A

Mini Project

On

#### A SUPERVISED LEARNING ALGORITHM FOR CREDIT CARD FRAUD DETECTION

(Submitted in partial fulfillment of the requirements for the award of Degree)

BACHELOR OF TECHNOLOGY

in

COMPUTER SCIENCE AND ENGINEERING (AI&ML)

by

K. SRAVAN KUMAR (217R5A6609)

P. VAISHNAVI (207R1A66A7)

CH. B. ABHISHEKAM (207R1A6671)

Under the Guidance of

**Mr. D. BABU RAO**

(Associate Professor)

#### DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING

#### (AI&ML)

#### 

#### CMR TECHNICAL CAMPUS

#### UGC AUTONOMOUS

(Accredited by NAAC, 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.

#### 2020-2024

**DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING (AI&ML)**



#### CERTIFICATE

This is to certify that the project entitled **“A SUPERVISED LEARNING ALGORITHM FOR CREDIT CARD FRAUD DETECTION”** being submitted by **K. SRAVAN KUMAR (217R5A6609), P. VAISHNAVI (207R1A66A7) & CH. B. ABHISHEKAM (207R1A6671)** in partial fulfillment of the requirements for the award of the degree of B.Tech in Computer Science and Engineering (AI&ML) to the Jawaharlal Nehru Technological University Hyderabad, is a record of bonafide work carried out by them under our guidance and supervision during the year 2022-23.

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

**Mr. D. Babu Rao Dr. S Rao Chintalapudi**

Associate Professor HOD CSE(AI&ML)

INTERNAL GUIDE

**EXTERNAL EXAMINER**

**Submitted for viva voice Examination held on**

#### 

#### ACKNOWLEDGEMENT

Apart 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 **Mr. D. Babu Rao,** Associate 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 the Project Review Committee (PRC) **Dr. G. Vinoda Reddy, Dr. K. Mahesh, Md. Hafeena & D. Babu Rao** for their cordial support, valuable information and guidance, which helped us in completing this task through various stages.

We are also thankful to **Dr. S Rao Chintalapudi,** Head, Department of Computer Science and Engineering (AI&ML) for providing encouragement and support for completing this project successfully.

We are obliged to **Dr. A. Raji Reddy,** Director for being cooperative throughout the course of this project. We also express our sincere gratitude to Sri. **Ch. Gopal Reddy,** Chairman for providing excellent infrastructure and a nice atmosphere throughout the course of this project.

The guidance and support received from all the members of **CMR Technical Campus** who contributed to the completion of the project. 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 completed. We sincerely acknowledge and thank all those who gave support directly and indirectly in the completion of this project.

**K. SRAVAN KUMAR (217R5A6609)**

**P. VAISHNAVI (207R1A66A7)**

**CH. B. ABHISHEKAM (207R1A6671)**

#### ABSTRACT

Credit card fraud detection, in this project we mainly focus on credit card fraud detection in real world. Here the credit card fraud detection is based on fraudulent transactions. Generally, credit card fraud activities can happen in both online and offline. But in today’s world online fraud transaction activities are increasing day by day. So, to find the online fraud transactions, we followed and have various methods used in existing system. In the existing system, we used the Dempster-Shafer theory to control the sensitive data and information, it created an initial belief, which was used to classify a transaction as normal or suspicious transaction in this fraud detection activities.

In proposed system we use random forest algorithm (RFA) for finding the fraudulent transactions and the accuracy of those transactions is high. This algorithm is based on supervised learning algorithm where it uses decision trees for classification of the dataset. After classification of dataset a confusion matrix is obtained. The performance of random forest algorithm (RFA) is evaluated based on the confusion matrix i.

#### LIST OF FIGURES/TABLES

|  |  |  |  |
| --- | --- | --- | --- |
| **FIGURE NO** | **FIGURE NAME** | | **PAGE NO** |
| Figure 3.1 | Project Architecture for Supervised Learning Algorithm for Credit Card Fraud Detection | | 6 |
|  |  | |  |
| Figure 3.2 | Use Case Diagram for Supervised Learning Algorithm for Credit Card Fraud Detection | | 7 |
| Figure 3.3 | Class Diagram for Supervised Learning Algorithm for Credit Card Fraud Detection | | 8 |
|  |  |
| Figure 3.4 | Sequence diagram for Supervised Learning Algorithm for Credit Card Fraud Detection | | 9 |
| Figure 3.5 | Activity diagram for Supervised Learning Algorithm for Credit Card Fraud Detection | | 10 |

#### LIST OF SCREENSHOTS

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **SCREENSHOT NO.** | | **SCREENSHOT NAME** | | **PAGE NO**. | |
| Screenshot 6.1 | | Uploaded Credit Card Dataset | | 16 | |
| Screenshot 6.2 | | Loading the Dataset and get below the screen | | 16 | |
| Screenshot 6.3 | | Loaded Dataset and Seen Normal and Fraud Transactions | | 17 | |
| Screenshot 6.4 | | Normalized and Split the Dataset | | 17 | |
| Screenshot 6.5 | | Extracting Encoder & Decoder for Prediction | | 18 | |
|  | |  | |  | |

Screenshot 6.6 MAE Histogram on Fraud 18

Transaction

Screenshot 6.7 Test Record of MAE Values 19

#### 

#### TABLE OF CONTENTS

**ABSTRACT** i

LIST OF FIGURES ii

LIST OF SCREENSHOTS iii

1.INTRODUCTION 1

* 1. PROJECT SCOPE 1
  2. PROJECT PURPOSE 1
  3. OVERVIEW OF PROJECT 1

2.SYSTEM ANALYSIS

* 1. INTRODUCTION 2
  2. EXISTING SYSTEM 2

2.2.1 DISADVANTAGES OF EXISTING SYSTEM 2

* 1. PROPOSED SYTEM 3

2.3.1 ADVANTAGES OF PROPOSED SYSTEM 3

2.4 FEASIBILITY STUDY 4

2.4.1 ECONOMIC FEASIBILITY 4

2.4.2 TECHNICAL FEASIBILITY 4

* + 1. SOCIAL FEASIBILITY 4

2.5 HARDWARE & SOFTWARE REQUIREMENTS 5

2.5.1 HARDWARE REQUIREMENTS 5

2.5.2 SOFTWARE REQUIREMENTS 5

3.ARCHITECTURE

* 1. PROJECT ARCHITECTURE 6
  2. DESCRIPTION 6
  3. USE CASE DIAGRAM 7
  4. CLASS DIAGRAM 8
  5. SEQUENCE DIAGRAM 9
  6. ACTIVITY DIAGRAM 10

#### 4.ALGORITHMS

4.1 LOGISTIC REGRESSION 11

4.2 DECISION TREES 11

* 1. RANDOM FOREST 11

5.IMPLEMENTATION

* 1. SAMPLE CODE FOR IMPLEMENTATION 12

6.SCREENSHOTS 17

7.TESTING

7.1 INTRODUCTION TO TESTING 21

7.2 TYPES OF TESTING 21

7.2.1 UNIT TESTING 21

7.2.2 INTEGRATING TESTING 21

7.2.3 FUNCTIONAL TESTING 22

#### 8.CONCLUSION & FUTURE SCOPE

* 1. PROJECT CONCLUSION 24
  2. FUTURE SCOPE 24

9.BIBLIOGRAPHY

9.1 REFERENCES 25

9.2 GITHUB 25

#### 

#### 

#### 1.INTRODUCTION

#### 

#### 1.INTRODUCTION

**1.1 PROJECT SCOPE**

This project is titled “A Supervised Learning Algorithm for Credit Card Fraud Detection”. Credit card frauds are increasing day by day. Modelling prior credit card transactions with data from ones that turned out to be fraudulent is part of the Credit Card Fraud Detection Problem. In Machine Learning the machine is trained at first to predict the output, so to predict the various bank transactions in various learning algorithms are used.

#### 1.2 PROJECT PURPOSE

This project has been developed to identify transactions fraud using credit card is one of the growing issue in the world of finance. A huge financial loss has significantly affected individuals using credit cards and further more vendors and banks. One of the most successful techniques to identify such fraud is Machine Learning. This paper proposes a fraud detection algorithm using Random Forest which can help in solving this real-world problem.

#### 1.3 OVERVIEW OF PROJECT

* Credit card fraud increasing day by day.
* There are two types of credit card transactions:

1. Online transactions
2. Offline transactions

The main motto of this project is to detect the credit card frauds that are happening and to protect the sensitive data. The standard model to recognize credit card fraud is “Supervised Learning Algorithm”.

CMRTC 1

#### 

#### 2.SYSTEM ANALYSIS

#### 2.SYSTEM ANALYSIS

#### 2.1 INTRODUCTION

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.

#### 2.2 EXISTING SYSTEM

Initially, in the existing system of Credit card fraud detection used theory is “The Dempster-Shafer theory”. In the Dempster-Shafer combined various evidential information and created an initial belief, which was used to classify a transaction as normal or suspicious. If a transaction was suspicious, the belief was further evaluated using transaction history from Bayesian Learning. A modified Fisher Discriminant function was used for credit card fraud detection. The modification made the traditional functions to become more sensitive to important instances. Although, linear discriminant functions are less complex classifiers and can work on high-dimensional problems like credit card fraud detection, they didn’t receive considerable attention so far.

#### 2.2.1 DISADVANTAGES OF EXISTING SYSTEM

Following are the disadvantages of existing system:

* There is no majority voting technique for credit card fraud detection.
* There is no Machine Learning techniques are used in the existing system.

CMRTC 2

#### 2.3 PROPOSED SYSTEM

This project presents a proposed system which consists a twelve machine learning algorithms are used for detecting credit card fraud. The algorithms range from standard neutral networks to deep learning models. They are evaluated using benchmark and real world credit card datasets, the data set is used in this paper is extracted from actual credit card transactions information over three months. In real time, the proposed system for fraud identification in credit card transactions is designed using “Random Forest Algorithm (RFA)”. This algorithm is used in combination of Decision Tree to solve the problems in credit card transactions.

#### 2.3.1 ADVANTAGES OF THE PROPOSED SYSTEM

Following are the advantages of proposed system:

* Robustness and reliability of the models can be evaluated, noise can be added to real-world dataset.
* The Ada Boost and majority voting methods are applied for forming hybrid models.
* The key contribution of this system is the evolution of a variety of machine learning models with a real-world credit card dataset for fraud detection.

#### 2.4 FEASIBILITY STUDY

The feasibility of the project is analyzed in this phase and a business proposal is put forth with a very general plan for the project and some cost estimates. During system analysis the feasibility study of the proposed system is to be carried out. This is to ensure that the proposed system is not a burden to the company. For feasibility analysis, some understanding of the major requirements for the system is essential.

CMRTC 3

Three key considerations involved in the feasibility analysis:

* ECONOMICAL FEASIBILIY
* TECHNICAL FEASIBILITY
* SOCIAL FEASIBILITY

#### 2.4.1 ECONOMIC FEASIBILITY

This study is carried out to check the economic impact that the system will have on the organization. The amount of fund that company can pour into research and development system as well within the budget and this was achieved because most of the technologies used are freely available. Only the customized products had to be purchased.

#### 2.4.2 TECHNICAL FEASIBILITY

This study is carried out to check the technical feasibility, that is, the technical requirements of the system. Any system developed must not have a high demand on the available technical resources. The developed system must have a modest requirement, as only minimal or null changes are required for implementing this system.

#### 2.4.3 SOCIAL FEASIBILITY

The aspect of study is to check the level of acceptance of the system by the user. This includes the process of training the user to use the system efficiently. The user must not feel threatened by the system, instead must accept it as a necessity. The level of acceptance by the users solely depends on the methods that are employed to educate the user about the system and to make him familiar with it. His level of confidence must be raised so that he is also able to make some constructive criticism, which is welcomed, as he is the final user of the system.

CMRTC 4

#### 2.5 HARDWARE & SOFTWARE REQUIREMENTS

#### 2.5.1 HARDWARE REQUIREMENTS:

For developing the application the following are the Hardware Requirements:

* + - * Processor: Intel Dual Core I5 and above
      * Hard disk: 128 GB
      * RAM: 4 GB
      * Input devices: Keyboard, mouse.

#### 2.5.2 SOFTWARE REQUIREMENTS:

For developing the application the following are the Software Requirements:

* Operating system: Windows XP/7/8
* Language: Python
* Tool: Anaconda
* Interface: Jupiter notebook

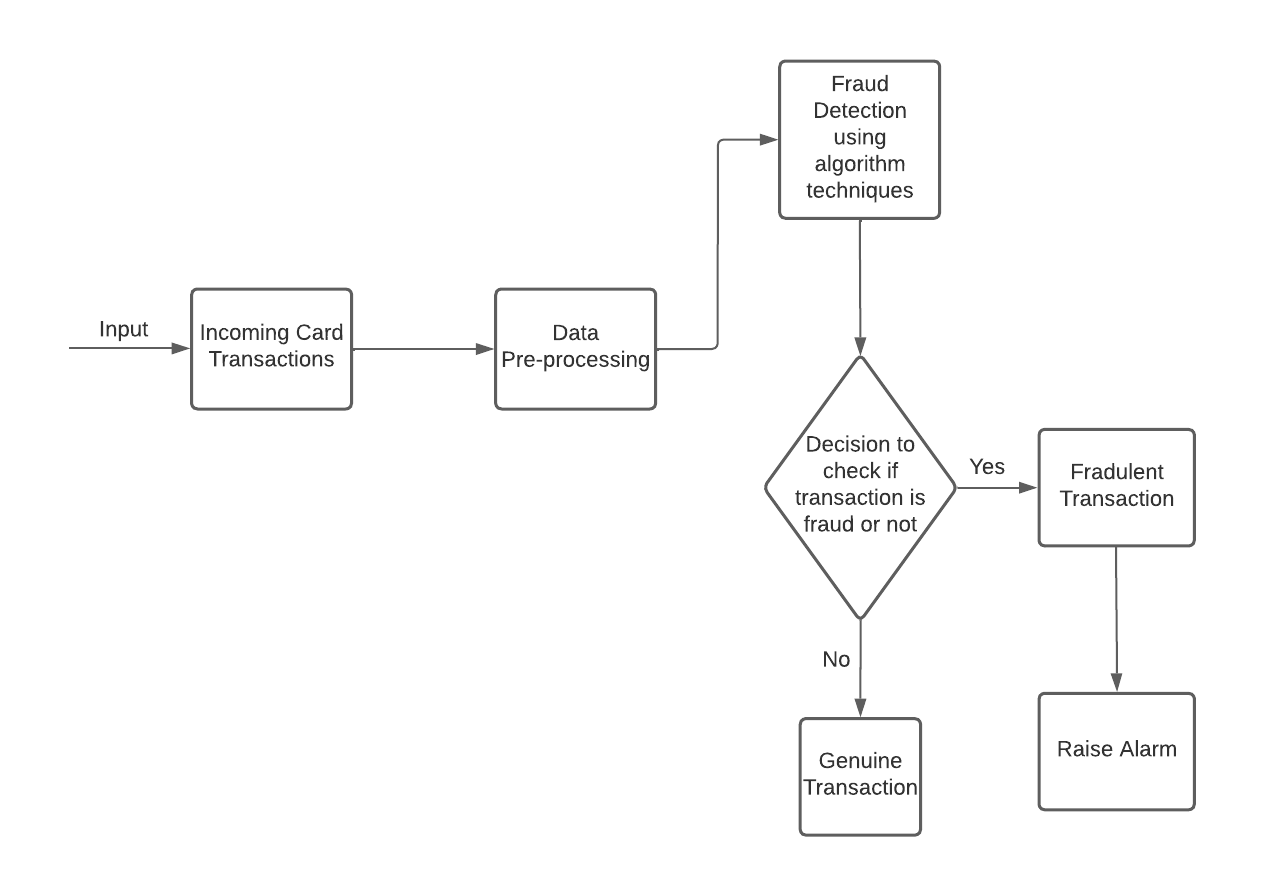
CMRTC 5

#### 3.ARCHITECTURE

#### 3.ARCHITECTURE

#### 3.1 PROJECT ARCHITECTURE

This project architecture shows the procedure followed for classification:

 Figure 3.1: Project Architecture of Supervised Learning Algorithm

For Credit Card Fraud Detection

#### 

#### 3.2 DESCRIPTION

Architecture defines the components, modules interfaces and data for a system to satisfy specified requirements. One should see as the applications of the systems theory to product development. System Architecture bridges the gap between the problem domain and the existing system in a manageable way. It is a phase where the SRS document is converted into a format that can be implemented and decides how the system will operate. In this phase, the complex activity of system development is divided into several smaller sub activities, which coordinate with each other to achieve main objective of the system development.

CMRTC 6

#### 3.3 USE CASE DIAGRAM

Use case diagram in the Unified Modeling Language (UML) is a type of behavioral diagram defined by and created from a Use-case analysis. Its 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.

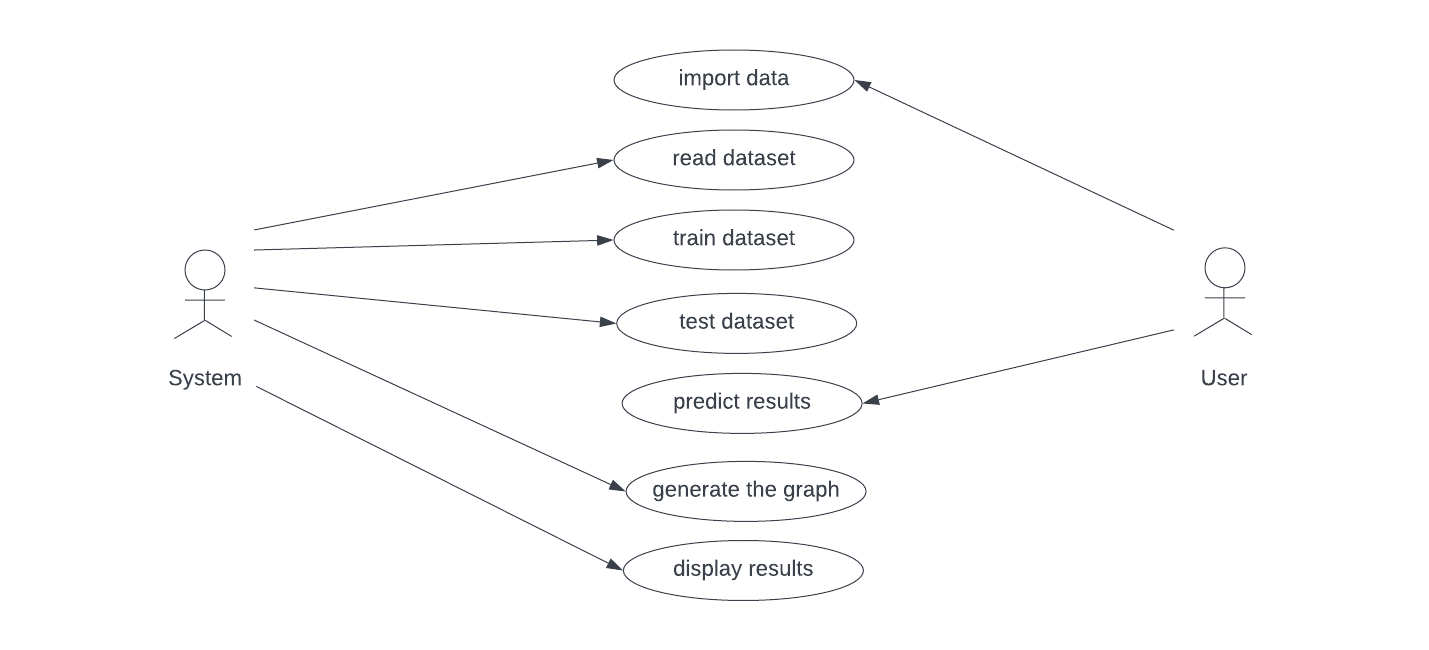


Figure 3.2: Use Case Diagram

CMRTC 7

#### 3.4 CLASS DIAGRAM

In software engineering, a class diagram in the Unified Modeling Language (UML) is a type of static structure diagram that describes the structure of a system by showing the system's classes, their attributes, operations (or methods), and the relationships among the classes. It explains which class contains information.

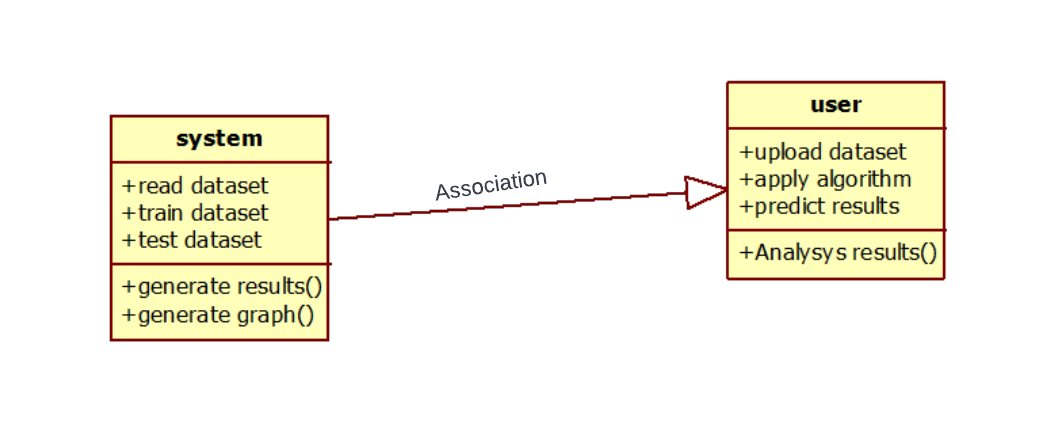


Figure 3.3: Class Diagram

CMRTC 8

#### 3.5 SEQUENCE DIAGRAM

A sequence diagram in Unified Modeling Language (UML) is a kind of interaction diagram that shows how processes operate with one another and in what order. It is a construct of a Message Sequence Chart. Sequence diagrams are sometimes called event diagrams, event scenarios, and timing diagrams.

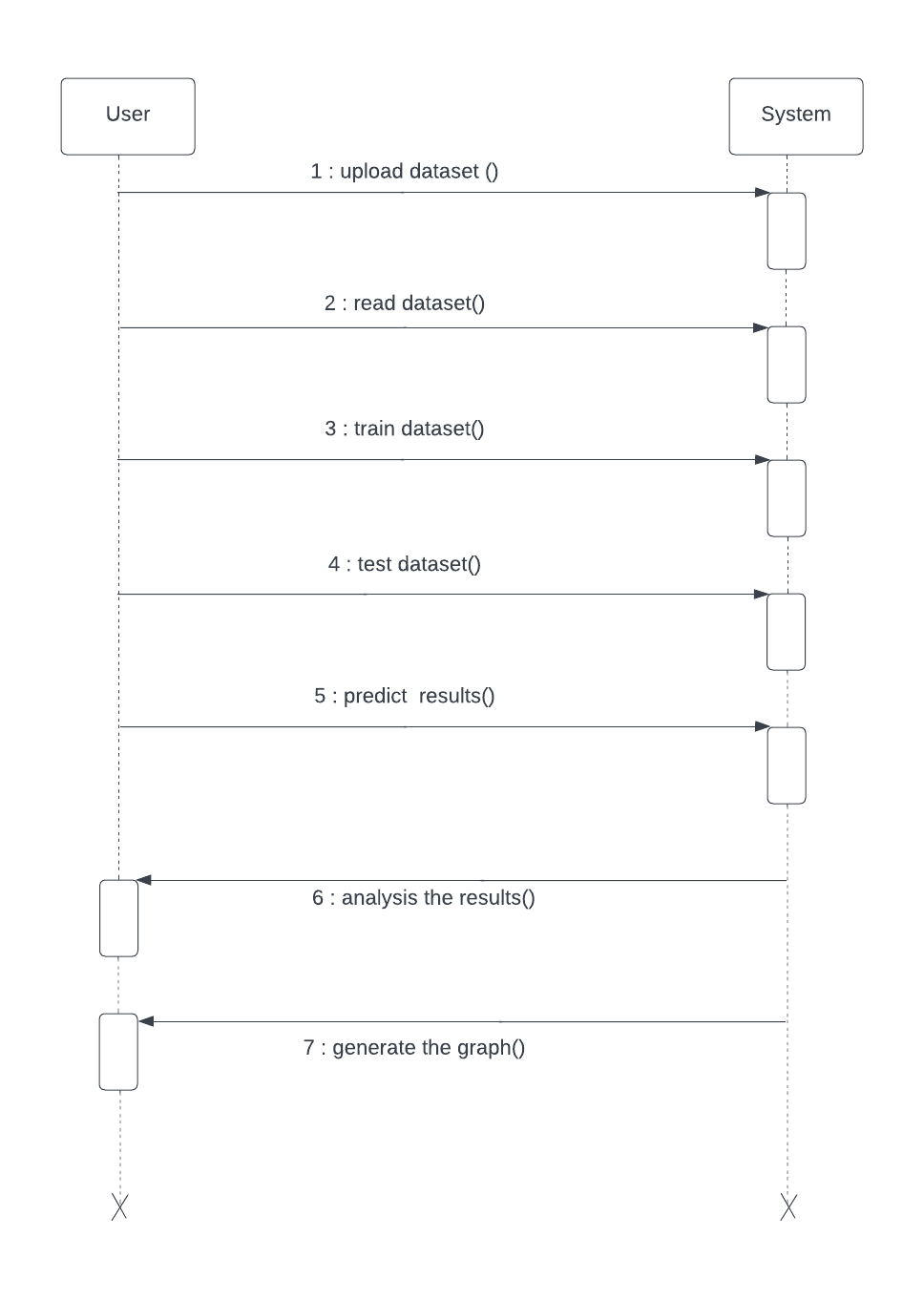


Figure 3.4: Sequence Diagram

CMRTC 9

#### 3.6 ACTIVITY DIAGRAM

Activity diagrams are graphical representations of workflows of stepwise activities and actions with support for choice, iteration and concurrency. In the Unified Modeling Language, activity diagrams can be used to describe the business and operational step-by-step workflows of components in a system. An activity diagram shows the overall flow of control.

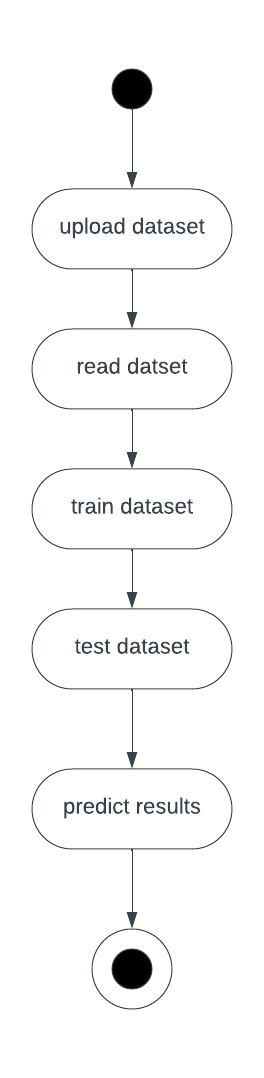


Figure 3.5: Activity Diagram

CMRTC 10

#### 

#### 4. ALGORITHMS

#### 4. ALGORITHMS

#### 4.1 LOGISTIC REGRESSION

Logistic Regression works with sigmoid function because the sigmoid can be used to classify the output that is dependent feature and it uses the probability for classification of the dependent feature. The sigmoid activation function in middle layer neuron then results in the value of 0.25 this will affect the accuracy of the module in the deep machine learning.

#### 4.2 DECISION TREES

Decision Tree can be used for the classification and regression problems working for both is same but different formulae. Classification problem uses the entropy and information gain for building of the decision tree model. Entropy tell about how the data is random and information gain tells about much information we can get from this feature. Regression problem uses gini and gini index for the building of the decision tree model.

#### 4.3 RANDOM FOREST

The Random Forest Algorithm (RFA) randomly selects the features that is independent variables and also randomly selects the rows by row sampling and the number of decision trees can be determined by using hyper parameter optimization. For classification problem statement the output is the maximum occurrence outputs from each decision tree models inside the random forest. This is one the widely used machine learning algorithm in real world scenarios and in deployed models. And in most of the Kaggle computation challenges, this algorithm is used to solve the problem statement.

CMRTC 11

#### 

#### 5. IMPLEMENTATION

#### 5.1 CODE FOR IMPLEMENTATION OF SUPERVISED LEARNING ALGORITHM FOR CREDIT CARD FRAUD DETECTION:

#### 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 pandas as pd

#### from sklearn.preprocessing import StandardScaler

#### from sklearn.metrics import accuracy\_score

#### import pickle

#### from sklearn.model\_selection import train\_test\_split

#### from keras.models import Model

#### from keras.layers import Input, Dense

#### from keras.callbacks import ModelCheckpoint, TensorBoard

#### from keras import regularizers

#### main = tkinter.Tk()

#### main.title("Credit Card Fraud Detection using AutoEncoder & Decoder") #designing main screen

#### main.geometry("1300x1200")

#### global filename

#### global cls

#### global X\_train, X\_test, y\_train, y\_test

#### global auto\_encoder

#### global dataset

#### global error\_df

#### def upload(): #function to upload tweeter profile

#### global filename

#### global dataset

#### #upload dataset file

#### filename = filedialog.askopenfilename(initialdir="dataset")

#### text.delete('1.0', END)

#### text.insert(END,filename+" loaded\n");

#### #reading dataset as CSV file

#### dataset = pd.read\_csv(filename)

#### dataset = dataset.drop(['Time'], axis=1)

#### lbl = dataset['Class']

#### #finding count of normal and fraud transaction

#### unique, count = np.unique(lbl,return\_counts=True)

CMRTC 12

#### text.insert(END,str(dataset.head()))

#### text.insert(END,"Total Normal Transaction: "+str(count[0])+"\n")

#### text.insert(END,"Total Fraud Transaction: "+str(count[1])+"\n")

#### #plotting normal and fraud transaction dataset

#### label = dataset.groupby('Class').size()

#### label.plot(kind="bar")

#### plt.show()

#### def normalizeDataset():

#### global X\_train, X\_test, y\_train, y\_test

#### global dataset

#### text.delete('1.0', END)

#### #using scaler class we are scaling or normalizing dataset

#### dataset['Amount'] = StandardScaler().fit\_transform(dataset['Amount'].values.reshape(-1, 1))

#### #splitting dataset into train and test

#### X\_train, X\_test = train\_test\_split(dataset, test\_size=0.2, random\_state=42)

#### X\_train = X\_train[X\_train.Class == 0]

#### X\_train = X\_train.drop(['Class'], axis=1)

#### y\_test = X\_test['Class']

#### X\_test = X\_test.drop(['Class'], axis=1)

#### X\_train = X\_train.values

#### X\_test = X\_test.values

#### text.insert(END,"Normalized Dataset\n\n")

#### text.insert(END,str(dataset.head())+"\n\n")

#### text.insert(END,"Total Training records after split: "+str(X\_train.shape[0])+"\n")

#### text.insert(END,"Total Testing records after split: "+str(X\_test.shape[0])+"\n")

#### 

#### def trainModel():

#### global X\_train, X\_test, y\_train, y\_test

#### global auto\_encoder

#### text.delete('1.0', END)

#### #defining input layer for encdoer

#### inputLayer = Input(shape=(X\_train.shape[1], ))

#### #first encoder layer with neuron or features filtration size as 14

#### encoder = Dense(14, activation="tanh", activity\_regularizer=regularizers.l1(10e-5))(inputLayer)

#### #another encoder with 7 filetrs

#### encoder = Dense(7, activation="relu")(encoder)

#### #decoder with 7 filters to give output

#### decoder = Dense(7, activation='tanh')(encoder)

#### #decoder to perform prediction with givenshape

#### decoder = Dense(X\_train.shape[1], activation='relu')(decoder)

#### #combining or extracting input layer and decoder layer to form auto encoding prediction layer

#### 

CMRTC 13

#### auto\_encoder = Model(inputs=inputLayer, outputs=decoder)

#### #compiling and trsining odel

#### auto\_encoder.compile(optimizer='adam', loss='mean\_squared\_error', metrics=['accuracy'])

#### auto\_encoder.load\_weights('model/autoencoder.h5')

#### text.insert(END,"Auto Encoder & Decoder model training completed\n\n")

#### f = open('model/history.pckl', 'rb')

#### data = pickle.load(f)

#### f.close()

#### text.insert(END,"Auto Encoder & Decoder model Accuracy: "+str(data['accuracy'][99]))

#### print(data)

#### #plot graph using auto encoder training LOSS

#### plt.plot(data['loss'])

#### plt.plot(data['val\_loss'])

#### plt.title('AutoEncoder-Decoder training & Validation Loss Graph')

#### plt.ylabel('Loss')

#### plt.xlabel('Epoch')

#### plt.legend(['Train Loss', 'Validation Loss'], loc='upper left')

#### plt.show()

#### 

#### def prediction():

#### global error\_df

#### global X\_train, X\_test, y\_train, y\_test

#### global auto\_encoder

#### text.delete('1.0', END)

#### #perform prediction on test data

#### predictions = auto\_encoder.predict(X\_test)

#### #calculate MAE error between origina Y value and predicted value

#### mae = np.mean(np.power(X\_test - predictions, 2), axis=1)

#### error\_df = pd.DataFrame({'mae': mae, 'true\_class': y\_test})

#### #calculate accuracy

#### threshold = 2.9

#### y\_pred = [1 if e > threshold else 0 for e in error\_df.mae.values]

#### acc = accuracy\_score(error\_df.true\_class, y\_pred)

#### text.insert(END,"Auto Encoder Decoder Accuracy on Test Data: "+str(acc))

#### #plot histogram on non fraud transaction where class label 0

#### fig = plt.figure()

#### ax = fig.add\_subplot(111)

#### normal\_error\_df = error\_df[(error\_df['true\_class']== 0) & (error\_df['mae'] < 10)]

#### \_ = ax.hist(normal\_error\_df.mae.values, bins=10)

#### plt.title("MAE histogram on Non-Fraudulent Transaction")

#### plt.show()

#### 

CMRTC 14

#### def graph():

#### #plot histogram on fraud transaction where class label 1

#### fig = plt.figure()

#### ax = fig.add\_subplot(111)

#### fraud\_error\_df = error\_df[error\_df['true\_class'] == 1]

#### \_ = ax.hist(fraud\_error\_df.mae.values, bins=10)

#### plt.title("MAE histogram on Fraudulent Transaction")

#### plt.show()

#### font = ('times', 16, 'bold')

#### title = Label(main, text='Credit Card Fraud Detection using AutoEncoder & Decoder')

#### 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', 13, 'bold')

#### uploadButton = Button(main, text="Upload Credit Card Dataset", command=upload)

#### uploadButton.place(x=50,y=550)

#### uploadButton.config(font=font1)

#### splitButton = Button(main, text="Normalize & Dataset Split", command=normalizeDataset)

#### splitButton.place(x=450,y=550)

#### splitButton.config(font=font1)

#### encoderButton = Button(main, text="Train AutoEncoder & Decoder Model", command=trainModel)

#### encoderButton.place(x=50,y=600)

#### encoderButton.config(font=font1)

#### predictButton = Button(main, text="Extract Encoder & Decoder for Prediction", command=prediction)

#### predictButton.place(x=450,y=600)

#### predictButton.config(font=font1)

#### graphButton = Button(main, text="MAE Histogram on Fraud Transaction", command=graph)

#### graphButton.place(x=50,y=650)

#### graphButton.config(font=font1)

#### main.config(bg='LightSkyBlue')

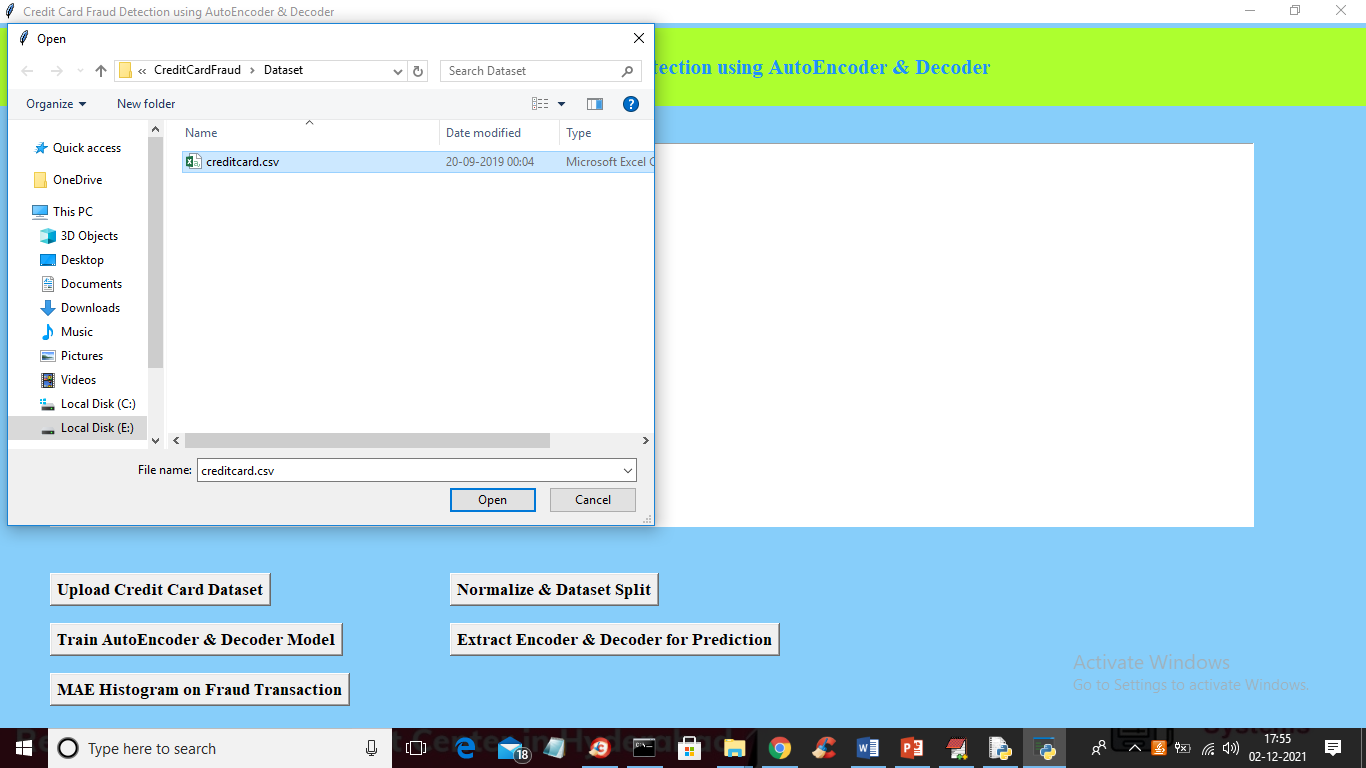
#### main.mainloop()

CMRTC 15

#### 6. SCREENSHOTS

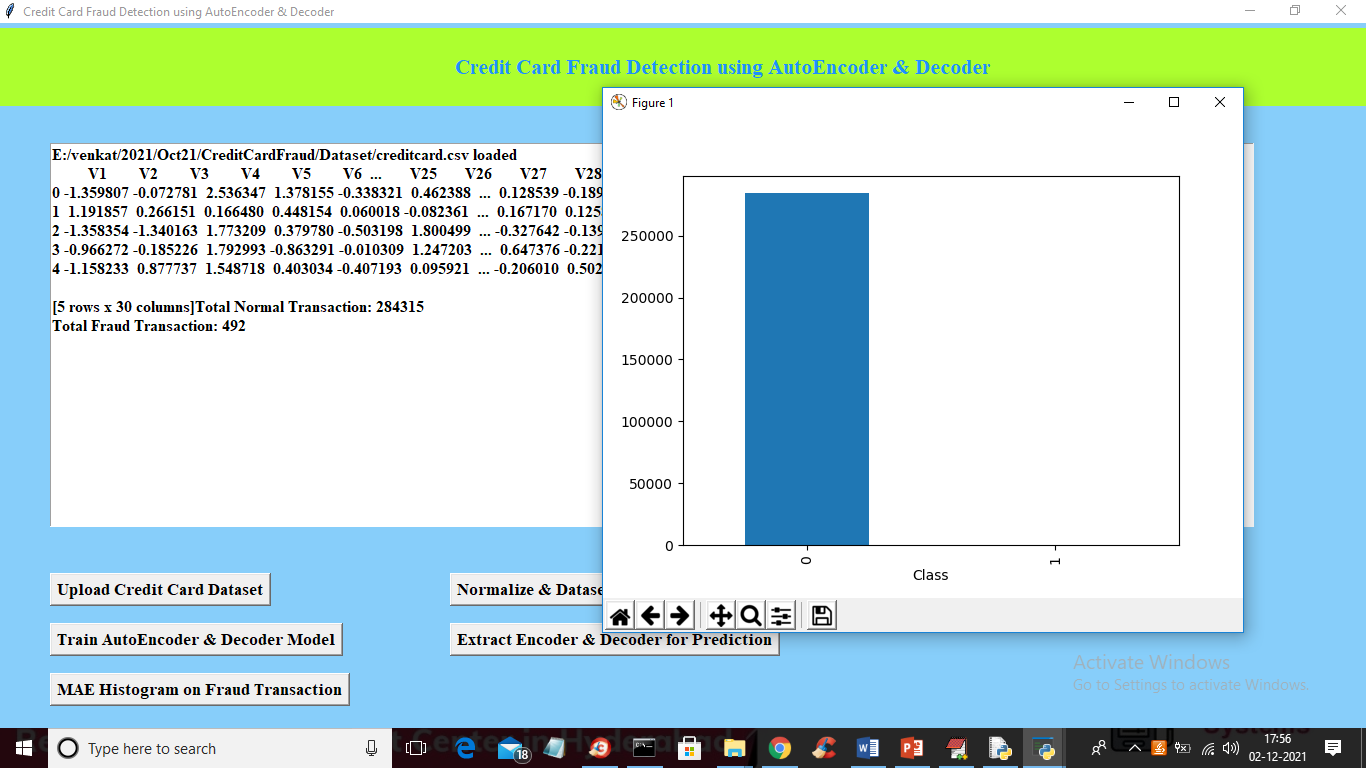
#### 

Screenshot 6.1: Upload Credit Card Dataset

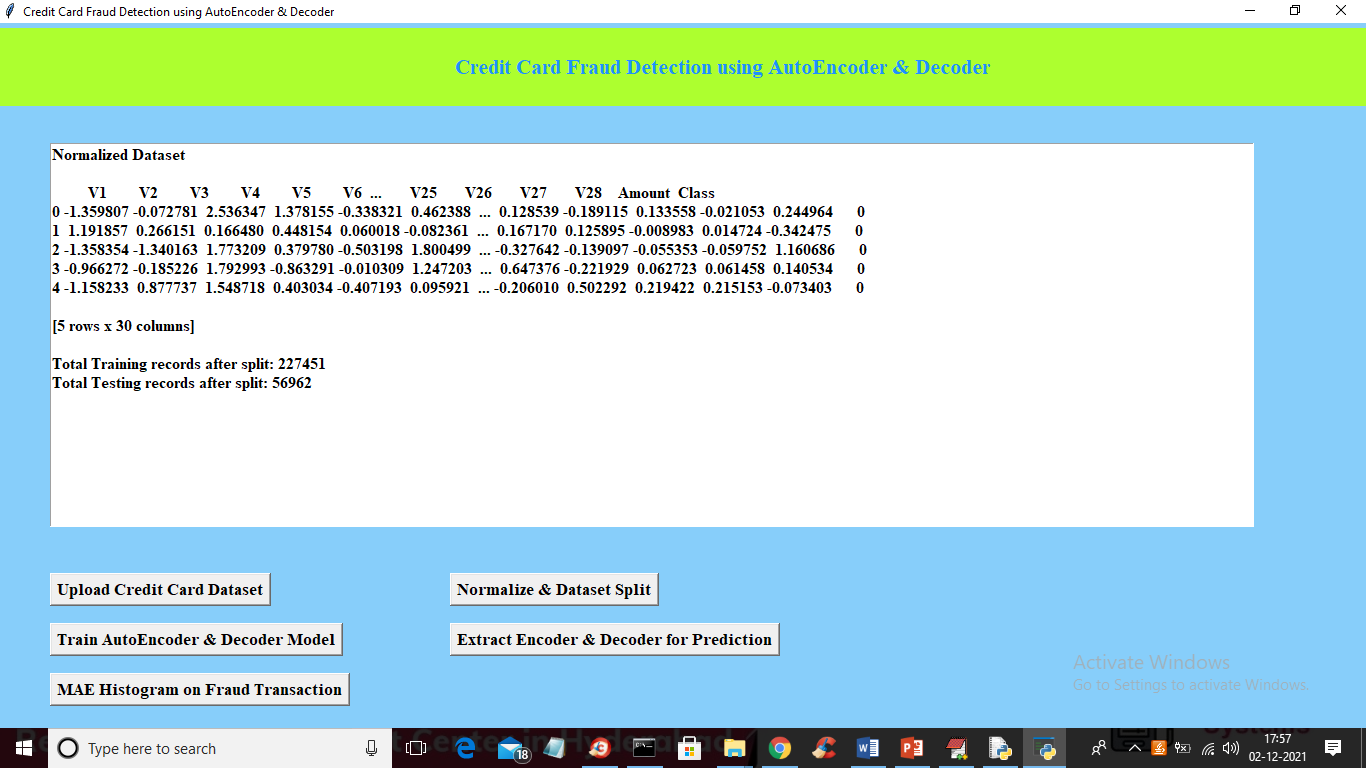


Screenshot 6.2: Loading the Dataset and get below the screen

CMRTC 16

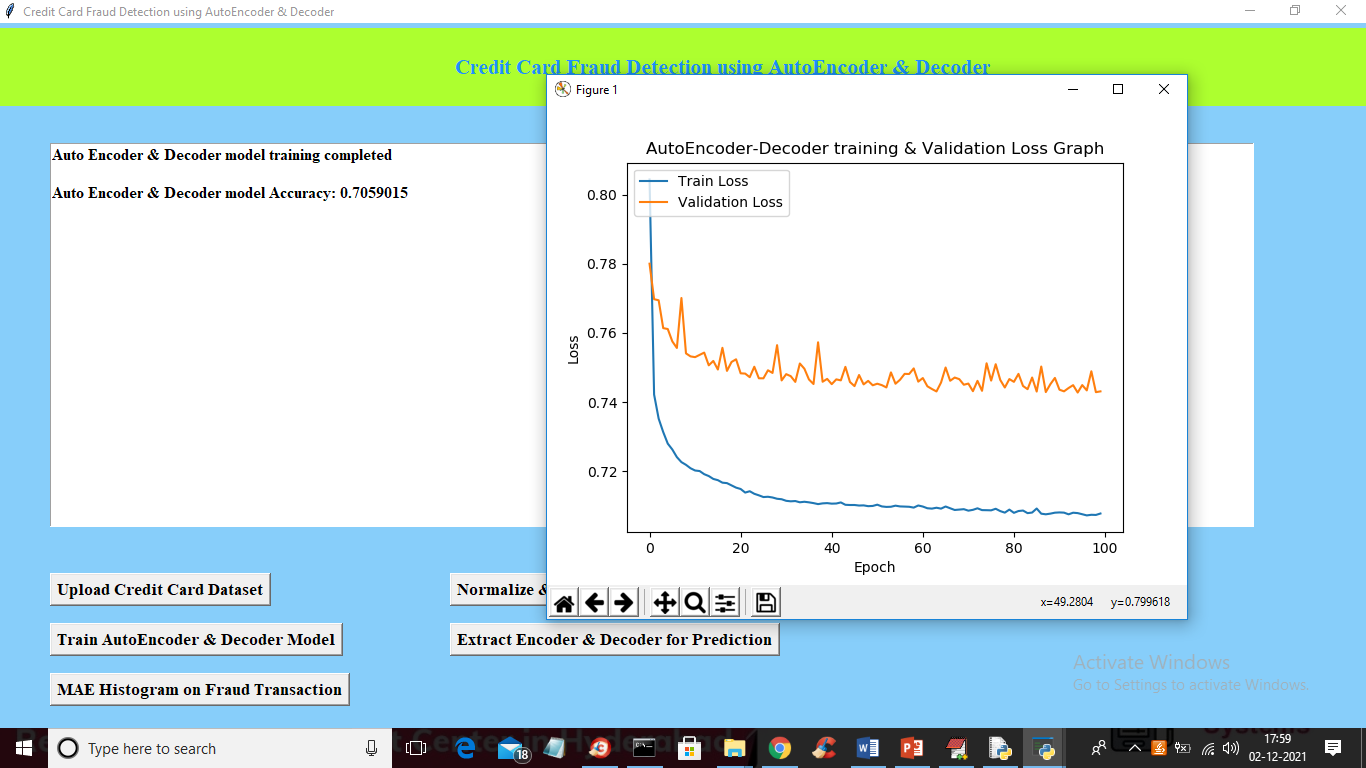


Screenshot 6.3: Loaded Dataset and Seen Normal and Fraud Transactions

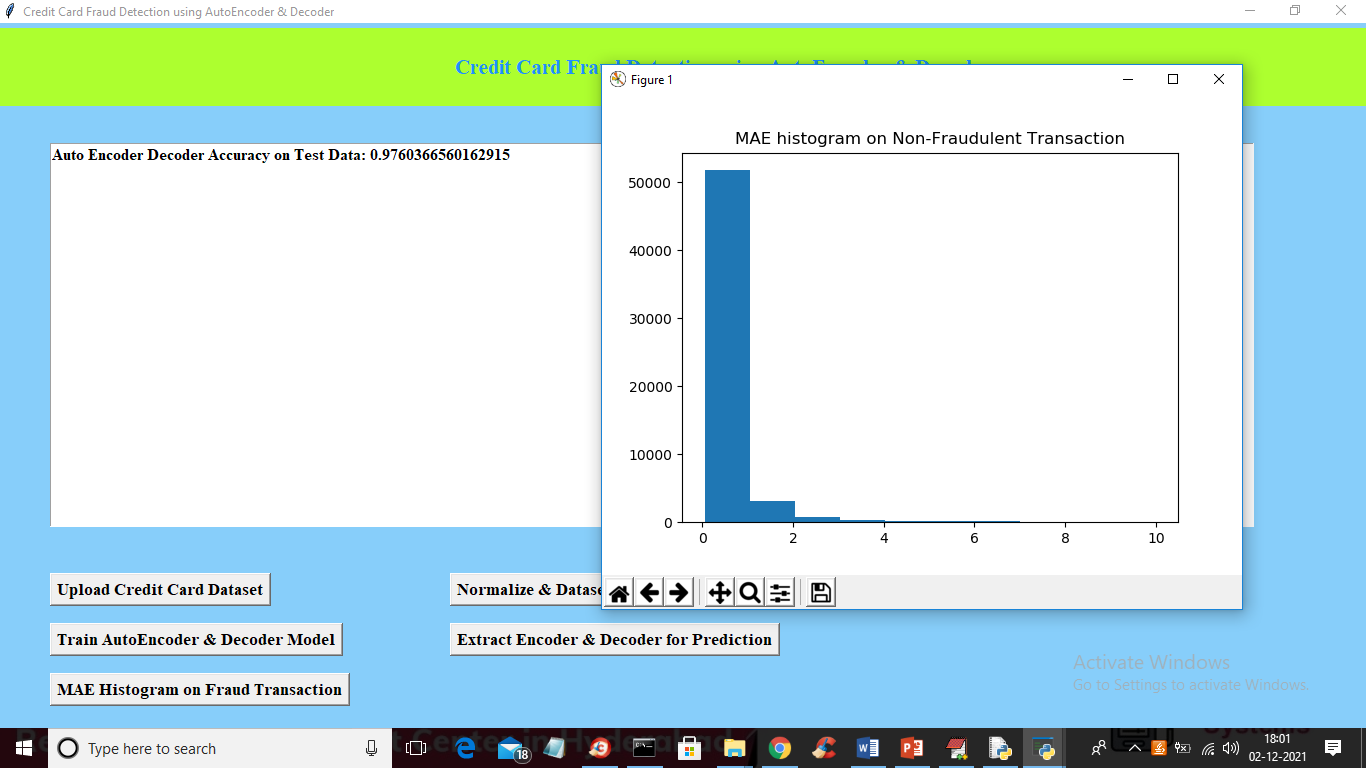


Screenshot 6.4: Normalized and Split the Dataset

CMRTC 17

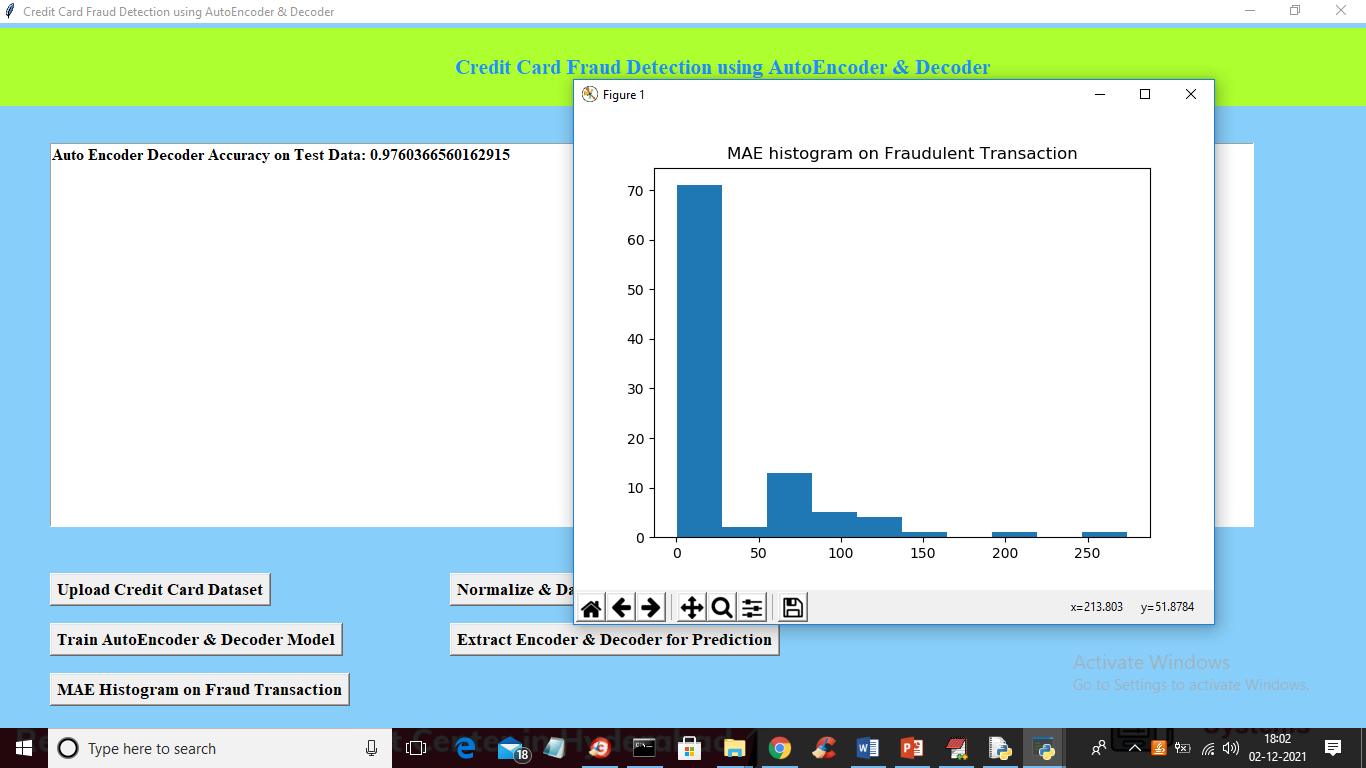


Screenshot 6.5: Extracting Encoder & Decoder for Prediction



Screenshot 6.6: MAE Histogram on Fraud Transaction

CMRTC 18



Screenshot 6.7: Test Record of MAE Values

CMRTC 19

#### 7. TESTING

#### 7.TESTING

#### 7.1 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, subassemblies, 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 tests. Each test type addresses a specific testing requirement.

#### 7.2 TYPES OF TESTING

#### 7.2.1 UNIT TESTING

Unit testing involves the design of test cases that validate that the internal program logic is functioning properly, and that program inputs produce valid outputs. All decision branches and internal code flow should be validated. It is the testing of individual software units of the application. It is done after the completion of an individual unit before integration. This is a structural testing that relies on knowledge of its construction and is invasive. Unit tests perform basic tests at component level and test a specific business process, application and/or system configuration. Unit tests ensure that each unique path of a business process performs accurately to the documented specifications and contains clearly defined inputs and expected results.

CMRTC 20

#### 7.2.2 INTEGRATION TESTING

Integration tests are designed to test integrated software components to determine if they actually run as one program. Integration tests demonstrate that although the components were individually satisfactory, 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.

#### 7.2.3 FUNCTIONAL TESTING

Functional tests provide systematic demonstrations that functions tested are available as specified by the business and technical requirements, system documentation, and user manuals.

Functional testing is centered on the following items:

Valid Input : identified classes of valid input must be accepted.

Invalid Input : identified classes of invalid input must be rejected.

Functions : identified functions must be exercised.

Output : identified classes of application outputs must be exercised.

Systems/Procedures : interfacing systems or procedures must be invoked.

#### 7.2.4 SYSTEM TEST

System testing ensures that the entire integrated software system meets requirements. It tests a configuration to ensure known and predictable results. An example of system testing is the configuration oriented system integration test. System testing is based on process descriptions and flows, emphasizing pre-driven process links and integration points.

CMRTC 21

#### 7.2.5 WHITE BOX TESTING

White Box Testing is a testing in which in which the software tester has knowledge of the inner workings, structure and language of the software, or at least its purpose. It is purpose. It is used to test areas that cannot be reached from a black box level.

#### 7.2.6 BLACK BOX TESTING

Black Box Testing is testing the software without any knowledge of the inner workings, structure or language of the module being tested. Black box tests, as most other kinds of tests, must be written from a definitive source document, such as specification or requirements document, such as specification or requirements document. It is a testing in which the software under test is treated, as a black box, you cannot “see” into it. The test provides inputs and responds to outputs without considering how the software works.

#### 7.2.7 UNIT TESTING

Unit testing is usually conducted as part of a combined code and unit test phase of the software lifecycle, although it is not uncommon for coding and unit testing to be conducted as two distinct phases.

**7.2.8 Test strategy and approach**

Field testing will be performed manually and functional tests will be written in detail.

#### 

CMRTC 22

#### 7.2.9 Test objectives

* + All field entries must work properly.
  + Pages must be activated from the identified link.
  + The entry screen, messages and responses must not be delayed.

#### 7.3 Features to be tested

* Verify that the entries are of the correct format
* No duplicate entries should be allowed
* All links should take the user to the correct page.

#### Integration Testing

Software integration testing is the incremental integration testing of two or more integrated software components on a single platform to produce failures caused by interface defects. The task of the integration test is to check that components or software applications, e.g. components in a software system or – one step up – software applications at the company level – interact without error.

#### 

#### Acceptance Testing

User Acceptance Testing is a critical phase of any project and requires significant participation by the end user. It also ensures that the system meets the functional requirements.

#### 7.4 Test Results

All the test cases mentioned above passed successfully. No defects encountered.

CMRTC 23

#### 8. CONCLUSION

#### 8.CONCLUSION & FUTURE SCOPE

#### 8.1 PROJECT CONCLUSION

The Random Forest Algorithm (RFA) will perform better with a larger number of training data, but speed during algorithm will suffer. The SVM algorithm still suffers from imbalanced dataset problem and requires more preprocessing to give better if more preprocessing have been done on the data. This model can further be improved with the addition of more 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.

#### 8.2 FUTURE SCOPE

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. This model can further be improved with 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. From this project the advancement to the supervised learning algorithm for credit card fraud detection can be a great use to see the fraudulent transactions in the different set of transactions.

CMRTC 24

#### 9. BIBLIOGRAPHY

#### 9. BIBLIOGRAPHY

* 1. **REFERENCES**
* S. Bhattacharya, S. Jha, K. Tharakunnel and J. C. Westland, “Data mining for credit card fraud: A comparative study”, Decis. Support Syst., vol. 50, no. 3, pp. 602-613, 2011
* K. Chaudhary, J. Yadav and B. Mallick, “A review of Fraud Detection Techniques: Credit Card”, Int. j. Comput. Appl., vol. 45, no.1, pp, 975-8887,2012
* F. N. Ogwueleka, Data Mining Application in credit card fraud detection system, vol. 6, no. 3, pp. 311-322,2011.
* M. Zareapoor and P. Shamsolmoali, “Application of credit card fraud detection: Based on bagging ensemble classifier”, Procedia Comput. Sci., vol. 48, no. C, pp. 679-686, 2015
* G. H. John and P. Langley, Estimating Continuous Distributions in Bayesian Classifiers, Feb. 2013.
* C. C. Lin, A. A. Chiu, S. Y. Huang and D. C. Yen, “Detecting the financial statement fraud”
* H. Nordberg, K. Bhatia, K. Wang and Z. Wand, “BioPig: a Hadoop-based analytic toolkit for large-scale sequence data”, Bioinformatics, vo. 29, no. 23, pp. 3014-3019, Dec. 2013.
* X.-Y . Liu, J. Wu and Z.-H. Zhou, “Exploratory Undersampling for ClassImbalance Learning, vol. 39, no. 2, 2009.”

#### 9.2 GITHUB LINK

CMRTC 25