A

Major Project

On

CREDIT CARD FRAUD DETECTION USING ADABOOSTAND MAJORITY VOTING

Submitted to

Jawaharlal Nehru Technological University, Hyderabad

In partial fulfillment of the requirements for the award of Degree

BACHELOR OF TECHNOLOGY
in

COMPUTER SCIENCE & ENGINEERING

By

V. Phanindra Shivaji	(187R1A05P5)
R. Bindhu Madhavi	(197R5A0516)
G. Naga Lakshmi	(187R1A05K2)
C G Sri Chakradhar Kishan	(187R5A0511)

Under the esteemed Guidance of

G. Pavan Kumar

(Assistant Professor)

DEPARTMENT OF COMPUTER SCIENCE & ENGINEERING

CMR TECHNICAL CAMPUS

UGC AUTONOMOUS

(Accredited byNAAC,NBA,Permanently Affiliated to JNTUH, Approved by AICTE, New Delhi)
Recognized Under Section 2(f)&12(B) of the UGC Act . 1956,Kandlakoya(V),
Medchal Road, Hyderabad-501401.

2018-2022

DEPARTMENT OF COMPUTER SCIENCE & ENGINEERING

CERTIFICATE

This is to certify that the project entitled "**CREDIT CARD FRAUD DETECTION USING ADABOOST AND MAJORITY VOTING**" being submitted by **V. PHANINDRA SHIVAJI (187R1A05P5), R. BINDHU MADHAVI (197R5A0516), G. NAGA LAKSHMI (187R1A05K2), C G SRI CHAKRADHAR KISHAN (187R5A0511)** in partial fulfillment of the requirements for the award of the degree of B. Tech in Computer Science and Engineering of the Jawaharlal Nehru Technological University Hyderabad, is a record of bonafide work carries out by him/her under our guidance and supervision during the year 2021-2022.

The results embodied in this have not been submitted to any other University or Institute for the award of any degree or diploma.

G. Pavan Kumar
Assistant Professor
INTERNAL GUIDE

Dr. A. Raji Reddy
DIRECTOR

Dr. K. Srujan Raju
HOD

EXTERNAL EXAMINER

Submitted on viva voice Examination held on__________________

ACKNOWLEDGEMENT

A part from the efforts of us, the success of any project depends largely on the encouragement and guidelines of many others. We take this opportunity to express our gratitude to the people who have been instrumental in the successful completion of this project.

We take this opportunity to express my profound gratitude and deep regard to my guide

G. Pavan Kumar, Assnt. Professor for his exemplary guidance, monitoring and constant encouragement throughout the project work. The blessing, help and guidance given by him shall carry us a long way in the journey of life on which we are about to embark.

We also take this opportunity to express a deep sense of gratitude to Project Review Committee (PRC) Coordinators: **Mr. A. Uday Kiran, Mr. J. Narasimha Rao, Dr. T. S. Mastan Rao, Mrs. G. Latha , Mr. A. Kiran Kumar** for their cordial support, valuable information and guidance, which helped us in completing this task through various stages.

We are also thankful to **Dr. K. Srujan Raju**, Head of the Department of Computer Science and Engineering for providing excellent infrastructure and a nice atmosphere for completing this project successfully.

We are obliged to our Director **Dr. A. Raji Reddy** for being cooperative throughout the courseof this project. We would like to express our sincere gratitude to our chairman Sri. **Ch. Gopal Reddy** for his encouragement throughout the course of this project.

The guidance and support received from all the members of **CMR TECHNICAL CAMPUS** who contributed for the completion of the project, was vital for the success of the project. We are grateful for their constant support and help.We are grateful for their constant support and help.

Finally, we would like to take this opportunity to thank our family for their constant encouragement, without which this assignment would not be possible. We sincerely acknowledge and thank all those who gave support directly and indirectly in completion of this project.

V.PHANINDRA SHIVAJI (187R1A05P5)
R. BINDHU MADHAVI (197R5A0516)
G.NAGA LAKSHMI (187R1A05K2)
SRI CHAKRADHAR KISHAN (187R5A0511)

ABSTRACT

In the financial services industry, credit card fraud is a big issue. Credit card fraud costs billions of rupees every year. Due to confidentiality concerns, there are few research studies on evaluating real-world credit card data. Machine learning techniques are employed to detect credit card fraud in this article. Standard models are employed first, and then After that, AdaBoost-based hybrid algorithms and majority voting are utilised. The model's performance is evaluated using publicly available credit card data. A real-world credit card data set from a financial institution is then used to analyse the data. In addition, noise is introduced into the data samples to test the algorithms robustness. The experimental findings show that the majority voting method detects credit card fraud situations with a high degree of accuracy.

LIST OF FIGURES

LIST OF SCREENSHOTS

TABLE OF CONTENTS

Page No’s

1. INTRODUCTION

1. INTRODUCTION

1.1 PROJECT SCOPE

The project titled as "Credit Card Fraud Detection Using Adaboost and Majority Voting” is a dynamic application.Using this application, We are able to detect fraud with a specific user's credit card. To use the application, the user must first launch it. After that, the user can check all transactions made with the specific account, along with all associated details, as well as recent fraud. The majority list of the fraud will be generated after the application has finished running.

1.2 PROJECT PURPOSE

The project's goal is to detect Credit Card Fraud by combining data from previous credit card transactions with data from those that turned out to be fraudulent. The model is then used to determine whether or not a new transaction is fraudulent.

1.3 PROJECT FEATURES

This project's features are built on the basis of sample fraudulent datasets. These are data items such as the customer account's age and value, as well as the credit card's origin. There are hundreds of features, each of which contributes to the likelihood of fraud to varied degrees.

2.SYSTEM ANALYSIS

2.SYSTEM ANALYSIS

SYSTEM ANALYSIS

System Analysis is the important phase in the system development process. The System is studied to the minute details and analyzed. The system analyst plays an important role of an interrogator and dwells deep into the working of the present system. In analysis, a detailed study of these operations performed by the system and their relationships within and outside the system is done. A key question considered here is, "what must be done to solve the problem?" The system is viewed as a whole and the inputs to the system are identified. Once analysis is completed the analyst has a firm understanding of what is to be done.

2.1 PROBLEM DEFINITION

Various fraudulent activity detection approaches have been implemented in credit card transactions, and strategies to construct models based on artificial intelligence, data mining, fuzzy logic, and machine learning have been retained in researcher thoughts. The identification of credit card fraud is a challenging but common problem to handle. Machine learning was used to build the credit card fraud detection in our suggested system. Machine learning techniques are becoming more advanced. Machine learning has been highlighted as a useful tool for detecting fraud. During online transaction operations, a great amount of data is sent, resulting in a binary result. Machine learning is used to detect credit card fraud by employing classification and regression algorithms. To classify fraudulent card transactions, we use supervised learning algorithms such as the Random forest technique. The Random Forest algorithm is a more advanced variant of the Decision Tree algorithm. Random forest outperforms the other machine learning algorithms in terms of efficiency and accuracy. By selecting only a subsample of the feature space at each split, random forest seeks to alleviate the previously mentioned correlation issue. Essentially, it seeks to de-correlate and prune the trees by establishing a node split stopping criteria, which I will go over in more depth later.

2.2 EXISTING SYSTEM

A study of a case study involving credit card fraud detection in which data normalisation is applied before Cluster Analysis and results obtained from the use of ClusterAnalysis and Artificial Neural Networks on fraud detection has shown that neuronal inputs can be minimised by clustering attributes in the existing system. Using normalised data withdata that has been MLP trained can also yield good results. Unsupervised learning was usedin this study. The purpose of this article was to develop new approaches for detecting fraudand to improve the accuracy of the results. The data set for this article is based on real-worldtransactional data from a large European corporation, with personal information maintained private.using data parameter value. An algorithm's accuracy is estimated to be around 50%.The purpose of this paper was to develop an algorithm and lower the cost measure. The outcome was a 23 percent increase.

2.2.1 LIMITATIONS OF EXISTING SYSTEM

- Low Accuracy
- A cost sensitive method
- Low Efficiency

2.3 PROPOSED SYSTEM

In proposed System, We use the random forest technique and Adaboost technique to classify the credit card dataset in the suggested system. Random Forest is a classification and regression algorithm. Random forest training is incredibly quick, even for big data sets with numerous characteristics and data instances because each tree is trained independently of the others.The binary classification's goal class is 'class,' which has a value of 1 for a positive case (fraud) and 0 for a negative instance (not fraud).To create hybrid models, the AdaBoost and majority voting methods are used. The evaluation of a range of machine learning models with a real-world credit card data set for fraud detection is the project's main contribution.

2.3.1 ADVANTAGES OF THE PROPOSED SYSTEM

- Easy to Understand and Implement.
- Require Low Computational Power.
- Provide Optimal Result.

2.4 FEASIBILITY STUDY

Preliminary investigation examine project feasibility, the likelihood the system willbe useful to the organization. The main objective of the feasibility study is to test the Technical, Social and Economical feasibility for adding new modules and debugging old running system. All system is feasible if they are unlimited resources and infinite time.Threekey considerations involved in the feasibility analysis are

- Economical Feasibility
- Technical Feasibility
- Social Feasibility

2.4.1 ECONOMICAL FEASIBILITY

This research is being carried out to determine the system's economic impact on the organisation. The amount of money the corporation has to invest in the system's research and development is limited. It is necessary to justify the spending. As a result, the produced system came in under budget, which was made possible by the fact that the majority of the technologies used were freely available. The customized products were only ones needed to be acquired.

2.4.2 TECHNICAL FEASIBILITY

This research is being carried out to determine the system's technological feasibility, or technical requirements. Any system that is created should not place a large burden on the available technical resources. As a result, there will be a lot of demand on the available technical resources. As a result, the client will be subjected to severe demands. Because very minor or no changes are necessary to implement this system, the designed system must have a low requirement.

2.4.3 SOCIAL FEASIBILITY

The purpose of the study is to determine the user's level of acceptance of the system. This covers the process of teaching the user how to effectively use the technology. The user should not be afraid of the system, but rather accept it as a need. The methods used to educate and familiarise the user with the system are totally responsible for the level of acceptance by the users. His self-esteem must be boosted so that he can offer constructive criticism, which is encouraged because he is the system's final user.

2.5 HARDWARE & SOFTWARE REQUIREMENTS

2.5.1 HARDWARE REQUIREMENTS:

Hardware interfaces specifies the logical characteristics of each interface between the software product and the hardware components of the system. The following are some hardware requirements.

Processor	:	intel core i3.
RAM	:	minimum 4GB.
Hard Disk	:	minimum 250GB.

2.5.2 SOFTWARE REQUIREMENTS:

Software Requirements specifies the logical characteristics of each interface and software components of the system.The following are some software requirements.

Operating System	:	Windows 7 & above.
Coding Language	:	Python 3.7.
Tool	:	Anaconda 3.7.

3. ARCHITECTURE

3. ARCHITECTURE

3.1 PROJECT ARCHITECTURE

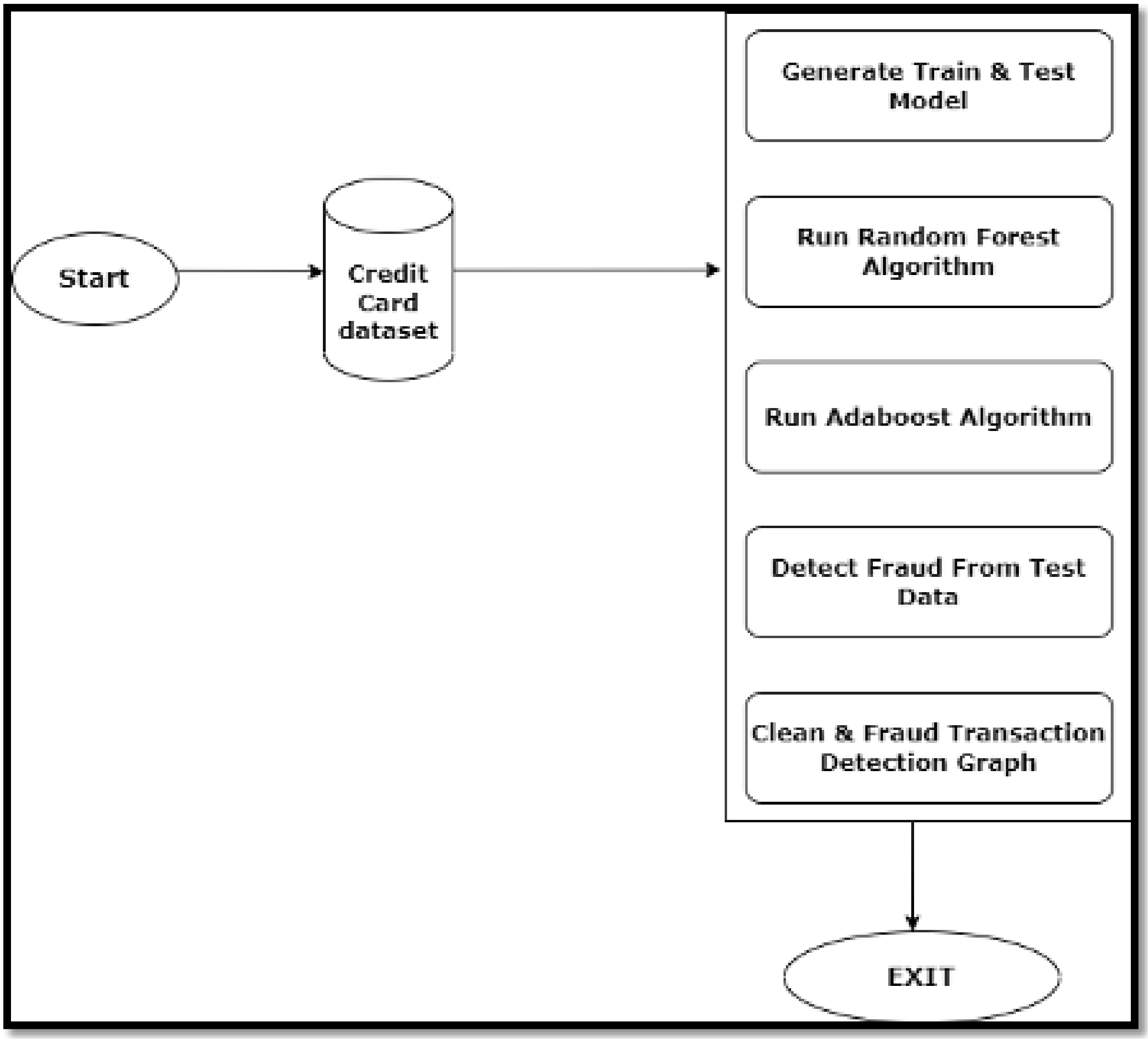

Start
Credit Card dataset
Generate Train & Test Model
Run Random Forest Algorithm
Run Adaboost Algorithm
Detect Fraud From Test Data
Clean & Fraud Transaction Detection Graph
EXIT

Fig.3.1 Project Architecture of Credit Card Fraud Detection.

3.2 MODULES DESCRIPTION

Modules

- Upload Credit Card Dataset
- Generate Train & Test Model
- Run Random Forest Algorithm
- Detect Fraud From Test Data
- Clean & Fraud Transaction Detection Graph

Upload Credit Card Dataset

In this module user upload Credit Card Dataset.

Generate Train & Test Model

In this module user train & test model through dataset.

Run Random Forest Algorithm

In this module random forest algorithm classify dataset.

Detect Fraud From Test Data

In this module fraud is detected from dataset.

Clean & Fraud Transaction Detection Graph

In this module clean & Fraud Transaction detection graph is shown.

3.3 USE CASE DIAGRAM

In the use case diagram, is a type of behavioral diagram defined by and created from a Use-case analysisIts purpose is to present a graphical overview of the functionality provided by a system in terms of actors, their goals (represented as use cases), and any dependencies between those use cases. The main purpose of a use case diagram is to show what system functions are performed for which actor. Roles of the actors in the system can be depicted.

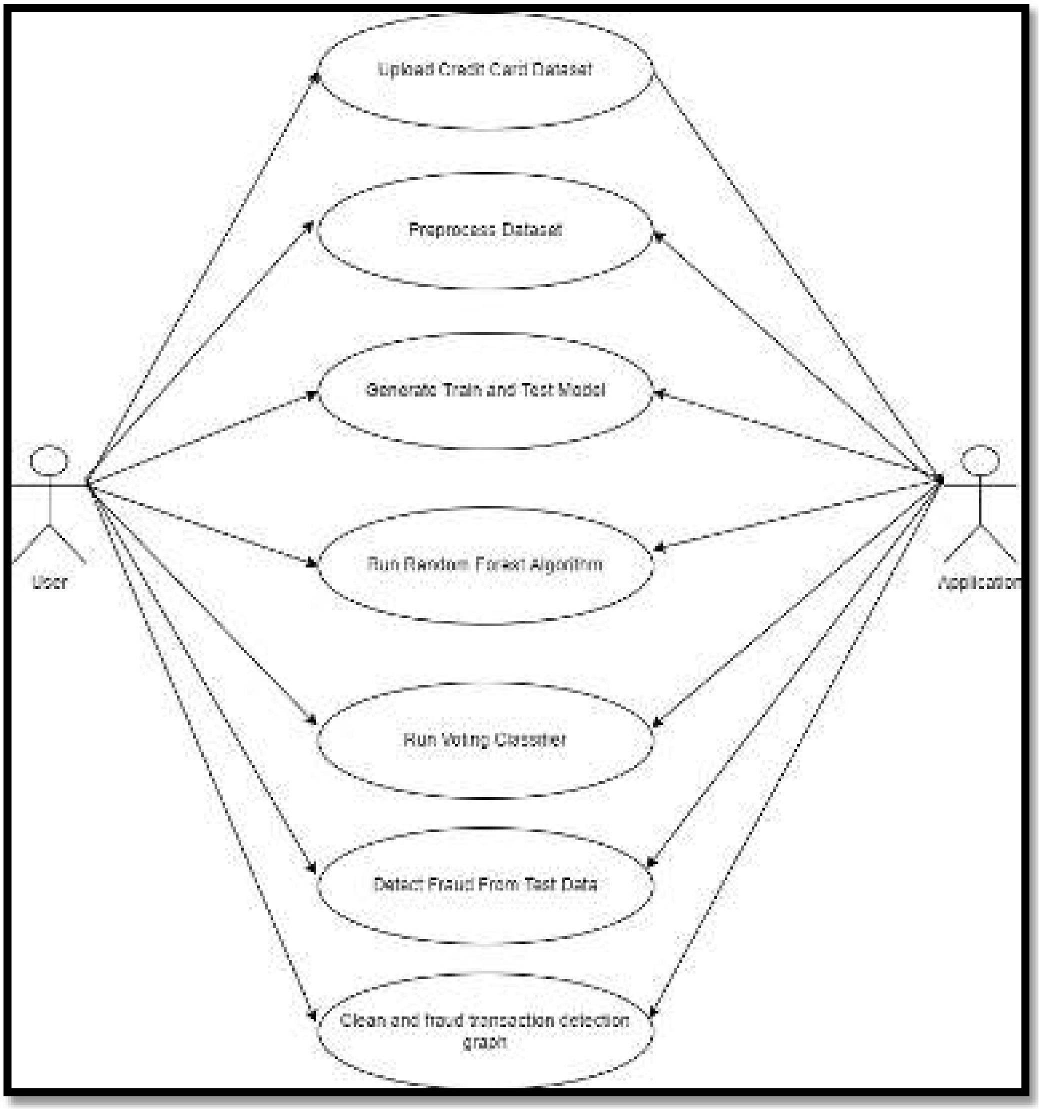

Upload Credit Card Dataset
Preprocess Dataset
Generate Train and Test Model
User
Run Random Forest Algorithm
Applicatior
Run Voting Classifier
Detect Fraud From Test Data
Clean and fraud transaction detection graph

Fig.3.3 Use Case Diagram

3.4 CLASS DIAGRAM

Class Diagram is a collection of classes and objects.

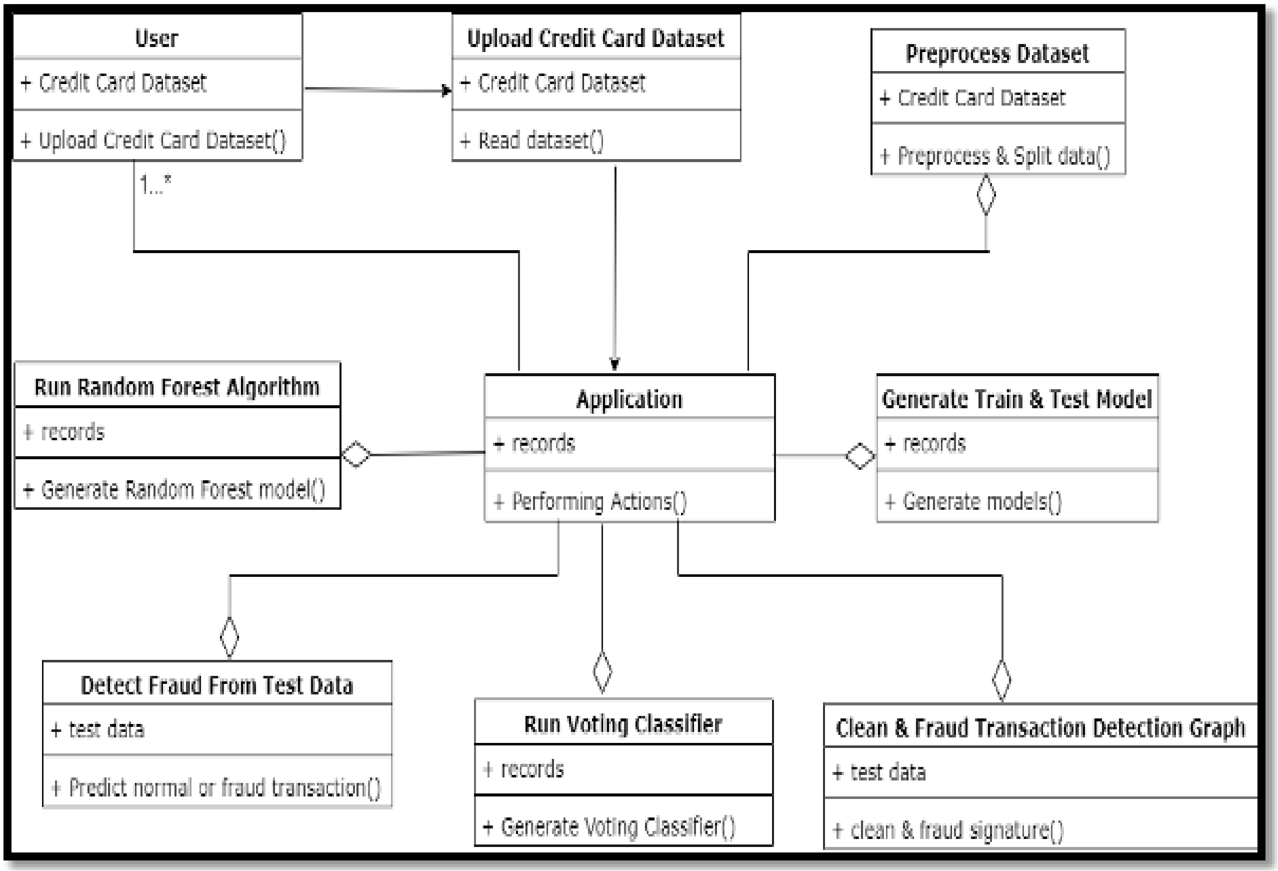

User
+ Credit Card Dataset
+ Upload Credit Card Dataset()
1...*
Upload Credit Card Dataset
+ Credit Card Dataset
+ Read dataset()
Preprocess Dataset
+ Credit Card Dataset
+ Preprocess & Split data()
Run Random Forest Algorithm
+ records
+ Generate Random Forest model()
Application
+ records
+ Performing Actions()
Generate Train & Test Model
+ records
+ Generate models()
Detect Fraud From Test Data
+ test data
+ Predict normal or fraud transaction()
Run Voting Classifier
+ records
+ Generate Voting Classifier()
Clean & Fraud Transaction Detection Graph
+ test data
+ clean & fraud signature()

Fig.3.4 Class Diagram

3.5 SEQUENCE DIAGRAM

The sequence diagram shows the sequence in which different tasks are being carried out by the actors.

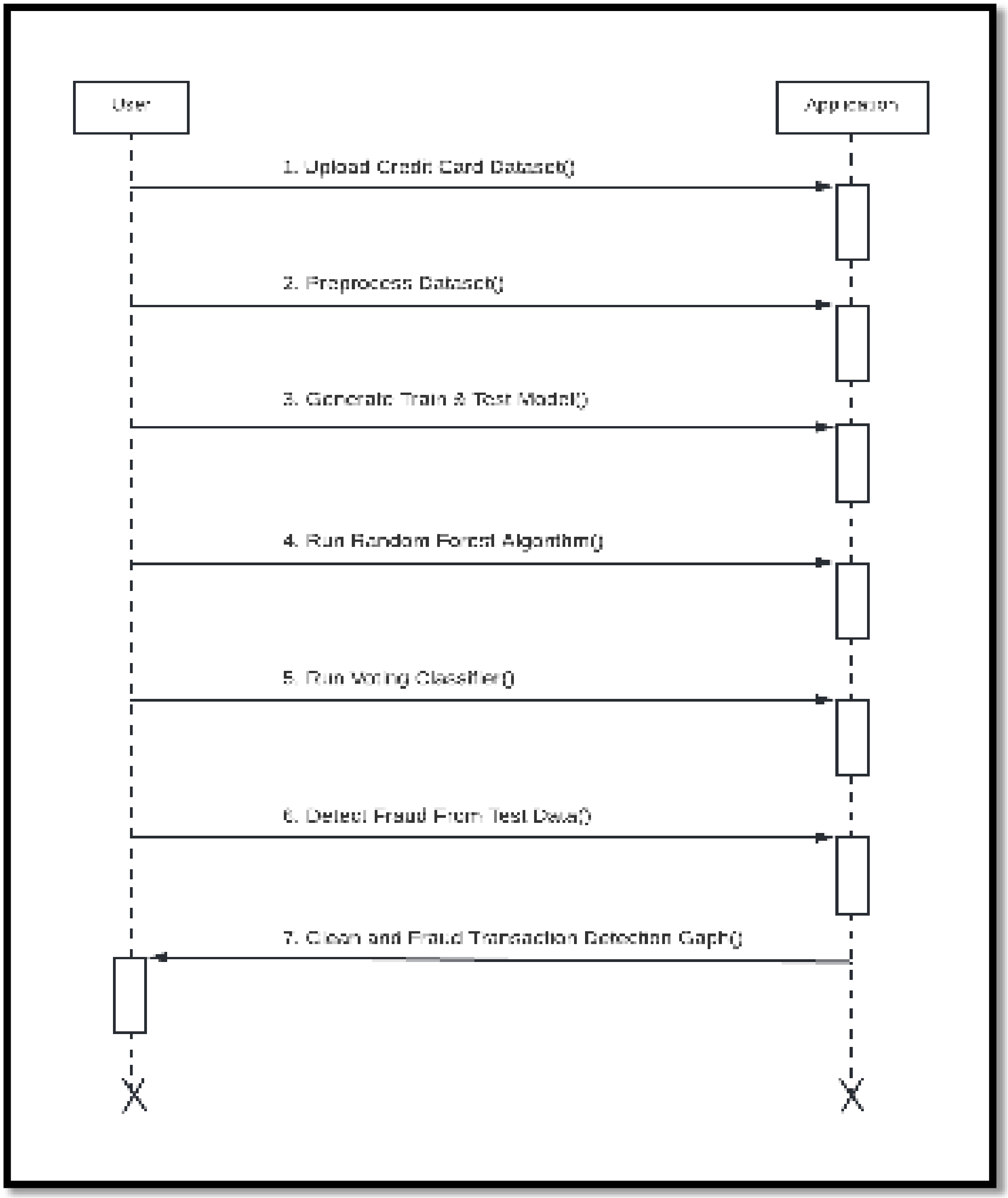

User
Application
1. Upload Credit Card Dataset()
2. Preprocess Dataset()
3. Generate Train & Test Model()
4. Run Random Forest Algorithm()
5. Run Voting Classifier()
6. Detect Fraud From Test Data()
7. Clean and Fraud Transaction Detection Gaph()

Fig.3.5 Sequence Diagram

3.6 ACTIVITY DIAGRAM

It describes the flow of activity states

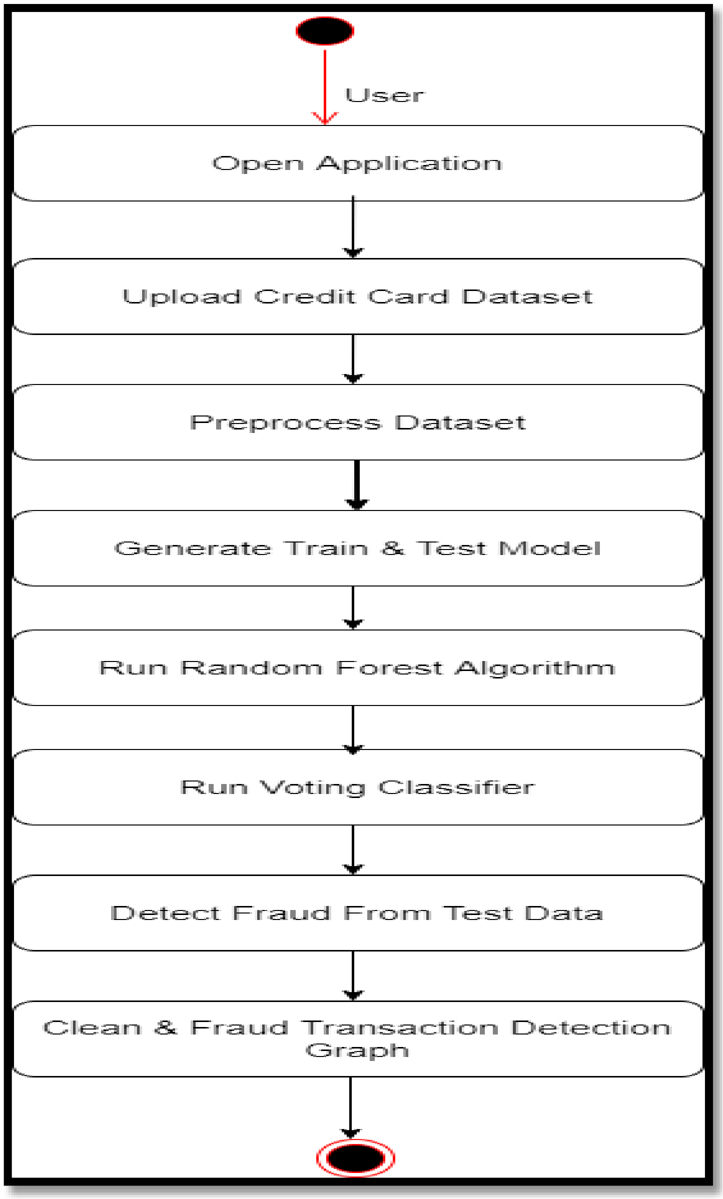

User
Open Application
Upload Credit Card Dataset
Preprocess Dataset
Generate Train & Test Model
Run Random Forest Algorithm
Run Voting Classifier
Detect Fraud From Test Data
Clean & Fraud Transaction Detection Graph

Fig 3.6 Activity Diagram

3.7 DATA-FLOW DIAGRAM

Data-Flow Diagram represents a flow pf data through a process

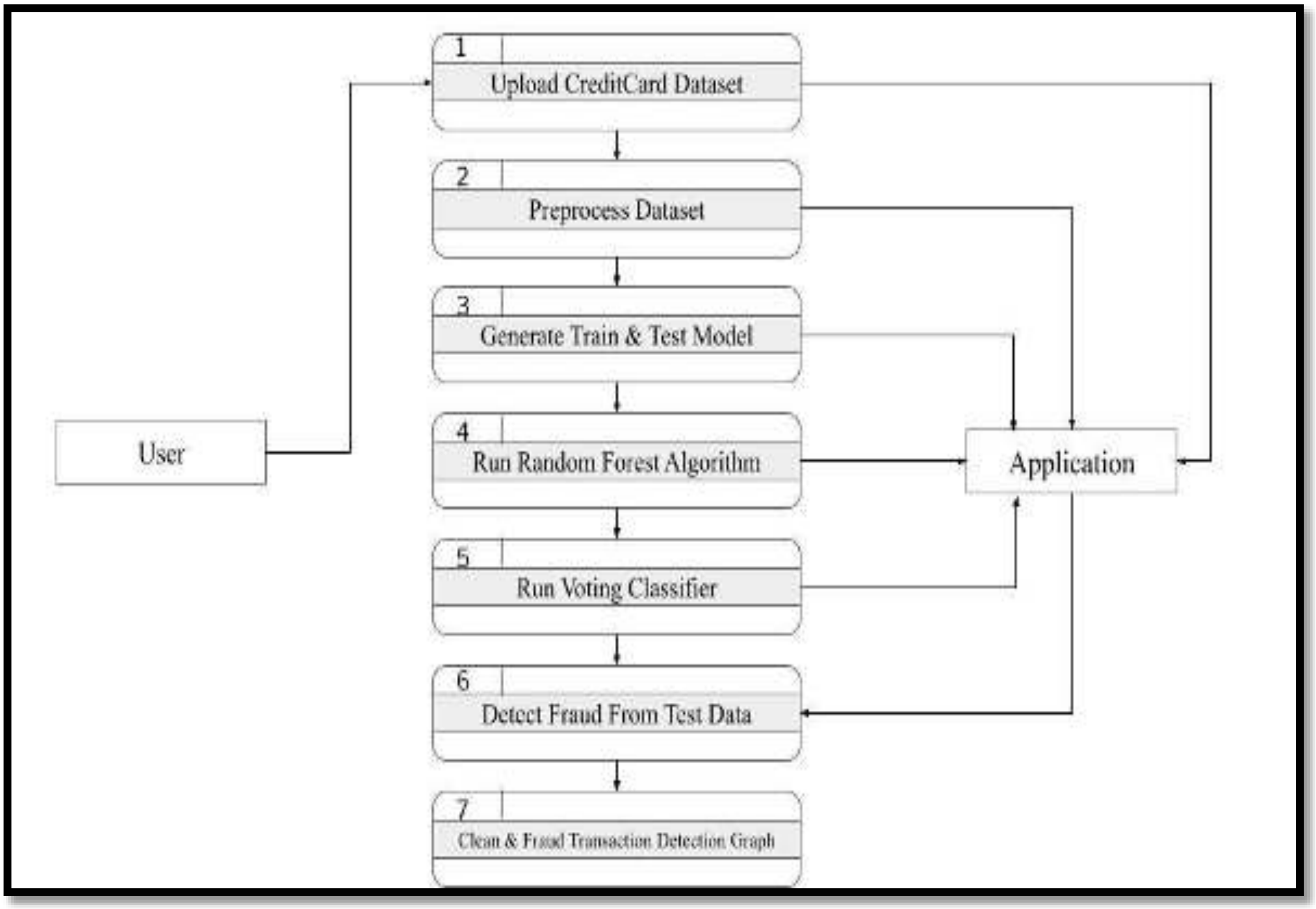

1
Upload CreditCard Dataset
2
Preprocess Dataset
3
Generate Train & Test Model
4
Run Random Forest Algorithm
5
Run Voting Classifier
6
Detect Fraud From Test Data
7
Clean & Fraud Transaction Detection Graph
User
Application

Fig 3.7 Dataflow Diagram

4.IMPLEMENTATION

4.IMPLEMENTATION

4.1 SAMPLE CODE

```
from tkinter import messagebox
from tkinter import *
from tkinter import simpledialog
import tkinter
from tkinter import filedialog
import matplotlib.pyplot as plt
import numpy as np
from tkinter.filedialog import askopenfilename
import numpy as np
import pandas as pd
from sklearn import *
from sklearn.model_selection import train_test_split
from sklearn.metrics import accuracy_score
from sklearn.metrics import classification_report
from sklearn.ensemble import RandomForestClassifier
        from sklearn.ensemble import AdaBoostClassifier
#from sklearn.tree import export_graphviz
#from IPython import display

main = tkinter.Tk()
main.title("Credit Card Fraud Detection") #designing main screen
main.geometry("1300x1200")

global filename
global cls
global X, Y, X_train, X_test, y_train, y_test
global random_acc # all global variables names define in above lines
global clean
global attack
global total def traintest(train):     #method to generate test and train data from dataset


 X = train.values[:, 0:29]
   Y = train.values[:, 30]
   print(X)
   print(Y)
   X_train, X_test, y_train, y_test = train_test_split(
   X, Y, test_size = 0.3, random_state = 0)
   return X, Y, X_train, X_test, y_train, y_test
```

```
def generateModel(): #method to read dataset values which contains all five features data
    global X, Y, X_train, X_test, y_train, y_test

    train = pd.read_csv(filename)
    X, Y, X_train, X_test, y_train, y_test = traintest(train)
    text.insert(END,"Train & Test Model Generated\n\n")
    text.insert(END,"Total Dataset Size : "+str(len(train))+"\n")
    text.insert(END,"Split Training Size : "+str(len(X_train))+"\n")
    text.insert(END,"Split Test Size : "+str(len(X_test))+"\n")



def upload(): #function to upload tweeter profile
    global filename
    filename = filedialog.askopenfilename(initialdir="dataset")
    text.delete('1.0', END)
    text.insert(END,filename+" loaded\n");



def prediction(X_test, cls):  #prediction done here
    y_pred = cls.predict(X_test)
    for i in range(50):
      print("X=%s, Predicted=%s" % (X_test[i], y_pred[i]))
    return y_pred

# Function to calculate accuracy
def cal_accuracy(y_test, y_pred, details):
    accuracy = accuracy_score(y_test,y_pred)*100
    text.insert(END,details+"\n\n")
    text.insert(END,"Accuracy : "+str(accuracy)+"\n\n")
    return accuracy


def runRandomForest():
    headers =
["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","Class"]
    global random_acc
    global cls
    global X, Y, X_train, X_test, y_train, y_test
    cls =
RandomForestClassifier(n_estimators=50,max_depth=2,random_state=0,class_weight='balance
d')
```

```
 cls.fit(X_train, y_train)
    text.insert(END,"Prediction Results\n\n")
    prediction_data = prediction(X_test, cls)
    random_acc = cal_accuracy(y_test, prediction_data,'Random Forest Accuracy')
    #str_tree = export_graphviz(cls, out_file=None, feature_names=headers,filled=True,
special_characters=True, rotate=True, precision=0.6)
    #display.display(str_tree)
def runada():
    headers =
["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","Class"]
    global random_acc
    global cls
    global X, Y, X_train, X_test, y_train, y_test
    cls = AdaBoostClassifier(n_estimators=100, random_state=0)
    cls.fit(X_train, y_train)
    text.insert(END,"Prediction Results\n\n")
    prediction_data = prediction(X_test, cls)
    random_acc = cal_accuracy(y_test, prediction_data,'Ada Boost')

def predicts():
    global clean
    global attack
    global total
    clean = 0;
    attack = 0;
    text.delete('1.0', END)
    filename = filedialog.askopenfilename(initialdir="dataset")
    test = pd.read_csv(filename)
    test = test.values[:, 0:29]
    total = len(test)
    text.insert(END,filename+" test file loaded\n");
    y_pred = cls.predict(test)
    for i in range(len(test)):
        if str(y_pred[i]) == '1.0':
            attack = attack + 1
            text.insert(END,"X=%s, Predicted = %s" % (test[i], 'Contains Fraud Transaction
Signature')+"\n\n")
        else:
            clean = clean + 1
            text.insert(END,"X=%s, Predicted = %s" % (test[i], 'Transaction Contains Cleaned
Signatures')+"\n\n")
```

```
def graph():
    height = [total,clean,attack]
    bars = ('Total Transactions','Normal Transaction','Fraud Transaction')
    y_pos = np.arange(len(bars))
    plt.bar(y_pos, height)
    plt.xticks(y_pos, bars)
    plt.show()

font = ('times', 16, 'bold')
title = Label(main, text='Credit Card Fraud Detection Using Random Forest Tree Based
Classifier')
title.config(bg='greenyellow', fg='dodger blue')
title.config(font=font)
title.config(height=3, width=120)
title.place(x=0,y=5)

font1 = ('times', 12, 'bold')
text=Text(main,height=20,width=150)
scroll=Scrollbar(text)
text.configure(yscrollcommand=scroll.set)
text.place(x=50,y=120)
text.config(font=font1)


font1 = ('times', 14, 'bold')
uploadButton = Button(main, text="Upload Credit Card Dataset", command=upload)
uploadButton.place(x=50,y=550)
uploadButton.config(font=font1)

modelButton = Button(main, text="Generate Train & Test Model", command=generateModel)
modelButton.place(x=350,y=550)
modelButton.config(font=font1)

runrandomButton = Button(main, text="Run Random Forest Algorithm",
command=runRandomForest)
runrandomButton.place(x=650,y=550)
runrandomButton.config(font=font1)

runadaButton = Button(main, text="Run Ada Boost Algorithm", command=runada)
runadaButton.place(x=950,y=550)
runadaButton.config(font=font1)

predictButton = Button(main, text="Detect Fraud From Test Data", command=predicts)
predictButton.place(x=50,y=600)
predictButton.config(font=font1)
```

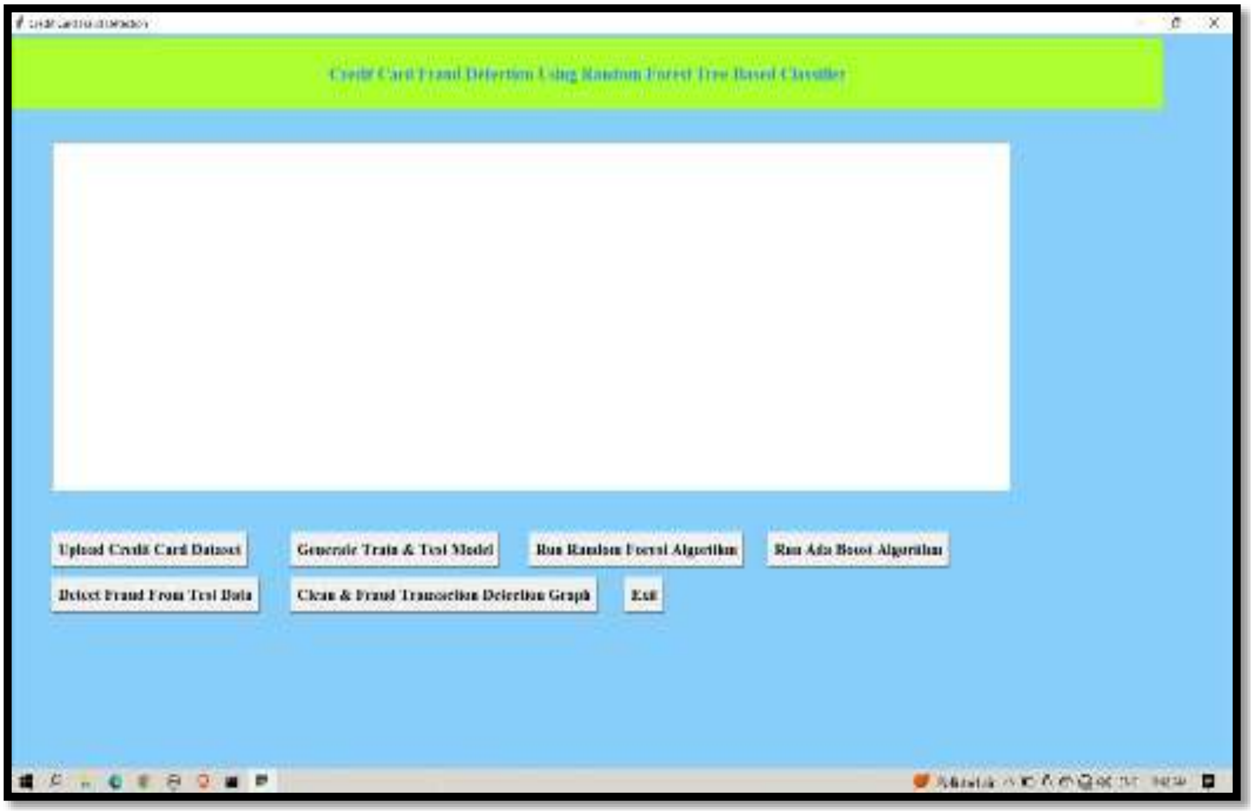
```
graphButton = Button(main, text="Clean & Fraud Transaction Detection Graph",
command=graph)
graphButton.place(x=350,y=600)
graphButton.config(font=font1)


exitButton = Button(main, text="Exit", command=exit)
exitButton.place(x=770,y=600)
exitButton.config(font=font1)

main.config(bg='LightSkyBlue')
main.mainloop()
```

5.RESULTS

5. RESULTS

5.1 HOME PAGE

Home page looks like this.



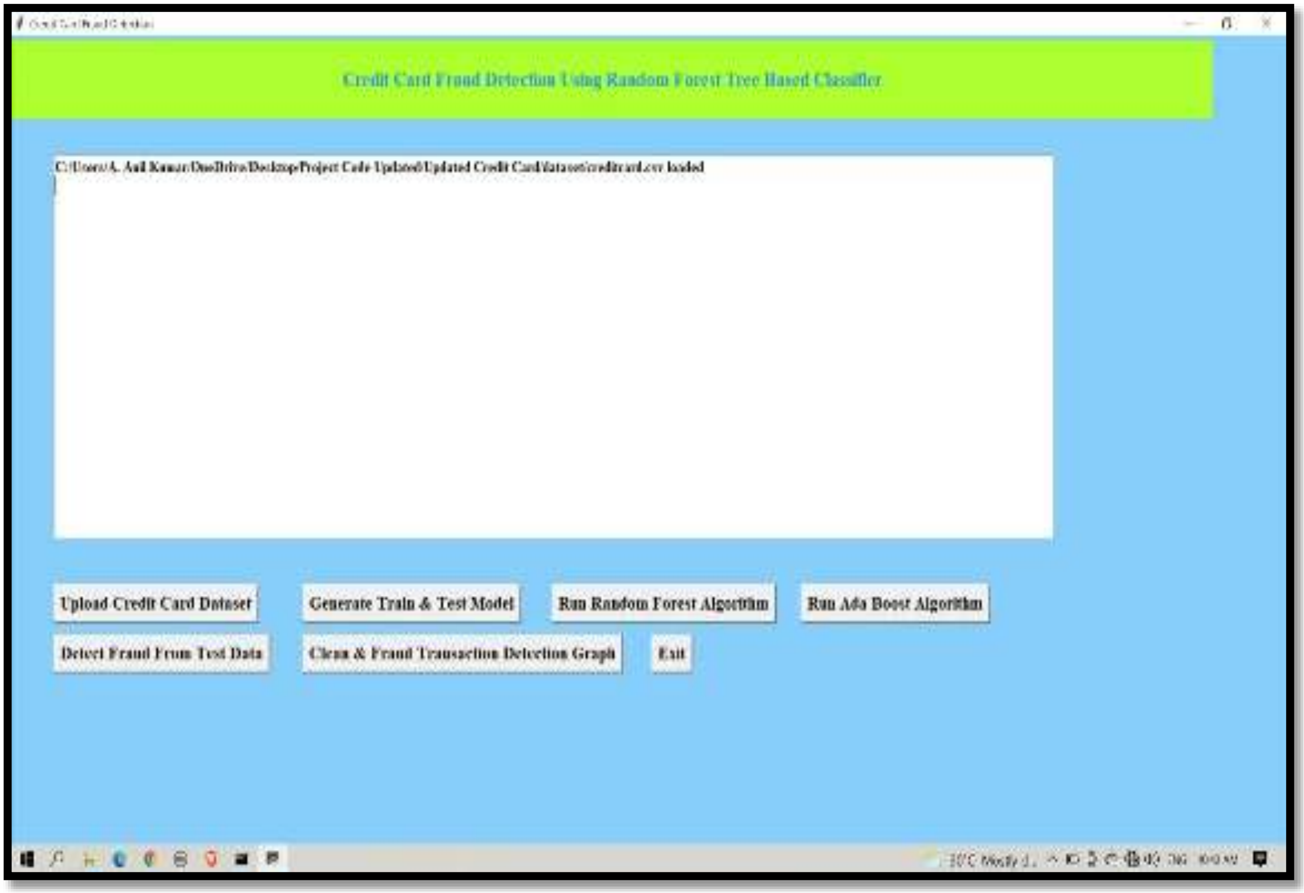
Credit Card Fraud Detection Using Random Forest Tree Based Classifier

Upload Credit Card Dataset
Generate Train & Test Model
Run Random Forest Algorithm
Run Ada Boost Algorithm
Detect Fraud From Test Data
Clean & Fraud Transaction Detection Graph
Exit

Screenshot 5.1 Home Page

5.2 Upload Dataset

Click on the “upload Credit Card Dataset” button to upload the data.

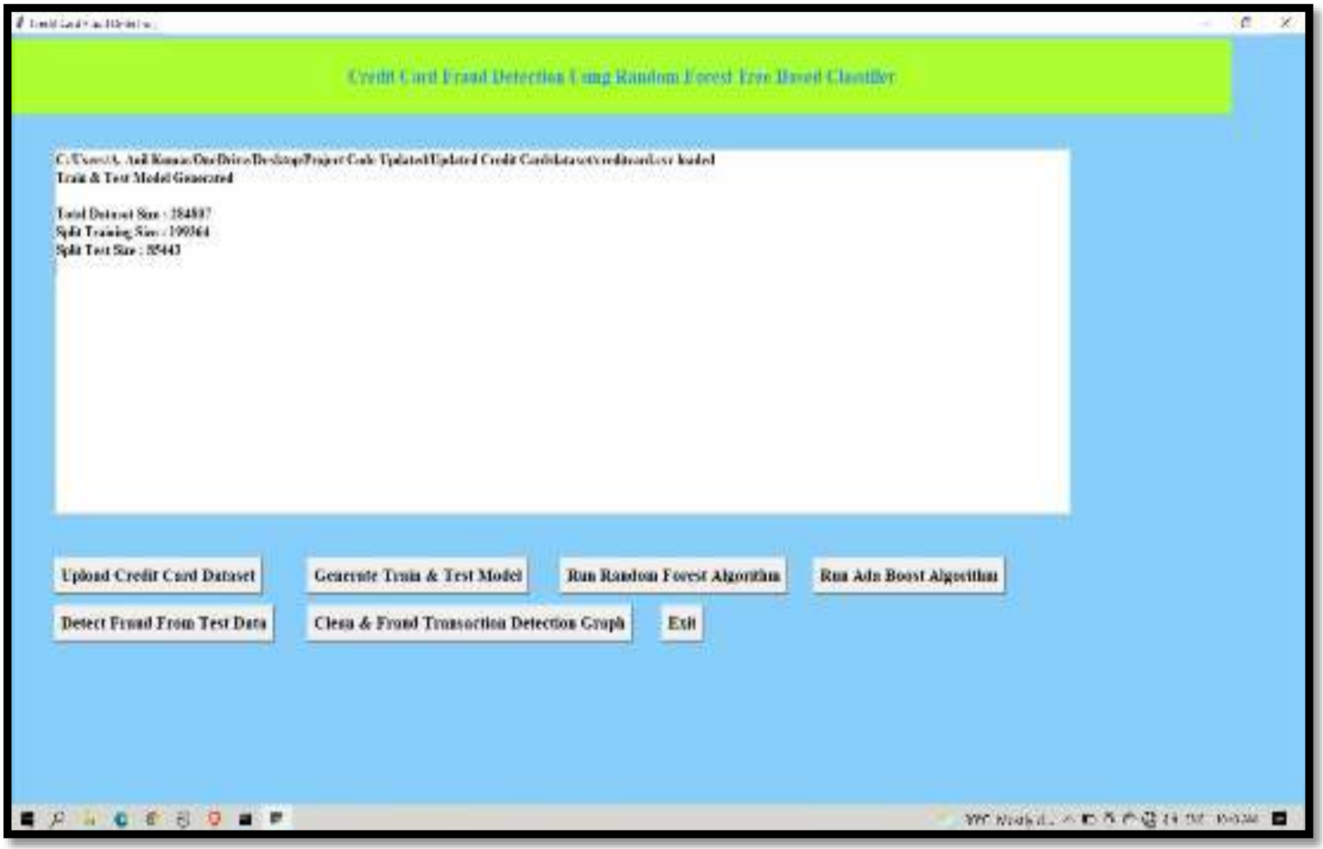

Credit Card Fraud Detection Using Random Forest Tree Based Classifier
Upload Credit Card Dataset
Generate Train & Test Model
Run Random Forest Algorithm
Run Ada Boost Algorithm
Detect Fraud From Test Data
Clean & Fraud Transaction Detection Graph
Exit

```
C:/Users/A. Anil Kumar/OneDrive/Desktop/Project Code Updated/Updated Credit Card/dataset/creditcard.csv loaded
```

Screenshot 5.2 Upload Dataset

5.3 Train and Test Model

To Train and Test the dataset click on the “Generate train and test model”. We can examine the total number of records in the dataset and then use how many records for training and testing in the application.

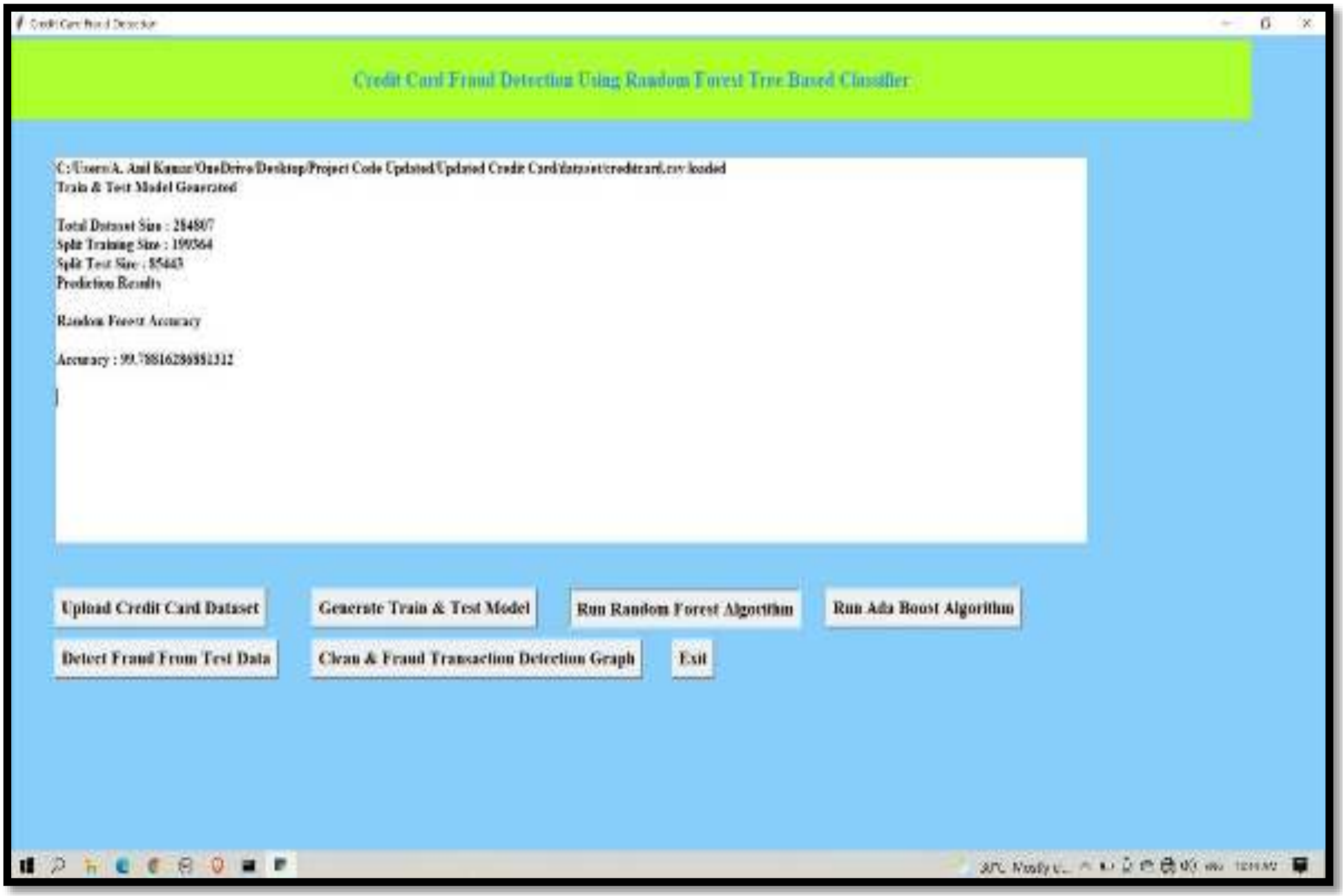

Credit Card Fraud Detection Using Random Forest Tree Based Classifier
Upload Credit Card Dataset
Generate Train & Test Model
Run Random Forest Algorithm
Run Ada Boost Algorithm
Detect Fraud From Test Data
Clean & Fraud Transaction Detection Graph
Exit

```
C:/Users/A. Anil Kumar/OneDrive/Desktop/Project Code Updated/Updated Credit Card/dataset/creditcard.csv loaded
Train & Test Model Generated

Total Dataset Size : 284807
Split Training Size : 199364
Split Test Size : 85443
```

Screenshot 5.3 Train And Test Model

5.4 Run RFA

Click on “Run Random Forest Algorithm” button to apply RFA Classifier.

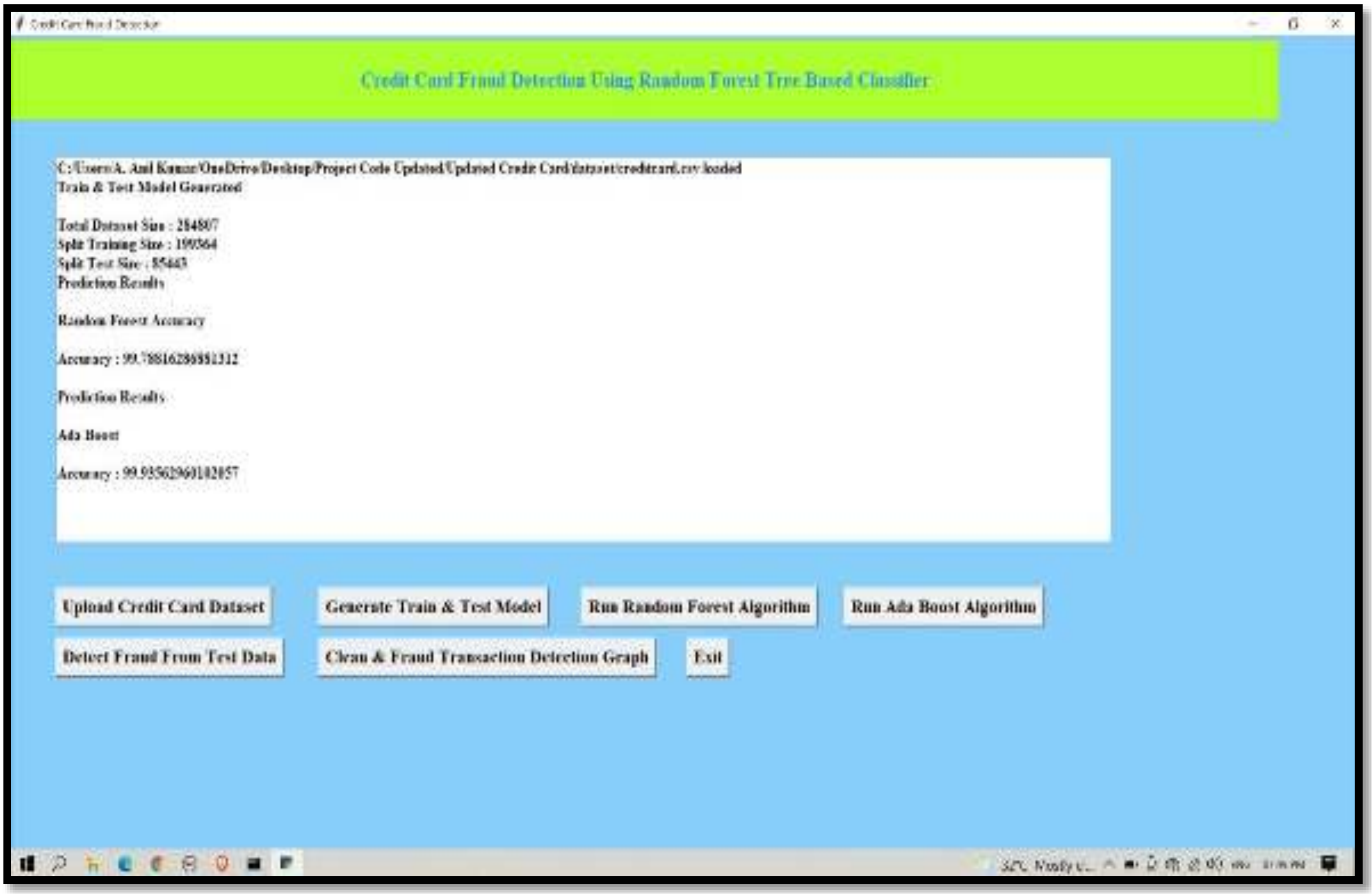

Credit Card Fraud Detection
Credit Card Fraud Detection Using Random Forest Tree Based Classifier
Upload Credit Card Dataset
Generate Train & Test Model
Run Random Forest Algorithm
Run Ada Boost Algorithm
Detect Fraud From Test Data
Clean & Fraud Transaction Detection Graph
Exit

```
C:/Users/A. Anil Kumar/OneDrive/Desktop/Project Code Updated/Updated Credit Card/dataset/creditcard.csv loaded
Train & Test Model Generated

Total Dataset Size : 284807
Split Training Size : 199364
Split Test Size : 85443
Prediction Results

Random Forest Accuracy

Accuracy : 99.78816286851312
```

Screenshot 5.4 Run RFA

5.5 Run Adaboost

Click on “Run Adaboost Algorithm” button to apply Adaboost Classifier.


Credit Card Fraud Detection
Credit Card Fraud Detection Using Random Forest Tree Based Classifier
Upload Credit Card Dataset
Generate Train & Test Model
Run Random Forest Algorithm
Run Ada Boost Algorithm
Detect Fraud From Test Data
Clean & Fraud Transaction Detection Graph
Exit

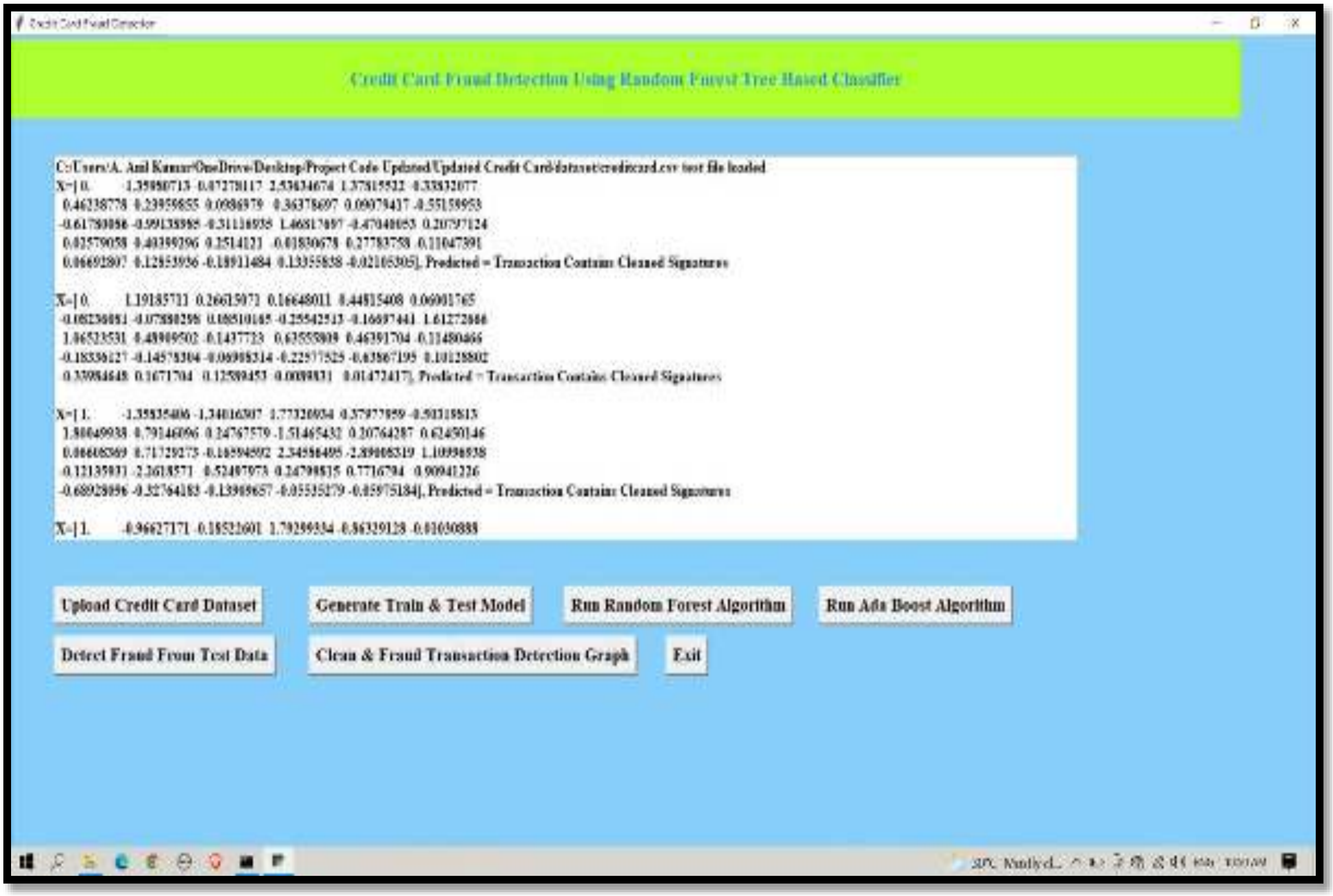
```
C:/Users/A. Anil Kumar/OneDrive/Desktop/Project Code Updated/Updated Credit Card/dataset/creditcard.csv loaded
Train & Test Model Generated

Total Dataset Size : 284807
Split Training Size : 199364
Split Test Size : 85443
Prediction Results

Random Forest Accuracy

Accuracy : 99.78816286881312

Prediction Results

Ada Boost

Accuracy : 99.93562960102057
```

Screenshot 5.5 Run AdaBoost

5.6 Detection of Fraud

Click on ‘Detect Fraud From Test Data’ button to upload test data and to predict whether test data contains normal or fraud transaction.

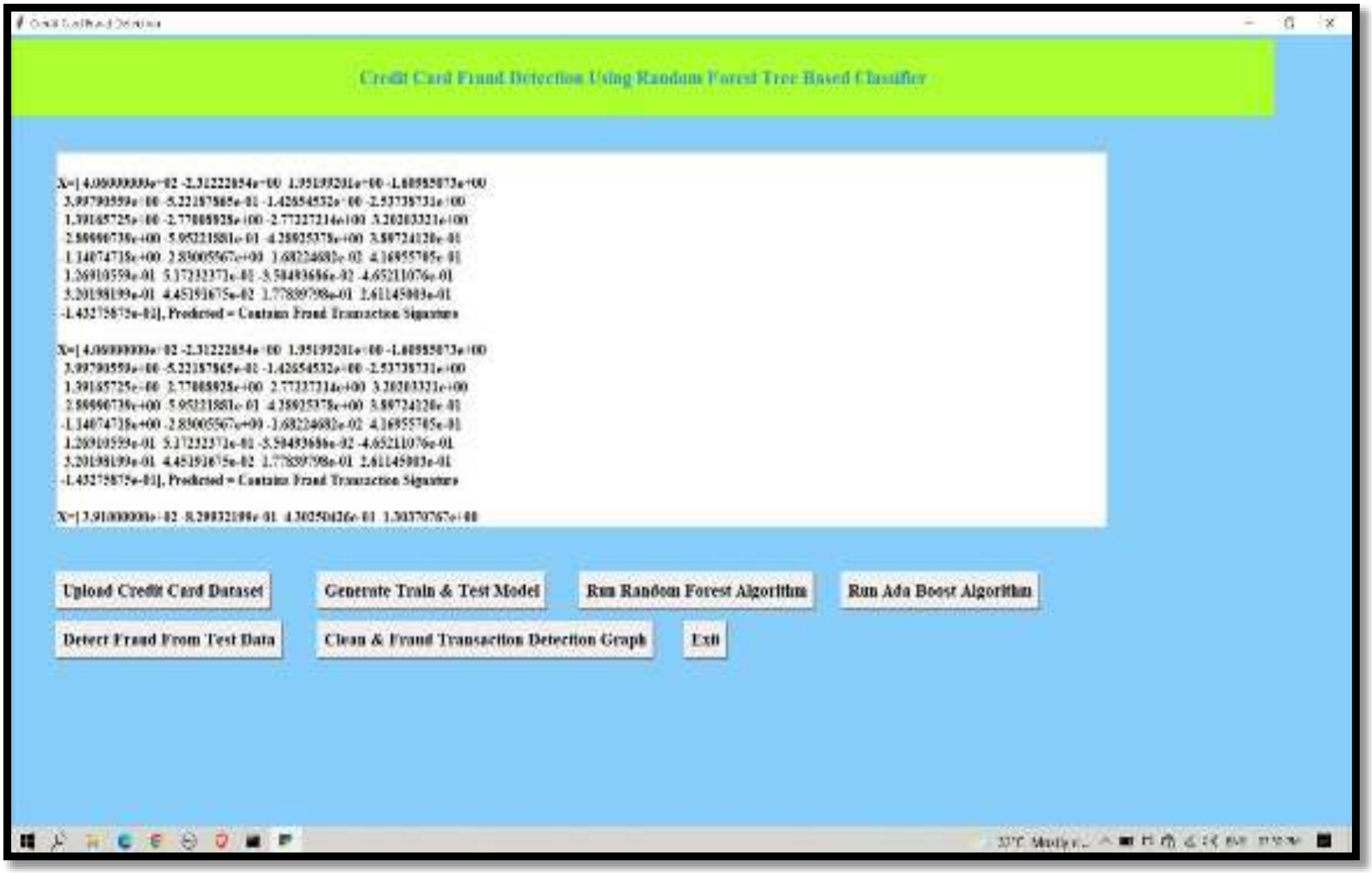
The below figure decribes the clean Signatures.


Credit Card Fraud Detection
Credit Card Fraud Detection Using Random Forest Tree Based Classifier
Upload Credit Card Dataset
Generate Train & Test Model
Run Random Forest Algorithm
Run Ada Boost Algorithm
Detect Fraud From Test Data
Clean & Fraud Transaction Detection Graph
Exit

```
C:/Users/A. Anil Kumar/OneDrive/Desktop/Project Code Updated/Updated Credit Card/dataset/creditcard.csv test file loaded
X=[0.        -1.35980713 -0.07278117  2.53634674  1.37815522 -0.33832077
  0.46238778  0.23959855  0.0986979   0.36378697  0.09079417 -0.55159953
 -0.61780086 -0.99138985 -0.31116935  1.46817697 -0.47040053  0.20797124
  0.02579058  0.40399296  0.2514121  -0.01830678  0.27783758 -0.11047391
  0.06692807  0.12853936 -0.18911484  0.13355838 -0.02105305], Predicted = Transaction Contains Cleaned Signatures

X=[0.         1.19185711  0.26615071  0.16648011  0.44815408  0.06001765
 -0.08236081 -0.07880298  0.08510165 -0.25542513 -0.16697441  1.61272666
  1.06523531  0.48909502 -0.1437723   0.63555809  0.46391704 -0.11480466
 -0.18336127 -0.14578304 -0.06908314 -0.22577525 -0.63867195  0.10128802
  0.33984648  0.1671704   0.12589453 -0.0089831   0.01472417], Predicted = Transaction Contains Cleaned Signatures

X=[1.        -1.35835406 -1.34016307  1.77320934  0.37977959 -0.50319813
  1.80049938  0.79146096  0.24767579 -1.51465432  0.20764287  0.62450146
  0.06608369  0.71729273 -0.16594592  2.34586495 -2.89008319  1.10996938
 -0.12135931 -2.2618571   0.52497973  0.24799815  0.7716794   0.90941226
 -0.68928096 -0.32764183 -0.13909657 -0.05535279 -0.05975184], Predicted = Transaction Contains Cleaned Signatures

X=[1.        -0.96627171 -0.18522601  1.79299334 -0.86329128 -0.01030888
```

Screenshot 5.6 Detection of Clean Signatures

The below figure decribes the Fraud Signatures.

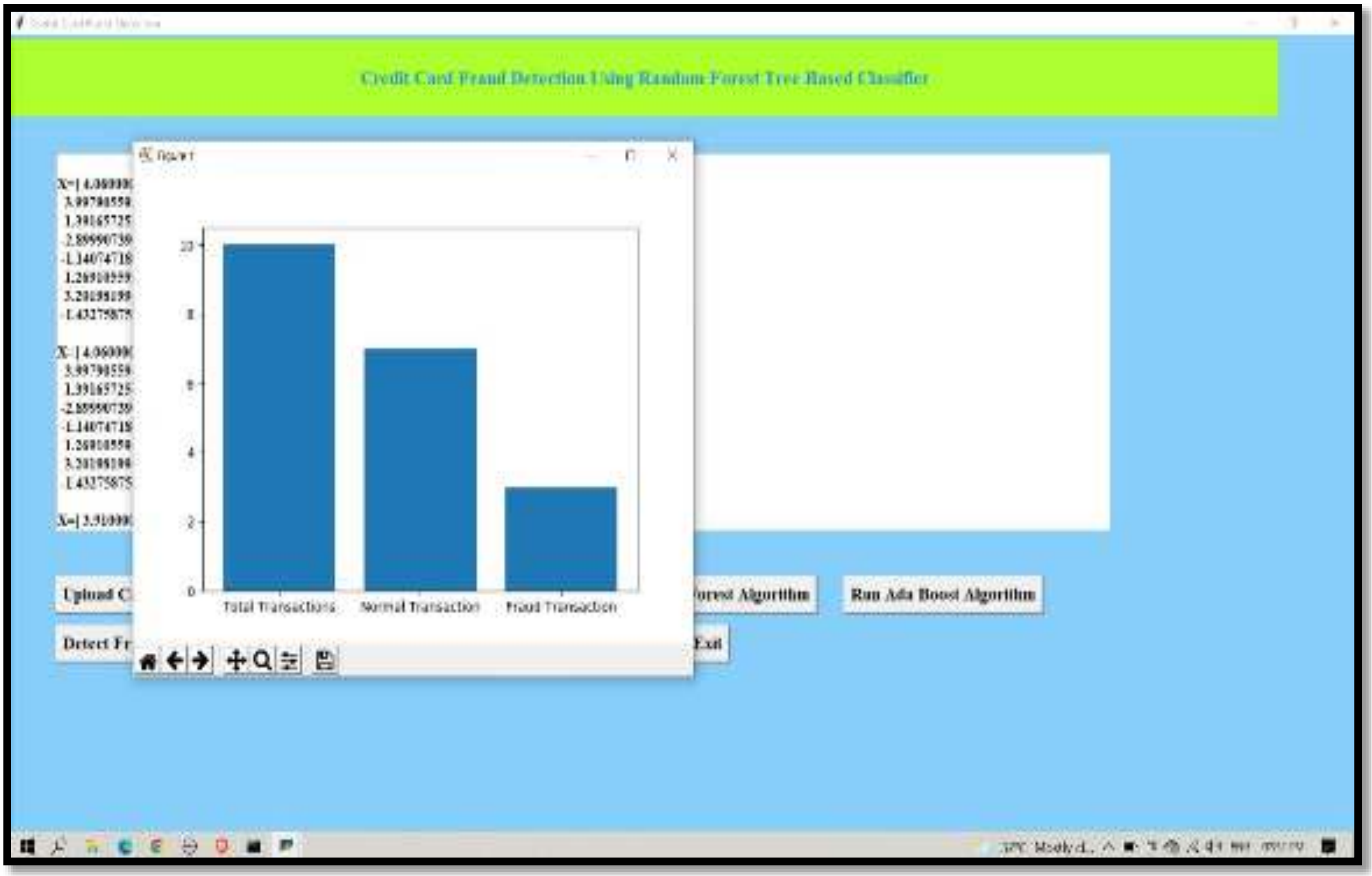

Credit Card Fraud Detection Using Random Forest Tree Based Classifier
Upload Credit Card Dataset
Generate Train & Test Model
Run Random Forest Algorithm
Run Ada Boost Algorithm
Detect Fraud From Test Data
Clean & Fraud Transaction Detection Graph
Exit

```
X=[ 4.06000000e+02 -2.31222654e+00  1.95199201e+00 -1.60985073e+00
  3.99790559e+00 -5.22187865e-01 -1.42654532e+00 -2.53738731e+00
  1.39165725e+00 -2.77008928e+00 -2.77227214e+00  3.20203321e+00
 -2.89990739e+00 -5.95221881e-01 -4.28925378e+00  3.89724120e-01
 -1.14074718e+00 -2.83005567e+00 -1.68224682e-02  4.16955705e-01
  1.26910559e-01  5.17232371e-01 -3.50493686e-02 -4.65211076e-01
  3.20198199e-01  4.45191675e-02  1.77839798e-01  2.61145003e-01
 -1.43275875e-01], Predicted = Contains Fraud Transaction Signature

X=[ 4.06000000e+02 -2.31222654e+00  1.95199201e+00 -1.60985073e+00
  3.99790559e+00 -5.22187865e-01 -1.42654532e+00 -2.53738731e+00
  1.39165725e+00 -2.77008928e+00 -2.77227214e+00  3.20203321e+00
 -2.89990739e+00 -5.95221881e-01 -4.28925378e+00  3.89724120e-01
 -1.14074718e+00 -2.83005567e+00 -1.68224682e-02  4.16955705e-01
  1.26910559e-01  5.17232371e-01 -3.50493686e-02 -4.65211076e-01
  3.20198199e-01  4.45191675e-02  1.77839798e-01  2.61145003e-01
 -1.43275875e-01], Predicted = Contains Fraud Transaction Signature

X=[ 3.91000000e+02 -8.29932199e-01  4.30250426e-01  1.30370767e+00
```

Screenshot 5.7 Detection of Fraud Signatures

5.8 Detection Graph

Click on 'Clean & Fraud Transaction Detection Graph' button to see the total test transaction of clean and fraud signatures in graphical format.The below X-Y graph represents the count of clean and fraud transactions.



```
X=[ 4.060000
  3.99790559
  1.39165725
 -2.89990739
 -1.14074718
  1.26910559
  3.20198199
 -1.43275875

X=[ 4.060000
  3.99790559
  1.39165725
 -2.89990739
 -1.14074718
  1.26910558
  3.20198198
 -1.43275875

X=[ 3.910000
```

Screenshot 5.8 Detection graph

6.TESTING

6.TESTING

6.1 INTRODUCTION TO TESTING

The purpose of testing is to discover errors. Testing is the process of trying to discover every conceivable fault or weakness in a work product. It provides a way to check the functionality of components, sub assemblies, assemblies and/or a finished product It is the process of exercising software with the intent of ensuring that the Software system meets its requirements and user expectations and does not fail in an unacceptable manner. There are various types of test. Each test type addresses a specific testing requirement.

6.2 TYPES OF TESTING

6.2.1 SYSTEM TESTING

Testing has become an integral part of any system or project especially in the field of information technology. The importance of testing is a method of justifying, if one is ready to move further, be it to be check if one is capable to with stand the rigors of a particular situation cannot be underplayed and that is why testing before development is so critical. When the software is developed before it is given to user to user the software must be tested whether it is solving the purpose for which it is developed. This testing involves various types through which one can ensure the software is reliable. The program was tested logically and pattern of execution of the program for a set of data are repeated. Thus the code was exhaustively checked for all possible correct data and the outcomes were also checked.

6.2.2 UNIT TESTING

Unit testing entails creating test cases to ensure that the program's internal logic is working properly and that programme inputs result in valid outputs.Validation should be performed on all decision branches and internal code flow. It is the testing of the application's individual software units. Before integration, it is done after each individual unit is completed. Unit tests guarantee that each individual path of a business process follows the published specifications and has clearly defined inputs and outputs.

6.2.3 INTEGRATION TESTING

Integration tests are designed to test integrated software components to determine if they actually run as one program. Testing is event driven and is more concerned with the basic outcome of screens or fields. Integration tests demonstrate that although the components were individually satisfaction, as shown by successfully unit testing, the combination of components is correct and consistent. Integration testing is specifically aimed at exposing the problems that arise from the combination of components.

6.2.4 ACCEPTANCE TESTING

When that user find no major problems with its accuracy, the system passes through a final acceptance test. This test confirms that the system needs the original goals, objectives and requirements established during analysis without actual execution which eliminates wastage of time and money acceptance tests on the shoulders of users and management, it is finally acceptable and ready for the operation.

6.3 TEST CASES

Test Case Id	Test Case Name	Description	Sample Input	Expected Output	Actual Output	Remarks
01	Upload Dataset	Credit Card dataset is added	Adding dataset to the application by user	Dataset loaded	Result shows is credit card.csv loaded	Pass
02	Train and test model	Click on train & test model to know the whole training size& test size	Uploaded dataset	Train & test model generated	Train & test model generated	Pass
03	Applying RFA & Adaboost Classifier	These classifiers are used to detect the fraud	Combined classifier used on dataset collected from credit card users	High accuracy	High accuracy	Pass
04	Detection of fraud	Detects the fraud	Detects the fraud & clean data from collected test data	Transaction clean & fraud	Result shows that transaction contain clean & fraud signatures	Pass
05	System Testing in various versions of OS	OS compatibility	Execute the program in windows XP\ Windows 7 or above	Performance is better in windows 7	Performance is better in windows 7	Pass

7.CONCLUSION

7.CONCLUSION & FUTURE ENHANCEMENT

7.1 PROJECT CONCLUSION

With more training data, the Random forest algorithm will perform better, but the application's pace will slow down during testing. More pre-processing procedures would also be beneficial. Individual (standard) models and hybrid models using AdaBoost and majority voting combination methods were evaluated using a publicly available credit card data set.

7.2 FUTURE ENHANCEMENT

In Future , privacy preserving techniques can be applied in distributed environment which will resolve the security related issues preventing private data access.

8.BIBLIOGRAPHY

8.BIBLIOGRAPHY

8.1 REFERENCES

1. W. Yu and N. Wang, "Research on Credit Card Fraud Detection Model Based on Distance Sum," 2009 International Joint Conference on Artificial Intelligence, Hainan Island, 2009, pp. 353-356.

2. Vijayshree B. Nipane, Poonam S. Kalinge, Dipali Vidhate, Kunal War, Bhagyashree P. Deshpande, Fraudulent Detection in Credit Card System Using SVM & Decision Tree.

3. Sitaram patel, Sunita Gond, Supervised Machine (SVM) Learning for Credit Card Fraud Detection.

4. Y. Sahin and E. Duman, Detecting Credit Card Fraud by Decision Trees and Support Vector Machines.

5. Snehal Patil, Harshada Somavanshi, Jyoti Gaikwad, Amruta Deshmane, Rinku Badgujar, Credit Card Fraud Detection Using Decision Tree Induction Algorithm.

6. E. Aleskerov, B. Freisleben, and B. Rao, “CARDWATCH: A neural network based database mining system for credit card fraud detection,” in Proc. IEEE/IAFE Computat. Intell. Financial Eng., Mar. 1997, pp. 220– 226.

7. C. Alippi, G. Boracchi, and M. Roveri, “A just-in-time adaptive classification system based on the intersection of confidence intervals rule,” Neural Netw., vol. 24, no. 8, pp. 791–800, 2011.

8. C. Alippi, G. Boracchi, and M. Roveri, “Hierarchical change-detection tests,” IEEE Trans. Neural Netw. Learn. Syst., vol. 28, no. 2, pp. 246– 258, Feb. 2016.

9. C. Alippi, G. Boracchi, and M. Roveri, “Just-in-time classifiers for recurrent concepts,” IEEE Trans. Neural Netw. Learn. Syst., vol. 24, no. 4, pp. 620–634, Apr. 2013.

10. B. Baesens, V. Van Vlasselaer, and W. Verbeke, Fraud Analytics Using Descriptive, Predictive, and Social Network Techniques: A Guide to Data Science for Fraud Detection. Hoboken, NJ, USA: Wiley, 2015.

11. A. O. Adewumi and A. A. Akinyelu, “A survey of machinelearning and nature-inspired based credit card fraud detection techniques,” International Journal of System Assurance Engineering and Management, vol. 8, pp. 937–953, 2017.

12.A. Srivastava, A. Kundu, S. Sural, A. Majumdar, “Credit card fraud detection using hidden Markov model,” IEEE Transactions on Dependable and Secure Computing, vol. 5, no. 1, pp. 37–48, 2008

13. S. Bhattacharyya, S. Jha, K. Tharakunnel, and J. C., “Data mining for credit card fraud: A comparative study,” Decision Support Systems, vol. 50, no. 3, pp. 602–613, 2011. [7] N. S. Halvaiee and M. K. Akbari, “A novel model for credit card fraud detection using Artificial Immune Systems,” Applied Soft Computing, vol. 24, pp. 40–49, 2014

14. E. Kirkos, C. Spathis, and Y. Manolopoulos, “Data mining techniques for the detection of fraudulent financial statements,” Expert Systems with Applications, vol. 32, no. 4, pp. 995–1003, 2007

15. C. Phua, K. Smith-Miles, V. Lee, and R. Gayler, “Resilient identity crime detection,” IEEE Transactions on Knowledge and Data Engineering, vol. 24, no. 3, pp. 533–546, 2012.

8.2 WEBSITES

1. **https://ieeexplore.ieee.org/document/8292883**
2. **https://www.ijert.org/research**
3. **https://researchbank.swinburne.edu.au**

8.3 GITHUB LINK

- **https://github.com/Bindhu1204/Creditcard-Frauddetection**