```
In [35]:  1  TP = conf_mat[1,1] # true positive
          2  TN = conf_mat[0,0] # true negatives
          3  FP = conf_mat[0,1] # false positives
          4  FN = conf_mat[1,0] # false negatives
```

In [36]:
```
print("Accuracy:-",metrics.accuracy_score(y_test, y_pred))
print("Precision: ",precision_score(y_test,y_pred))
print("Recall: ",recall_score(y_test,y_pred))
print("Sensitivity:-",TP / float(TP+FN))
print("Specificity:-", TN / float(TN+FP))
print("F1-Score:-", f1_score(y_test, y_pred))
```

```
Accuracy:- 0.999403110845827
Precision:  0.9078947368421053
Recall:  0.71875
Sensitivity:- 0.71875
Specificity:- 0.999876903597932
F1-Score:- 0.8023255813953488
```

In [37]:
```
print("CLASSIFICATION REPORT ON RANDOM FOREST : ")
print(classification_report(y_test, y_pred))
```

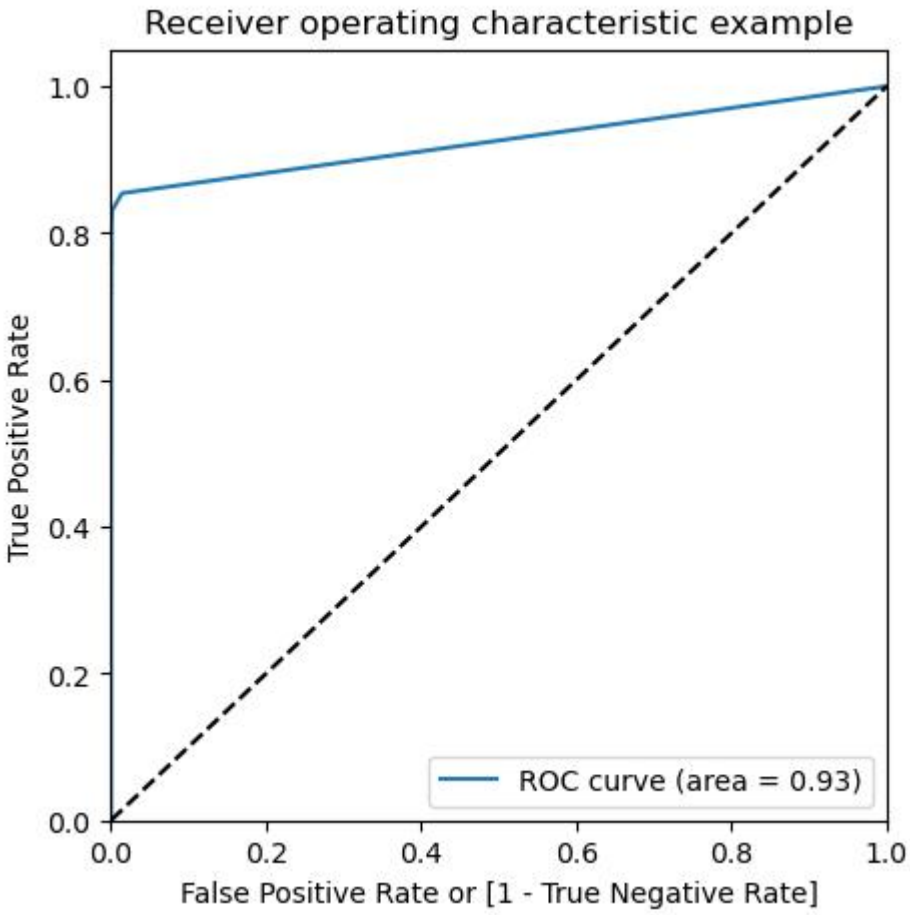
```
CLASSIFICATION REPORT ON RANDOM FOREST :
              precision    recall  f1-score   support

           0       1.00      1.00      1.00     56866
           1       0.91      0.72      0.80        96

    accuracy                           1.00     56962
   macro avg       0.95      0.86      0.90     56962
weighted avg       1.00      1.00      1.00     56962
```

In [38]:
```
y_test_pred_proba = rfc.predict_proba(x_test)[:,1]
```

In [39]:
```
auc = metrics.roc_auc_score(y_test, y_test_pred_proba)
auc
```

Out[39]:
```
0.9257899052157703
```

```
In [40]: 1 draw_roc(y_test, y_test_pred_proba)
```



LOGISTIC REGRESSION

```
In [41]: 1 from sklearn.linear_model import LogisticRegression
```

```
In [42]: 1 lr = LogisticRegression(C=0.01)
```

```
In [43]: 1 lr_model = lr.fit(x_train, y_train)
```

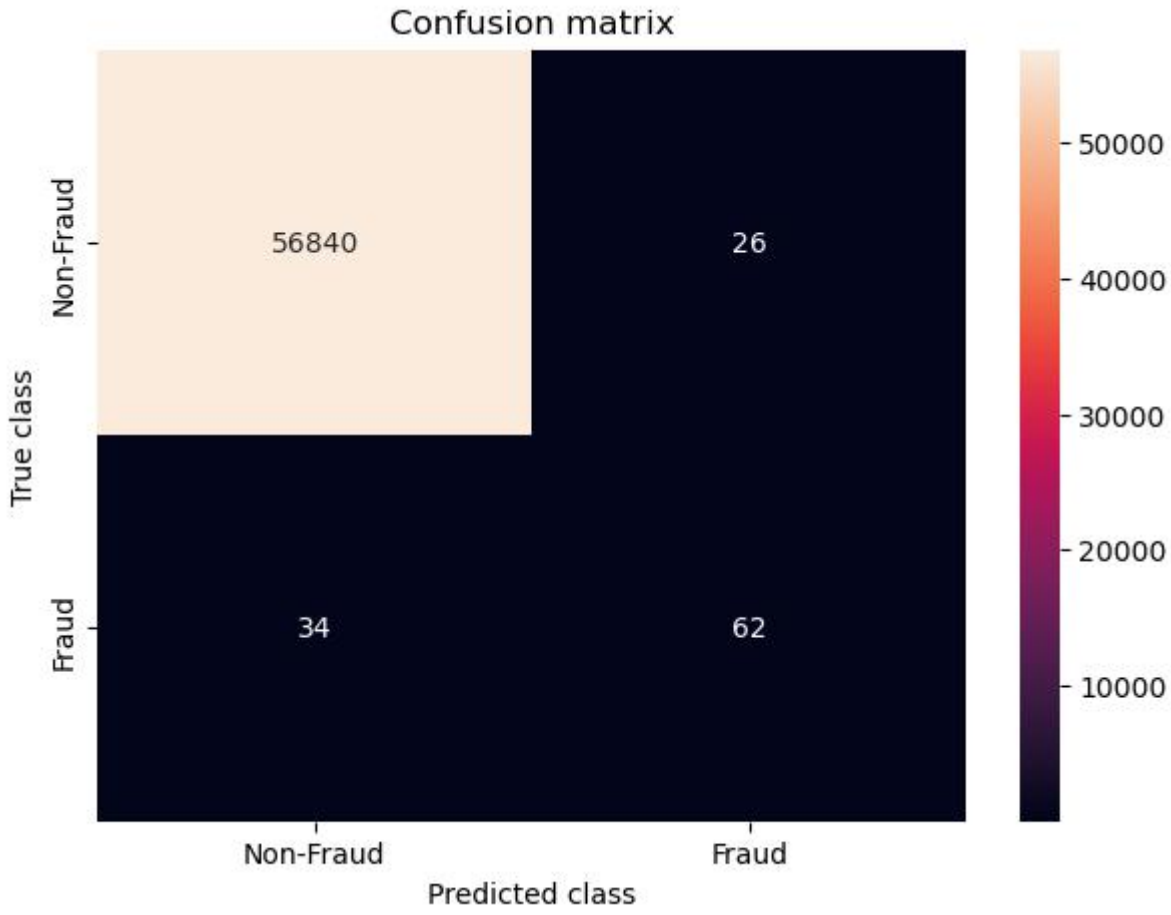
```
In [44]: 1 y_pred = lr_model.predict(x_test)
```

In [45]:
```
confusion = metrics.confusion_matrix(y_test, y_pred)
print(confusion)
```

```
[[56840    26]
 [   34    62]]
```

In [46]:
```
TP = confusion[1,1] # true positive
TN = confusion[0,0] # true negatives
FP = confusion[0,1] # false positives
FN = confusion[1,0] # false negatives
```

In [47]:
```
labels = ['Non-Fraud', 'Fraud']
plt.figure(figsize = (7,5))
sns.heatmap(confusion, xticklabels = labels, yticklabels = labels, annot = True, fmt ="d")
plt.title("Confusion matrix")
plt.ylabel('True class')
plt.xlabel('Predicted class')
plt.show()
```



In [48]:
```
print("Accuracy:-",metrics.accuracy_score(y_test, y_pred))
print("Precision: ",precision_score(y_test,y_pred))
print("Recall: ",recall_score(y_test,y_pred))
print("Sensitivity:-",TP / float(TP+FN))
print("Specificity:-", TN / float(TN+FP))
print("F1-Score:-", f1_score(y_test, y_pred))
```

```
Accuracy:- 0.9989466661985184
Precision:  0.7045454545454546
Recall:  0.6458333333333334
Sensitivity:- 0.6458333333333334
Specificity:- 0.9995427847923188
F1-Score:- 0.6739130434782609
```

In [49]:
```
print("CLASSIFICATION REPORT ON LOGISTIC REGRESSION : ")
print(classification_report(y_test, y_pred))
```

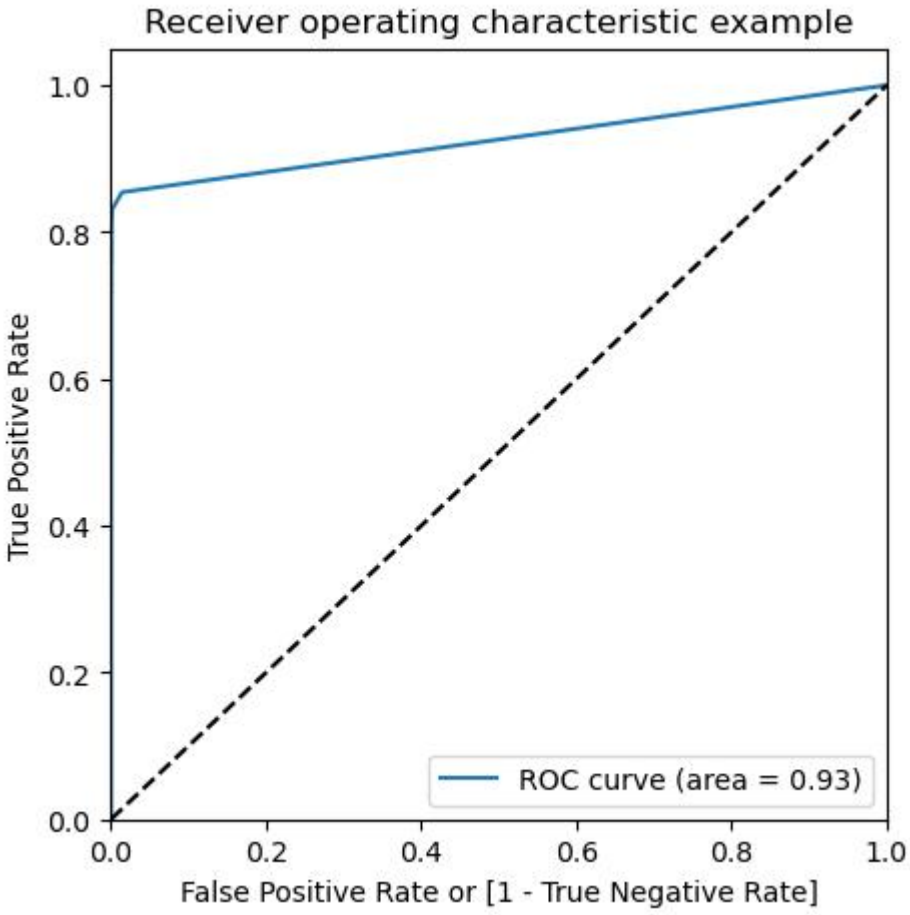
```
CLASSIFICATION REPORT ON LOGISTIC REGRESSION :
              precision    recall  f1-score   support

           0       1.00      1.00      1.00     56866
           1       0.70      0.65      0.67        96

    accuracy                           1.00     56962
   macro avg       0.85      0.82      0.84     56962
weighted avg       1.00      1.00      1.00     56962
```

In [50]:
```
y_test_pred_prob = lr.predict_proba(x_test)[:,1]
```

In [51]:
```
auc = metrics.roc_auc_score(y_test, y_test_pred_prob)
auc
```

Out[51]: 0.928778473370145

```
In [52]:  1  draw_roc(y_test, y_test_pred_proba)
```



6.TESTING

ML testing is necessary to develop a model that performs how it's expected. Just like traditional software testing, vulnerabilities can cause chaos and damage reputations. For example, a vulnerability in cybersecurity automation could spell disaster for the model.

Thorough testing eliminates bugs and errors that would otherwise frustrate end users and ensures the final product is to the user's requirements.

But, ML doesn't just care about models. ML models produce predictions based on input data. So, the quality of the input data will directly impact the quality of model predictions. Therefore, the data used to train the model must be of high quality, to ensure the best possible predictions.

Test the quality of your data. Consider the various tests available and go through several iterations of testing to ensure seamless and high performance from your ML model

Different Testing Techniques in Machine Learning:

✧ Regression Testing

Regression testing covers already tested software to ensure it doesn't suddenly break, even after a change of component or module dialer after making a feature upgrade. If the software didn't work after several modifications, then it would be called a regression.

✧ Unit Testing

Here, the program is split into blocks, or units, and each section is tested separately. Each unit is called and validated to ensure the individual components of the model meet the user requirements.

✧ Beta Testing

Beta testing, or usability testing, gives selected target users an almost finished version of the program. It helps finding bugs and is carried out to match and validate the program with the user's requirements.

Beta tests are typically deployed several times to achieve this. For example, giving select users access to new versions of video conferencing platforms, like Jitsi or an alternative to Zoom.

✧ Alpha Testing

This type of testing is done just before the product launches. It's similar to beta testing, where users test the program. But this time it's done in-house with the testing team. Alpha testing aims to find and fix bugs that weren't discovered through previous tests

✧ Integration Testing

In integration testing, the result set is taken from unit testing and groups of modules combined to see how they work together. The main purpose of integration testing is to ensure that modules interact correctly when combined and that the standards of the system and model are met.

For example, consider a UCaaS model that combines multiple communication channels in an app. An UCaaS provider will test each channel on its own and with other channels. If there were issues when combined, the system wouldn't be fit for the user's need.

7. RESULTS

The data set contains transactions made by credit cardholders in September 2013 by European countries. This data set contains transactions that occurred in two days, which has 492 frauds out of 2,84,807 transactions. The data set is highly unbalanced, the fraud transactions are of 0.172%.

NON-FRAUD (0)	284315
FRAUD (1)	492
TOTAL	284,807

Fig -5.1: Transactions in the data set

Data set is divided into training(80%) and testing(20%). 20% of data contains 56962 transactions.

RANDOM FOREST

NON-FRAUD (0)	56855
FRAUD (1)	69

LOGISTIC REGRESSION

NON-FRAUD (0)	56840
FRAUD (1)	62

Fig -5.2: Testing data set results

❖ **CONFUSION MATRIX**

✧ **RANDOM FOREST**

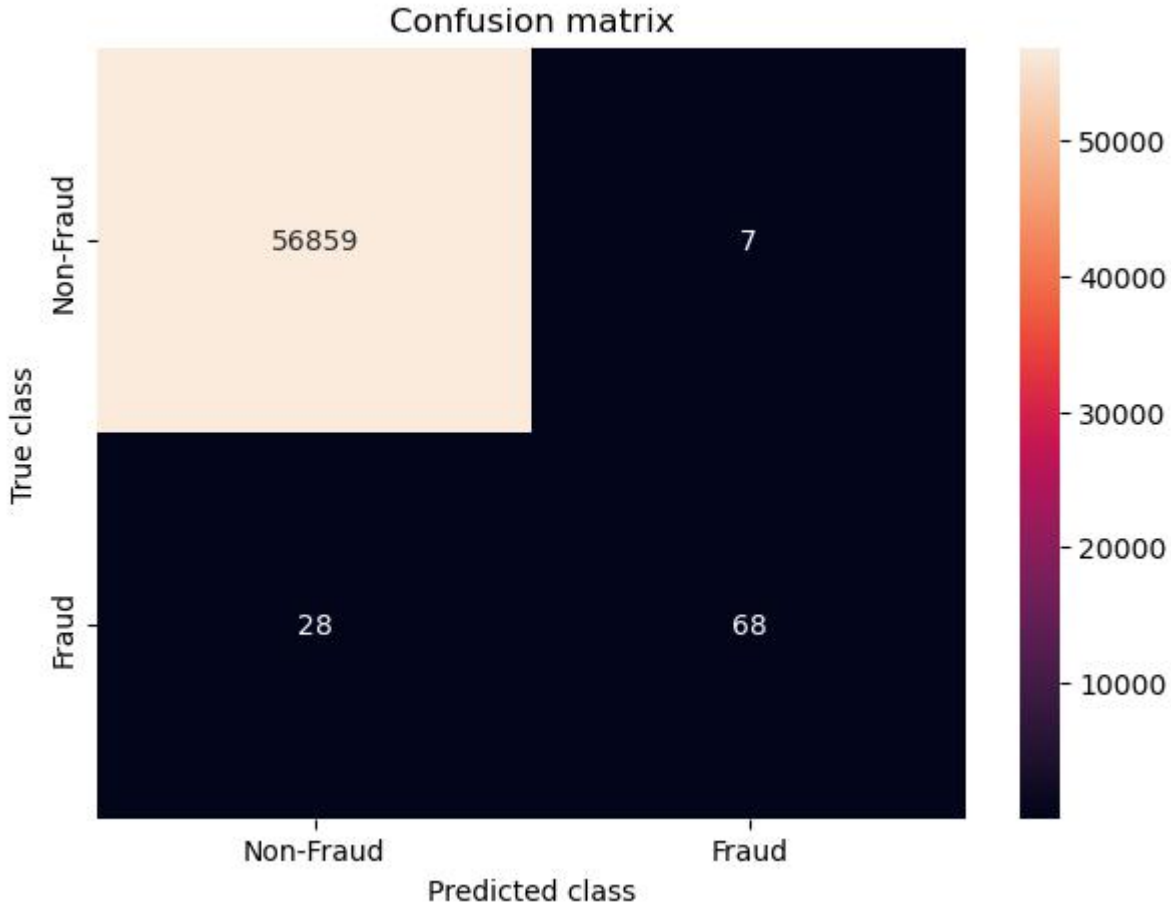


Fig –6.1: Confusion matrix of Random Forest

✧ **LOGISTIC REGRESSION**

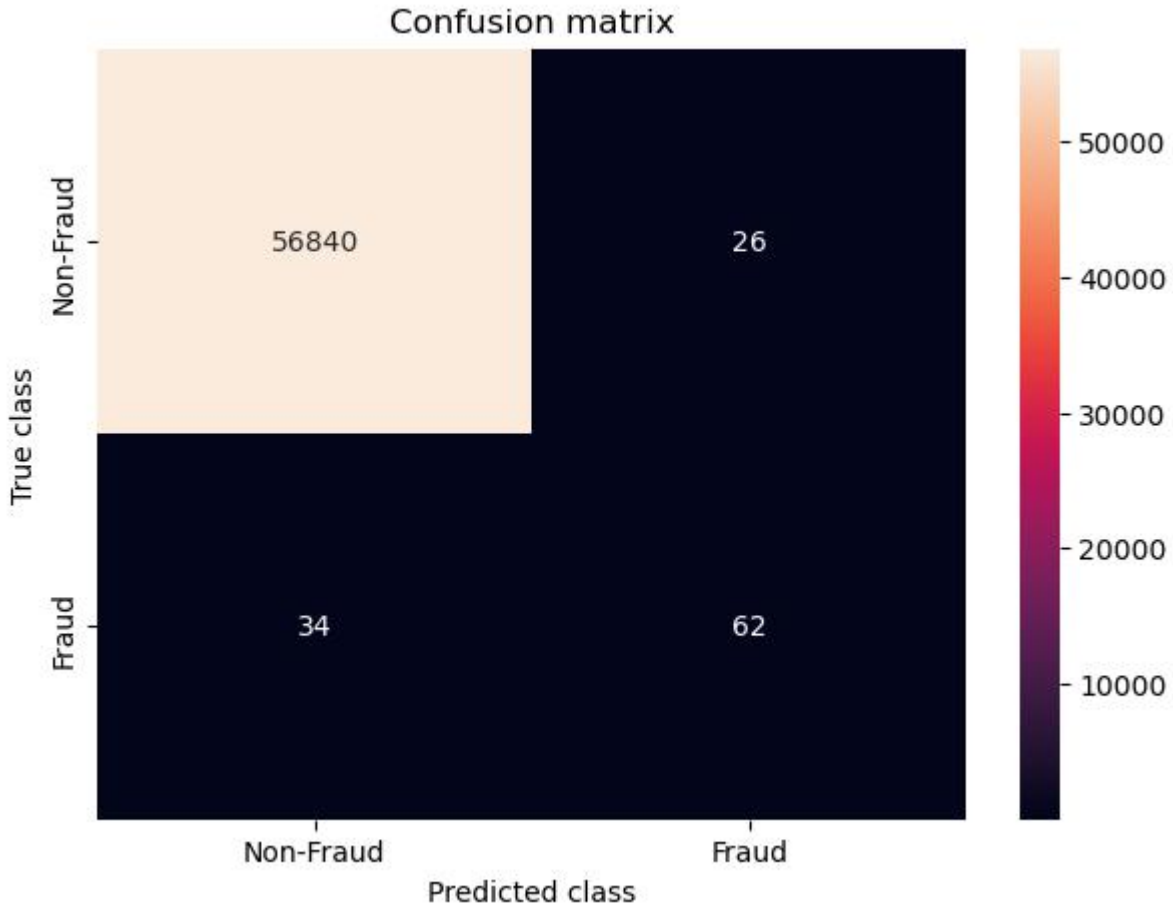


Fig –6.2: Confusion matrix of Logistic Regression

❖ **ACCURACY:**

The sum of true positives and true negatives divided by the total samples.

$$\text{Accuracy} = \frac{\text{True Positives + True Negatives}}{\text{All Samples}}$$

RANDOM FOREST	LOGISTIC REGRESSION
0.9993328	0.9989466

Fig –7.1: Accuracy in percentage

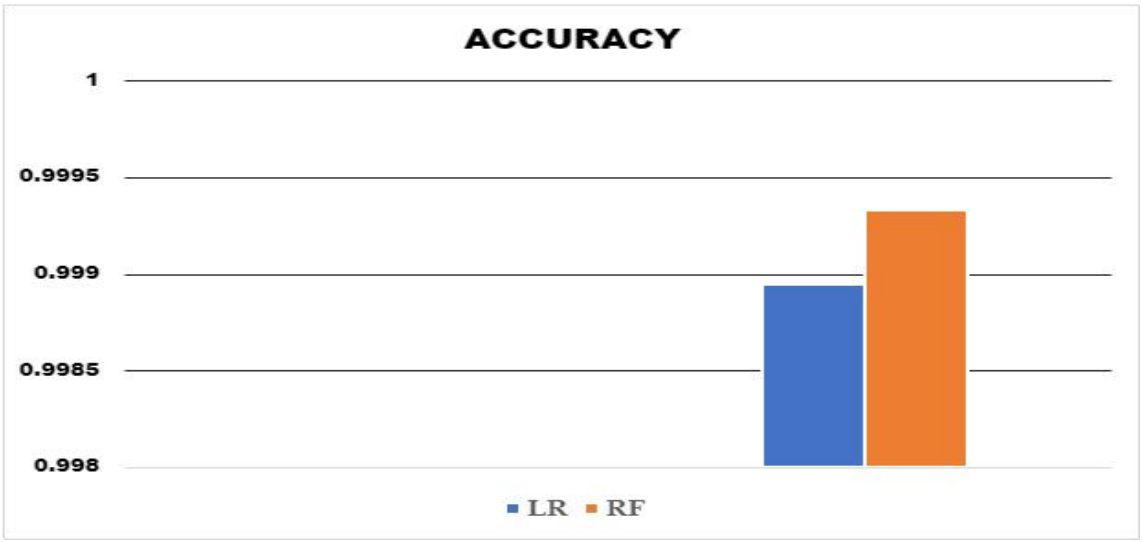


Fig –7.2: Accuracy by graph

❖ **PRECISION:**

Precision is defined as the ratio of true positives to the sum of true and false positives.

$$\text{Precision} = \frac{\text{True Positives}}{\text{True Positives + False Positives}}$$

RANDOM FOREST	LOGISTIC REGRESSION
0.8625	0.7045454

Fig –8.1: Precision in percentage

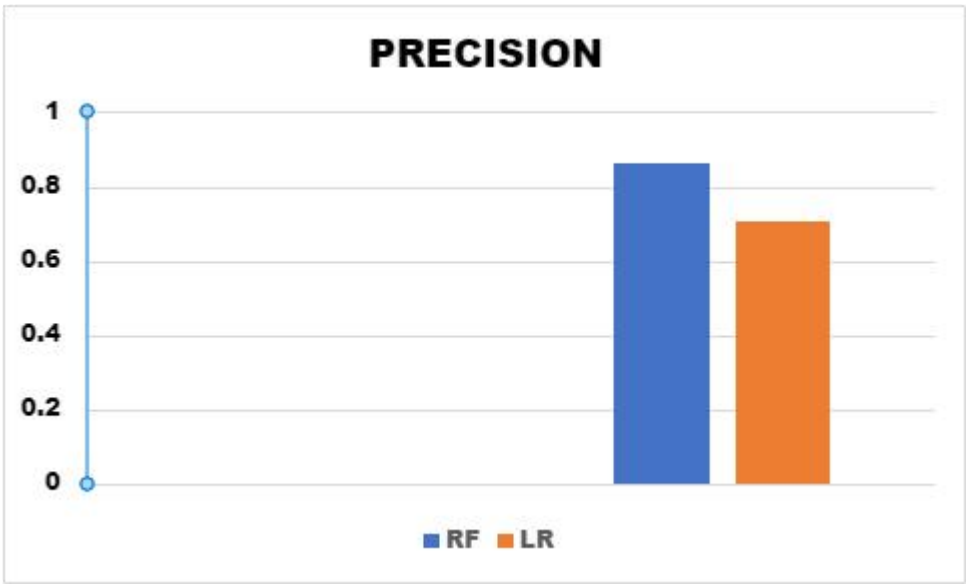


Fig –8.2: Precision by graph

❖ **RECALL:**

Recall is defined as the ratio between the true positives to the sum of true positives and false negatives

$$\text{Recall} = \frac{\text{True Positives}}{\text{Total } \textit{Actual} \text{ Positives}}$$

RANDOM FOREST	LOGISTIC REGRESSION
0.71875	0.64583

Fig -9.1: Recall in percentage

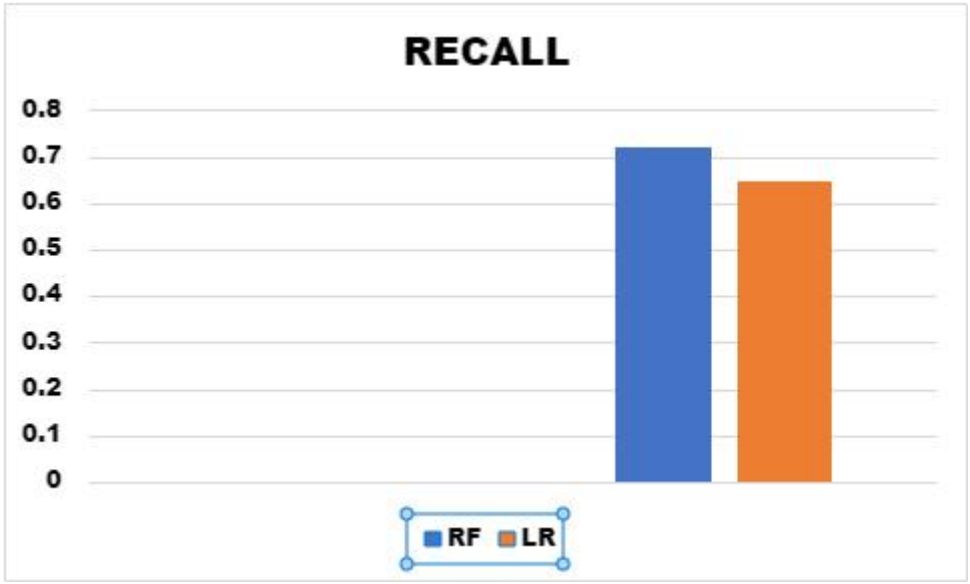


Fig –9.2: Recall by graph

❖ **F1 SCORE:**

The F1 is the mean of precision and recall. The closer the value of the F1 score to 1.0, is the better expected performance of the model.

$$F1 = 2 \times \frac{Precision * Recall}{Precision + Recall}$$

RANDOM FOREST	LOGISTIC REGRESSION
0.78409	0.6739

Fig –10.1: F1 Score in percentage

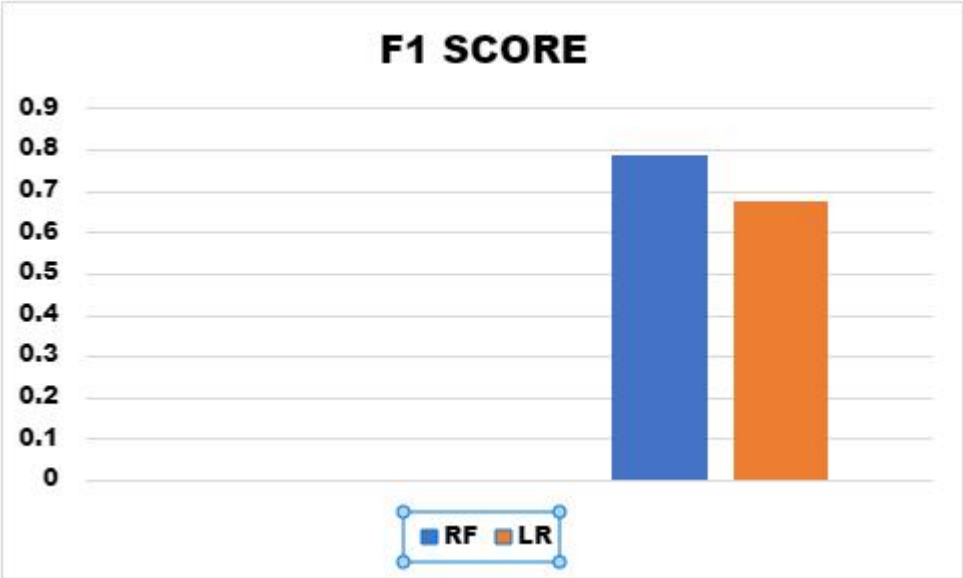


Fig –10.2: F1 Score by graph

❖ **CLASSIFICATION REPORT:**

✧ **RANDOM FOREST:**

CLASSIFICATION REPORT ON RANDOM FOREST :

	precision	recall	f1-score	support
0	1.00	1.00	1.00	56866
1	0.88	0.71	0.79	96
accuracy			1.00	56962
macro avg	0.94	0.85	0.89	56962
weighted avg	1.00	1.00	1.00	56962

Fig –11.1: Classification report on Random Forest

✧ **LOGISTIC REGRESSION:**

CLASSIFICATION REPORT ON LOGISTIC REGRESSION :

	precision	recall	f1-score	support
0	1.00	1.00	1.00	56866
1	0.70	0.65	0.67	96
accuracy			1.00	56962
macro avg	0.85	0.82	0.84	56962
weighted avg	1.00	1.00	1.00	56962

Fig –11.2: Classification report on Logistic Regression

8. CONCLUSION

Credit card fraud is the thing without a doubt an act of criminal dishonesty. Here, we listed out the methods of fraud detection methods and reviewed recent findings in this field. We explained in detail, how machine learning can be applied to get better results in fraud detection along with the algorithm and its implementation and experimentation results.

While the "Random Forest" reach over 99.9% accuracy, its precision remains at 86.2% when the 20% of the data set is taken into test consideration. However, when the "Random Forest" reach over 99.8% accuracy, its precision remains only at 70.4%

8. FUTURE ENHANCEMENT

While we couldn't reach out goal of 100% accuracy in fraud detection, we did end up creating a system that can, with enough time and data, get very close to that goal. As with any such project, there is some room for improvement here.

The very nature of this project allows for multiple algorithms to be integrated together as modules and their results can be combined to increase the accuracy of the final result.

This model can further be improved with the addition of more algorithms into it. However, the output of these algorithms needs to be in the same format as the others. Once that condition is satisfied, the modules are easy to add as done in the code. This provides a great degree of modularity and versatility to the project.

More room for improvement can be found in the dataset. As demonstrated before, the precision of the algorithms increases when the size of dataset is increased. Hence, more data will surely make the model more accurate in detecting frauds and reduce the number of false positives. However, this requires official support from the banks themselves.

10.REFERENCES

[1]. Maltseva SM et al. To the question of protection against financial cyber fraud // Azimuth of scientific research: economics and management. - 2020. - T. 9. - No. 2. - S. 332-324.

[2]. Credit Card Fraud Detection Based on Transaction Behavior-by John Richard D. Kho, Larry A. Vea" published by Proc. of the 2017 IEEE Region 10Conference (TENCON), Malaysia, November 5-8,20107.

[3]. "Research on Credit Card Fraud Detection Model Based on Distance Sum–by Wen-Fang YU and Na Wang" published by 2009 International Joint Conference on Artificial Intelligence.

[4]. "Fraud Detection in Credit Card using Machine Learning Techniques by Mr. Manohar. s, Arvind Bedi, Shashank Kumar and Shounak kr" Singh published by International Research Journal of Engineering and Technology (IRJET) Volume: 07Issue: 04, Apr 2020.

[5]. "Credit Card Fraud Detection through Par enclitic Network Analysis-By Massimiliano Zanin, Miguel Romance, Regino Criado, and Santiago Moral" published by Hindavi Complexity Volume 2018, Article ID 5764370, 9 pages.

[6]. "Credit Card Fraud Detection Using Machine Learning Models and Collating Machine Learning Models by Navanshu Khare and Saad Yunus Sait" published by International Journal of Pure and Applied Mathematics, Volume 118 No. 20, 2018

[7]. "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, March2014.

[8]. CLIFTON PHUA1, VINCENT LEE1, KATE SMITH1 & ROSS GAYLER2 " A Comprehensive Survey of Data Mining- based Fraud Detection Research" published by School of Business Systems, Facultyof Information Technology, Monash University, Wellington Road, Clayton, Victoria 3800, Australia.

[9]. "Credit Card Fraud Detection: A Realistic Modelling and a Novel Learning Strategy"
published by IEEE TRANSACTIONS ON NEURALNETWORKS AND LEARNING SYSTEMS, VOL. 29,NO. 8, AUGUST 2018.