# CREDIT CARD FRAUD DETECTION USING MACHINE LEARNING

**A PROJECT REPORT**

***Submitted by***

**IRFANA K V (720917104020)**

**NASEELA K T (720917104036)**

**AKSHAY SIVARAJ (720917104701)**

***in partial fulfillment for the award of the degree of***

**BACHELOR OF ENGINEERING**

**IN**

**COMPUTER SCIENCE AND ENGINEERING**

**JCT COLLEGE OF ENGINEERING AND TECHNOLOGY, PICHANUR, COIMBATORE**

**ANNA UNIVERSITY: CHENNAI 600 025**

APRIL 2021

# ANNA UNIVERSITY: CHENNAI 600 025

## BONAFIDE CERTIFICATE

Certified that this project report **"CREDIT CARD FRAUD DETECTION USING MACHINE LEARNING"** is the bonafide work of "**IRFANA K V, NASEELA K T and AKSHAY SIVARAJ"** who carried out the project work under my supervision.

**SIGNATURE**

**Dr.G Rajiv Suresh Kumar,**
**M.E, (Ph.D), HEAD OF THE DEPARTMENT**
Computer Science and Engineering
JCT College of Engineering and
Technology, Pichanur, Coimbatore

**SIGNATURE**

**Mr.G Deeban Chakkarawarthi,**
**M.E, MBA, ASSISTANT PROFESSOR**
Computer Science and Engineering,
JCT College of Engineering and
Technology, Pichanur, Coimbatore

# ACKNOWLEDGEMENT

We thank almighty god for the blessings that has been showered upon us to complete this project successfully.

We would like greatly indebted to our Principal Dr.G.Ramesh, M.E., Ph.D. who has been the motivating force behind all our deeds.

And our Vice Principal Dr.V J Arul Karthick, M.E, Ph.D. who was a great support for our ideas.

We earnestly express our sincere thanks to our Head of the Department Dr. G. Rajiv Suresh Kumar, M.E., (Ph.D), for his immense encouragement and support throughout the project as our project coordinator.

We are very much obliged to express our sincere thanks and gratitude to our beloved guide Mr.G Deeban Chakkarawarthi, M.E, MBA, Assistant Professor of Computer Science and Engineering who gave us valuable suggestions, constructive criticisms and encouragement that has enables us to complete our project successfully.

Above all we extend our heartfelt gratitude to our parents and friends and those who supported directly and indirectly to complete the project successfully.

## VISION

To emerge as a Premier Institute for developing industry ready engineers with competency, initiative and character to meet the challenges in global environment.

## MISSION

- To impart state-of-the-art engineering and professional education through strong theoretical basics and hands on training to students in their choice of field.
- To serve our students by teaching them leadership, entrepreneurship, teamwork, values, quality, ethics and respect for others.
- To provide opportunities for long-term interaction with academia and industry.
- To create new knowledge through innovation and research

**JCT COLLEGE OF ENGINEERING AND TECHNOLOGY**

**(Approved by AICTE, New Delhi&**

**Affiliated to Anna University, Chennai)**

**Pichanur, Coimbatore – 641 105**

# DEPARTMENT OF COMPUTER SCIENCE & ENGINEERING

## VISION

To produce the leaders in the field of Computer Science and Engineering, evolving as a Centre of Excellence for Learning and Research.

## MISSION

**Computer Science and Engineering Department is committed,**

- To develop globally competent engineers capable of providing secure and Out-of-the Box computing and cutting-edge technology solutions.
- To provide state-of-art laboratories and quality learning environment.
- To educate students with ethical values and to serve society with innovative, intelligent products and services.

# DEPARTMENT OF COMPUTER SCIENCE & ENGINEERING

## Programme Educational Objectives (PEO)

**PEO1:** Graduates shall exhibit their sound theoretical, practical skills and knowledge for successful employments or higher studies or research or entrepreneurial assignments.

**PEO2**: Graduates shall have lifelong learning skills, professional ethics and good communication capabilities along with leadership skills, so that they can succeed in their life.

**PEO3:** Graduates shall become leaders, innovators and entrepreneurs by devising software solutions for social issues and problems, thus caring for the society.

# Programme Outcomes (PO)

## Engineering Graduates will be able to:

1. **Engineering knowledge**: Apply the knowledge of mathematics, science, engineering fundamentals, and an engineering specialization to the solution of complex engineering problems.

2. **Problem analysis**: Identify, formulate, review research literature, and analyze complex engineering problems reaching substantiated conclusions using first principles of mathematics, natural sciences, and engineering sciences.

3. **Design/development of solutions**: Design solutions for complex engineering problems and design system components or processes that meet the specified needs with appropriate consideration for the public health and safety, and the cultural, societal, and environmental considerations.

4. **Conduct investigations of complex problems**: Use research-based knowledge and research methods including design of experiments, analysis and interpretation of data, and synthesis of the information to provide valid conclusions.

5. **Modern tool usage**: Create, select, and apply appropriate techniques, resources, and modern engineering and IT tools including prediction and modelling to complex engineering activities with an understanding of the limitations.

6. **The engineer and society**: Apply reasoning informed by the contextual knowledge to assess societal, health, safety, legal and cultural issues and the consequent responsibilities relevant to the professional engineering practice.

7. **Environment and sustainability**: Understand the impact of the professional engineering solutions in societal and environmental contexts, and demonstrate the knowledge of, and need for sustainable development.

8. **Ethics**: Apply ethical principles and commit to professional ethics and responsibilities and norms of the engineering practice.

9. **Individual and team work**: Function effectively as an individual, and as a member or leader in diverse teams, and in multidisciplinary settings.

10. **Communication**: Communicate effectively on complex engineering activities with the engineering community and with society at large, such as, being able to comprehend and write effective reports and design documentation, make effective presentations, and give and receive clear instructions.

11. **Project management and finance**: Demonstrate knowledge and understanding of the engineering and management principles and apply these to one's own work, as a member and leader in a team, to manage projects and in multidisciplinary environments.

12. **Life-long learning**: Recognize the need for, and have the preparation and ability to engage in independent and life-long learning in the broadest context of technological change.

# Program specific outcomes (PSO)

**PSO1:** Have capabilities to successfully qualify in national level competitive examinations for higher studies and employment.

**PSO2:** Have abilities to apply their knowledge in the domain of Design and Analysis of Algorithms, Computer Networks, Artificial Intelligence, Information Security, Data Science, Data Structure, Grid and Cloud Computing, Software Engineering, Machine Learning, Operating Systems.

# ABSTRACT

Credit card has an important role in our day- to-day lives. It is now being widely used all over the world for transactions, irrespective of the geographical boundaries People use credit card for their purchases, it allow them to pay it later. Credit card fraud happens when someone steals the information or loses their card. Criminals may be using technologies such as Trojan or phishing to get card details. Therefore, an effective fraud detection method is important since it can identify a fraud in time when a criminal uses a stolen card to consume. One method is to make full use of the historical transaction data including normal transactions and fraud ones to obtain normal/fraud behavior features based on machine learning techniques, and then utilize these features to check if a transaction is fraud or not. In this paper, Machine learning algorithm is used to train the behavior features of normal and abnormal transactions. We implement this using Random forest machine learning algorithm in openCV and analyze the performance on credit fraud detection.

# TABLE OF CONTENTS

## LIST OF FIGURES

# CHAPTER 1
# INTRODUCTION

## 1.1 Description

Credit card have made life easy, they have also managed to make life easy for the crooks. While it made fraudsters to guzzle away money that is not truly theirs. There is a growing trends of transaction fraud resulting in a great loss of money every year. It is estimated that losses are increased yearly at double digit rates by2020. since the physical card is no needed in the online transaction environment and the card's information is enough to complete a payment, it is easier to conduct a fraud than before. The key objective of credit card fraud detection system is to identify suspicious events and report them to an analyst while letting normal transactions be automatically processed.

There are two kinds of methods for fraud detection, misuse detection and anomaly detection. This paper is about misuse method. We use Random forest to train the normal and fraud behavior features. Random Forest is a classification algorithm that is comprised of many Decision Trees.

## 1.2 Objective

The objective of this project is to develop a Machine Learning model that can classify the dataset into normal and fraud transaction based on the features of dataset.

## 1.3 Scope

Credit card fraud detection is a very popular but also a very difficult problem to solve. The problem has many constraints, firstly the data set is not easily accessible for public and the result of researches are often hidden and censored, making the result inaccessible due to this it is challenging to benchmarking for the models build. Secondly, the improvement of methods is more difficult by the fact by the security

concern imposes a limitation to exchange of ideas and methods in fraud in credit card fraud detection. Lastly, the data sets are contentiously evolving and changing making the profiles of normal and fraudulent behaviors always different, that is, a legit transaction in the past may be a fraud in present or vice versa. We examine the performance of Random forest model, which identifying the fraud transactions occurring during the transactions made by the card holder.

## 1.4 Existing System

The existing systems are done with Markov model, Cost sensitive decision tree(CSDT), Support vector machine(SVM), Neural Networks, SMOTE technique and Whale optimization algorithm. Each model has their own way to identify the fraud transactions. Along with that these methods having issues for detecting the fraudulent one.

**Disadvantages:**

1. Markov models are generally inappropriate over sufficiently short time intervals.
2. CSDT are unstable, meaning that a small change in the data can lead to a large change in the structure of the optimal decision tree.
3. SVM algorithm is not suitable for large data sets. They do not perform very well when the data set has more noise i.e., target classes are overlapping.
4. Neural Networks are hardware dependent and there is no specific rule for determining the structure of artificial neural networks.
5. SMOTE is not very practical for high dimensional data.

## 1.5 Proposed system

These are the proposed techniques used in this paper for detecting the frauds in credit card system. There are two kinds of methods for fraud detection, misuse detection and anomaly detection. This paper is about misuse method.

We use Random forest to train the normal and fraud behavior features. The random forest algorithm for fraud detection and prevention has two cardinal factors that make it good at predicting things. The first one is randomness, meaning that the rows and columns of data chosen randomly from the data set and fit into different decision trees. The second factor is diversity, meaning that there is a forest of trees that contribute to the final decision instead of just one decision tree.

The main objective of Random forest algorithm to classify the collected data as normal and abnormal. Machine learning algorithms are employed to analyze all the authorized transactions and report the suspicious ones.

**Advantages:**

1. It reduces overfitting in decision trees and helps to improve the accuracy.
2. It works well with both categorical and continuous values.
3. It automates missing values present in the data.

# CHAPTER 2
# LITERATURE SURVEY

**[1] S P Maniraj, Aditya Saini and Swarna Deep Sarkar**

Proposed a method of detecting fraud along with their detection methods and reviewed recent findings in this field. This paper has also explained in detail, how machine learning can be applied to get better results in fraud detection along with the algorithm, pseudocode, explanation its implementation and experimentation results. **"Credit card fraud detection using Machine Learning and Data Science"- International Journal of Engineering Research & Technology (IJERT), ISSN: 2278-0181, Vol. 8 Issue 09, September-2019.**

**[2] Sahayasakila.V, D.Kavya, Monisha, Aishwarya, Sikhakolli and VenkatavisalakshiseshsaiYasaswi**

Proposed a system that aims in identifying the fraud transactions occurring during the transactions made by the card holder. The system also aims to improve the convergence speed and solves the data imbalance. The receiver operating characteristics (ROC) shows that the relation between the true positive rate and false positive rate. **"Credit Card Fraud Detection System using Smote Technique and Whale Optimization Algorithm"- International Journal of Engineering and Advanced Technology (IJEAT) ISSN: 2249-8958, Volume-8 Issue-5, June 2019.**

**[3] Andrea Dal Pozzolo, Giacomo Boracchi, Olivier Caelen, Cesare Alippi, Gianluca Bontempi**

a formalization of the fraud-detection problem that realistically describes the operating conditions of FDSs that everyday analyze massive streams of credit card transactions. It illustrates the most appropriate performance measures to be used for fraud-detection purposes. Design and assess a novel learning strategy that effectively addresses class imbalance, concept drift, and verification latency. **"Credit Card**

**Fraud Detection: A Realistic Modeling and a Novel Learning Strategy"- IEEE TRANSACTIONS ON NEURAL NETWORKS AND LEARNING SYSTEMS, VOL. 29, NO. 8, AUGUST 2018.**

**[4] Massimiliano Zanin, Miguel Romance, Santiago Moral, and Regino Criado**

It explains the possibility of using complex networks as a way of improving credit card fraud detection. Specifically, networks are used to synthesize complex features representing card transactions, relying on the recently proposed approach of parenclitic networks. Afterwards, their relevance is evaluated by means of a large dataset of real transactions, by comparing the yielded increase in the classification score when compared to the use of a standard ANN algorithm. Additionally, show that the combined data mining/complex networks approach is able to outperform a commercial system in some specific situations. **"Credit Card Fraud Detection through Par enclitic Network Analysis"- Hindawi Complexity Volume 2018, Article ID 5764370.**

**[5] Anushree Naik, Kalyani Phulmamdikar, Shreya Pradhan, Sayali Thorat and Prof. Sachin V. Dhande**

Proposed system in this paper is a real time system which is feasible and can be implemented. The use of such systems in the Bank Server can handle crucial frauds related to Credit Cards. Evaluation confirmed that including the real cost by creating cost sensitive system using a Bayes minimum risk classifier, gives rise to much better fraud detection results in the sense of higher savings. **"Real Time Credit Card Transaction Analysis"- International Engineering Research Journal (IERJ) Volume 1 Issue 11 Page 1663-1666, 2016, ISSN 2395-1621.**

# CHAPTER 3

# SYSTEM SPECIFICATION

## 3.1 Introduction

The requirements specification is a technical specification of requirements for the software products. It is the first step in the requirement analysis process. It lists the requirements of a particular software system including functional, non-functional and specific requirements.

## 3.2 Functional Requirements

- The model should be able to give accurate and trustworthy predictions.
- It must show the graphical visualization of the prediction.

## 3.3 Non-Functional Requirements

Non-functional requirements will describe how a system should behave and what limits are constrained on its functionality.

- Availability: The system should be available to any transaction.
- 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 maximum number of banking system needs.

## 3.4 Specific Requirements

### 3.4.1 Hardware Requirements:

1. RAM: 4-GB
2. STORAGE: 80 GB Hard Disk

### 3.4.2 Software Requirements:

3. Language: Python
4. Operating System: Windows, Linux
5. Back End Software: Anaconda, Spyder

### 3.5 Software Description

#### 3.5.1 Python

Python is an interpreted, object-oriented, high-level programming language with dynamic semantics. Its high-level built-in data structures, combined with dynamic typing and dynamic binding, make it very attractive for Rapid Application Development, as well as for use as a scripting or glue language to connect existing components together. Python's simple, easy to learn syntax emphasizes readability and therefore reduces the cost of program maintenance. Python supports modules and packages, which encourages program modularity and code reuse. The Python interpreter and the extensive standard library are available in source or binary form without charge for all major platforms, and can be freely distributed.

#### 3.5.2 Anaconda

Anaconda is a distribution of the Python and R programming languages for scientific computing (data science, machine learning applications, large-scale data processing, predictive analytics, etc.), that aims to simplify package management and deployment. The distribution includes data-science packages suitable for Windows, Linux, and macOS. It is developed and maintained by Anaconda, Inc., which was founded by Peter Wang and Travis Oliphant in 2012. As an Anaconda, Inc. product, it is also known as Anaconda Distribution or Anaconda Individual Edition, while other products from the company are Anaconda Team Edition and Anaconda Enterprise Edition, both of which are not free.

**Anaconda Navigator:**

Anaconda Navigator is a desktop graphical user interface (GUI) included in Anaconda distribution that allows users to launch applications and manage conda packages, environments and channels without using command-line commands. Navigator can search for packages on Anaconda Cloud or in a local Anaconda Repository, install them in an environment, run the packages and update them. It is available for Windows, macOS and Linux.

The following applications are available by default in Navigator:

- JupyterLab
- Jupyter Notebook
- QtConsole
- Spyder
- Glue
- Orange
- RStudio
- Visual Studio Code

### 3.5.3 Spyder

Spyder is an open-source cross-platform integrated development environment (IDE) for scientific programming in the Python language. Spyder integrates with a number of prominent packages in the scientific Python stack, including NumPy, SciPy, Matplotlib, pandas, IPython, SymPy and Cython, as well as other open-source software. It is released under the MIT license. Spyder is extensible with first-party and third-party plugins, includes support for interactive tools for data inspection and embeds Python-specific code quality assurance and introspection instruments, such as Pyflakes, Pylint and Rope. It is available cross-platform through Anaconda, on Windows, on macOS through MacPorts, and on major Linux distributions such as Arch Linux, Debian, Fedora, Gentoo Linux, openSUSE and Ubuntu.

# CHAPTER 4
# SYSTEM ANALYSIS AND DESIGN

## 4.1 System Architecture

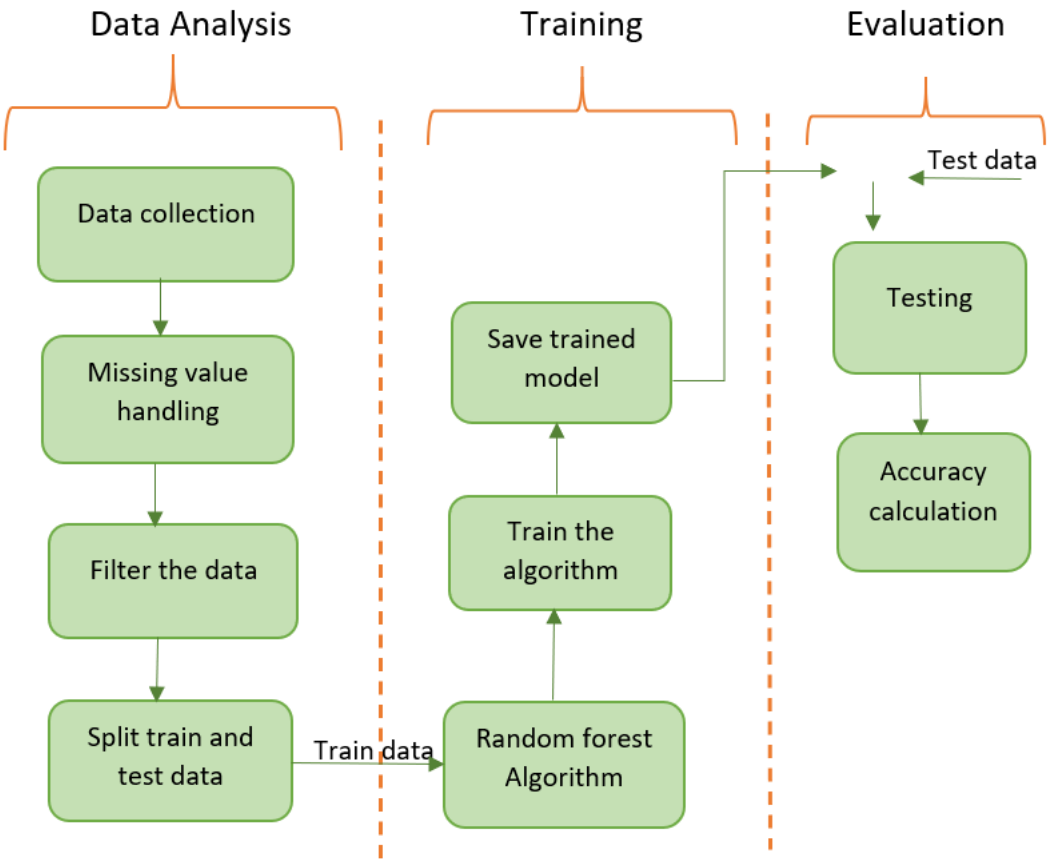



**Fig 4.1: Architectural Diagram**

This architecture explains the steps to develop the system. Mainly it is divided into three, Data analysis training and evaluation. In the Data analysis part, first we want to collect a set of data. The dataset includes the user details and their transactional details. Analyze these data and find the missing value field and make them filled. Then we have to divide the dataset into two, i.e., training and testing.

Next phase, the train data is given to the Machine Learning algorithm that we are using, i.e., Random Forest algorithm. Using the train data, the algorithm builds a model that can predict which transaction is normal and which one is abnormal.

Then, the saved model and the test data is given to testing purpose. In that phase, the created model is tested and the performance is calculated. Finding the accuracy and comparing with the existing system.

## 4.2 DFD Diagrams

A Data Flow Diagram (DFD) is a traditional visual representation of the information flows within a system. A neat and clear DFD can depict the right amount of the system requirement graphically. It can be manual, automated, or a combination of both. It shows how data enters and leaves the system, what changes the information, and where data is stored.

The objective of a DFD is to show the scope and boundaries of a system as a whole. It may be used as a communication tool between a system analyst and any person who plays a part in the order that acts as a starting point for redesigning a system. The DFD is also called as a data flow graph or bubble chart.

**Level 0**


**Fig 4.2: DFD Diagram Level 0**

Here, it explains that all the user details and their transactional details are given to the system for identifying the Fraudulent one.

**Level 1**

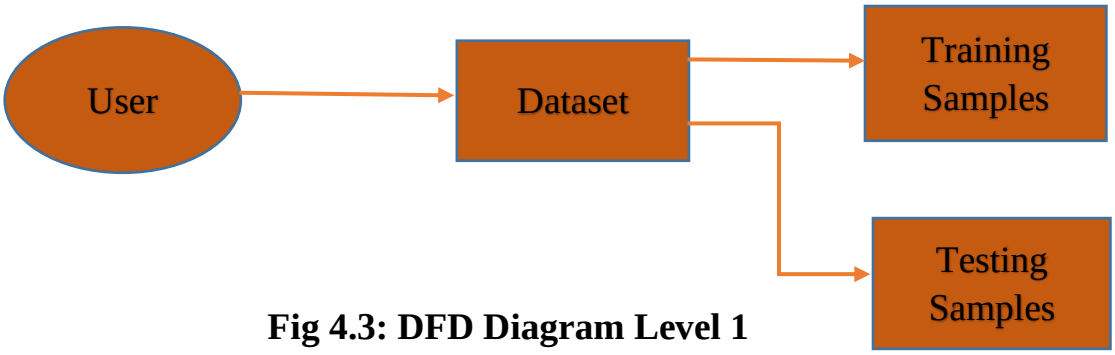



**Fig 4.3: DFD Diagram Level 1**

Collecting the dataset that containing the user data and dividing the dataset into two, i.e., Training and Testing.

**Level 2**


**Fig 4.4: DFD Diagram Level 2**

The system is processing the dataset that we are divided. Almost 80% datas are given for training and remaining for testing.

**Level 2.1**

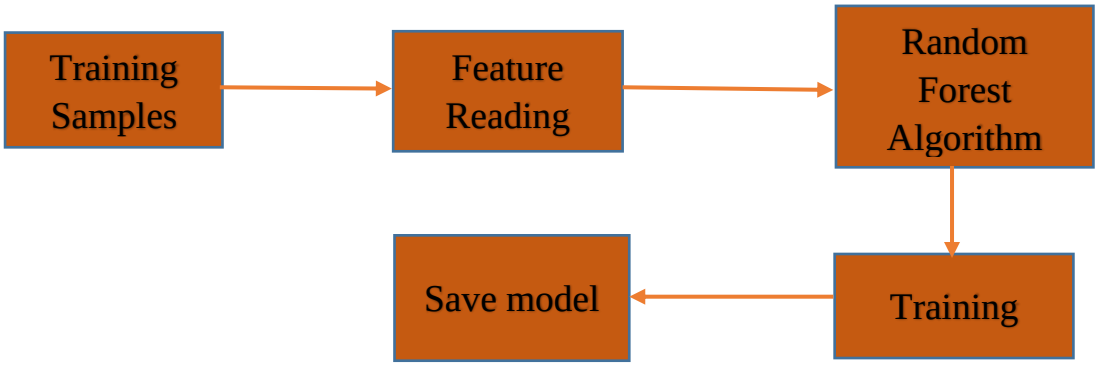



**Fig 4.5: DFD Diagram Level 2.1**

The training samples are further reading the features in the dataset and then it is given to Random Forest Algorithm. Using these train data, it makes a model and the model is saved.

**Level 2.2**

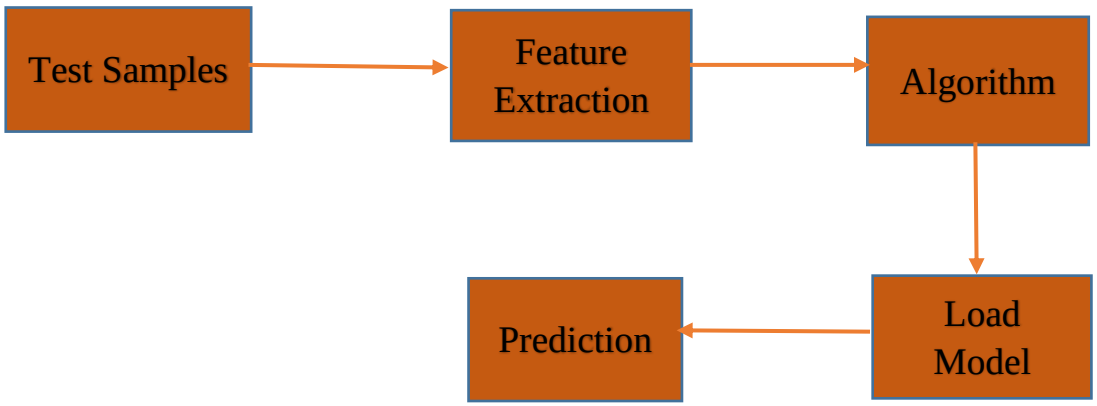



**Fig 4.6: DFD Diagram Level 2.2**

Test samples are collected and their features are extracted. Then these features are added to the algorithm and load the model which already created. Then perform prediction using the model.

# CHAPTER 5

# IMPLEMENTATION

## 5.1 Modules

The implementation of the proposed system includes five modules. They are,

1. Data collection and analysis
2. Data Splitting
3. Algorithm building
4. Training
5. Testing and Evaluation

### 5.1.1 Data Collection and Analysis

In this project, we have to collect the data from the bank. But bank does not provide transaction data. So, we collect data from internet which already uploaded by some companies.

Data analysis is a process in which data from one or more sources is cleaned, transformed and enriched to improve the quality of data prior to its issues. The collected data where then preprocessed to fill the missing data and made compatible for further processing.

### 5.1.2 Data Splitting

Data splitting is the act of partitioning available into two portions, usually for cross-validatory purposes. One portion of the data is used to develop a predictive model. And the other to evaluate the model's performance.ie training and testing. Our machine learning model will try to understand any correlation in our training set and then we will test the models on our test set to examine how accurately it will predict.

### 5.1.3 Algorithm Building

**Random Forest**

Random forests or random decision forests are an ensemble learning method for classification, regression and other tasks that operate by constructing a multitude of decision trees at training time and outputting the class that is the mode of the classes or mean/average prediction of the individual trees. The capacity of random forest not only depends on the strength of individual tree but also the correlation between different trees. The stronger the strength of single tree and the less the correlation of different tress, the better the performance of random forest. The variation of trees comes from their randomness which involves bootstrapped samples and randomly selects a subset of data attributes. Although there possibly exist some mislabeled instances in our dataset, random forest is still robust to noise and outliers.

**Random-tree-based Random Forest**

A base classifier of random forest I, which is a simple implement of decision tree, is called a random tree. The training set of each tree is a collection of bootstrapped samples selected randomly from the standard training set with replacement. At each internal node, it randomly selects a subset of attributes and computes the centers of different classes of the data in current node. The centers of class 0 and 1 are denoted as leftCenter and rightCenter, respectively. The kth element of a center is computed based on the following equations.

$$\text{lef tCenter}[k] = \frac{1}{n}\sum_{i=1}^{n} x_{ik} I(y = 0) \qquad (1)$$

$$\text{rightCenter}[k] = \frac{1}{n}\sum_{i=1}^{n} x_{ik} I(y = 1) \qquad (2)$$

where I (y = 0) and I (y = 1) are the dictator functions. At the current node, each record of the dataset is allocated to the corresponding class according to the Manhattan distance between the record and the center as shown in (3).

$$Distance(center,record)=\sum_{i\in sub} center[i]-record[i] \quad (3)$$

Note, sub is the subset of attributes randomly selected from X whose size is the square root of m = |X|. Each tree grows fully without pruning.

**Algorithm:**

Input: Dataset D and the number of trees NT

Output: A random forest

For i = 1 to NT:

1) Draw a bootstrap sample Di from the training set D whose size is n.

2) Construct a binary tree of the bootstrapped data recursively from root node. Repeatedly perform the following steps until all records of current node belong to a class.

a) Randomly select a subset of $\sqrt{m}$ attributes.

b) For j = 1 to $\sqrt{m}$:

i) Compute lef tCenter[j] and rightCenter[j].

c) For k = 1 to $|D_{ic}|$:

i) Compute the Manhattan distance $dL_k$ and $dR_k$ between the $record_k$ and each center.

ii) if $dL_k \leq dR_k$

Allocate $record_k$ to the left child of the current node.

else

Allocate $record_k$ to the right child of the current node.

d) Split the node into a left child and a right child.

where $D_{ic}$ is the subset of $D_i$ in the current node.

A simple example of random tree is shown in figure. The internal nodes are represented by circles. The variables in a circle are attributes randomly chosen from X = {x1, x2, x3, x4}. The decisions are made according to their values. Each terminal node is represented by a rectangle and corresponds to a class. The number in a terminal node represents which class the node belongs to.

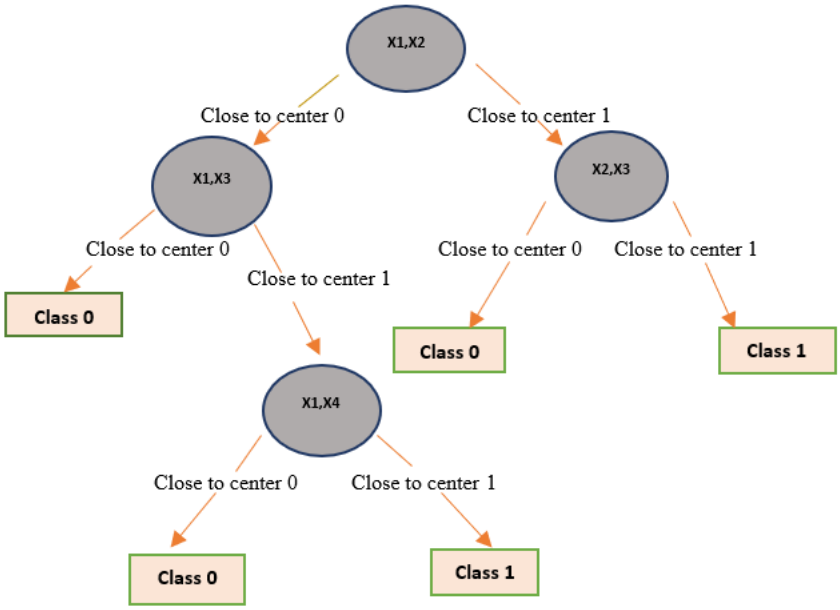



**Fig 5.1: Illustration of decision tree**

### 5.1.4 Training

Training a model simply means learning(determining) good values for all the weights and the bias from labeled examples.

### 5.1.5 Testing and Evaluation

Testing these datas to find the amount of normal and abnormal transaction. accuracy rate is not enough to measure the performance of a random forest model when the data is significantly imbalanced. For instance, a default prediction of all instances into the majority class will also have a high value of accuracy. Therefore, we need to consider other measures. Positive corresponds to fraud instances and Negative corresponds to normal instances. Evaluate these datas and check its accuracy.

# CHAPTER 6

# TESTING AND EVALUATION

## 6.1 Source code

```
# import the necessary packages
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
import seaborn as sns
from sklearn.model_selection import train_test_split
from sklearn.ensemble import RandomForestClassifier
from sklearn.metrics import accuracy_score
from sklearn.metrics import precision_score, confusion_matrix
from IPython.display import Image
from sklearn.tree import export_graphviz
import pydot

data = pd.read_excel('credit card.xlsx')
data.head()
```

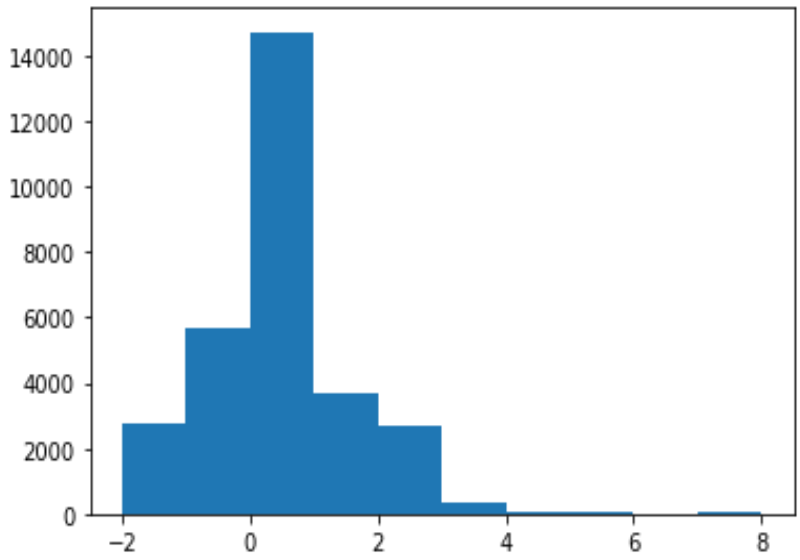
| | ID | X1 | X2 | X3 | X4 | X5 | X6 | X7 | X8 | X9 | X10 | X11 | X12 | X13 | X14 | X15 | X16 | X17 | X18 | X19 | X20 | X21 | X22 | X23 | Y |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| 0 | 1 | 20000 | 2 | 2 | 1 | 24 | 2 | 2 | -1 | -1 | -2 | -2 | 3913 | 3102 | 689 | 0 | 0 | 0 | 0 | 689 | 0 | 0 | 0 | 0 | 1 |
| 1 | 2 | 120000 | 2 | 2 | 2 | 26 | -1 | 2 | 0 | 0 | 0 | 2 | 2682 | 1725 | 2682 | 3272 | 3455 | 3261 | 0 | 1000 | 1000 | 1000 | 0 | 2000 | 1 |
| 2 | 3 | 90000 | 2 | 2 | 2 | 34 | 0 | 0 | 0 | 0 | 0 | 0 | 29239 | 14027 | 13559 | 14331 | 14948 | 15549 | 1518 | 1500 | 1000 | 1000 | 1000 | 5000 | 0 |
| 3 | 4 | 50000 | 2 | 2 | 1 | 37 | 0 | 0 | 0 | 0 | 0 | 0 | 46990 | 48233 | 49291 | 28314 | 28959 | 29547 | 2000 | 2019 | 1200 | 1100 | 1069 | 1000 | 0 |
| 4 | 5 | 50000 | 1 | 2 | 1 | 57 | -1 | 0 | -1 | 0 | 0 | 0 | 8617 | 5670 | 35835 | 20940 | 19146 | 19131 | 2000 | 36681 | 10000 | 9000 | 689 | 679 | 0 |

```
print(data.shape)
print(data.describe())
```

```
(30000, 25)
                ID             X1  ...           X23             Y
count  30000.000000    30000.000000  ...  30000.000000  30000.000000
mean   15000.500000   167484.322667  ...   5215.502567      0.221200
std     8660.398374   129747.661567  ...  17777.465775      0.415062
min        1.000000    10000.000000  ...      0.000000      0.000000
25%     7500.750000    50000.000000  ...    117.750000      0.000000
50%    15000.500000   140000.000000  ...   1500.000000      0.000000
75%    22500.250000   240000.000000  ...   4000.000000      0.000000
max    30000.000000  1000000.000000  ...  528666.000000      1.000000

[8 rows x 25 columns]
```

*# distribution of Amount*

plt.hist(data.X6)




print('No Frauds', round(data['Y'].value_counts()[0]/len(data) * 100,2), '% of the datas')

print('Frauds', round(data['Y'].value_counts()[1]/len(data) * 100,2), '% of the datas')

```
No Frauds 77.88 % of the datas
Frauds 22.12 % of the datas
```

*# Determine number of fraud cases in dataset*

```
fraud = data[data['Y'] == 1]
valid = data[data['Y'] == 0]
outlierFraction = len(fraud)/float(len(valid))
print(outlierFraction)
print('Fraud Cases: {}'.format(len(data[data['Y'] == 1])))
print('Valid Transactions: {}'.format(len(data[data['Y'] == 0])))
```

```
0.2840267077555213
Fraud Cases: 6636
Valid Transactions: 23364
```

```
print('Amount details of the fraudulent transaction')
fraud.X1.describe()
```

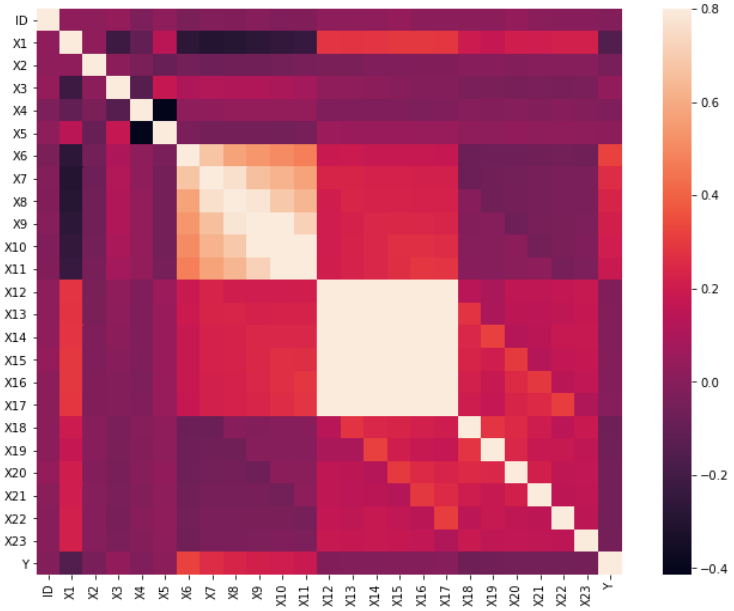
```
Amount details of the fraudulent transaction
count      6636.000000
mean     130109.656420
std      115378.540571
min       10000.000000
25%       50000.000000
50%       90000.000000
75%      200000.000000
max      740000.000000
Name: X1, dtype: float64
```

```
print('details of valid transaction')
valid.X1.describe()
```

```
details of valid transaction
count      23364.000000
mean      178099.726074
std       131628.359660
min        10000.000000
25%        70000.000000
50%       150000.000000
75%       250000.000000
max      1000000.000000
Name: X1, dtype: float64
```

*#Correlationmatrix*

```
corrmat = data.corr()
fig = plt.figure(figsize = (12, 9))
sns.heatmap(corrmat, vmax = .8, square = True)
plt.show()
```

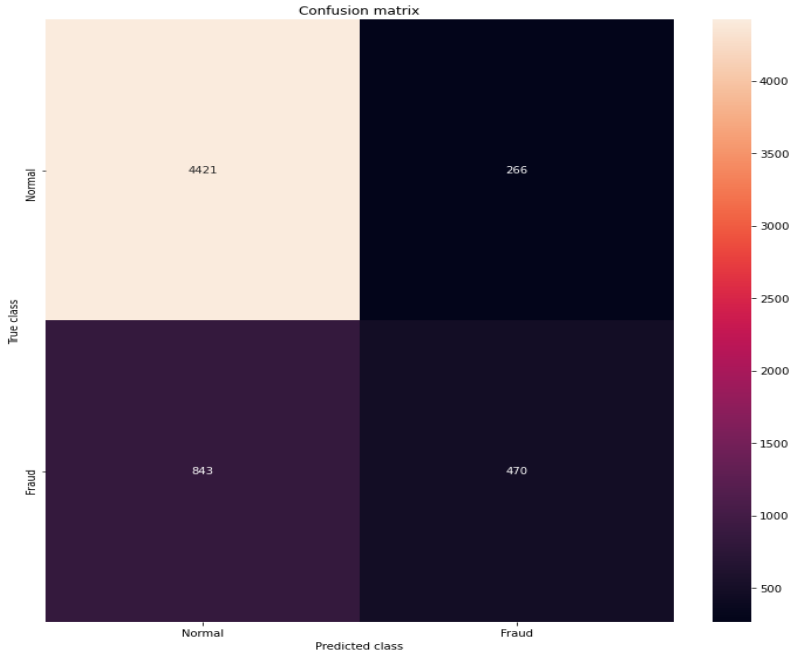


```
# dividing the X and the Y from the dataset
X = data.drop(['Y'], axis = 1)
Y = data['Y']
xData = X.values
yData = Y.values


# Split the data into training and testing sets
xTrain, xTest, yTrain, yTest = train_test_split(
        xData, yData, test_size = 0.2, random_state = 42)

# random forest model creation
rfc = RandomForestClassifier()
rfc.fit(xTrain, yTrain)
yPred = rfc.predict(xTest)
# Evaluating the classifier
print('The model used is Random Forest classifier')
acc = accuracy_score(yTest, yPred)
print('The accuracy is {}'.format(acc))
prec = precision_score(yTest, yPred)
print('The precision is {}'.format(prec))
```

```
The model used is Random Forest classifier
The accuracy is 0.815
The precision is 0.6373477672530447
```

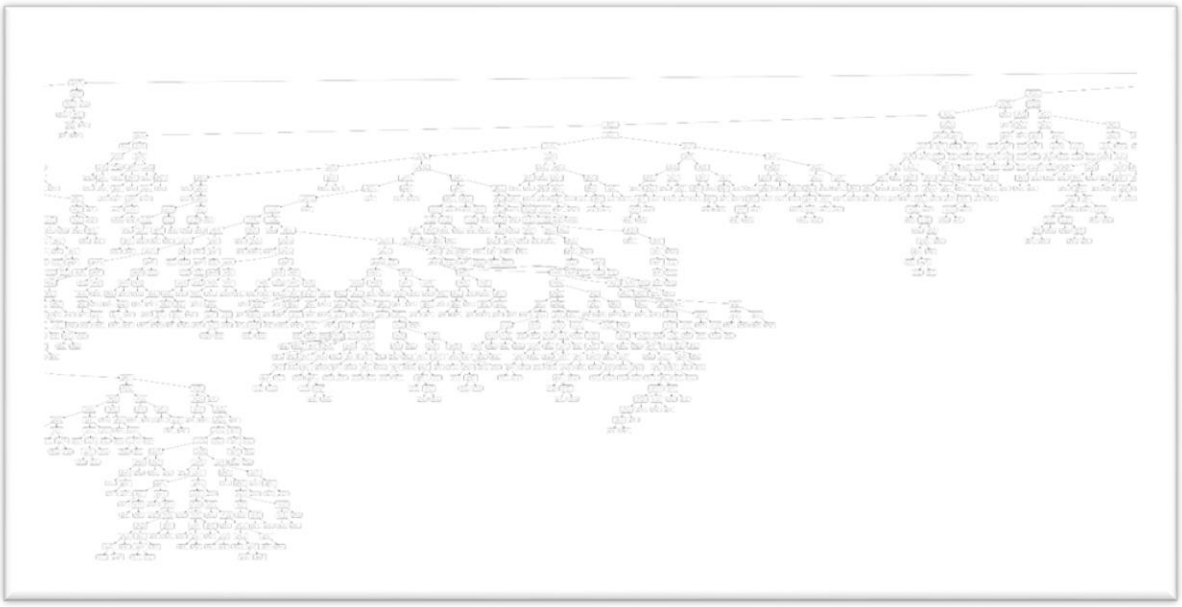
```
# printing the confusion matrix
LABELS = ['Normal', 'Fraud']
conf_matrix = confusion_matrix(yTest, yPred)
plt.figure(figsize =(12, 12))
```

```
sns.heatmap(conf_matrix, xticklabels = LABELS,
            yticklabels = LABELS, annot = True, fmt ="d");
plt.title('Confusion matrix')
plt.ylabel('True class')
plt.xlabel('Predicted class')
plt.show()
```




*#visualizing the random tree*

```
feature_list = list(X.columns)
tree = rfc.estimators_[5]
export_graphviz(tree, out_file = 'tree.dot', feature_names = feature_list, rounded =
True, precision = 1)
(graph, ) = pydot.graph_from_dot_file('tree.dot')
display(Image(graph.create_png()))
```



# CHAPTER 7
# CONCLUSION & ENHANCEMENT

## 7.1 Conclusion

Credit card fraud has increased exponentially in recent years. Accordingly, one of the main tasks of the financial industries is to develop an accurate and easy credit card fraud detection system. There are different categories of transaction scam as well as several methods to detect them. However, all these methods remain insufficient.

This paper has examined the performance of Random forest model, which identifying the fraud transactions occurring during the transactions made by the card holder. It is an effective technique that helps implement a credit card fraud detection system.

### 7.2 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. One we can change is that use large amount of data and apply multiple algorithms. 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.

# CHAPTER 8
# REFERENCES

[1] “Credit Card Fraud Detection using Machine Learning and Data Science” published by International Journal of Engineering Research & Technology (IJERT) ISSN: 2278- 0181 Vol. 8 Issue 09, September-2019

[2] “Credit Card Fraud Detection System using Smote Technique and Whale Optimization Algorithm” published by International Journal of Engineering and Advanced Technology (IJEAT) ISSN: 2249-8958, Volume-8 Issue-5, June 2019

[3]“Credit Card Fraud Detection: A Realistic Modeling and a Novel Learning Strategy” published by IEEE TRANSACTIONS ON NEURAL NETWORKS AND LEARNING SYSTEMS, VOL. 29, NO. 8, AUGUST 2018

[4] Credit Card Fraud Detection through Par enclitic Network Analysis By Massimiliano Zanin, Miguel Romance, Regino Criado, and SantiagoMoral” published by Hindawi Complexity Volume 2018, Article ID 5764370

[5] A. Naik, K. Phulmamdikar, S. Pradhan, and S. Thorat, “Real Time Credit Card Transaction Analysis,” vol. 1, no. 11, pp. 1663–1666, 2016