A

Mini Project

On

COMPARATIVE STUDY OF MACHINE LEARNING ALGORITHMS FOR FRAUD DETECTION IN BLOCKCHAIN

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

BACHELOR OF TECHNOLOGY

In

COMPUTER SCIENCE AND ENGINEERING

By

K. SURYABHASKAR (207R1A0522)

K. YASHMINE (207R1A0523)

Under the Guidance of

Dr. J. NARSIMHARAO

(Associate Professor)

DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING

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 UGCAct.1956, Kandlakoya (V), Medchal Road, Hyderabad-501401.

2020-2024

DEPARTMENT OF COMPUTER SCIENCE ENGINEERING

CERTIFICATE

This is to certify that the project entitled **"COMPARATIVE STUDY OF MACHINE LEARNING ALGORITHMS FOR FRAUD DETECTION IN BLOCKCHAIN"** being submitted by **K. SURYABHASKAR (207R1A0522) & K. YASHMINE (207R1A0523)** partial fulfilment of the requirement for the award of the degree of B.Tech in Computer Science and Engineering to the Jawaharlal Nehru Technological University Hyderabad, record of bonafide work carried out by them under our guidance and supervision during year 2023-2024.

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

Dr. J. Narsimharao
(Associate Professor)
INTERNAL GUIDE

Dr. A. Raji Reddy
DIRECTOR

Dr. K. Srujan Raju
HOD

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 competition of this project.

We take this opportunity to Express my profound gratitude and deep regard To my guide **Dr. J. Narsimharao**, Associate professor for his exemplary guidance, monitoring and constant encouragement throughout the project work. The blessing, Help and guidance given by him cell carriers a long way in the journey of life on which we are about to embark.

We also take the opportunity to express a deep sense of gratitude to the project review committee (PRC) **G. Vinesh Shanker, Dr. J. Narasimharao, Ms. Shilpa, & Dr. K. Maheswari** 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, Department of Computer Science and Engineering for providing encouragement and support for completing of 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. SURYABHASKAR	**(207R1A0522)**
K. YASHMINE	**(207R1A0523)**

ABSTRACT

Fraudulent transactions have a huge impact on the economy and trust of a blockchain network. Consensus algorithms like proof of work or proof of stake can verify the validity of the transaction but not the nature of the users involved in the transactions or those who verify the transactions. This makes a blockchain network still vulnerable to fraudulent activities. One of the ways to eliminate fraud is by using machine learning techniques. Machine learning can be of supervised or unsupervised nature. In this paper, we use various supervised machine learning techniques to check for fraudulent and legitimate transactions. We also provide an extensive comparative study of various supervised machine learning techniques like decision trees, Naive Bayes, logistic regression, multilayer perceptron, and so on for the above task.

LIST OF FIGURES/TABLES

LIST OF SCREENSHOTS

TABLE OF CONTENTS

1.INTRODUCTION

1.INTRODUCTION

1.1 PROJECT SCOPE

This project aims to conduct a comparative study of various machine learning algorithms to evaluate their effectiveness in detecting fraud within blockchain transactions. The project will involve the development of a system that can analyse blockchain data and apply machine learning techniques for fraud detection.

1.2 PROJECT PURPOSE

The primary purpose of this project is to address a critical and growing concern within the blockchain ecosystem—fraudulent activities and security breaches. As blockchain technology gains widespread adoption in various industries, it becomes increasingly important to ensure the integrity and security of blockchain transactions. Fraudulent activities such as unauthorized access, double spending, and other forms of financial misconduct can have significant financial and reputational consequences for organizations and individuals using blockchain technology. The purpose of this project is to explore the capabilities of machine learning algorithms in effectively identifying and mitigating fraud in blockchain transactions.

1.3 PROJECT FEATURES

The main features of this project are that this model it collects and preprocesses the data. It implements and evaluates multiple machine learning algorithms for fraud detection. It trains, validates and fine-tune models using historical data. It allows users to choose and configure algorithms. It enables users to upload blockchain data for analysis. It displays fraud detection results and statistics. It generates comprehensive reports on comparative study results. It ensures data and algorithm security.

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.

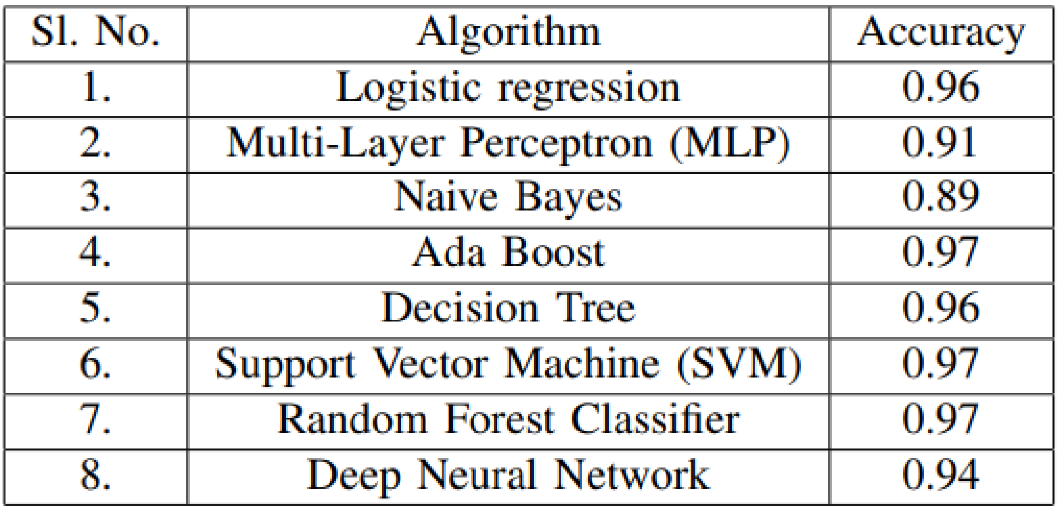
2.1 PROBLEM DEFINITION

The problem addressed by the "Comparative Study of Machine Learning Algorithms for Fraud Detection in Blockchain" project revolves around the increasing vulnerability of blockchain systems to fraudulent activities and security breaches. As blockchain technology gains widespread adoption, its inherent transparency and immutability also attract malicious actors seeking to exploit vulnerabilities for unauthorized access, double spending, and other fraudulent activities. Traditional fraud detection methods often prove insufficient in this unique context, necessitating a novel approach. This project aims to address this pressing issue by conducting a comprehensive study of machine learning algorithms, offering an effective means to detect and mitigate fraud within blockchain transactions. Through this research, the project seeks to empower organizations and individuals with valuable insights and data-driven solutions to enhance the security, trustworthiness, and adoption of blockchain technology in various industries.

2.2 EXISTING SYSTEM

We applied eight different supervised learning algorithms to the dataset.

The dataset contains information about trust on different nodes or ratings given to them. This information is useful in detecting if a certain node's transaction can be fraudulent or not. The following table summarizes the accuracy obtained in each case.



Sl. No.	Algorithm	Accuracy
1.	Logistic regression	0.96
2.	Multi-Layer Perceptron (MLP)	0.91
3.	Naive Bayes	0.89
4.	Ada Boost	0.97
5.	Decision Tree	0.96
6.	Support Vector Machine (SVM)	0.97
7.	Random Forest Classifier	0.97
8.	Deep Neural Network	0.94

We observed that using Ada Boost, SVM, and Random Forest classifier gave the best results among the seven different algorithms. Also, since these algorithms already provide an accuracy of 97%, we would like to build a fraud detector that will use the scores and decisions from the three algorithms together to decide if a transaction is fraudulent or not finally.

2.2.1 DISADVANTAGES OF EXISTING SYSTEM

Following are the disadvantages of existing system:

- Lack of Regulation
- Difficulty in Tracking Transaction
- Limited Data Availability
- False Positives
- Limited Ability to Detect Complex Fraud

2.3 PROPOSED SYSTEM

The workflow for detecting fraudulent activity is summarised in Figure 1. Essentially, after the Blockchain network has approved a transaction after all basic checks, our proposed system kicks in and does additional checks to detect if the transaction can be fraudulent. This approach makes sure that there is no extra overhead of even checking the transactions that the Blockchain network itself can easily invalidate.

The work done can be divided mainly into three phases:

1. Preprocessing phase
2. Building and training various models
3. Performance evaluation of all the models.

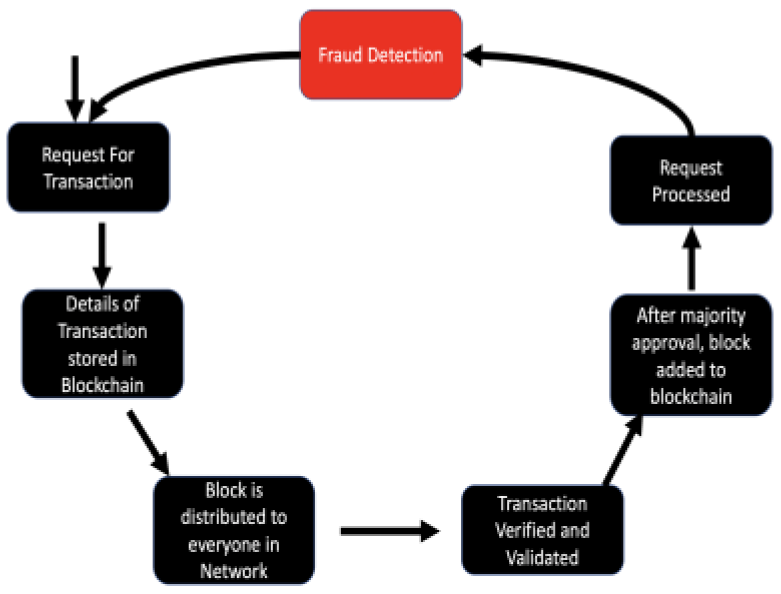

Fraud Detection
Request For Transaction
Details of Transaction stored in Blockchain
Block is distributed to everyone in Network
Transaction Verified and Validated
After majority approval, block added to blockchain
Request Processed

Fig. 1. Workflow of applying check for Fraud Detection

We preprocess using node-embedding in the network using the node2vec algorithm. Then, we read and convert the shorter version of concatenated rating dataset into a data frame. Then, we create a function for the perception store features. This function extracts the features of a node using the" source" and" target" columns of the dataset. These features are then stored in a CSV file. We then run the node2vec algorithm in python and create a dictionary of nodes and corresponding embeddings. We also create a network edge list file and then reduce embeddings dimensionality for 2D projections. This dimensionality reduction can be obtained using algorithms like t-SNE.

We then normalize the features extracted from the node2vec algorithm and create a file that contains the normalized values. We assign a score of 1 if the transaction is rated badly (fraud) and 0 otherwise. We then calculate the mean and standard deviation of the node features and save it to a CSV file. We then divide all our obtained data into train and test sets.

Phase II- Building various models, training and testing them.

We divide our data into train (0.8) and test (0.2) data. We then check the ratio of fraudulent and honest transactions in our train and test sets. We use machine following machine learning and deep learning models to predict if a transaction is fraudulent:

1. Logistic Regression: This is a simple linear classifier. Logistic regression works well for binary classification problems.

2. Multilayer Perceptron: Multilayer perceptron helps in separation data that cannot be classified using a linear classifier by introducing nonlinearity.

3. Naive Bayes: This model uses the Bayes theorem to calculate the probability of a transaction being fraudulent.

4. Adaboost: This is an ensemble learning method to boost the performance of binary classifiers.

5. Decision Tree: This classifier has a sequence of conditions and questions on data based on various features.

6. SVM: It uses a kernel method to transform the data in the dataset, and based on these transitions, it finds a boundary between all possible outputs.

7. Random Forest Classifier: This classifier fits a number of decision trees on small batches of the dataset.

8. Neural Network: This model consists of six dense layers and four hidden layers. Relu and sigmoid were used as activation functions.

Phase III-Evaluation of models on test set

We evaluate all our classification models using bootstrap sampling. In machine learning, bootstrap sampling involves drawing sample data with replacement from the dataset to estimate a parameter. So, we first choose the number of bootstrap samples.

Then, we choose the sample size. Then, for each bootstrap sample, we draw a sample with chosen bootstrap size (with replacement) and test the sample's data. For this purpose, we use the accuracy metric, which is a standard metric used in machine learning problems. We then take the mean of all accuracies obtained in this fashion to evaluate the skill of our model.

2.3.1 ADVANTAGES OF PROPOSED SYSTEM

- Improved Accuracy
- Reduces False Positives
- Cost Efficient
- Scalability
- Flexibility

2.4 FEASIBILITY STUDY

The feasibility of the project is analyzed in this phase and 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. Three considerations involved in the feasibility analysis:

- Economic Feasibility
- Technical Feasibility
- Social Feasibility

2.4.1 ECONOMICAL FEASIBILITY

This study is carried out to check the economic impact that the system will have on the organization. The amount of fund that the company can pour into the research and development of the system is limited. The expenditures must be justified. Thus, the developed 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. This will lead to high demands on the available technical resources. This will lead to high demands being placed on the client. 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.

2.5 HARDWARE & SOFTWARE REQUIREMENTS

2.5.1 HARDWARE REQUIREMENTS:

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

• System	: Pentium IV 2.4 GHz.
• RAM	: 512MB.
• HDD	: 40 GB.
• Floppy Drive	: 1.44 M

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 (or) higher
• Coding Languages	: Python

3. ARCHITECTURE

3.ARCHITECTURE

3.1 PROJECT ARCHITECTURE

This project architecture shows the procedure followed for classification, starting from input to final prediction.

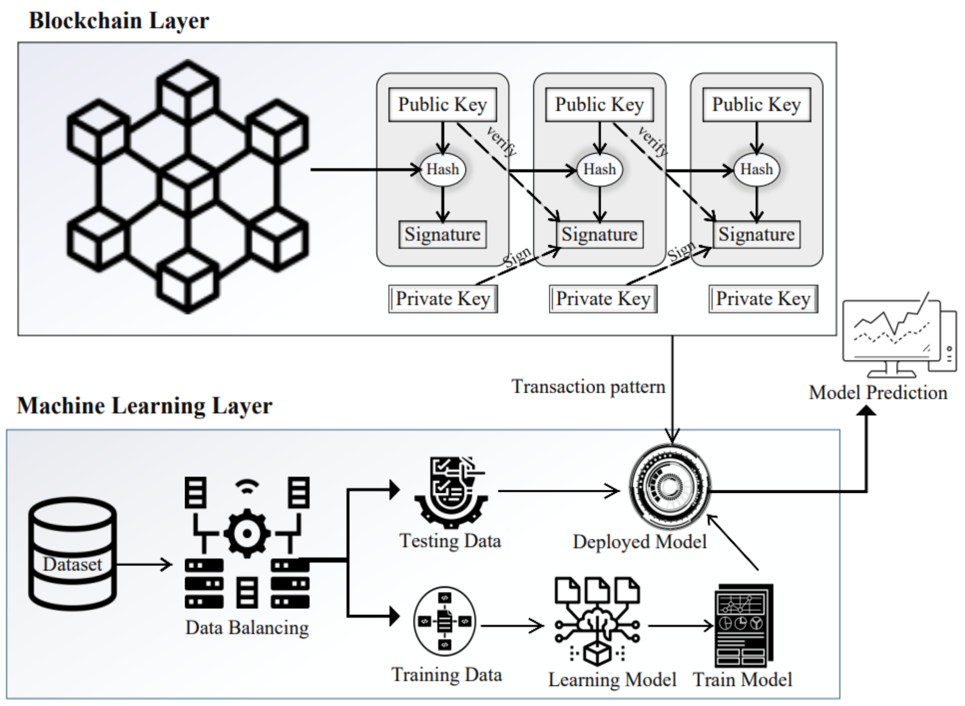


Figure 3.1: Architecture of Comparative Study of Machine Learning Algorithms for Fraud Detection in Block Chain.

3.2 DESCRIPTION:

The project involves conducting a comparative study of machine learning algorithms to enhance fraud detection within blockchain transactions. It includes data collection, preprocessing, and the development of machine learning models. The system provides a user-friendly interface for uploading blockchain data, analysing it, and presenting fraud detection results. Security measures are implemented to protect data and algorithms. The project aims to bridge the gap between technology and security in the blockchain ecosystem, ultimately promoting trust and adoption of blockchain technology by addressing security concerns.

3.3 USECASE DIAGRAM

A 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.

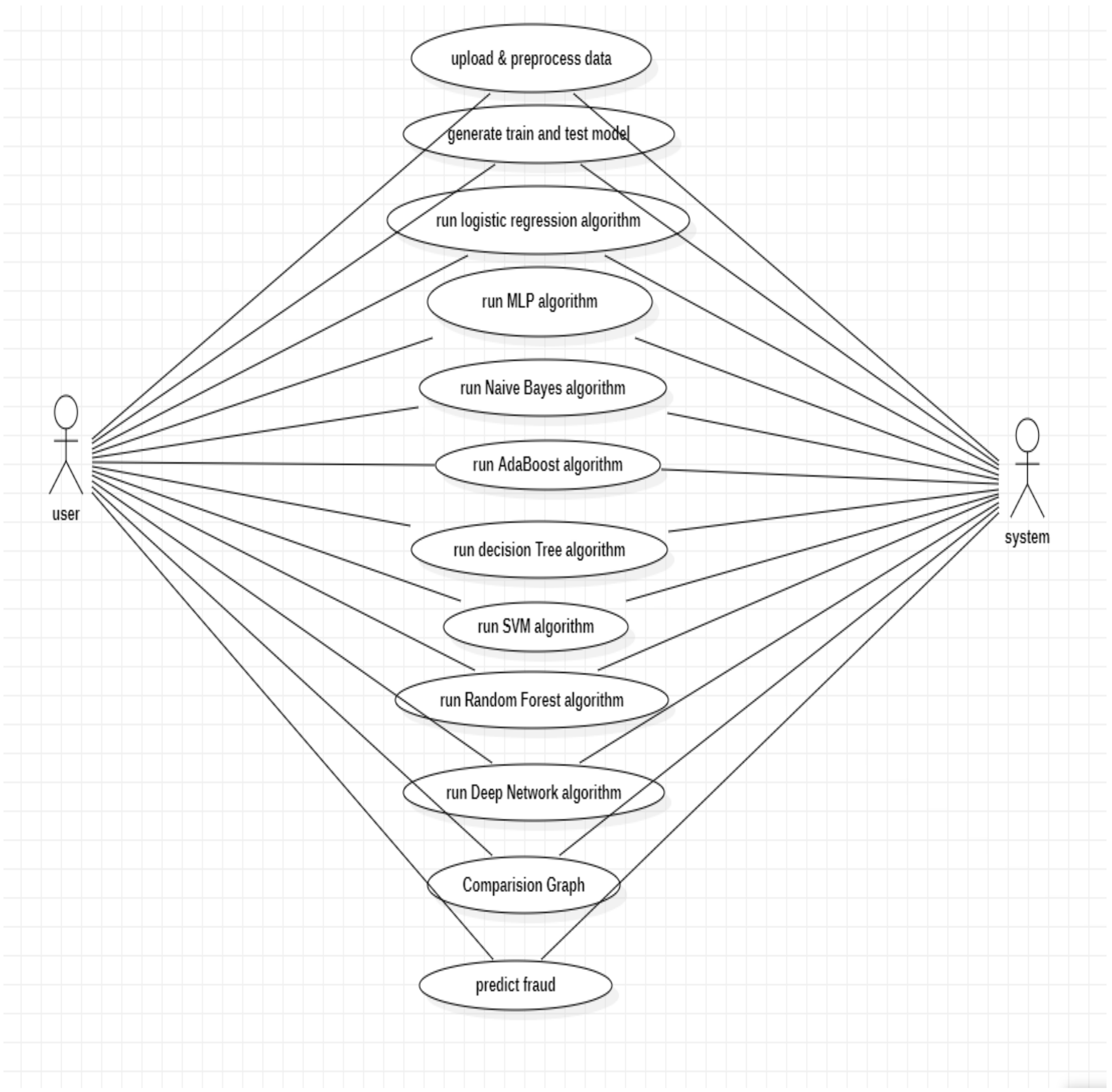

upload & preprocess data
generate train and test model
run logistic regression algorithm
run MLP algorithm
run Naive Bayes algorithm
run AdaBoost algorithm
run decision Tree algorithm
run SVM algorithm
run Random Forest algorithm
run Deep Network algorithm
Comparision Graph
predict fraud
user
system

Figure 3.2: Use Case Diagram for Comparative Study of Machine Learning Algorithms for Fraud Detection in Blockchain.

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.

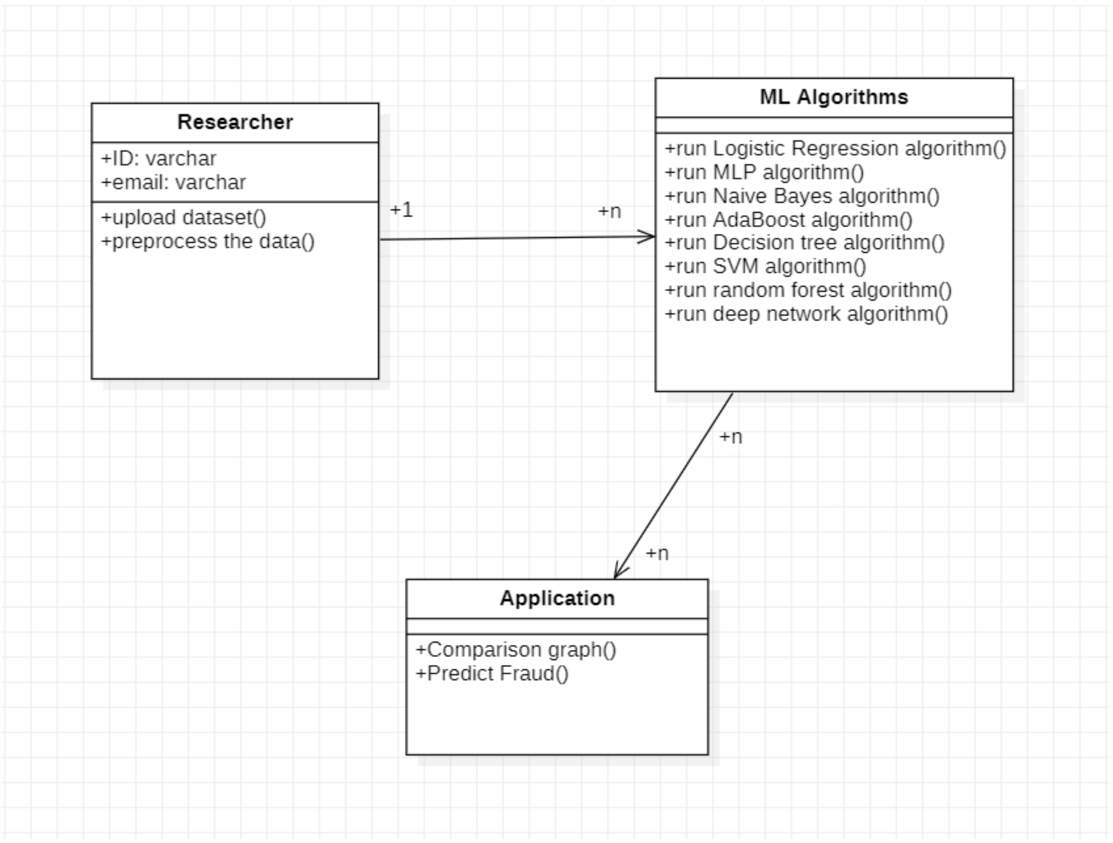


Figure 3.3 Class Diagram for Comparative Study of Machine Learning Algorithms for Fraud Detection in Blockchain

3.5 SEQUENCE DIAGRAM

A sequence diagram in Unified Modelling Language (UML) is 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.

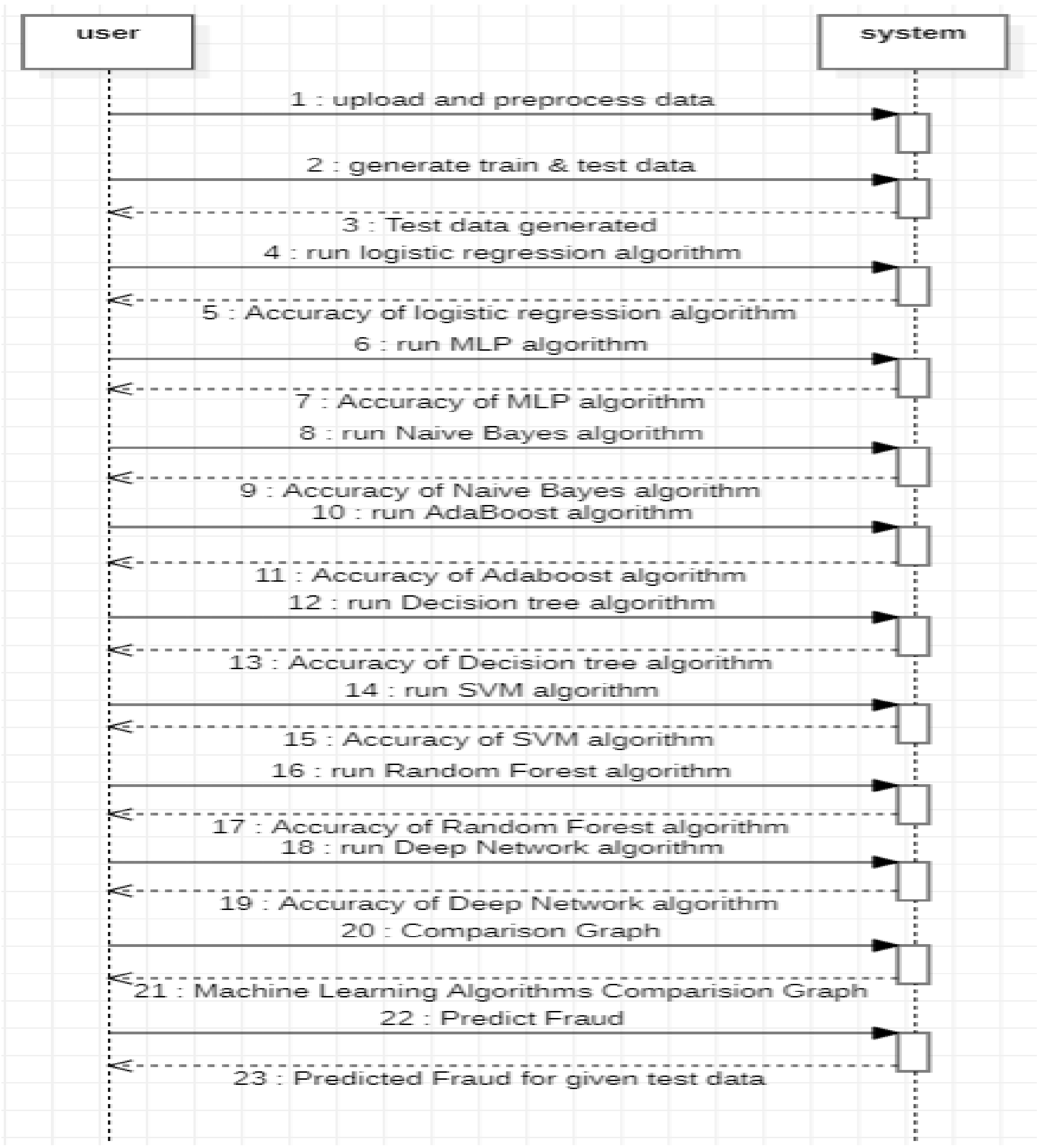

user
system
1 : upload and preprocess data
2 : generate train & test data
3 : Test data generated
4 : run logistic regression algorithm
5 : Accuracy of logistic regression algorithm
6 : run MLP algorithm
7 : Accuracy of MLP algorithm
8 : run Naive Bayes algorithm
9 : Accuracy of Naive Bayes algorithm
10 : run AdaBoost algorithm
11 : Accuracy of Adaboost algorithm
12 : run Decision tree algorithm
13 : Accuracy of Decision tree algorithm
14 : run SVM algorithm
15 : Accuracy of SVM algorithm
16 : run Random Forest algorithm
17 : Accuracy of Random Forest algorithm
18 : run Deep Network algorithm
19 : Accuracy of Deep Network algorithm
20 : Comparison Graph
21 : Machine Learning Algorithms Comparision Graph
22 : Predict Fraud
23 : Predicted Fraud for given test data

Figure 3.4 Sequence Diagram for Comparative Study of Machine Learning Algorithms for Fraud Detection in Blockchain

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 Modelling 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.

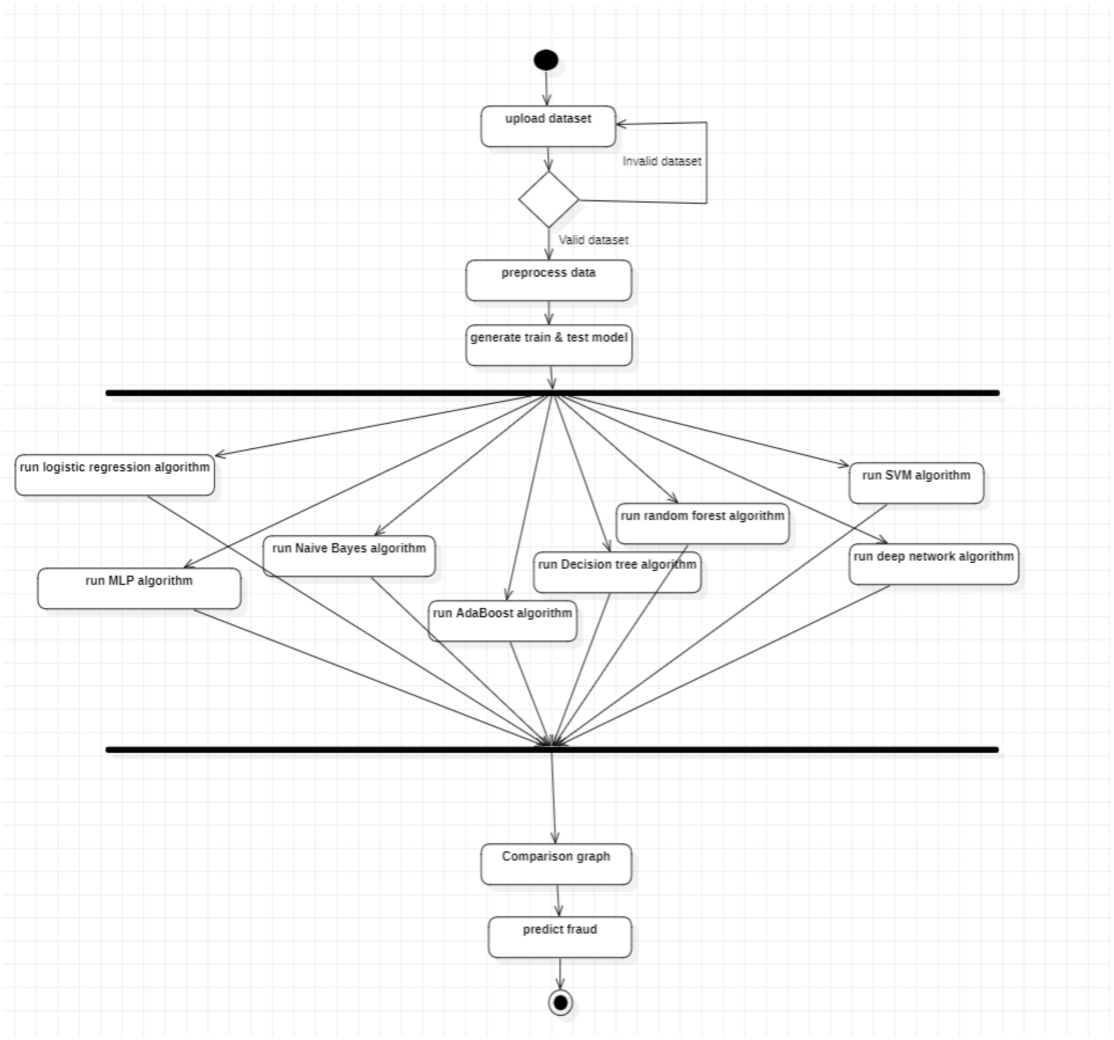

upload dataset
Invalid dataset
Valid dataset
preprocess data
generate train & test model
run logistic regression algorithm
run SVM algorithm
run random forest algorithm
run Naive Bayes algorithm
run MLP algorithm
run Decision tree algorithm
run deep network algorithm
run AdaBoost algorithm
Comparison graph
predict fraud

Figure 3.5 Activity Diagram for Comparative Study of Machine Learning Algorithms for Fraud Detection in Blockchain.

4. IMPLEMENTATION

4.1 SAMPLE CODE:

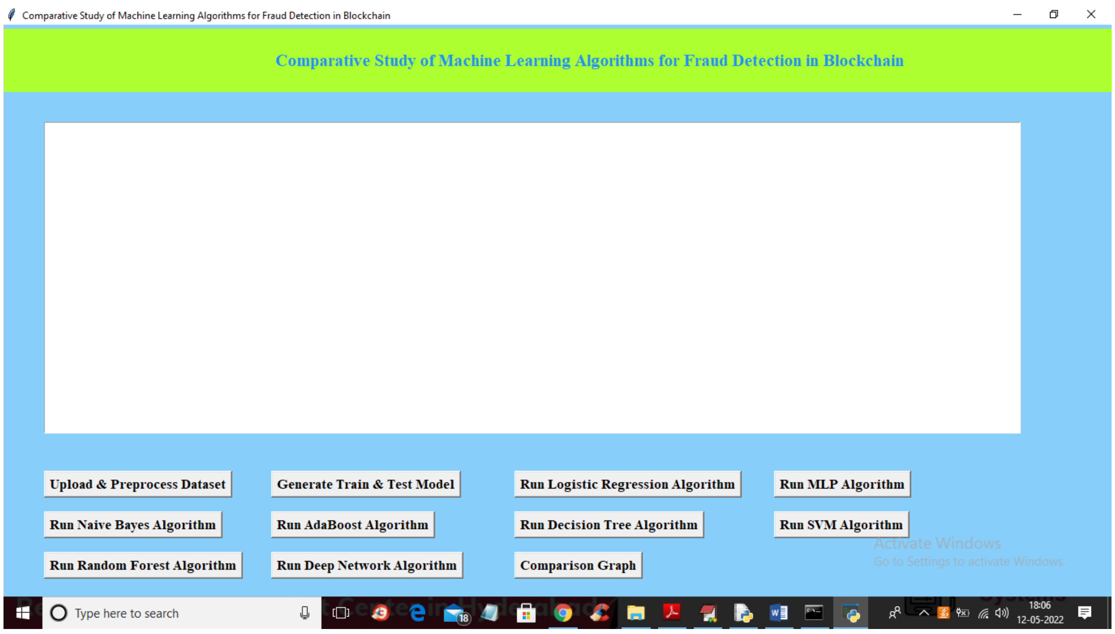
```
from tkinter import messagebox
from tkinter import *
from tkinter import simpledialog
import tkinter
from tkinter import filedialog
from tkinter.filedialog import askopenfilename
import numpy as np
import matplotlib.pyplot as plt
import pandas as pd
from sklearn.preprocessing import normalize
from sklearn.metrics import accuracy_score
from sklearn.model_selection import train_test_split
import os
from sklearn.metrics import confusion_matrix
from sklearn.metrics import accuracy_score
from sklearn.tree import DecisionTreeClassifier
from sklearn.naive_bayes import GaussianNB
from sklearn import svm
from sklearn.metrics import precision_score
from sklearn.metrics import recall_score
from sklearn.metrics import f1_score
import seaborn as sns
import webbrowser
from sklearn.ensemble import RandomForestClassifier
from sklearn.linear_model import LogisticRegression
from sklearn.neural_network import MLPClassifier
from sklearn.ensemble import AdaBoostClassifier
from keras.utils.np_utils import to_categorica
from keras.layers import  MaxPooling2D
from keras.layers import Dense, Dropout, Activation, Flatten
from keras.layers import Convolution2D
from keras.models import Sequential
from keras.models import model_from_json
import pickle
global filename
global X,Y
global dataset
global main
global text
accuracy = []
precision = []
recall = []
fscore = []
global X_train, X_test, y_train, y_test, predict_cls
global classifier
main = tkinter.Tk()
main.title("Comparative Study of Machine Learning Algorithms for Fraud Detection
in Blockchain") #designing main screen
main.geometry("1300x1200")
#fucntion to upload dataset
def uploadDataset():
    global filename
    global dataset
    text.delete('1.0', END)
    filename = filedialog.askopenfilename(initialdir="Dataset")
    text.insert(END,filename+" loaded\n\n")
    dataset = pd.read_csv(filename)
    text.insert(END,"Dataset before preprocessing\n\n")
    text.insert(END,str(dataset.head())
    text.update_idletasks()
    label = dataset.groupby('FLAG').size()
    label.plot(kind="bar")
    plt.title("Blockchain Fraud Detection Graph 0 means Normal & 1        means
Fraud")
    plt.show()


#function to perform dataset preprocessing
def trainTest():
    global X,Y
    global dataset
    global X_train, X_test, y_train, y_test
    text.delete('1.0', END)
    #replace missing values with 0
    dataset.fillna(0, inplace = True)
    Y = dataset['FLAG'].ravel()
    dataset = dataset.values
    X = dataset[:,4:dataset.shape[1]-2]
    X = normalize(X)
    indices = np.arange(X.shape[0])
    np.random.shuffle(indices)
    X = X[indices]
    Y = Y[indices]
    X = X[0:5000]
    Y = Y[0:5000]
    print(Y)
    print(X)
    text.insert(END,"Dataset after features normalization\n\n")
    text.insert(END,str(X)+"\n\n")
    text.insert(END,"Total records found in dataset : "+str(X.shape[0])+"\n")
    text.insert(END,"Total features found in dataset: "+str(X.shape[1])+"\n\n")
    X_train, X_test, y_train, y_test = train_test_split(X, Y, test_size=0.2)
    text.insert(END,"Dataset Train and Test Split\n\n")
    text.insert(END,"80% dataset records used to train ML algorithms : "+str(X_train.shape[0])+"\n")
    text.insert(END,"20% dataset records used to train ML algorithms : "+str(X_test.shape[0])+"\n")

def calculateMetrics(algorithm, predict, y_test):
    a = accuracy_score(y_test,predict)*100
    p = precision_score(y_test, predict,average='macro') * 100
    r = recall_score(y_test, predict,average='macro') * 100
    f = f1_score(y_test, predict,average='macro') * 100
    accuracy.append(a)
    precision.append(p)
    recall.append(r)
    fscore.append(f)
    text.insert(END,algorithm+" Accuracy  :  "+str(a)+"\n")
    text.insert(END,algorithm+" Precision : "+str(p)+"\n")
    text.insert(END,algorithm+" Recall    : "+str(r)+"\n")
    text.insert(END,algorithm+" FScore    : "+str(f)+"\n\n")

def runLogisticRegression():
    global X,Y, X_train, X_test, y_train, y_test
    global accuracy, precision,recall, fscore
    accuracy.clear()
    precision.clear()
    recall.clear()
    fscore.clear()
    text.delete('1.0', END)
    lr = LogisticRegression()
    lr.fit(X, Y)
    predict = lr.predict(X_test)
    calculateMetrics("Logistic Regression", predict, y_test)

def runMLP():
    mlp = MLPClassifier()
    mlp.fit(X_train, y_train)
    predict = mlp.predict(X_test)
    calculateMetrics("MLP", predict, y_test)

def runNaiveBayes():
    cls = GaussianNB()
    cls.fit(X_train, y_train)
    predict = cls.predict(X_test)
    calculateMetrics("Naive Bayes", predict, y_test)

def runAdaBoost():
    cls = AdaBoostClassifier()
    cls.fit(X_train, y_train)
    predict = cls.predict(X_test)
    calculateMetrics("AdaBoost", predict, y_test)

def runDT():
    global predict_cls
    cls = DecisionTreeClassifier()
    cls.fit(X_train, y_train)
    predict = cls.predict(X_test)
    calculateMetrics("Decision Tree", predict, y_test)

def runSVM():
    cls = svm.SVC()
    cls.fit(X_train, y_train)
    predict = cls.predict(X_test)
    calculateMetrics("SVM", predict, y_test)

def runRF():
    global predict_cls
    rf = RandomForestClassifier()
    rf.fit(X_train, y_train)
    predict = rf.predict(X_test)
    predict_cls = rf
    calculateMetrics("Random Forest", predict, y_test)

def predict():
    global predict_cls
    text.delete('1.0', END)
    filename = filedialog.askopenfilename(initialdir="Dataset")
    dataset = pd.read_csv(filename)
    dataset.fillna(0, inplace = True)
    dataset = dataset.values
    X = dataset[:,4:dataset.shape[1]-2]
    X1 = normalize(X)
    prediction = predict_cls.predict(X1)
    print(prediction)
    for i in range(len(prediction)):
        if prediction[i] == 0:
            text.insert(END,"Test DATA : "+str(X[i])+" ===> PREDICTED AS
NORMAL\n\n")
        else:
            text.insert(END,"Test DATA : "+str(X[i])+" ===> PREDICTED AS
FRAUD\n\n")
```

```
def runDeepNetwork():
    global X, Y
    X = np.reshape(X, (X.shape[0], X.shape[1], 1, 1))
    Y = to_categorical(Y)
    X_train, X_test, y_train, y_test = train_test_split(X, Y, test_size=0.2)
    if os.path.exists('model/model.json'):
        with open('model/model.json', "r") as json_file:
            loaded_model_json = json_file.read()
            classifier = model_from_json(loaded_model_json)
        json_file.close()
        classifier.load_weights("model/model_weights.h5")
        classifier._make_predict_function()
    else:
        classifier = Sequential()
        classifier.add(Convolution2D(32, 1, 1, input_shape = (X_train.shape[1],
X_train.shape[2], X_train.shape[3]), activation = 'relu'))
        classifier.add(MaxPooling2D(pool_size = (1, 1)))
        classifier.add(Convolution2D(32, 1, 1, activation = 'relu'))
        classifier.add(MaxPooling2D(pool_size = (1, 1)))
        classifier.add(Flatten())
        classifier.add(Dense(output_dim = 256, activation = 'relu'))
        classifier.add(Dense(output_dim = Y.shape[1], activation = 'softmax'))
        print(classifier.summary())
        classifier.compile(optimizer = 'adam', loss = 'categorical_crossentropy', metrics =
['accuracy'])
        hist = classifier.fit(X, Y, batch_size=16, epochs=10, shuffle=True, verbose=2)
        classifier.save_weights('model/model_weights.h5')
        model_json = classifier.to_json()
        with open("model/model.json", "w") as json_file:
            json_file.write(model_json)
        json_file.close()
    predict = classifier.predict(X_test)
    predict = np.argmax(predict, axis=1)
    y_test = np.argmax(y_test, axis=1)
    calculateMetrics("Deep Neural Network", predict, y_test)


def graph():
    output = "<html><body><table align=center border=1><tr><th>Algorithm Name</th><th>Accuracy</th><th>Precision</th><th>Recall</th>"
    output+="<th>FSCORE</th></tr>"
    output+="<tr><td>Logistic Regression Algorithm</td><td>"+str(accuracy[0])+"</td><td>"+str(precision[0])+"</td><td>"+str(recall[0])+"</td><td>"+str(fscore[0])+"</td></tr>"
    output+="<tr><td>MLP Algorithm</td><td>"+str(accuracy[1])+"</td><td>"+str(precision[1])+"</td><td>"+str(recall[1])+"</td><td>"+str(fscore[1])+"</td></tr>"
    output+="<tr><td>Naive Bayes Algorithm</td><td>"+str(accuracy[2])+"</td><td>"+str(precision[2])+"</td><td>"+str(recall[2])+"</td><td>"+str(fscore[2])+"</td></tr>"
    output+="<tr><td>AdaBoost Algorithm</td><td>"+str(accuracy[3])+"</td><td>"+str(precision[3])+"</td><td>"+str(recall[3])+"</td><td>"+str(fscore[3])+"</td></tr>"
    output+="<tr><td>Decision Tree Algorithm</td><td>"+str(accuracy[4])+"</td><td>"+str(precision[4])+"</td><td>"+str(recall[4])+"</td><td>"+str(fscore[4])+"</td></tr>"
    output+="<tr><td>SVM Algorithm</td><td>"+str(accuracy[5])+"</td><td>"+str(precision[5])+"</td><td>"+str(recall[5])+"</td><td>"+str(fscore[5])+"</td></tr>"
    output+="<tr><td>Random Forest Algorithm</td><td>"+str(accuracy[6])+"</td><td>"+str(precision[6])+"</td><td>"+str(recall[6])+"</td><td>"+str(fscore[6])+"</td></tr>"
    output+="<tr><td>Deep Neural Network Algorithm</td><td>"+str(accuracy[7])+"</td><td>"+str(precision[7])+"</td><td>"+str(recall[7])+"</td><td>"+str(fscore[7])+"</td></tr>"
    output+="</table></body></html>"
    f = open("table.html", "w")
    f.write(output)
    f.close()

    webbrowser.open("table.html",new=2)

    df = pd.DataFrame([['Logistic Regression','Precision',precision[0]],['Logistic Regression','Recall',recall[0]],['Logistic Regression','F1 Score',fscore[0]],['Logistic Regression','Accuracy',accuracy[0]],

                      ['MLP','Precision',precision[1]],['MLP','Recall',recall[1]],['MLP','F1 Score',fscore[1]],['MLP','Accuracy',accuracy[1]],

                      ['Naive Bayes','Precision',precision[2]],['Naive Bayes','Recall',recall[2]],['Naive Bayes','F1 Score',fscore[2]],['Naive Bayes','Accuracy',accuracy[2]],


['AdaBoost','Precision',precision[3]],['AdaBoost','Recall',recall[3]],['AdaBoost','F1 Score',fscore[3]],['AdaBoost','Accuracy',accuracy[3]],

                      ['Decision Tree','Precision',precision[4]],['Decision Tree','Recall',recall[4]],['Decision Tree','F1 Score',fscore[4]],['Decision Tree','Accuracy',accuracy[4]],

                      ['SVM','Precision',precision[5]],['SVM','Recall',recall[5]],['SVM','F1 Score',fscore[5]],['SVM','Accuracy',accuracy[5]],

                      ['Random Forest','Precision',precision[6]],['Random Forest','Recall',recall[6]],['Random Forest','F1 Score',fscore[6]],['Random Forest','Accuracy',accuracy[6]],

                      ['Deep Neural Network','Precision',precision[7]],['Deep Neural Network','Recall',recall[7]],['Deep Neural Network','F1 Score',fscore[7]],['Deep Neural Network','Accuracy',accuracy[7]],

                     ],columns=['Parameters','Algorithms','Value'])

    df.pivot("Parameters", "Algorithms", "Value").plot(kind='bar')

    plt.show()

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

title = Label(main, text='Comparative Study of Machine Learning Algorithms for Fraud Detection in Blockchain')

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)
```

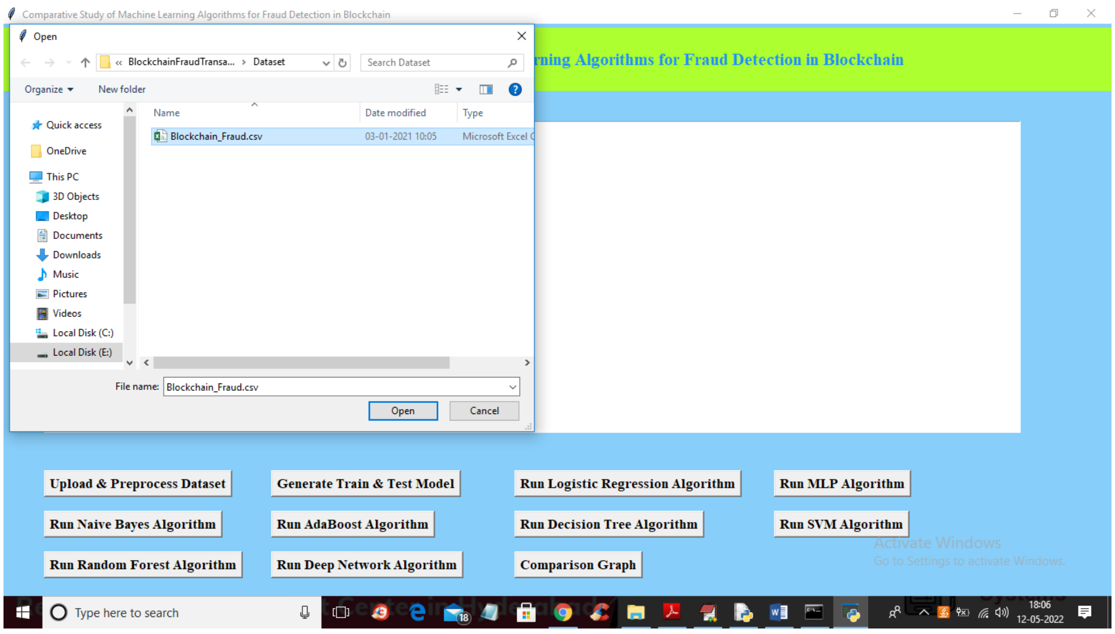
```
scroll=Scrollbar(text)

text.configure(yscrollcommand=scroll.set)

text.place(x=50,y=120)

text.config(font=font1)

font1 = ('times', 13, 'bold')


uploadButton = Button(main, text="Upload & Preprocess Dataset",
command=uploadDataset)

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

uploadButton.config(font=font1)


traintestButton = Button(main, text="Generate Train & Test Model",
command=trainTest)

traintestButton.place(x=330,y=550)

traintestButton.config(font=font1)


lrButton = Button(main, text="Run Logistic Regression Algorithm",
command=runLogisticRegression)

lrButton.place(x=630,y=550)

lrButton.config(font=font1)


mlpButton = Button(main, text="Run MLP Algorithm", command=runMLP)

mlpButton.place(x=950,y=550)

mlpButton.config(font=font1)


nbButton = Button(main, text="Run Naive Bayes Algorithm",
command=runNaiveBayes)

nbButton.place(x=50,y=600)

nbButton.config(font=font1)


adaboostButton = Button(main, text="Run AdaBoost Algorithm",
command=runAdaBoost)
```

```
adaboostButton.place(x=330,y=600)
adaboostButton.config(font=font1)

dtButton = Button(main, text="Run Decision Tree Algorithm", command=runDT)
dtButton.place(x=630,y=600)
dtButton.config(font=font1)

svmButton = Button(main, text="Run SVM Algorithm", command=runSVM)
svmButton.place(x=950,y=600)
svmButton.config(font=font1)

rfButton = Button(main, text="Run Random Forest Algorithm", command=runRF)
rfButton.place(x=50,y=650)
rfButton.config(font=font1)

dnButton = Button(main, text="Run Deep Network Algorithm",
command=runDeepNetwork)
dnButton.place(x=330,y=650)
dnButton.config(font=font1)

graphButton = Button(main, text="Comparison Graph", command=graph)
graphButton.place(x=630,y=650)
graphButton.config(font=font1)

predictButton = Button(main, text="Predict Fraud ", command=predict)
predictButton.place(x=950,y=650)
predictButton.config(font=font1)

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

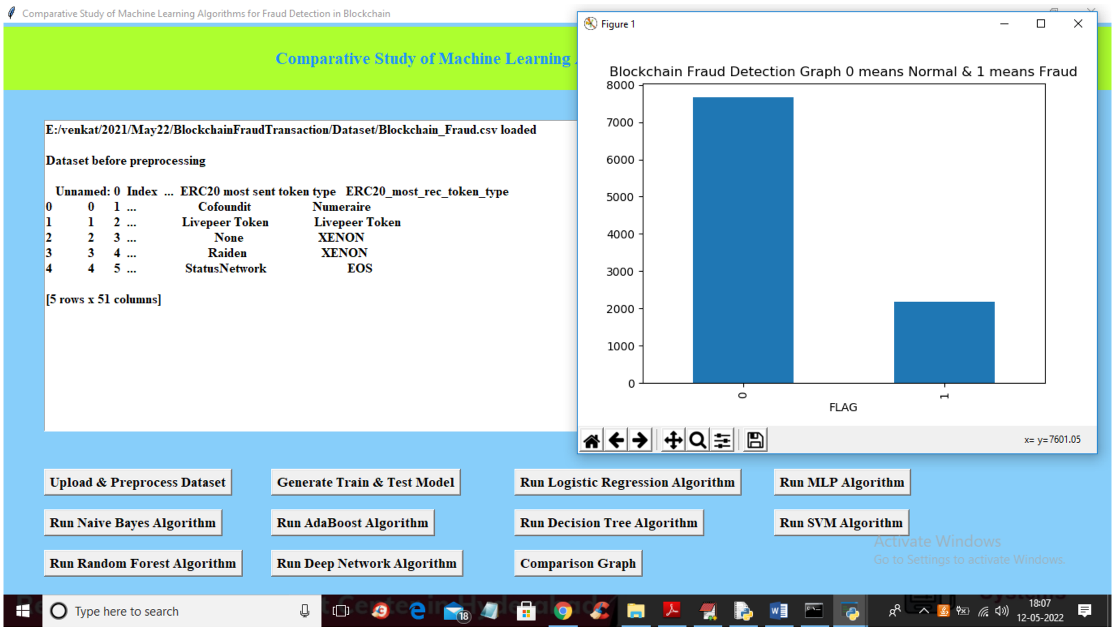
5.SCREENSHOTS


Comparative Study of Machine Learning Algorithms for Fraud Detection in Blockchain
Comparative Study of Machine Learning Algorithms for Fraud Detection in Blockchain
Upload & Preprocess Dataset
Generate Train & Test Model
Run Logistic Regression Algorithm
Run MLP Algorithm
Run Naive Bayes Algorithm
Run AdaBoost Algorithm
Run Decision Tree Algorithm
Run SVM Algorithm
Run Random Forest Algorithm
Run Deep Network Algorithm
Comparison Graph

Screenshot 5.1 Initial User Interface


Comparative Study of Machine Learning Algorithms for Fraud Detection in Blockchain
Open
BlockchainFraudTransa... > Dataset
Blockchain_Fraud.csv
File name: Blockchain_Fraud.csv
Open
Cancel
Upload & Preprocess Dataset
Generate Train & Test Model
Run Logistic Regression Algorithm
Run MLP Algorithm
Run Naive Bayes Algorithm
Run AdaBoost Algorithm
Run Decision Tree Algorithm
Run SVM Algorithm
Run Random Forest Algorithm
Run Deep Network Algorithm
Comparison Graph

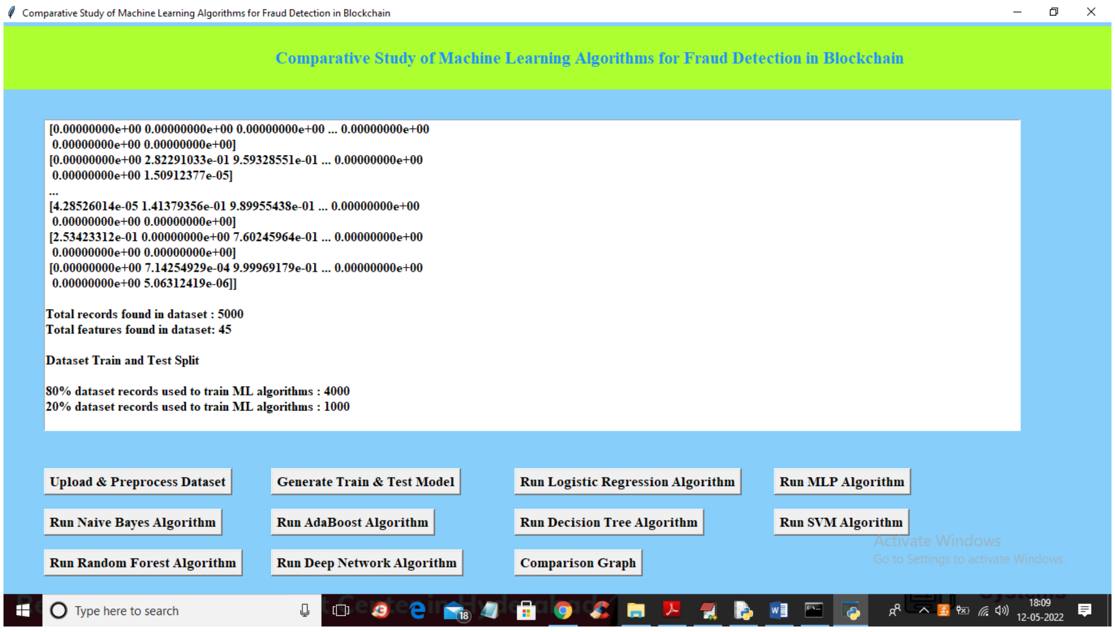
Screenshot 5.2 Upload and preprocess the dataset.



```
E:/venkat/2021/May22/BlockchainFraudTransaction/Dataset/Blockchain_Fraud.csv loaded

Dataset before preprocessing

  Unnamed: 0  Index  ...  ERC20 most sent token type   ERC20_most_rec_token_type
0          0      1  ...            Cofoundit              Numeraire
1          1      2  ...        Livepeer Token          Livepeer Token
2          2      3  ...              None               XENON
3          3      4  ...             Raiden              XENON
4          4      5  ...       StatusNetwork                  EOS

[5 rows x 51 columns]
```

Comparative Study of Machine Learning Algorithms for Fraud Detection in Blockchain
Figure 1
Blockchain Fraud Detection Graph 0 means Normal & 1 means Fraud
8000
7000
6000
5000
4000
3000
2000
1000
0
0
1
FLAG
x= y=7601.05
Upload & Preprocess Dataset
Generate Train & Test Model
Run Logistic Regression Algorithm
Run MLP Algorithm
Run Naive Bayes Algorithm
Run AdaBoost Algorithm
Run Decision Tree Algorithm
Run SVM Algorithm
Run Random Forest Algorithm
Run Deep Network Algorithm
Comparison Graph
Type here to search
18:07
12-05-2022

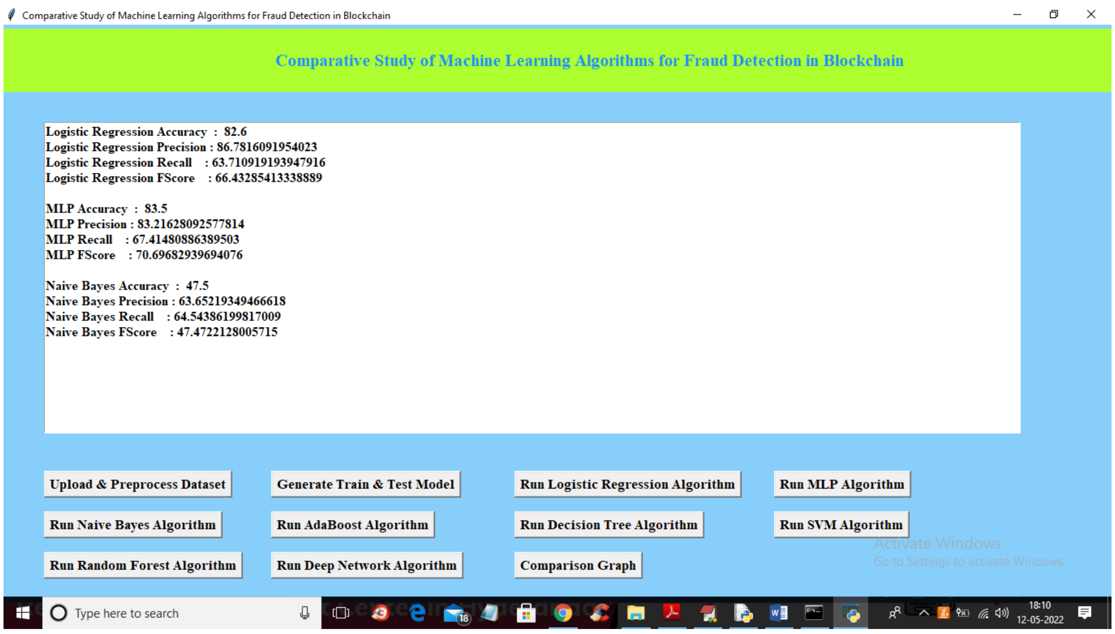
Screenshot 5.3 Generate the train and test model.



```
[0.00000000e+00 0.00000000e+00 0.00000000e+00 ... 0.00000000e+00
 0.00000000e+00 0.00000000e+00]
[0.00000000e+00 2.82291033e-01 9.59328551e-01 ... 0.00000000e+00
 0.00000000e+00 1.50912377e-05]
...
[4.28526014e-05 1.41379356e-01 9.89955438e-01 ... 0.00000000e+00
 0.00000000e+00 0.00000000e+00]
[2.53423312e-01 0.00000000e+00 7.60245964e-01 ... 0.00000000e+00
 0.00000000e+00 0.00000000e+00]
[0.00000000e+00 7.14254929e-04 9.99969179e-01 ... 0.00000000e+00
 0.00000000e+00 5.06312419e-06]]

Total records found in dataset : 5000
Total features found in dataset: 45

Dataset Train and Test Split

80% dataset records used to train ML algorithms : 4000
20% dataset records used to train ML algorithms : 1000
```

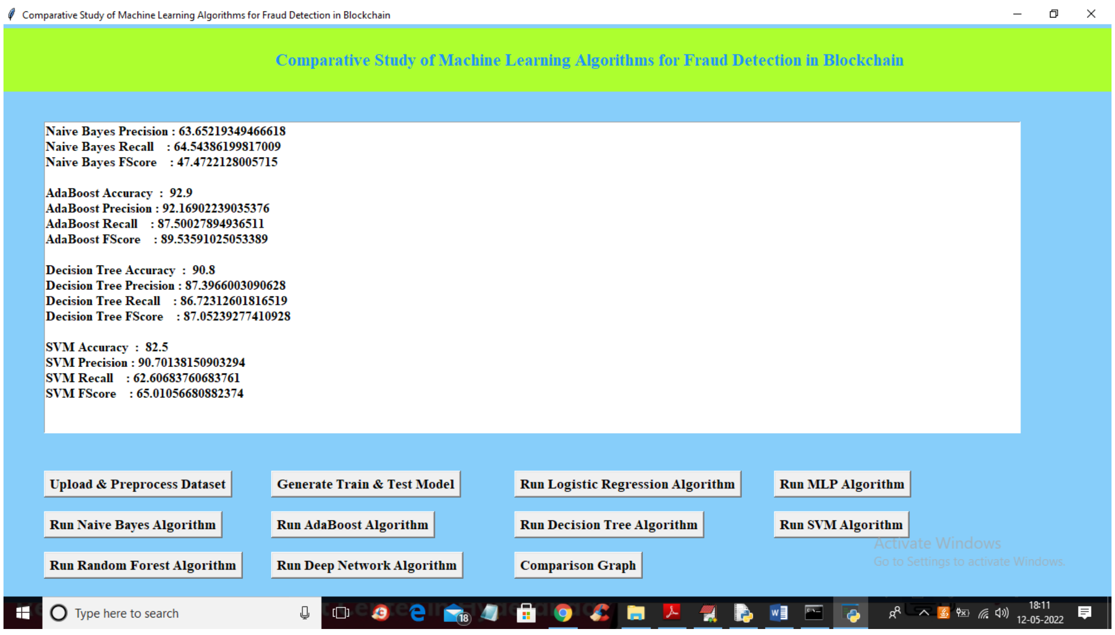
Comparative Study of Machine Learning Algorithms for Fraud Detection in Blockchain
Comparative Study of Machine Learning Algorithms for Fraud Detection in Blockchain
Upload & Preprocess Dataset
Generate Train & Test Model
Run Logistic Regression Algorithm
Run MLP Algorithm
Run Naive Bayes Algorithm
Run AdaBoost Algorithm
Run Decision Tree Algorithm
Run SVM Algorithm
Run Random Forest Algorithm
Run Deep Network Algorithm
Comparison Graph
Type here to search
18:09
12-05-2022

Screenshot 5.4 Dataset train and test split.


Comparative Study of Machine Learning Algorithms for Fraud Detection in Blockchain
Comparative Study of Machine Learning Algorithms for Fraud Detection in Blockchain
Upload & Preprocess Dataset
Generate Train & Test Model
Run Logistic Regression Algorithm
Run MLP Algorithm
Run Naive Bayes Algorithm
Run AdaBoost Algorithm
Run Decision Tree Algorithm
Run SVM Algorithm
Run Random Forest Algorithm
Run Deep Network Algorithm
Comparison Graph

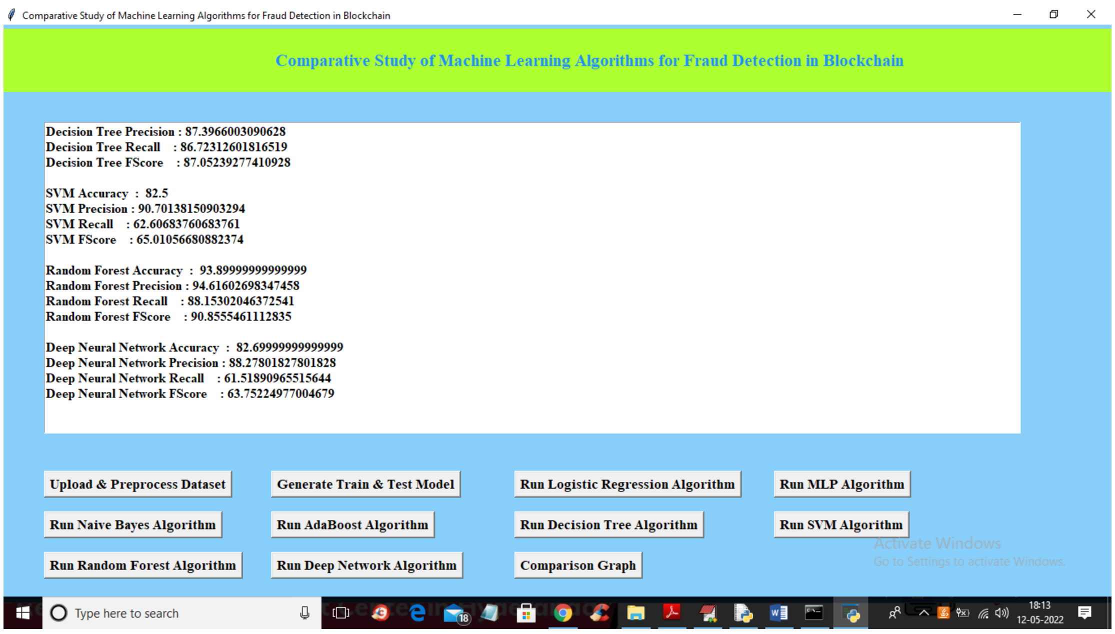
```
Logistic Regression Accuracy  :  82.6
Logistic Regression Precision : 86.7816091954023
Logistic Regression Recall    : 63.710919193947916
Logistic Regression FScore    : 66.43285413338889

MLP Accuracy  :  83.5
MLP Precision : 83.21628092577814
MLP Recall    : 67.41480886389503
MLP FScore    : 70.69682939694076

Naive Bayes Accuracy  :  47.5
Naive Bayes Precision : 63.65219349466618
Naive Bayes Recall    : 64.54386199817009
Naive Bayes FScore    : 47.4722128005715
```

Screenshot 5.5 Accuracy of Logistic Regression, MLP Algorithm, Naïve Bayes Algorithm.


Comparative Study of Machine Learning Algorithms for Fraud Detection in Blockchain
Comparative Study of Machine Learning Algorithms for Fraud Detection in Blockchain
Upload & Preprocess Dataset
Generate Train & Test Model
Run Logistic Regression Algorithm
Run MLP Algorithm
Run Naive Bayes Algorithm
Run AdaBoost Algorithm
Run Decision Tree Algorithm
Run SVM Algorithm
Run Random Forest Algorithm
Run Deep Network Algorithm
Comparison Graph

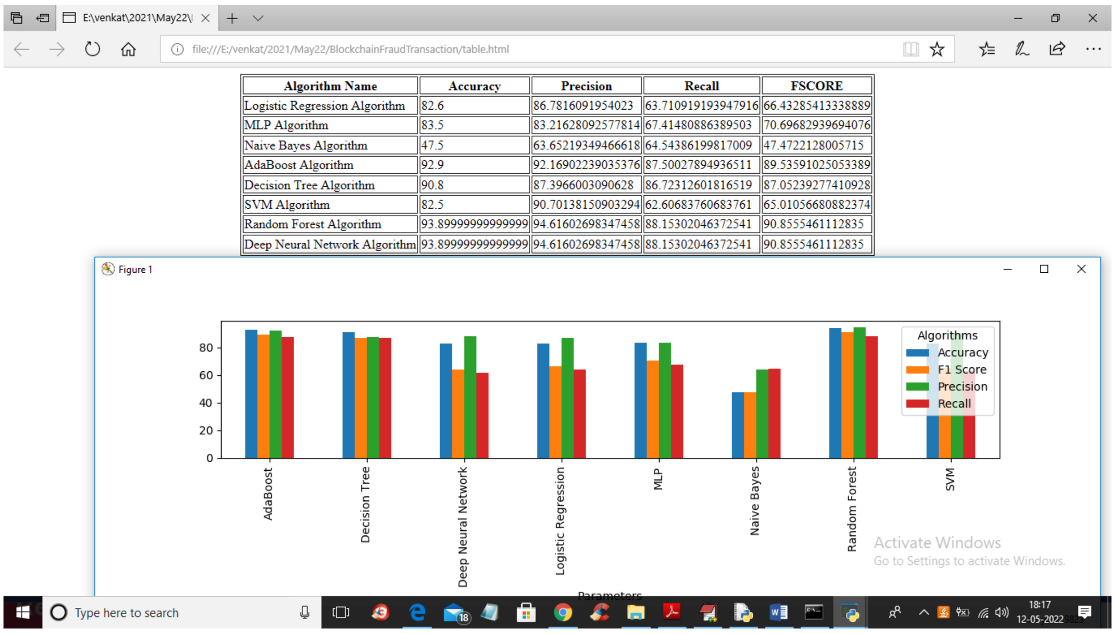
```
Naive Bayes Precision : 63.65219349466618
Naive Bayes Recall    : 64.54386199817009
Naive Bayes FScore    : 47.4722128005715

AdaBoost Accuracy  :  92.9
AdaBoost Precision : 92.16902239035376
AdaBoost Recall    : 87.50027894936511
AdaBoost FScore    : 89.53591025053389

Decision Tree Accuracy  :  90.8
Decision Tree Precision : 87.3966003090628
Decision Tree Recall    : 86.72312601816519
Decision Tree FScore    : 87.05239277410928

SVM Accuracy  :  82.5
SVM Precision : 90.70138150903294
SVM Recall    : 62.60683760683761
SVM FScore    : 65.01056680882374
```

Screenshot 5.6 Accuracy of Ada Boost Algorithm, Decision tree, SVM Algorithm.


Comparative Study of Machine Learning Algorithms for Fraud Detection in Blockchain
Comparative Study of Machine Learning Algorithms for Fraud Detection in Blockchain
Upload & Preprocess Dataset
Generate Train & Test Model
Run Logistic Regression Algorithm
Run MLP Algorithm
Run Naive Bayes Algorithm
Run AdaBoost Algorithm
Run Decision Tree Algorithm
Run SVM Algorithm
Run Random Forest Algorithm
Run Deep Network Algorithm
Comparison Graph
Type here to search

```
Decision Tree Precision : 87.3966003090628
Decision Tree Recall    : 86.72312601816519
Decision Tree FScore    : 87.05239277410928

SVM Accuracy  :  82.5
SVM Precision : 90.70138150903294
SVM Recall    : 62.60683760683761
SVM FScore    : 65.01056680882374

Random Forest Accuracy  :  93.89999999999999
Random Forest Precision : 94.61602698347458
Random Forest Recall    : 88.15302046372541
Random Forest FScore    : 90.8555461112835

Deep Neural Network Accuracy  :  82.69999999999999
Deep Neural Network Precision : 88.27801827801828
Deep Neural Network Recall    : 61.51890965515644
Deep Neural Network FScore    : 63.75224977004679
```

Screenshot 5.7 Accuracy of Random Forest, Deep Neural Networks.



Algorithm Name	Accuracy	Precision	Recall	FSCORE
Logistic Regression Algorithm	82.6	86.7816091954023	63.710919193947916	66.43285413338889
MLP Algorithm	83.5	83.21628092577814	67.41480886389503	70.69682939694076
Naive Bayes Algorithm	47.5	63.65219349466618	64.54386199817009	47.4722128005715
AdaBoost Algorithm	92.9	92.16902239035376	87.50027894936511	89.53591025053389
Decision Tree Algorithm	90.8	87.3966003090628	86.72312601816519	87.05239277410928
SVM Algorithm	82.5	90.70138150903294	62.60683760683761	65.01056680882374
Random Forest Algorithm	93.89999999999999	94.61602698347458	88.15302046372541	90.8555461112835
Deep Neural Network Algorithm	93.89999999999999	94.61602698347458	88.15302046372541	90.8555461112835

Screenshot 5.8 Comparison 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 tests. Each test type addresses a specific testing requirement.

6.2 TYPES OF TESTING

6.2.1 UNIT 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.

6.2.2 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.

6.3.3 FUNCTIONAL TEST:

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.

Organization and preparation of functional tests is focused on requirements, key functions, or special test cases. In addition, systematic coverage pertaining to identify Business process flows; data fields, predefined processes, and successive processes must be considered for testing. Before functional testing is complete, additional tests are identified and the effective value of current tests is determined.

6.3 TEST CASES

6.3.1 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.

Test strategy and approach

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

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.

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.

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

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.

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

7.CONCLUSION

7.CONCLUSION & FUTURE SCOPE

7.1 PROJECT CONCLUSION

A method has been proposed for the detection of fraudulent transactions in a blockchain network using machine learning. In this method, various supervised learning approaches like support vector machines, decision trees, logistic regression, and dense neural networks were analysed. A thorough comparative analysis of all the approaches is performed through accuracy. This work can be extended for the comparative study of unsupervised algorithms like clustering. In the future, we also plan to do an exhaustive study on fraudulent activities in a private blockchain.

7.2 FUTURE SCOPE

- Hybrid Approaches: Combining the strengths of both blockchain and machine learning can lead to powerful fraud detection systems. Future research may focus on developing hybrid models that leverage blockchain for data integrity and security while using advanced machine learning for fraud detection.
- Scalability: As the volume of blockchain transactions increases, the scalability of fraud detection systems becomes crucial. Future projects may explore ways to optimize and scale machine learning algorithms for large-scale blockchain networks.
- Cross-Industry Applications: The skills and knowledge gained from a comparative study of machine learning algorithms for fraud detection in blockchain can be applied to various industries beyond finance, such as healthcare, supply chain, and more.

8.BIBLIOGRAPHY

8. BIBLIOGRAPHY

8.1 REFERENCES

[1] Cai, Y., Zhu, D. Fraud detections for online businesses: a perspective from blockchain technology. Financ Innov 2, 20 (2016). https://doi.org/10.1186/s40854-016-0039-4

[2] Hyvarinen, H., Risius, M. & Friis, G. A Blockchain-Based Approach ¨ Towards Overcoming Financial Fraud in Public Sector Services. Bus Inf Syst Eng 59, 441–456 (2017). https://doi.org/10.1007/s12599-017-0502- 4

[3] Xu, J.J. Are blockchains immune to all malicious attacks?. Finance Innov 2, 25 (2016). https://doi.org/10.1186/s40854-016-0046-5

[4] Ostapowicz M., Zbikowski K. (2019) Detecting Fraudulent Accounts on ˙ Blockchain: A Supervised Approach. In: Cheng R., Mamoulis N., Sun Y., Huang X. (eds) Web Information Systems Engineering – WISE 2019. WISE 2020. Lecture Notes in Computer Science, vol 11881. Springer, Cham. https://doi.org/10.1007/978-3-030-34223-4 2

[5] Podgorelec, B., Turkanovic, M. and Karakati ´ c, S., 2020. A Machine ˇ Learning-Based Method for Automated Blockchain Transaction Signing Including Personalized Anomaly Detection. Sensors, 20(1), p.147.

[6] Farrugia S, Ellul J, Azzopardi G. Detection of illicit accounts over the Ethereum blockchain. Expert Systems with Applications. 2020 Jul 15;150:113318.

[7] Pham, Thai, and Steven Lee. ”Anomaly detection in bitcoin network using unsupervised learning methods.” arXiv preprint arXiv:1611.03941 (2016).

[8] Monamo, Patrick, Vukosi Marivate, and Bheki Twala. ”Unsupervised learning for robust Bitcoin fraud detection.” 2016 Information Security for South Africa (ISSA). IEEE, 2016.

[9] Shi, Fa-Bin, et al. ”Anomaly detection in Bitcoin market via price return analysis.” PloS one 14.6 (2019): e0218341.

[10] Li, Ji, et al. ”A Survey on Blockchain Anomaly Detection Using Data Mining Techniques.” International Conference on Blockchain and Trustworthy Systems. Springer, Singapore, 2019.

[11] P. N. Sureshbhai, P. Bhattacharya and S. Tanwar, ”KaRuNa: A Blockchain-Based Sentiment Analysis Framework for Fraud Cryptocurrency Schemes,” 2020 IEEE International Conference on Communications Workshops (ICC Workshops), Dublin, Ireland, 2020, pp. 1-6, doi: 10.1109/ICCWorkshops49005.2020.9145151.

[12] Brenig, Christian, and Gunter M ¨ uller. ”Economic analysis of cryptocur- ¨ rency backed money laundering.” (2015).

[13] Lorenz, Joana, et al. ”Machine learning methods to detect money laundering in the Bitcoin blockchain in the presence of label scarcity.” arXiv preprint arXiv:2005.14635 (2020).

[14] Bartoletti, Massimo, Barbara Pes, and Sergio Serusi. ”Data mining for detecting Bitcoin Ponzi schemes.” 2018 Crypto Valley Conference on Blockchain Technology (CVCBT). IEEE, 2018.

8.2 GITHUB LINK

https://github.com/207R1A0522/Fruad_Detection_in_Blockchain