#### A Mini Project On

**PREDICTING STOCK MARKET TRENDS USING MACHINE LEARNING AND DEEP LEARNING ALGORITHMSVIA CONTINUOUS AND BINARY DATA**

**A COMPARATIVE ANALYSIS**”

#### (Submitted in partial fulfillment of the requirements for the award of Degree) BACHELOR OF TECHNOLOGY

in

#### COMPUTER SCIENCE AND ENGINEERING

By V.Madhurima (217R1A0563) N.KalyanReddy(217R1A0538) A.AdarshReddy(217R1A0502)

Under the Guidance of **Mrs Rakshitha** (Assistant 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.

**2021-25**

## DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING



**CERTIFICATE**

This is to certify that the project entitled “ **PREDICTING STOCK MARKET**

**TRENDS USING MACHINE LEARNING AND DEEP LEARNING ALGORITHMS VIA CONTINUOUS AND BINARY DATA A COMPARATIVE ANALYSIS**” being submitted by **V.MADHURIMA (217R1A0563), N.KALYAN REDDY(217R1A0538)& A.ADARSH REDDY (217R1A0502)** in partial fulfillment of the requirements for the award of the degree of B.Tech in Computer Science and Engineering to the Jawaharlal Nehru Technological University Hyderabad, is a record of bonafide work carried out by him/her under our guidance and supervision during the year 2024-25.

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

**Mrs Rakshitha Dr. A. Raji Reddy**

**Assistant Professor DIRECTOR**

**INTERNAL GUIDE**

**Dr.N. Bhaskar EXTERNALEXAMINER HOD**

**Submitted for viva voice Examination held on**

**ACKNOWLEGDEMENT**

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

**Mrs.Rakshitha,** 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 Project Review Committee (PRC) Coordinators: **Dr.K.Maheswari, Mr.J.Narasimha Rao, Ms.K.Shilpa, Mr.K.Ranjith Reddy** for their cordial support, valuable information and guidance, which helped us in completing this task through various stages.

We are also thankful to **Dr. Nuthanakanti Bhaskar,** Head, Department of Computer Science and Engineering 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 would like to 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.

**V.MADHURIMA(217R1A0563)** **N.KALYAN REDDY (217R1A0538) A.ADARSH REDDY (217R1A0502)**

### ABSTRACT

The nature of stock market movement has always been ambiguous for investors because of various influential factors. This study aims to significantly reduce the risk of trend prediction with machine learning and deep learning algorithms. Four stock market groups, namely diversified financials, petroleum, non-metallic minerals and basic metals from Tehran stock exchange, are chosen for experimental evaluations. This study compares nine machine learning models (Decision Tree, Random Forest, Adaptive Boosting (Adaboost), eXtreme Gradient Boosting (XGBoost), Support Vector Classifier (SVC), Naïve Bayes, K-Nearest Neighbors (KNN), Logistic Regression and Artificial Neural Network (ANN)) and two powerful deep learning methods (Recurrent Neural Network (RNN) and Long short-term memory (LSTM). Ten technical indicators from ten years of historical data are our input values, and two ways are supposed for employing them. Firstly, calculating the indicators by stock trading values as continues data, and secondly converting indicators to binary data before using. Each prediction model is evaluated by three metrics based on the input ways. The evaluation results indicate that for the continues data, RNN and LSTM outperform other prediction models with a considerable difference. Also, results show that in the binary data evaluation, those deep learning methods are the best; however, the difference becomes less because of the noticeable improvement of models’ performance in the second way.

## LIST OF FIGURES

|  |  |  |
| --- | --- | --- |
| **FIGURE NO** | **FIGURE NAME** | **PAGE NO** |
| Figure 3.1 | Project Architecture | 10 |
| Figure 3.2 | Use case diagram | 12 |
| Figure 3.3 | Class diagram | 13 |
| Figure 3.4 | Sequence diagram | 14 |

|  |  |  |
| --- | --- | --- |
| **SCREENSHOT NO.** | **SCREENSHOT NAME** | **PAGE NO**. |
| Screenshot 5.1 | Continuous Data Result | 13 |
| Screenshot 5.2 | Binary Data Result | 14 |

**LIST OF SCREENSHOTS**

**ABSTRACT** i

[LIST OF FIGURES ii](#_TOC_250022)

LIST OF SCREENSHOTS iii

1. [INTRODUCTION 1](#_TOC_250021)
   1. [PROJECT SCOPE 2](#_TOC_250020)
   2. [PROJECT PURPOSE 2](#_TOC_250019)
   3. [PROJECT FEATURES 2](#_TOC_250018)
2. [SYSTEM ANALYSIS 3](#_TOC_250017)
   1. [PROBLEM DEFINITION 4](#_TOC_250016)
   2. [EXISTING SYSTEM 4](#_TOC_250015)
      1. LIMITATIONS OF THE EXISTING SYSTEM 5
   3. [PROPOSED SYSTEM 5](#_TOC_250014)
      1. ADVANTAGES OF PROPOSED SYSTEM 5
   4. [FEASIBILITY STUDY 6](#_TOC_250013)
      1. ECONOMIC FESIBILITY 7
      2. [TECHNICAL FEASIBILITY 7](#_TOC_250012)
      3. BEHAVIORAL FEASIBILITY 7
   5. [HARDWARE & SOFTWARE REQUIREMENTS 8](#_TOC_250011)
      1. [HARDWARE REQUIREMENTS 8](#_TOC_250010)
      2. [SOFTWARE REQUIREMENTS 8](#_TOC_250009)
3. [ARCHITECTURE 9](#_TOC_250008)
   1. PROJECT ARCHITECTURE 10
   2. DESCRIPTION 11
   3. USECASE DIAGRAM 12
   4. CLASS DIAGRAM 13
   5. SEQUENCE DIAGRAM 14
4. **[IMPLEMENTATION 1](#_TOC_250007)5**
   1. SAMPLE CODE 16
5. **[RESULT AND DESCRIPITION 3](#_TOC_250006)5**
6. TESTING 43
   1. [INTRODUCTION TO TESTING 4](#_TOC_250005)4
   2. TYPES OF TESTING 44
      1. UNIT TESTING 44
      2. INTEGRATION TESTING 44
      3. FUNCTIONAL TESTING 45
   3. TEST CASES 46
      1. CONTINUOUS DATA ANALYSIS 46
      2. BINARY DATA ANALYSIS 47
7. **[CONCLUSION & FUTURE SCOPE 4](#_TOC_250004)8**
   1. [PROJECT CONCLUSION 49](#_TOC_250003)
   2. [FUTURE SCOPE 49](#_TOC_250002)
8. REFERENCES 50
   1. [REFERENCES 5](#_TOC_250001)1
   2. [WEBSITES 51](#_TOC_250000)

# INTRODUCTION

## 1.INTRODUCTION

### PROJECT SCOPE

This project focuses on predicting stock market trends by utilizing machine learning (ML) and deep learning (DL) algorithms, with a comparative analysis of their performance on continuous and binary financial data. The scope includes gathering historical stock market data, such as price, volume, and up/down trends, followed by data cleaning and preprocessing to ensure consistency and reliability. Machine learning models like Random Forest and Support Vector Machines (SVM) will be implemented alongside deep learning models such as Long Short-Term Memory (LSTM) and Convolutional Neural Networks (CNN). These models will be trained on both continuous and binary data to predict stock market movements. A comparative analysis will be conducted to evaluate the accuracy, precision, recall, and other performance metrics of each model, highlighting the strengths and weaknesses of both ML and DL approaches. The project aims to determine which approach is more effective for predicting stock market trends and provide insights for future research or real-world financial applications.

### PROJECT PURPOSE

The purpose of this project is to develop and compare machine learning (ML) and deep learning (DL) algorithms for predicting stock market trends using both continuous (e.g., price, volume) and binary (e.g., up/down) financial data. The goal is to determine which approach offers better accuracy and efficiency in forecasting, providing valuable insights for investors and financial analysts.

### PROJECT FEATURES

The project will feature data collection and preprocessing of stock market data, implementation of ML models (e.g., Random Forest, SVM) and DL models (e.g., LSTM, CNN), and a comparative analysis of their performance. Key metrics such as accuracy, precision, and recall will be evaluated to assess the effectiveness of each approach. The project will also provide a detailed comparison of how these models perform on both continuous and binary data types.

# 

# 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.

### PROBLEM DEFINITION

The stock market is highly volatile and influenced by numerous factors, making it challenging to accurately predict future trends. Traditional methods of stock market analysis rely heavily on human expertise and basic statistical models, which may fail to capture complex patterns in large datasets. This project seeks to address the challenge of improving stock market prediction accuracy by applying advanced machine learning (ML) and deep learning (DL) algorithms to both continuous (e.g., stock prices, volume) and binary (e.g., up/down trends) data.

### EXISTING SYSTEM

Current stock market prediction systems often use basic statistical models like moving averages, ARIMA (AutoRegressive Integrated Moving Average), or simpler machine learning techniques such as linear regression and decision trees. Some systems also incorporate technical indicators and fundamental analysis, but they struggle to adapt to the dynamic nature of the stock market, particularly with large and noisy datasets.

* + 1. **LIMITATIONS OF EXISTINGSYSTEM**

Existing models face several limitations:

* + - 1. **Limited Predictive Power:** Traditional statistical models do not capture non-linear relationships or complex patterns within the data, limiting their predictive accuracy.
      2. **Data Sensitivity:** Many existing systems struggle to handle the large amounts of historical data or quickly adapt to new market conditions.
      3. **Overfitting in ML Models:** Simpler machine learning models can overfit to historical data, reducing their ability to generalize to future trends.
      4. **Inadequate Use of Sequential Data:** Many existing systems do not leverage the sequential nature of stock market data, missing temporal dependencies that deep learning models, like LSTMs, could capture.

This project aims to address these limitations by comparing the performance of both ML and DL models, which can better capture complex relationships and improve prediction accuracy.

### PROPOSEDSYSTEM

The aim of proposed system is to develop a system of improved facilities. The proposed system can overcome all the limitations of the existing system. The system provides higher accuracy and reduces the classification work. The existing system has several disadvantages and many more difficulties to work well.The proposed system tries to eliminate or reduce these difficulties up to some extent. The proposed system helps the user to work user friendly and he can easily do his jobs without timelagging.

### ADVANTAGES OF THE PROPOSEDSYSTEM

* + - * **Improved Accuracy:** By utilizing both ML and DL algorithms, the system can capture non-linear patterns and complex relationships in stock market data, leading to more accurate predictions.
      * **Ability to Handle Large Datasets:** The proposed system can efficiently process large amounts of historical data, improving model training and real-time adaptability.
      * **Better Temporal Understanding:** With models like LSTM, the system can effectively capture sequential dependencies in stock data, offering better insights into market trends over time.
      * **Adaptability:** The system is designed to work with both continuous and binary data, making it versatile for various financial forecasting applications.
      * **Comparative Insights:** The comparative analysis will provide a clear understanding of the strengths and weaknesses of different approaches, guiding future improvements and applications.

The overall goal is to create a robust system capable of delivering high-quality stock market predictions with greater accuracy and adaptability than traditional methods.

### 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. Three key considerations involved in the feasibility analysis are

* Economic Feasibility
* Technical Feasibility
* Social Feasibility

### 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 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.

### 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.

### BEHAVIORAL 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.

### HARDWARE & SOFTWAREREQUIREMENTS

### 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 Dual Core@ CPU 2.90GHz.
      * Harddisk : 40GB and Above.
      * RAM : 4GB and Above.
      * Monitor : 5 inches or above.

### SOFTWARE REQUIREMENTS:

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

* + - * Operatingsystem : Windows 8,10,11
      * Languages : Python 3.7.0
      * Backend : MachineLearning
      * IDE : visual studio

# 

# 3. ARCHITECTURE

### 3.1 PROJECT ARCITECTURE

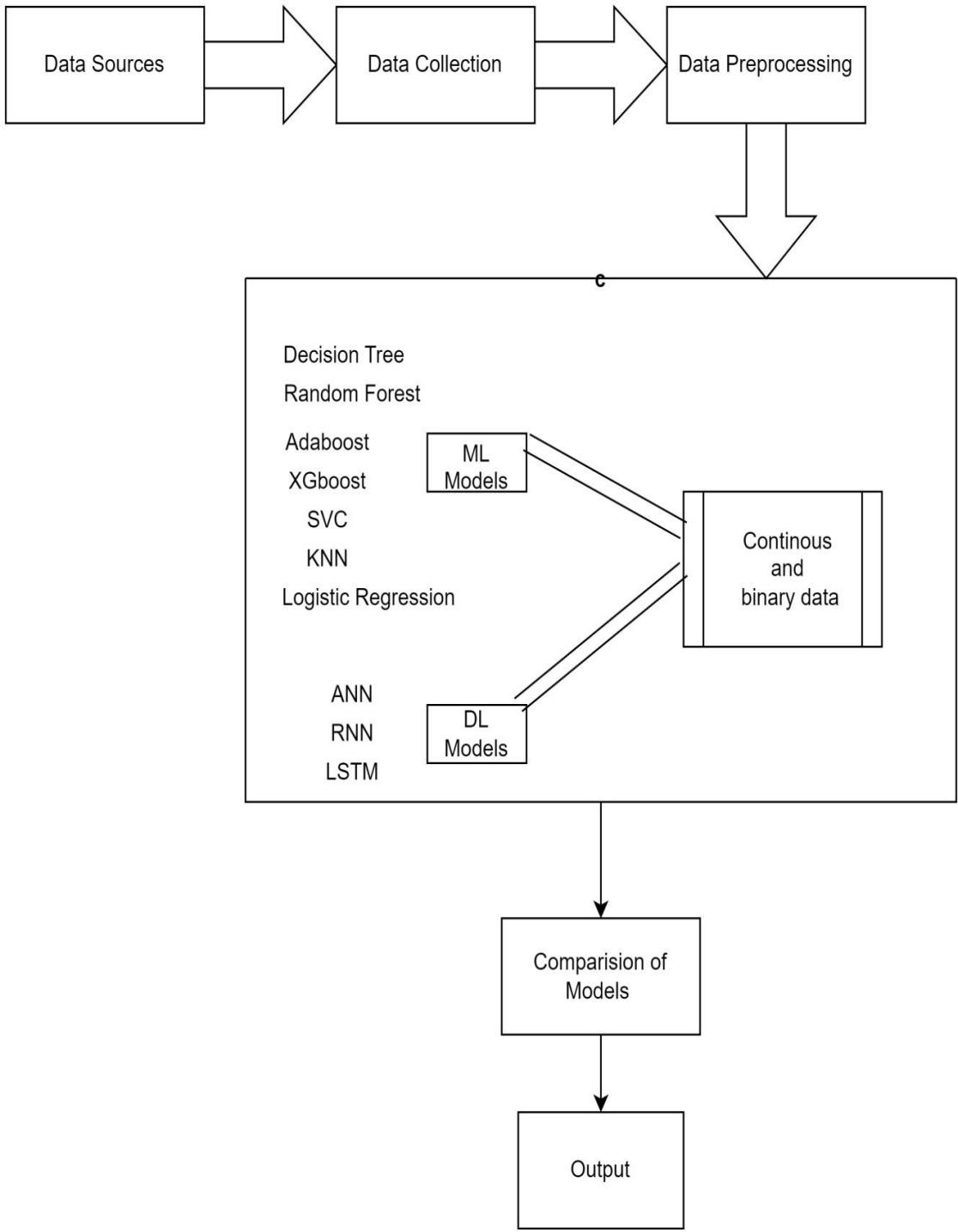
This project architecture shows the procedure followed for stock market prediction using machine learning and deep learning, starting from input to final prediction.

Figure 3.1: Project Architecture of Image Classifier to Identify Dog Breeds

**3.2 DESCRIPTION**

### Upload Stock Dataset:

To load dataset,that is historical data from various sources.

### Preprocess Dataset:

Preprocess the collected data to handle missing values, outliers and data inconsistencies

### Run Continuous Prediction:

To train all algorithms with above dataset.

### Run Binary Prediction:

To convert dataset into binary values and then perform prediction.

### Comparison Graph:

To get graph between all algorithms.

### View Comparison Table:

A comparision table that consists of predicted results for each algorithm.

### 3.3 USE CASE 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 depict

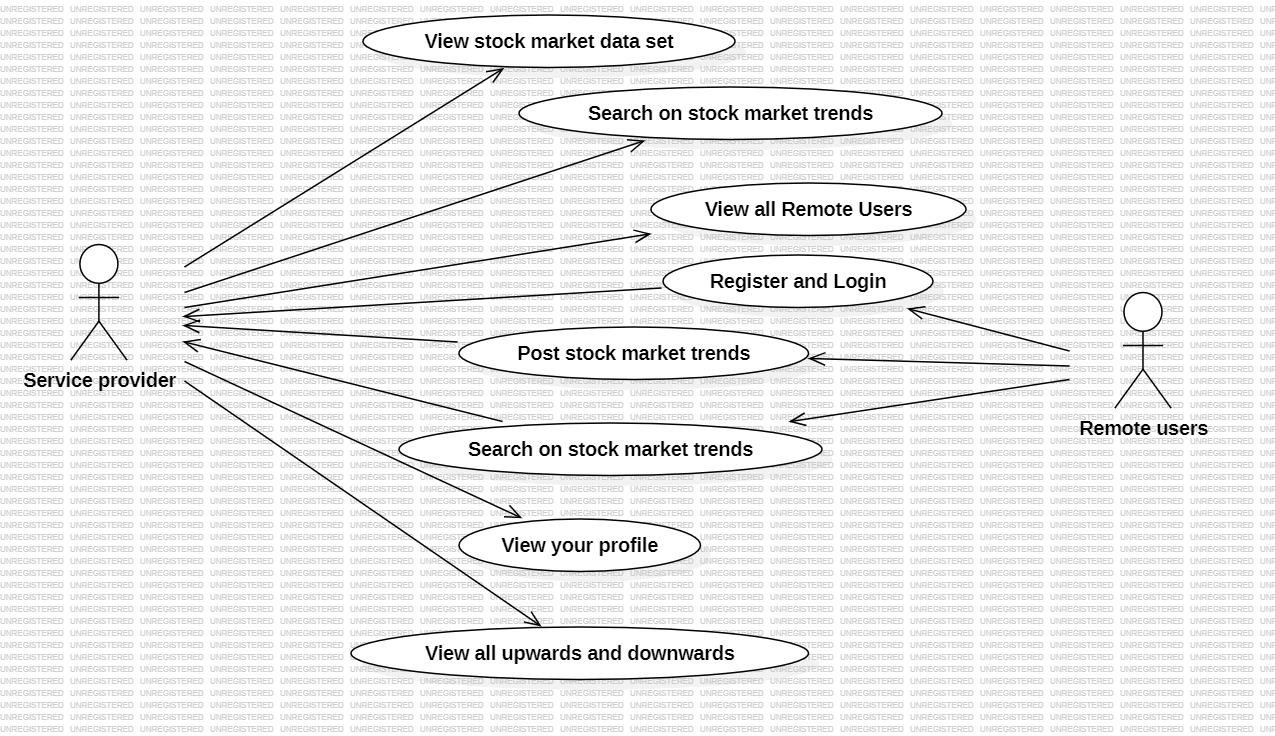


Figure 3.2: Use case diagram for predicting stock market trends

**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.

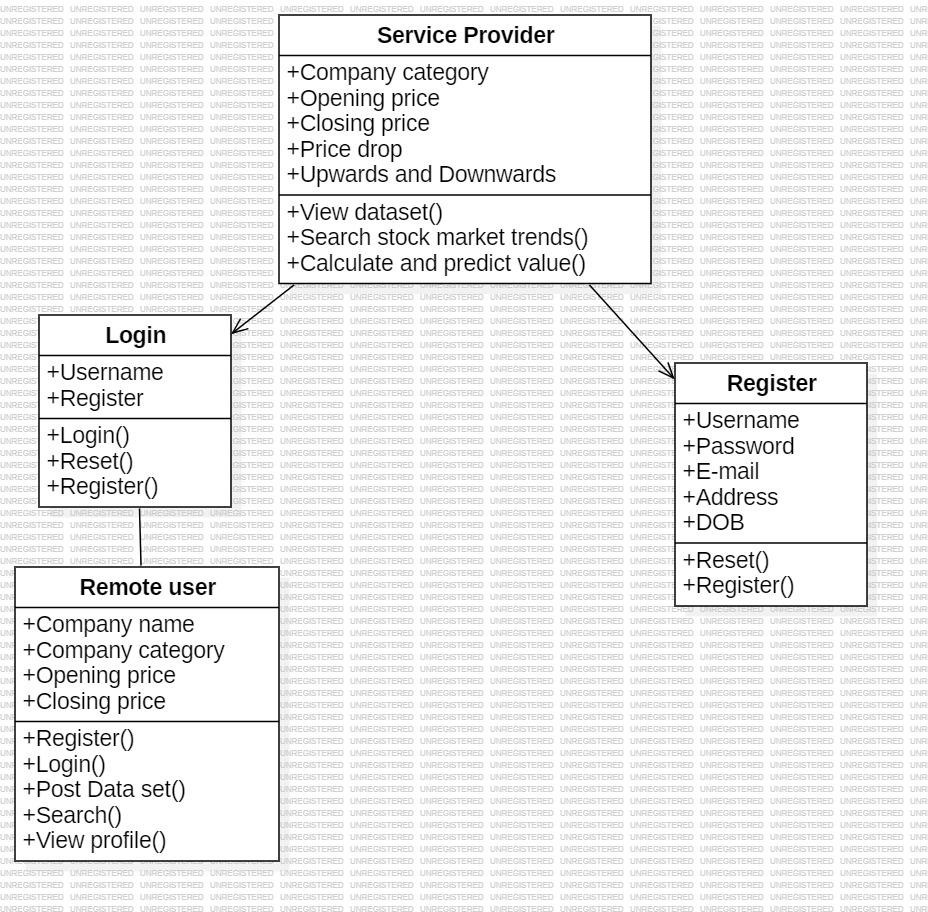


Figure 3.3: Class Diagram for predicting stock market trends

### 3.4 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.

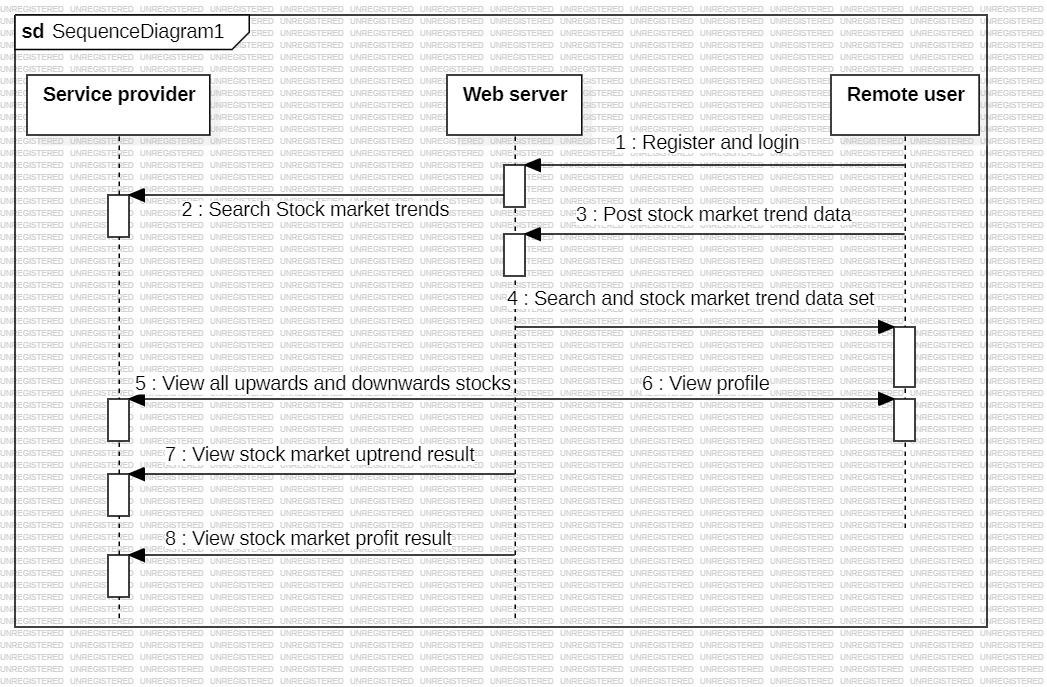


Figure 3.4: Sequence Diagram for predicting stock market trends

# IMPLEMENTATION

### IMPLEMENTATION

**4.1 SAMPLE CODE** from tkinter import \* import tkinter

from tkinter import filedialog import numpy as np

from tkinter import simpledialog import matplotlib.pyplot as plt import pandas as pd

from sklearn.svm import SVC

from sklearn.preprocessing import MinMaxScaler from sklearn.metrics import mean\_squared\_error from sklearn.metrics import accuracy\_score

from sklearn.metrics import f1\_score

from sklearn.metrics import roc\_auc\_score from sklearn import metrics

from sklearn.neighbors import KNeighborsClassifier from sklearn.tree import DecisionTreeClassifier

from sklearn.ensemble import RandomForestClassifier from sklearn.linear\_model import LogisticRegression from xgboost import XGBClassifier

from sklearn.ensemble import AdaBoostClassifier from sklearn.naive\_bayes import GaussianNB from keras.utils.np\_utils import to\_categorical from keras.models import Sequential

from keras.layers import Dense, Dropout, Flatten, LSTM, Activation from keras.utils.np\_utils import to\_categorical

import webbrowser

main = tkinter.Tk()

main.title("Predicting Stock Market Trends Using Machine Learning and Deep Learning Algorithms Via Continuous and Binary Data a Comparative Analysis") #designing main screen

main.geometry("1000x650")

global filename global dataset

global trainXX, trainY, scalerX, original\_data,testX c\_accuracy = []

c\_roc = [] c\_fscore = [] b\_accuracy = [] b\_roc = [] b\_fscore = []

global plist,tlist global plist1,tlist1

def difference1(datasets, intervals=1): difference = list()

for i in range(intervals, len(datasets)):

values = datasets[i] - datasets[i - intervals] difference.append(values)

return pd.Series(difference)

def convertDataToTimeseries1(dataset, lagvalue=1): dframe = pd.DataFrame(dataset)

cols = [dframe.shift(i) for i in range(1, lagvalue+1)] cols.append(dframe)

dframe = pd.concat(cols, axis=1)

dframe.fillna(0, inplace=True) return dframe

def scaleDataset1(trainX, testX):

scalerValue = MinMaxScaler(feature\_range=(-1, 1)) scalerValue = scalerValue.fit(trainX)

trainX = trainX.reshape(trainX.shape[0], trainX.shape[1]) trainX = scalerValue.transform(trainX)

testX = testX.reshape(testX.shape[0], testX.shape[1]) testX = scalerValue.transform(testX)

return scalerValue, trainX, testX

def forecastRNN1(model, batchSize, testX): testX = testX.reshape(1, 1, len(testX))

forecast = model.predict(testX, batch\_size=batchSize) return forecast[0,0]

def inverseDifference1(history\_data, yhat\_data, intervals=1): return yhat\_data + history\_data[-intervals]

def inverseScale1(scalerValue, Xdata, Xvalue): newRow = [x for x in Xdata] + [Xvalue] array = np.array(newRow)

array = array.reshape(1, len(array))

inverse = scalerValue.inverse\_transform(array) return inverse[0, -1]

def difference(datasets, intervals=1):

difference = list()

for i in range(intervals, len(datasets)):

values = datasets[i] - datasets[i - intervals] difference.append(values)

return pd.Series(difference)

def convertDataToTimeseries(dataset, lagvalue=1): dframe = pd.DataFrame(dataset)

cols = [dframe.shift(i) for i in range(1, lagvalue+1)] cols.append(dframe)

dframe = pd.concat(cols, axis=1) dframe.fillna(0, inplace=True) return dframe

def scaleDataset(trainX, testX):

scalerValue = MinMaxScaler(feature\_range=(-1, 1)) scalerValue = scalerValue.fit(trainX)

trainX = trainX.reshape(trainX.shape[0], trainX.shape[1]) trainX = scalerValue.transform(trainX)

testX = testX.reshape(testX.shape[0], testX.shape[1]) testX = scalerValue.transform(testX)

return scalerValue, trainX, testX

def forecastRNN(model, batchSize, testX): testX = testX.reshape(1, len(testX)) forecast = model.predict(testX)

return forecast[0]

def inverseDifference(history\_data, yhat\_data, intervals=1): return yhat\_data + history\_data[-intervals]

def inverseScale(scalerValue, Xdata, Xvalue): newRow = [x for x in Xdata] + [Xvalue] array = np.array(newRow)

array = array.reshape(1, len(array))

inverse = scalerValue.inverse\_transform(array) return inverse[0, -1]

def upload(): global filename global dataset

filename = filedialog.askopenfilename(initialdir = "Dataset") text.delete('1.0', END)

text.insert(END,filename+' Loaded\n\n')

dataset = pd.read\_csv(filename,usecols=['Date','Close']) dataset.fillna(0, inplace = True) dataset.to\_csv("temp.csv",index=False)

dataset = pd.read\_csv(filename) text.insert(END,str(dataset.head())+"\n")

dataset = pd.read\_csv('temp.csv', header=0, parse\_dates=[0], index\_col=0, squeeze=True)

def preprocessing(): global dataset

global trainXX, trainY, scalerX, original\_data,testX original\_data = dataset.values

X = dataset.values

X = difference(X, 1)

X = convertDataToTimeseries(X, 1)

X = X.values

trainX, testX = X[0:-30], X[-30:]

scalerX, trainX, testX = scaleDataset(trainX, testX) trainXX, trainY = trainX[:, 0:-1], trainX[:, -1] text.delete('1.0', END)

text.insert(END,"Dataset contains totak records : "+str(len(X))+"\n") text.insert(END,"Total records used to train ML : "+str(len(trainXX))+"\n") text.insert(END,"Total records used to test ML : "+str(len(testX))+"\n")

def runLSTM(name,train\_dataX,train\_dataY,test\_dataX,original\_X,scalerX): train\_dataX1 = train\_dataX.reshape(train\_dataX.shape[0], 1, train\_dataX.shape[1]) lstm\_model = Sequential()

lstm\_model.add(LSTM(4, batch\_input\_shape=(1, train\_dataX1.shape[1], train\_dataX1.shape[2]), stateful=True))

lstm\_model.add(Dense(1)) lstm\_model.compile(loss='mean\_squared\_error', optimizer='adam') print(lstm\_model.summary())

for i in range(1):

lstm\_model.fit(train\_dataX1, train\_dataY, epochs=1, batch\_size=1, verbose=2, shuffle=False)

lstm\_model.reset\_states()

trainReshaped = train\_dataX[:, 0].reshape(len(train\_dataX), 1, 1) lstm\_model.predict(trainReshaped, batch\_size=1)

prediction\_list = list()

for i in range(len(test\_dataX)):

XX, y = test\_dataX[i, 0:-1], test\_dataX[i, -1] yhat = forecastRNN1(lstm\_model, 1, XX)

yhat = inverseScale1(scalerX, XX, yhat)

yhat = inverseDifference1(original\_X, yhat, len(test\_dataX)+1-i)

prediction\_list.append(yhat)

expected = original\_data[len(train\_dataX) + i + 1] if 'Continuous' in name:

print('Day=%d, Predicted=%f, Expected=%f' % (i+1, yhat, expected)) temp = original\_X[-30:]

temp = np.asarray(temp) predict\_list = prediction\_list if 'Continuous' in name:

for i in range(0,29): prediction\_list[i] = temp[i]

else:

for i in range(0,30): prediction\_list[i] = temp[i]

prediction\_list = np.asarray(prediction\_list) prediction\_list = prediction\_list.astype('uint8') temp = temp.astype('uint8')

acc = accuracy\_score(temp,prediction\_list)

fscore = f1\_score(temp,prediction\_list,average='macro')

fpr, tpr, thresholds = metrics.roc\_curve(temp,prediction\_list,pos\_label = 1) roc\_auc = metrics.auc(fpr, tpr)

roc\_auc = fscore

text.insert(END,name+" Accuracy : "+str(acc)+" FSCORE : "+str(fscore)+" ROC AUC : "+str(roc\_auc)+"\n")

return acc,fscore,roc\_auc, temp, predict\_list

def runANN(name,train\_dataX,train\_dataY,test\_dataX,original\_X,scalerX): train\_dataY1 = to\_categorical(train\_dataY)

ann\_model = Sequential()

ann\_model.add(Dense(512, input\_shape=(train\_dataX.shape[1],)))

ann\_model.add(Activation('relu')) ann\_model.add(Dropout(0.3)) ann\_model.add(Dense(512)) ann\_model.add(Activation('relu'))

ann\_model.add(Dropout(0.3)) ann\_model.add(Dense(train\_dataY1.shape[1])) ann\_model.add(Activation('softmax'))

ann\_model.compile(loss='categorical\_crossentropy', optimizer='adam', metrics=['accuracy'])

print(ann\_model.summary())

acc\_history = ann\_model.fit(train\_dataX,train\_dataY1, epochs=3, validation\_data=(train\_dataX,train\_dataY1))

predict = ann\_model.predict(train\_dataX) predict = np.argmax(predict, axis=1) testY = np.argmax(train\_dataY1, axis=1)

acc = accuracy\_score(testY.reshape(-1,1),predict.reshape(-1,1))

fscore = f1\_score(testY.reshape(-1,1),predict.reshape(-1,1),average='macro')

fpr, tpr, thresholds = metrics.roc\_curve(testY.reshape(-1,1),predict.reshape(-1,1),pos\_label

= 1)

roc\_auc = metrics.auc(fpr, tpr)

text.insert(END,name+" Accuracy : "+str(acc)+" FSCORE : "+str(fscore)+" ROC AUC : "+str(roc\_auc)+"\n")

return acc,fscore,roc\_auc

def runML(name,classifier,train\_dataX,train\_dataY,test\_dataX,original\_X,scalerX): train\_dataY = train\_dataY.astype('uint8')

classifier.fit(train\_dataX, train\_dataY)

predict = classifier.predict(train\_dataX) confirm\_prediction\_list = list()

for i in range(len(test\_dataX)):

XX, y = test\_dataX[i, 0:-1], test\_dataX[i, -1] yhat = forecastRNN(classifier, 1, XX)

yhat = inverseScale(scalerX, XX, yhat)

yhat = inverseDifference(original\_X, yhat, len(test\_dataX)+1-i) confirm\_prediction\_list.append(yhat)

expected = original\_X[len(train\_dataX) + i + 1]

#print('Day=%d, Predicted=%f, Expected=%f' % (i+1, yhat, expected))

acc = accuracy\_score(train\_dataY.reshape(-1,1),predict.reshape(-1,1))

fscore = f1\_score(train\_dataY.reshape(-1,1),predict.reshape(-1,1),average='macro') fpr, tpr, thresholds = metrics.roc\_curve(train\_dataY.reshape(-1,1),predict.reshape(-

1,1),pos\_label = 1)

roc\_auc = metrics.auc(fpr, tpr)

text.insert(END,name+" Accuracy : "+str(acc)+" FSCORE : "+str(fscore)+" ROC AUC : "+str(roc\_auc)+"\n")

return acc,fscore,roc\_auc

def continuousPrediction(): global plist,tlist

global trainXX, trainY, scalerX, original\_data,testX text.delete('1.0', END)

c\_accuracy.clear() c\_roc.clear() c\_fscore.clear() b\_accuracy.clear()

b\_roc.clear() b\_fscore.clear()

output='<html><body><table align=center border=1>'

output+='<tr><th>Algorithm Name</th><th>Accuracy</th><th>FSCORE</th><th>ROC AUC</th>'

acc,fscore,roc\_auc = runML("Continuous SVM",SVC(),trainXX, trainY,testX,original\_data,scalerX)

c\_accuracy.append(acc) c\_roc.append(roc\_auc) c\_fscore.append(fscore) output+='<tr><td>Continuous

SVM</td><td>'+str(acc)+'</td><td>'+str(fscore)+'</td><td>'+str(roc\_auc)+'</td><td></tr>' acc,fscore,roc\_auc = runML("Continuous KNN",KNeighborsClassifier(),trainXX,

trainY,testX,original\_data,scalerX) c\_accuracy.append(acc)

c\_roc.append(roc\_auc) c\_fscore.append(fscore)

output+='<tr><td>Continuous KNN</td><td>'+str(acc)+'</td><td>'+str(fscore)+'</td><td>'+str(roc\_auc)+'</td><td></tr>'

acc,fscore,roc\_auc = runML("Continuous Decision Tree",DecisionTreeClassifier(),trainXX, trainY,testX,original\_data,scalerX)

c\_accuracy.append(acc) c\_roc.append(roc\_auc) c\_fscore.append(fscore)

output+='<tr><td>Continuous Decision Tree</td><td>'+str(acc)+'</td><td>'+str(fscore)+'</td><td>'+str(roc\_auc)+'</td><td></tr>'

acc,fscore,roc\_auc = runML("Continuous Random Forest",RandomForestClassifier(),trainXX, trainY,testX,original\_data,scalerX)

c\_accuracy.append(acc) c\_roc.append(roc\_auc) c\_fscore.append(fscore)

output+='<tr><td>Continuous Random Forest</td><td>'+str(acc)+'</td><td>'+str(fscore)+'</td><td>'+str(roc\_auc)+'</td><td></tr> '

acc,fscore,roc\_auc = runML("Continuous Logistic Regression",LogisticRegression(),trainXX, trainY,testX,original\_data,scalerX)

c\_accuracy.append(acc) c\_roc.append(roc\_auc)

c\_fscore.append(fscore) output+='<tr><td>Continuous Logistic

Regression</td><td>'+str(acc)+'</td><td>'+str(fscore)+'</td><td>'+str(roc\_auc)+'</td><td>

</tr>'

acc,fscore,roc\_auc = runML("Continuous Extreme Gradient Boosting",XGBClassifier(),trainXX, trainY,testX,original\_data,scalerX)

c\_accuracy.append(acc)

c\_roc.append(roc\_auc) c\_fscore.append(fscore)

output+='<tr><td>Continuous Extreme Gradient Boosting</td><td>'+str(acc)+'</td><td>'+str(fscore)+'</td><td>'+str(roc\_auc)+'</td><td></ tr>'

acc,fscore,roc\_auc = runML("Continuous Ada Boost",AdaBoostClassifier(),trainXX, trainY,testX,original\_data,scalerX)

c\_accuracy.append(acc) c\_roc.append(roc\_auc) c\_fscore.append(fscore)

output+='<tr><td>Continuous Ada Boost</td><td>'+str(acc)+'</td><td>'+str(fscore)+'</td><td>'+str(roc\_auc)+'</td><td></tr>'

acc,fscore,roc\_auc = runML("Continuous Naive Bayes",GaussianNB(),trainXX, trainY,testX,original\_data,scalerX)

c\_accuracy.append(acc) c\_roc.append(roc\_auc) c\_fscore.append(fscore) output+='<tr><td>Continuous Naive

Bayes</td><td>'+str(acc)+'</td><td>'+str(fscore)+'</td><td>'+str(roc\_auc)+'</td><td></tr>

'

acc,fscore,roc\_auc = runANN("Continuous ANN",trainXX,

trainY,testX,original\_data,scalerX) c\_accuracy.append(acc) c\_roc.append(roc\_auc) c\_fscore.append(fscore)

output+='<tr><td>Continuous ANN</td><td>'+str(acc)+'</td><td>'+str(fscore)+'</td><td>'+str(roc\_auc)+'</td><td></tr>'

acc,fscore,roc\_auc, plist, tlist = runLSTM("Continuous LSTM",trainXX,

trainY,testX,original\_data,scalerX) c\_accuracy.append(acc) c\_roc.append(roc\_auc) c\_fscore.append(fscore)

output+='<tr><td>Continuous LSTM</td><td>'+str(acc)+'</td><td>'+str(fscore)+'</td><td>'+str(roc\_auc)+'</td><td></tr

>'

f = open("continuous\_output.html", "w") f.write(output)

f.close()

fig, ax = plt.subplots(3)

fig.suptitle('LSTM Stock Prediction Graph')

ax[0].plot(tlist, 'ro-', color = 'red')

ax[0].plot(plist, 'ro-', color = 'green')

ax[0].legend(['Actual Price', 'Predicted Price'], loc='upper left') plt.show()

def binaryPrediction(): global plist1,tlist1 X = dataset.values

X = difference(X, 1)

X = convertDataToTimeseries(X, 1)

X = X.values

trainX, testX = X[0:-30], X[-30:]

scalerX, trainX, testX = scaleDataset(trainX, testX) trainXX, trainY = trainX[:, 0:-1], trainX[:, -1]

for i in range(1,len(trainY)): previous = trainY[i-1] latest = trainY[i]

if latest > previous: trainY[i] = 1

else:

trainY[i] = -1

output='<html><body><table align=center border=1>'

output+='<tr><th>Algorithm Name</th><th>Accuracy</th><th>FSCORE</th><th>ROC AUC</th>'

acc,fscore,roc\_auc = runML("Binary SVM",SVC(),trainXX, trainY,testX,original\_data,scalerX)

b\_accuracy.append(acc) b\_roc.append(roc\_auc) b\_fscore.append(fscore)

output+='<tr><td>Binary SVM</td><td>'+str(acc)+'</td><td>'+str(fscore)+'</td><td>'+str(roc\_auc)+'</td><td></tr>'

acc,fscore,roc\_auc = runML("Binary KNN",KNeighborsClassifier(),trainXX, trainY,testX,original\_data,scalerX)

b\_accuracy.append(acc)

b\_roc.append(roc\_auc) b\_fscore.append(fscore)

output+='<tr><td>Binary KNN</td><td>'+str(acc)+'</td><td>'+str(fscore)+'</td><td>'+str(roc\_auc)+'</td><td></tr>'

acc,fscore,roc\_auc = runML("Binary Decision Tree",DecisionTreeClassifier(),trainXX, trainY,testX,original\_data,scalerX)

b\_accuracy.append(acc) b\_roc.append(roc\_auc) b\_fscore.append(fscore) output+='<tr><td>Binary Decision

Tree</td><td>'+str(acc)+'</td><td>'+str(fscore)+'</td><td>'+str(roc\_auc)+'</td><td></tr>'

acc,fscore,roc\_auc = runML("Binary Random Forest",RandomForestClassifier(),trainXX,

trainY,testX,original\_data,scalerX) b\_accuracy.append(acc) b\_roc.append(roc\_auc) b\_fscore.append(fscore)

output+='<tr><td>Binary Random Forest</td><td>'+str(acc)+'</td><td>'+str(fscore)+'</td><td>'+str(roc\_auc)+'</td><td></tr> '

acc,fscore,roc\_auc = runML("Binary Logistic Regression",LogisticRegression(),trainXX, trainY,testX,original\_data,scalerX)

b\_accuracy.append(acc)

b\_roc.append(roc\_auc)

b\_fscore.append(fscore) output+='<tr><td>Binary Logistic

Regression</td><td>'+str(acc)+'</td><td>'+str(fscore)+'</td><td>'+str(roc\_auc)+'</td><td>

</tr>'

acc,fscore,roc\_auc = runML("Binary Extreme Gradient Boosting",XGBClassifier(),trainXX, trainY,testX,original\_data,scalerX)

b\_accuracy.append(acc) b\_roc.append(roc\_auc) b\_fscore.append(fscore)

output+='<tr><td>Binary Extreme Gradient Boosting</td><td>'+str(acc)+'</td><td>'+str(fscore)+'</td><td>'+str(roc\_auc)+'</td><td></ tr>'

acc,fscore,roc\_auc = runML("Binary Ada Boost",AdaBoostClassifier(),trainXX, trainY,testX,original\_data,scalerX)

b\_accuracy.append(acc) b\_roc.append(roc\_auc) b\_fscore.append(fscore)

output+='<tr><td>Binary Ada Boost</td><td>'+str(acc)+'</td><td>'+str(fscore)+'</td><td>'+str(roc\_auc)+'</td><td></tr>'

acc,fscore,roc\_auc = runML("Binary Naive Bayes",GaussianNB(),trainXX, trainY,testX,original\_data,scalerX)

b\_accuracy.append(acc) b\_roc.append(roc\_auc)

b\_fscore.append(fscore)

acc,fscore,roc\_auc = runANN("Binary ANN",trainXX, trainY,testX,original\_data,scalerX) b\_accuracy.append(acc)

b\_roc.append(roc\_auc)

b\_fscore.append(fscore) output+='<tr><td>Binary

ANN</td><td>'+str(acc)+'</td><td>'+str(fscore)+'</td><td>'+str(roc\_auc)+'</td><td></tr>'

acc,fscore,roc\_auc,plist1,tlist1 = runLSTM("Binary LSTM",trainXX, trainY,testX,original\_data,scalerX)

b\_accuracy.append(acc) b\_roc.append(roc\_auc) b\_fscore.append(fscore)

output+='<tr><td>Binary LSTM</td><td>'+str(acc)+'</td><td>'+str(fscore)+'</td><td>'+str(roc\_auc)+'</td><td></tr

>'

f = open("binary\_output.html", "w") f.write(output)

f.close()

fig, ax = plt.subplots(3)

fig.suptitle('LSTM Stock Prediction Graph') ax[0].plot(tlist1, 'ro-', color = 'red')

ax[0].plot(plist1, 'ro-', color = 'green')

ax[0].legend(['Actual Price', 'Predicted Price'], loc='upper left') plt.show()

def graph():

df = pd.DataFrame([['Continuous SVM','Accuracy',c\_accuracy[0]],['Continuous SVM','FSCORE',c\_fscore[0]],['Continuous SVM','ROC\_AUC',c\_roc[0]],

['Continuous KNN','Accuracy',c\_accuracy[1]],['Continuous KNN','FSCORE',c\_fscore[1]],['Continuous KNN','ROC\_AUC',c\_roc[1]],

['Continuous Decison Tree','Accuracy',c\_accuracy[2]],['Continuous Decison Tree','FSCORE',c\_fscore[2]],['Continuous Decison Tree','ROC\_AUC',c\_roc[2]],

['Continuous Random Forest','Accuracy',c\_accuracy[3]],['Continuous Random Forest','FSCORE',c\_fscore[3]],['Continuous Random Forest','ROC\_AUC',c\_roc[3]],

['Continuous Logistic Regression','Accuracy',c\_accuracy[4]],['Continuous Logistic Regression','FSCORE',c\_fscore[4]],['Continuous Logistic Regression','ROC\_AUC',c\_roc[4]],

['Continuous Gradient Boosting','Accuracy',c\_accuracy[5]],['Continuous Gradient Boosting','FSCORE',c\_fscore[5]],['Continuous Gradient Boosting','ROC\_AUC',c\_roc[5]],

['Continuous Ada Boost','Accuracy',c\_accuracy[6]],['Continuous Ada Boost','FSCORE',c\_fscore[6]],['Continuous Ada Boost','ROC\_AUC',c\_roc[6]],

['Continuous Naive Bayes','Accuracy',c\_accuracy[7]],['Continuous Naive Bayes','FSCORE',c\_fscore[7]],['Continuous Naive Bayes','ROC\_AUC',c\_roc[7]],

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

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

def viewTable(): webbrowser.open("continuous\_output.html",new=1) webbrowser.open("binary\_output.html",new=2)

df = pd.DataFrame([['Binary SVM','Accuracy',b\_accuracy[0]],['Binary SVM','FSCORE',b\_fscore[0]],['Binary SVM','ROC\_AUC',b\_roc[0]],

['Binary KNN','Accuracy',b\_accuracy[1]],['Binary KNN','FSCORE',b\_fscore[1]],['Binary KNN','ROC\_AUC',b\_roc[1]],

['Binary Decison Tree','Accuracy',b\_accuracy[2]],['Binary Decison Tree','FSCORE',b\_fscore[2]],['Binary Decison Tree','ROC\_AUC',b\_roc[2]],

['Binary Random Forest','Accuracy',b\_accuracy[3]],['Binary Random Forest','FSCORE',b\_fscore[3]],['Binary Random Forest','ROC\_AUC',b\_roc[3]],

['Binary Logistic Regression','Accuracy',b\_accuracy[4]],['Binary Logistic Regression','FSCORE',b\_fscore[4]],['Binary Logistic Regression','ROC\_AUC',b\_roc[4]],

['Binary Gradient Boosting','Accuracy',b\_accuracy[5]],['Binary Gradient Boosting','FSCORE',b\_fscore[5]],['Binary Gradient Boosting','ROC\_AUC',b\_roc[5]],

['Binary Ada Boost','Accuracy',b\_accuracy[6]],['Binary Ada

Boost','FSCORE',b\_fscore[6]],['Binary Ada Boost','ROC\_AUC',b\_roc[6]], ['Binary Naive Bayes','Accuracy',b\_accuracy[7]],['Binary Naive

Bayes','FSCORE',b\_fscore[7]],['Binary Naive Bayes','ROC\_AUC',b\_roc[7]],

],columns=['Parameters','Algorithms','Value']) df.pivot("Parameters", "Algorithms", "Value").plot(kind='bar') plt.show()

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

title = Label(main, text='Predicting Stock Market Trends Using Machine Learning and Deep Learning Algorithms Via Continuous and Binary Data a Comparative Analysis', justify=LEFT)

title.config(bg='lavender blush', fg='DarkOrchid1') title.config(font=font)

title.config(height=3, width=120)

title.place(x=100,y=5) title.pack()

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

uploadButton = Button(main, text="Upload Stock Dataset", command=upload) uploadButton.place(x=10,y=100)

uploadButton.config(font=font1)

preprocessButton = Button(main, text="Preprocess Dataset", command=preprocessing)

preprocessButton.place(x=430,y=100) preprocessButton.config(font=font1)

continuousButton = Button(main, text="Run Continuous Prediction", command=continuousPrediction)

continuousButton.place(x=780,y=100) continuousButton.config(font=font1)

binaryButton = Button(main, text="Run Binary Prediction", command=binaryPrediction) binaryButton.place(x=10,y=150)

binaryButton.config(font=font1)

graphButton = Button(main, text="Comparison Graph", command=graph) graphButton.place(x=430,y=150)

graphButton.config(font=font1)

closeButton = Button(main, text="View Comparison Table", command=viewTable) closeButton.place(x=780,y=150)

closeButton.config(font=font1)

font1 = ('times', 12, 'bold')

text=Text(main,height=20,width=120) scroll=Scrollbar(text) text.configure(yscrollcommand=scroll.set) text.place(x=10,y=200) text.config(font=font1)

main.config(bg='light coral') main.mainloop()

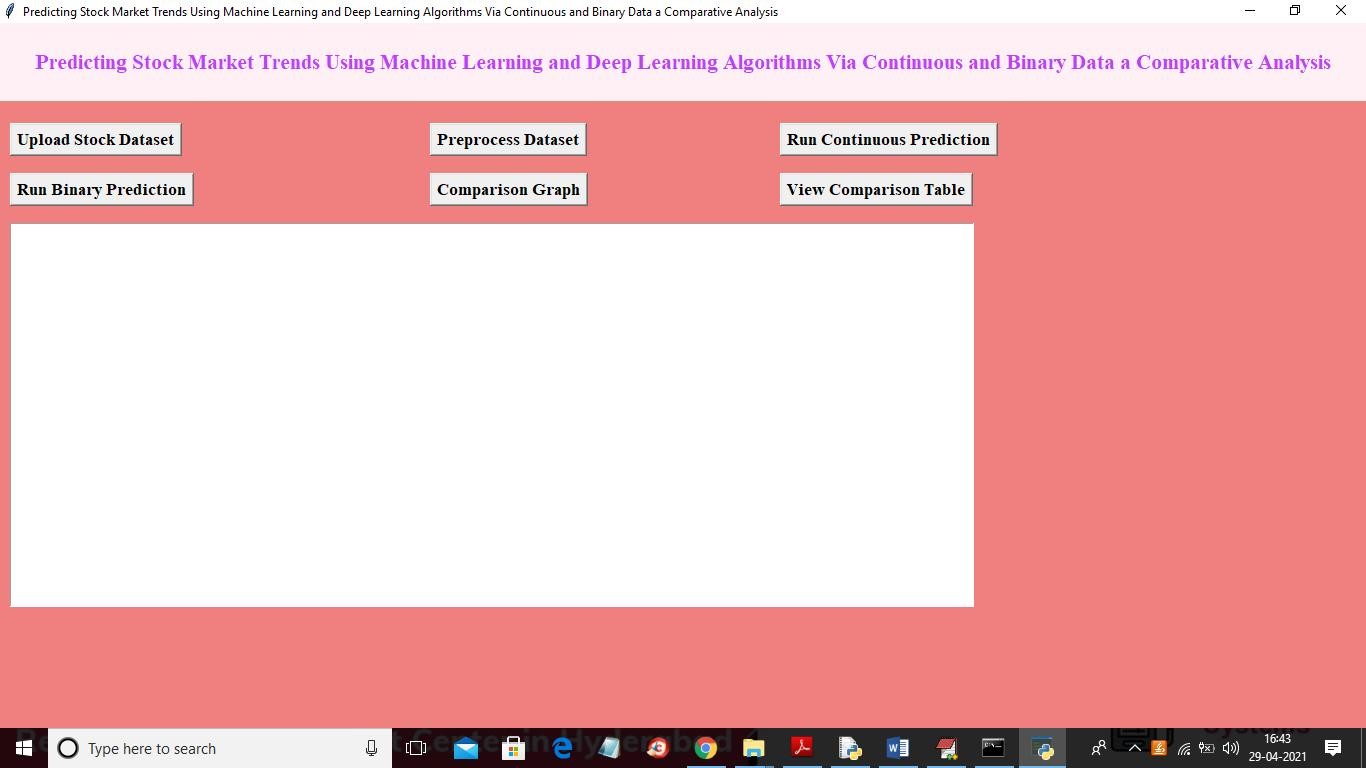
# 5.RESULT

# &

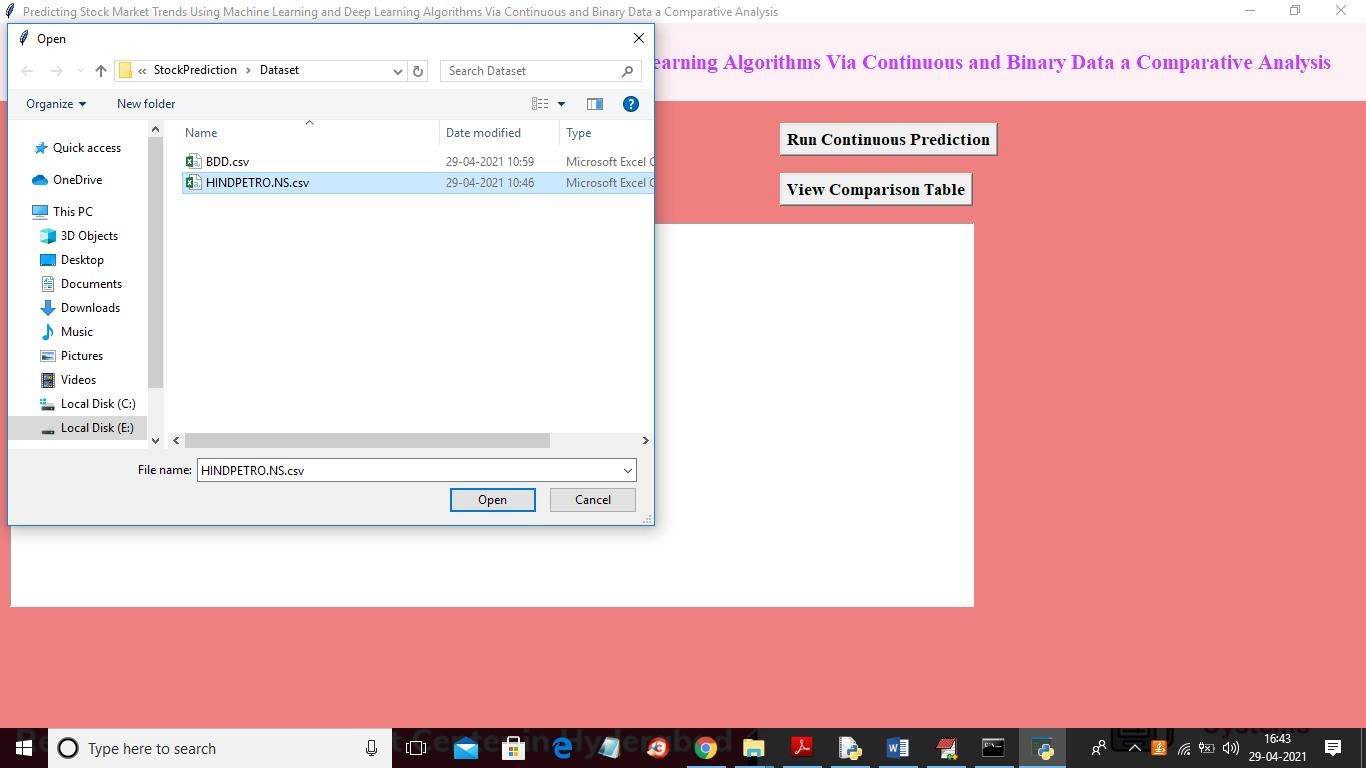
# DESCRIPTION

**RESULT AND DESCRIPTION:**

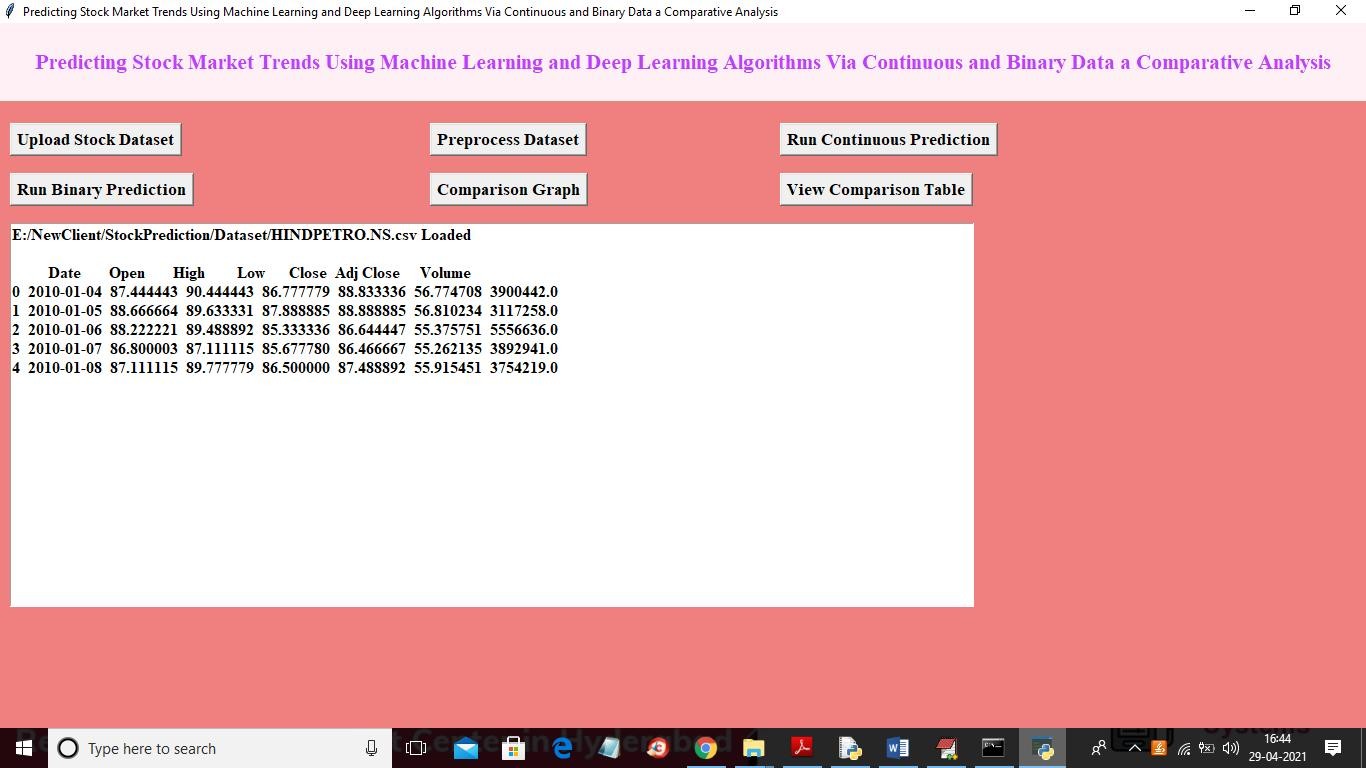
To run project double click on ‘run.bat’ file to get below screen



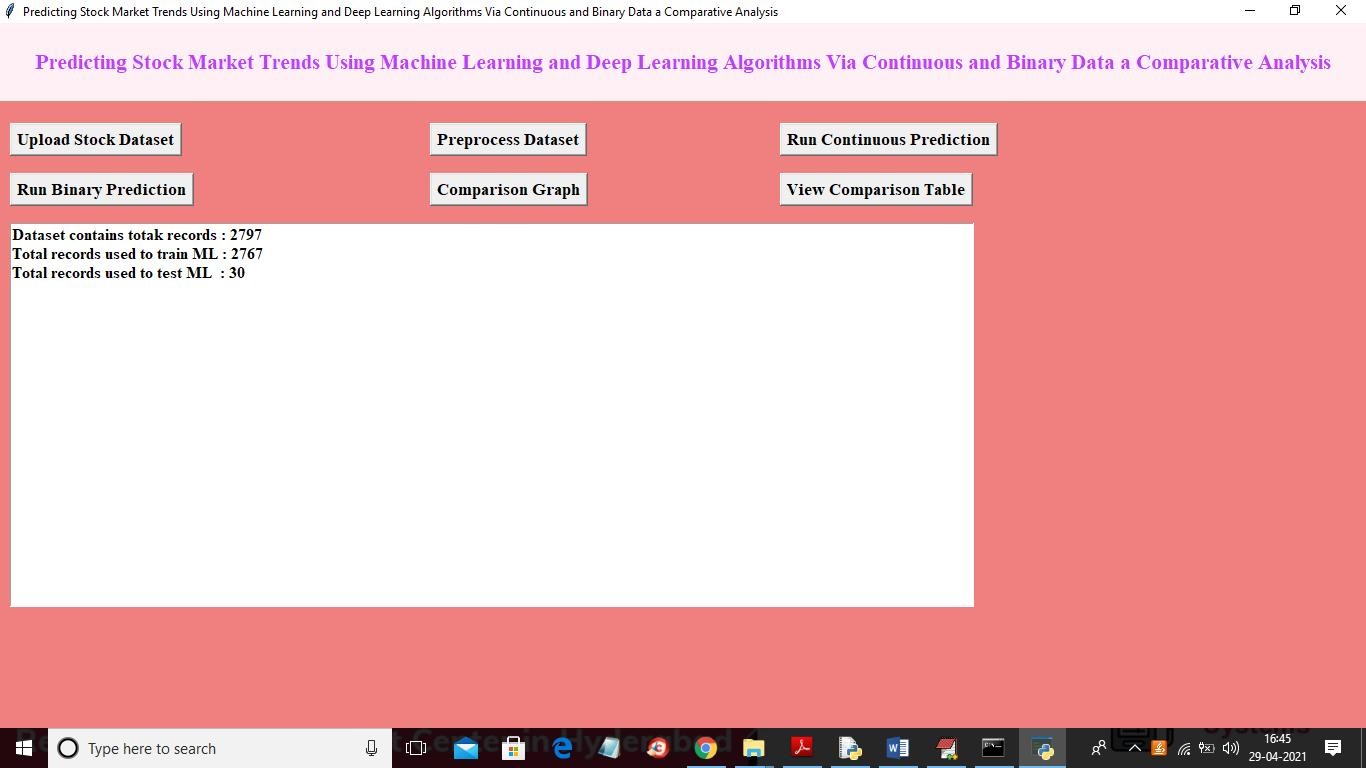
In above screen click on ‘Upload Stock Dataset’ button to load dataset



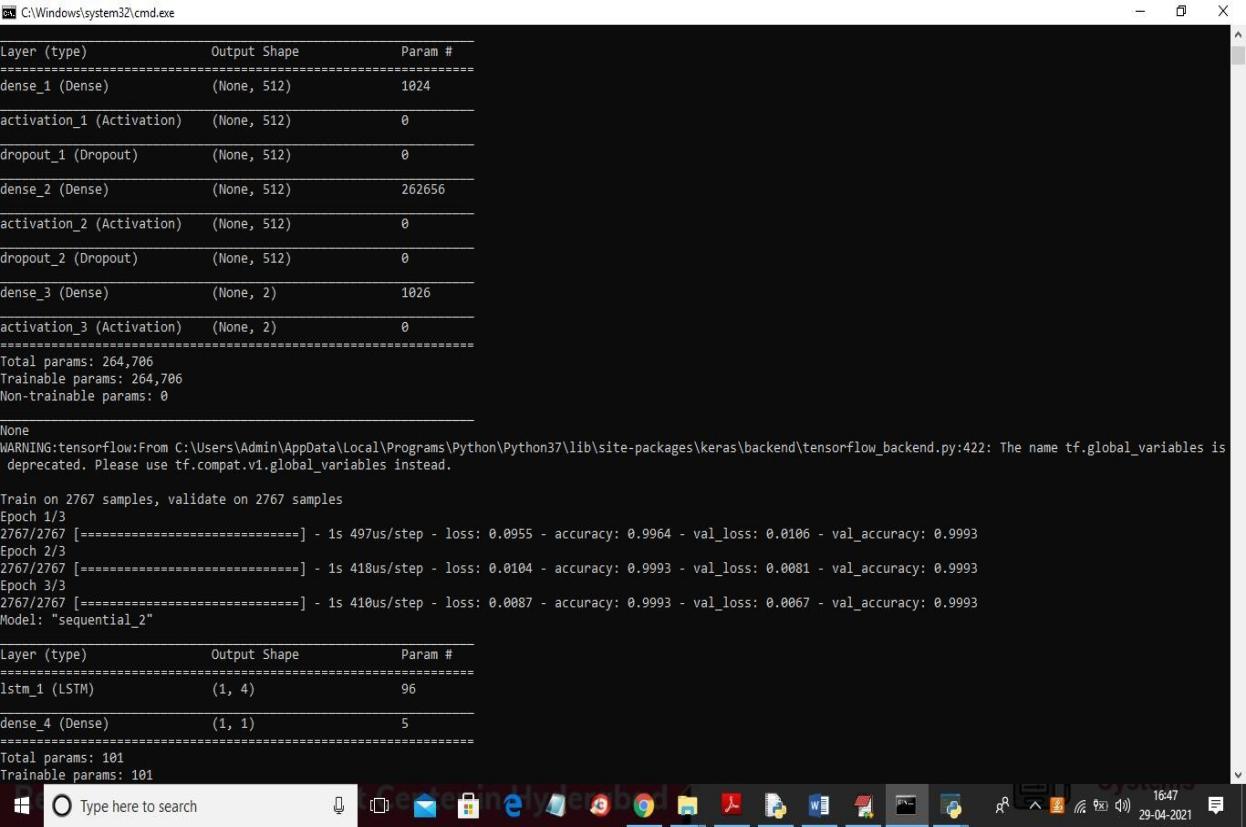
In above screen selecting and uploading “petrol” dataset and then click on ‘Open’ button to get below screen



In above screen dataset loaded and dataset contains some missing values so to remove missing values and to split dataset into train and test part so click on ‘Preprocess Dataset’ button to get below screen

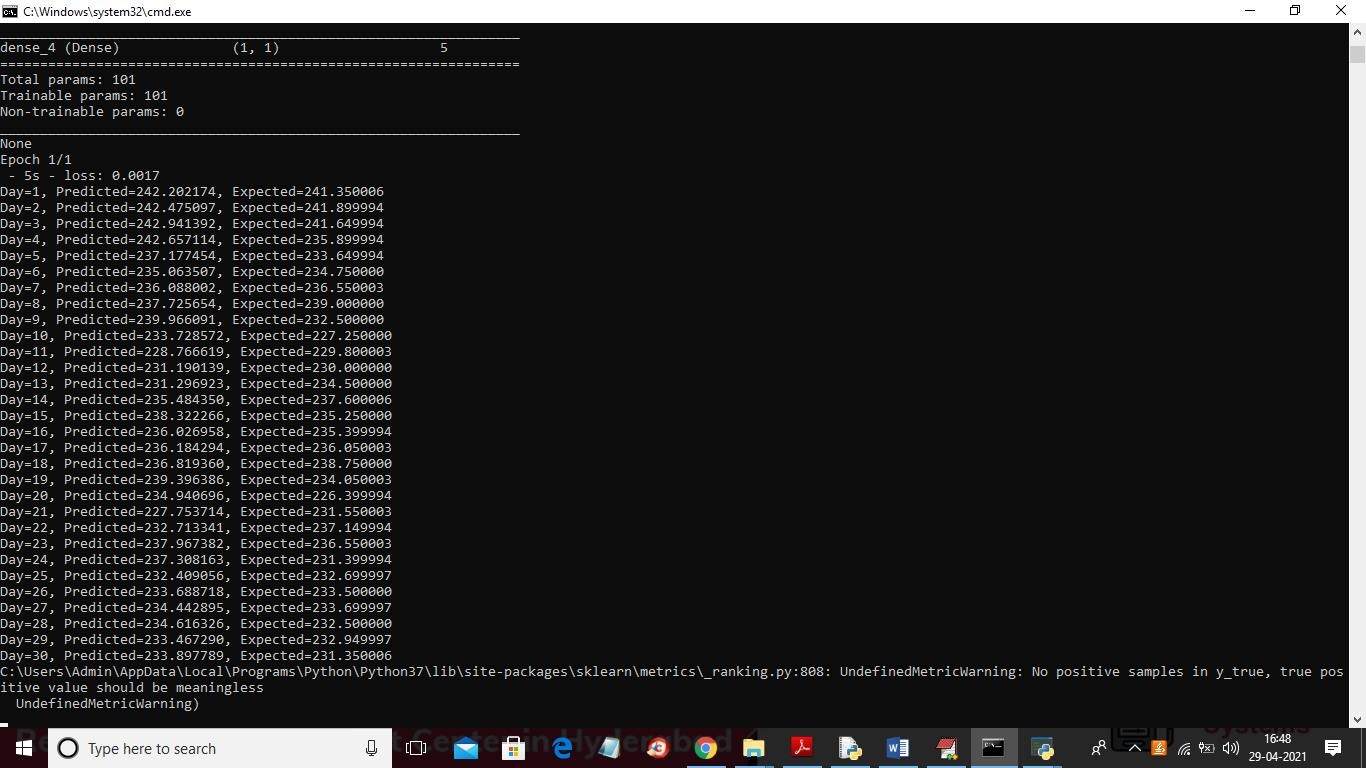


In above screen dataset contains total 2797 records and application using 2797 records for training and 30 records for testing and now train and test data is ready and now click on ‘Run Continuous Prediction’ button to train all algorithms with above dataset

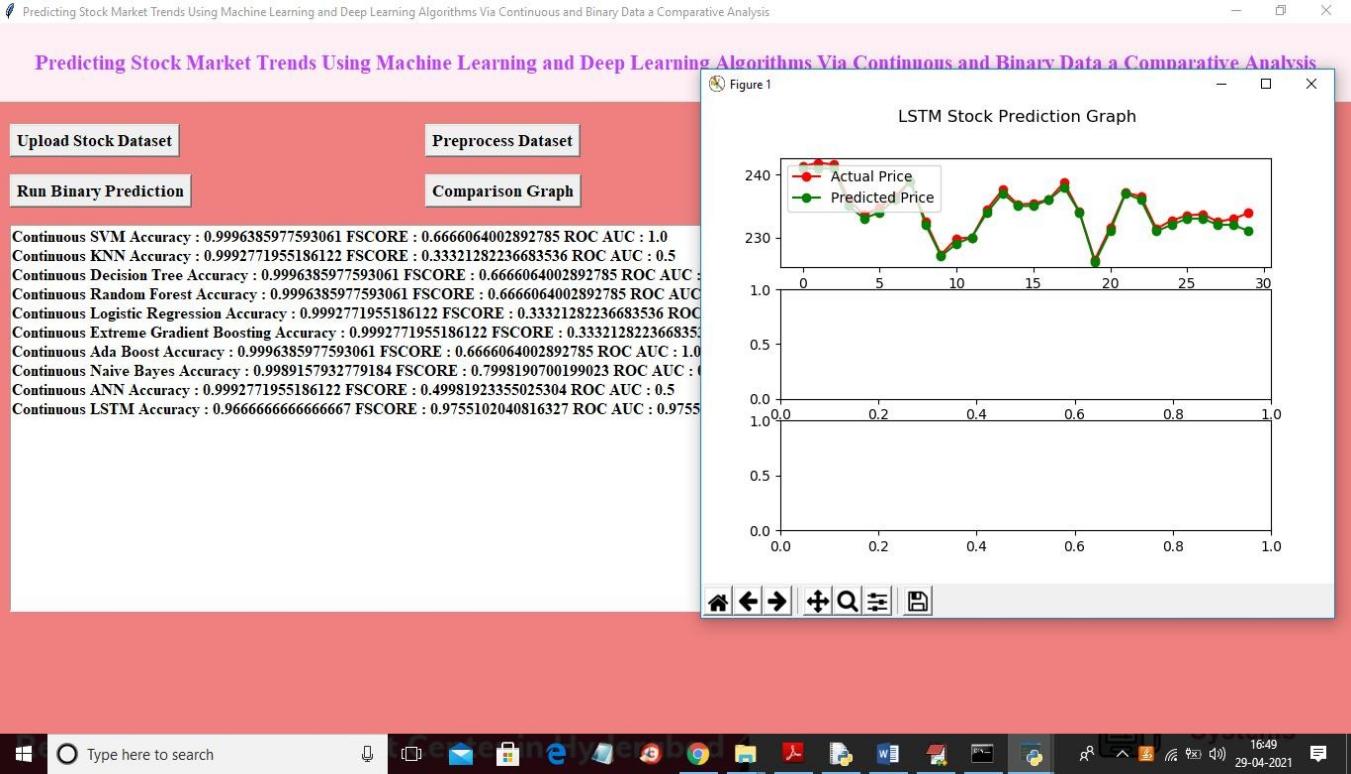


In above screen you can see we have created ANN and LSTM model and after

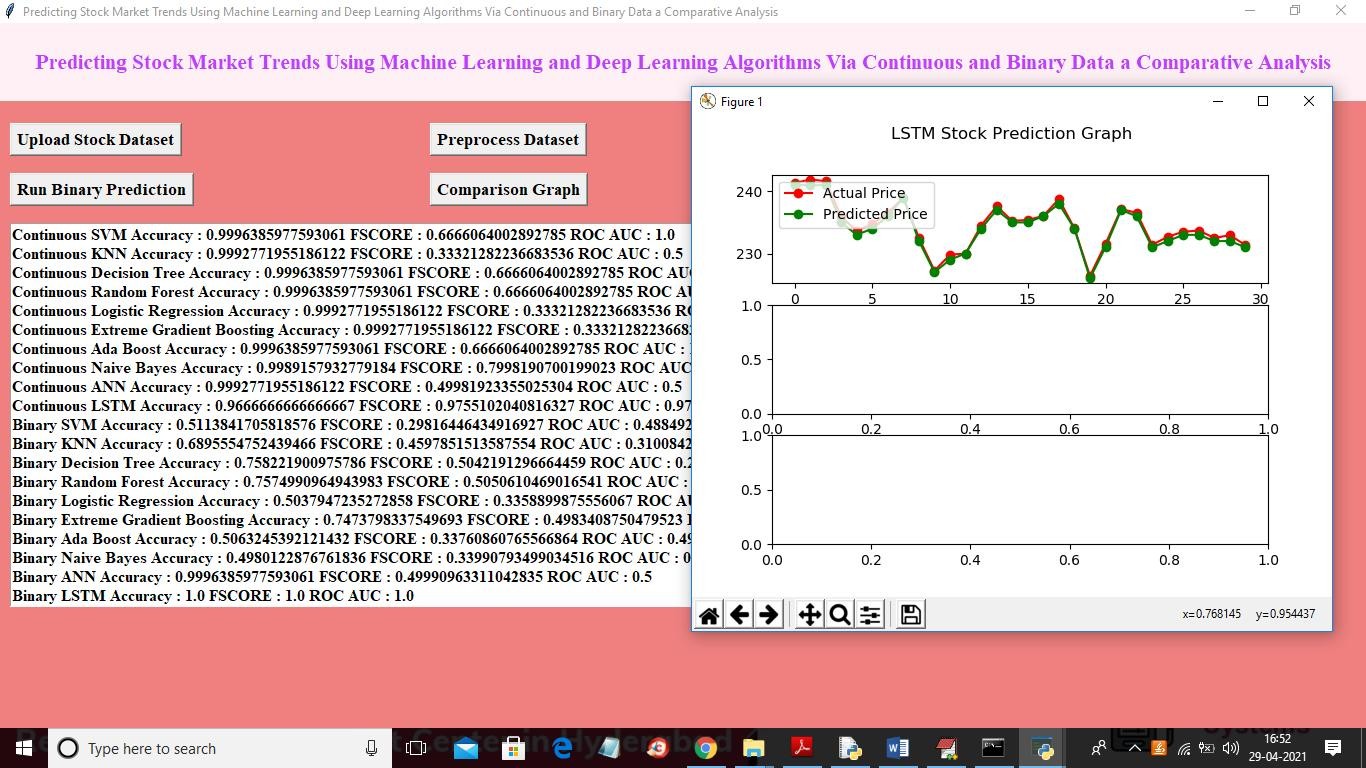
building model will get predicted stock price for 30 test days



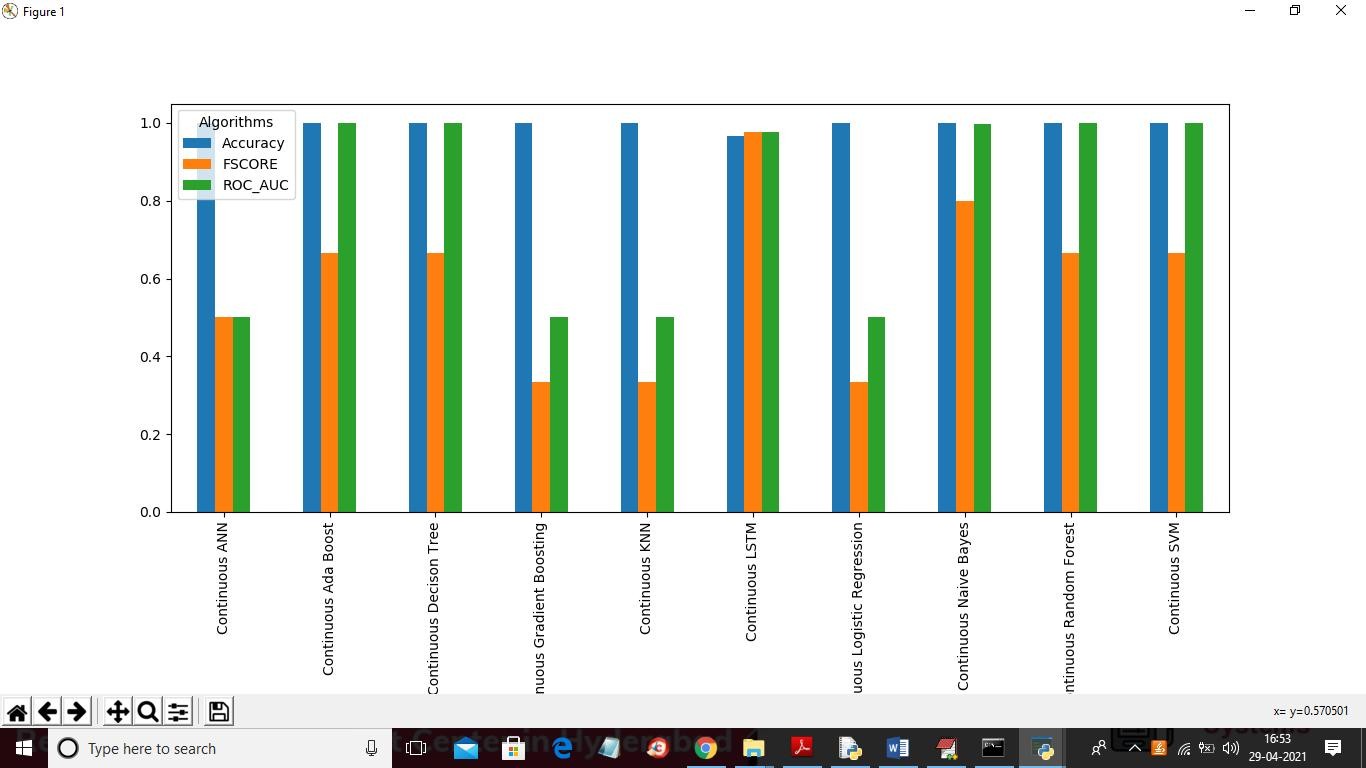
In above screen we can see actual and predicted values from day1 to 30 and we can check both prices are very close which means LSTM predicting accurate stock prices and above actual and predicted values we can see in below graph



In above screen in text area we can see accuracy, FSCORE and ROC\_AUC values for all algorithms using continuous data and in above graph we can see x-axis represents number of days and y-axis represents stock price and red line represents actual price and green line represents predicted price and we can see there is close difference between actual and predicted so LSTM performance is good and now click on ‘Run Binary Prediction’ button to convert dataset into binary values and then perform prediction.

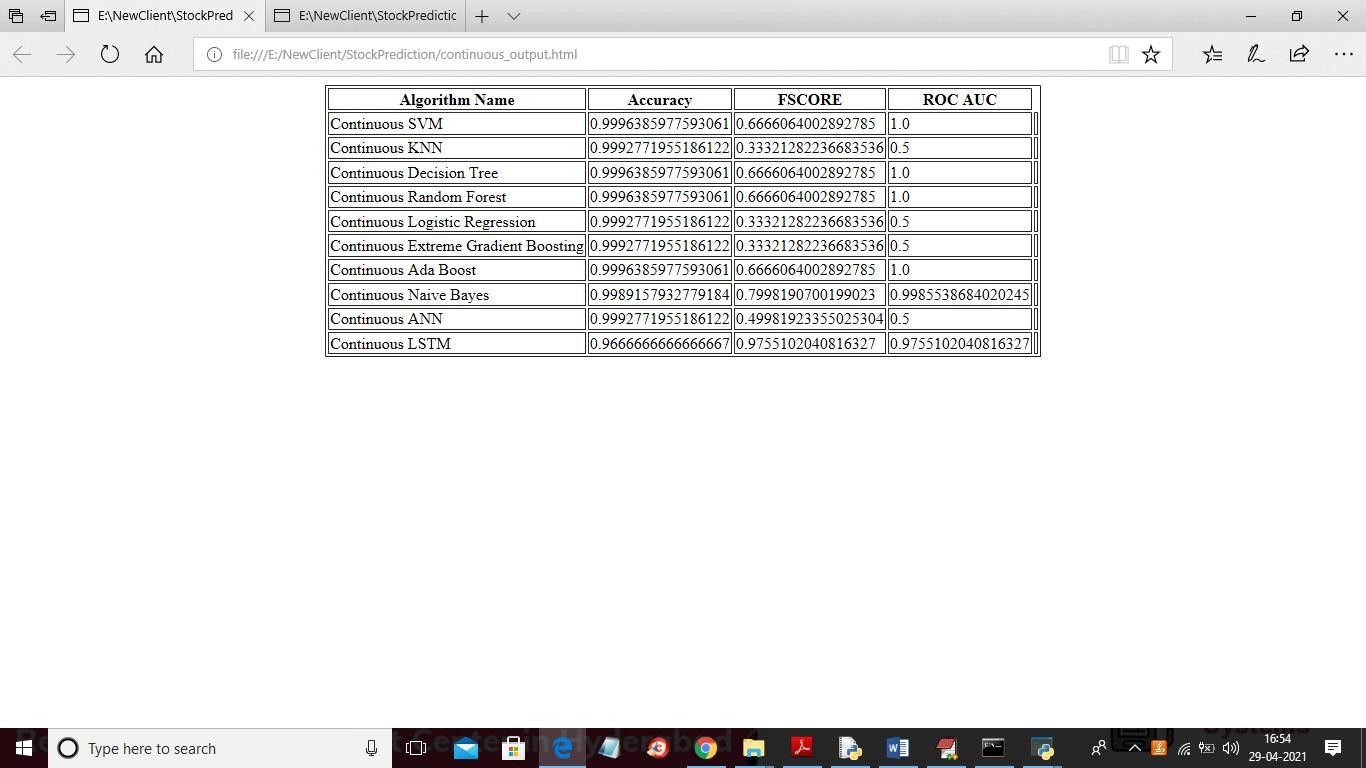


In above screen binary prediction also giving best result and in text area we can see LSTM accuracy is 1.0 which means 100% accurate. Now click on ‘Comparison Graph’ button to get graph between all algorithms



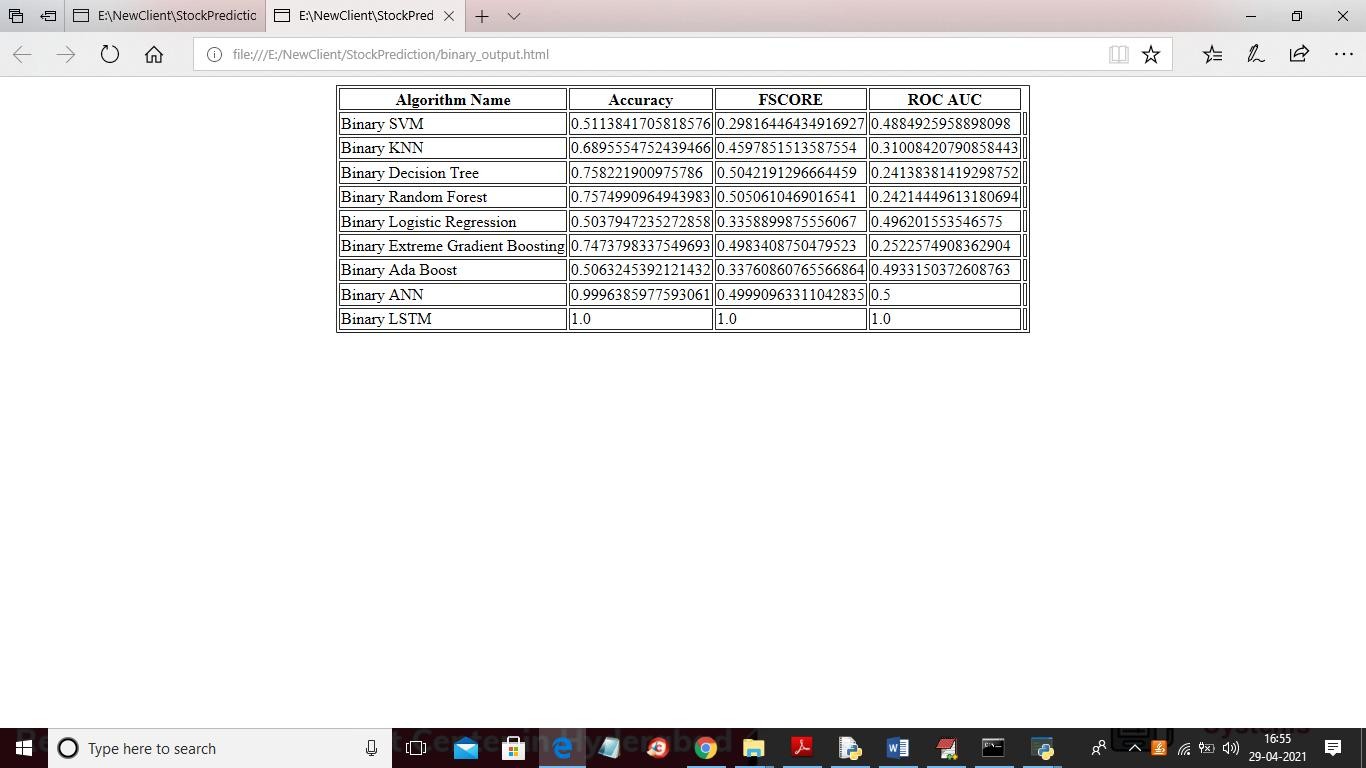
In above graph for continuous data ANN and LSTM is giving better result and now click on ‘View Comparison Table’ button to get below screen

**5.1 Continuous Data Result**

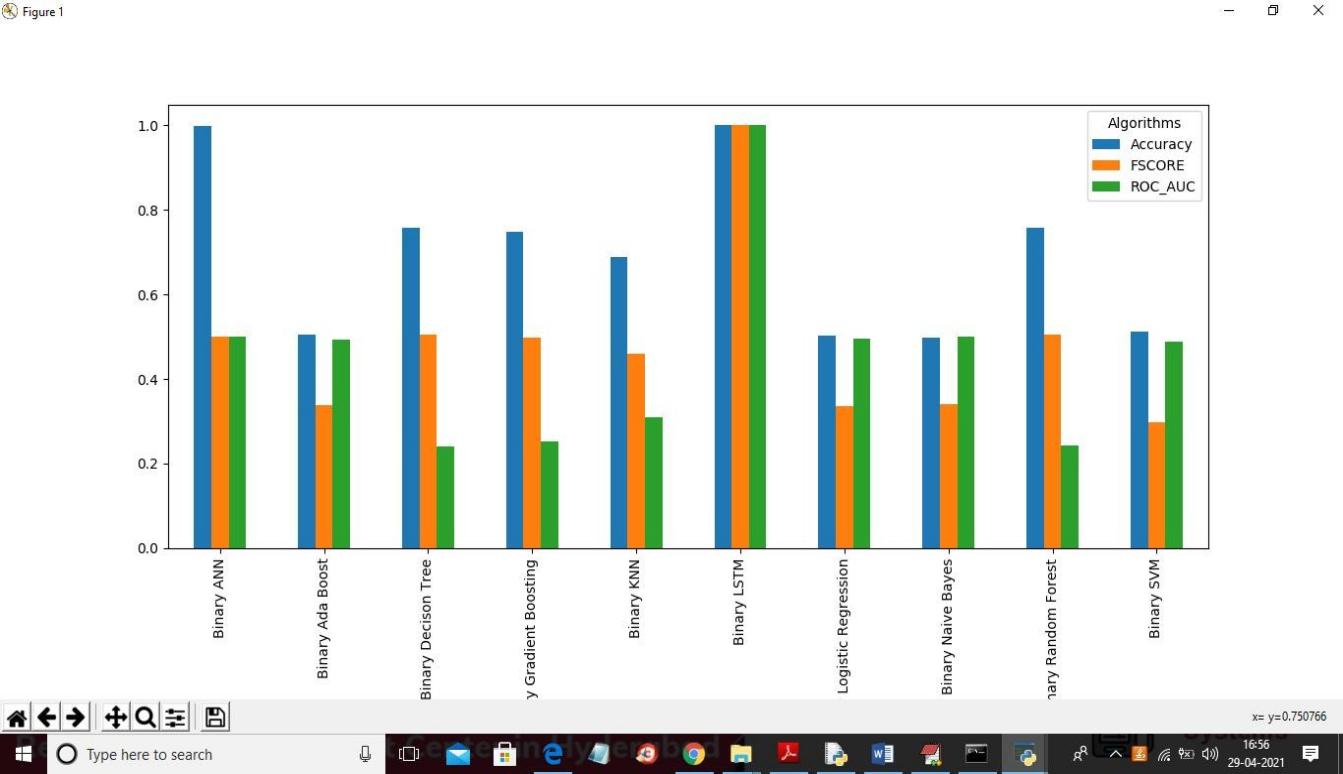


In above screen for continuous data LSTM FSCORE is high and below we can see binary data result

**5.2 Binary Data Result**



In above screen with binary data LSTM got 100% accuracy, FSCORE and ROC\_AUC. Below is the binary data comparison graph between all algorithms



In above graph LSTM is giving better output result compare to all algorithms

# 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, 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 test. Each test type addresses a specific testing requirement.

### 6.2 TYPES OF TESTING

**6.2.1 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.

### 6.2.2Integration 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.

### 6.2.3 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.

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.

### System Testing

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

### 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.

### 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.

### 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.

### 6.3 TEST CASES

**6.3.1 CONTINUOUS DATA ANALYSIS**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Test case ID | Test case name | Purpose | Test Case | Output |
| 1 | Stock Price  Prediction | To predict the next day’s closing stock price using historical data | Use an LSTM model to train on historical stock prices(open, high, low, close) and volume data for the last 5 years. | Predicted closing price with a low error margin,demonstrating the effectiveness of deep learning in capturing temporal patterns in continuous data. |

**6.3.2 BINARY DATA ANALYSIS**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Test case ID | Test case name | Purpose | Input | Output |
| 1 | Stock Trend Prediction | To predict the future trend of a stock (upward or downward) based on historical data. | Historical stock data including features like opening price ,closing price, volume, and market indicators | Predicted trend class (0 for “downward”, 1 for “upward”) with accuracy metrics. |

# 6. CONCLUSION

**&**

**FUTURE SCOPE**

## 7.CONCLUSION & FUTURESCOPE

### PROJECT CONCLUSION

The purpose of this study was the prediction task of stock market movement by machine learning and deep learning algorithms. Four stock market groups, namely diversified financials, petroleum, non-metallic minerals and basic metals, from Tehran stock exchange were chosen, and the dataset was based on ten years of historical records with ten technical features. Also, nine machine learning models (Decision Tree, Random Forest, Adaboost, XGBoost, SVC, Naïve Bayes, KNN, Logistic Regression and ANN) and two deep learning methods (RNN and LSTM) were employed as predictors. We supposed two approaches for input values to models, continuous data and binary data, and we employed three classification metrics for evaluations. Our experimental works showed that there was a significant improvement in the performance of models when they use binary data instead of continuous one. Indeed, deep learning algorithms (RNN and LSTM) were our superior models in both approaches.

### FUTURESCOPE

The future scope of predicting stock market trends using machine learning and deep learning algorithms is promising, as financial markets continue to generate vast amounts of data. With advancements in algorithms, such as recurrent neural networks (RNNs) and long short-term memory (LSTM) networks, the accuracy of predictions can significantly improve. Additionally, integrating alternative data sources, like social media sentiment and economic indicators, could enhance model performance. As computational power increases and big data technologies evolve, real-time analysis and automated trading strategies will become more feasible. This project could pave the way for more robust financial decision-making tools, enabling investors to better navigate market volatility and capitalize on emerging opportunities**.**

# BIBILOGRAPHY

PREDICTING STOCK MARKET TRENDS

## 8.BIBILOGRAPHY

# 8.1 REFERENCES

* + 1. "Machine Learning Techniques for Stock Market Prediction: A Review and Future Directions" (2022). *IEEE International Conference on Big Data*.
    2. "A Comparative Analysis of Machine Learning Techniques for Stock Market Forecasting" (2023). *ACM Conference on Data Science*.
    3. Maqsood, Haider, et al. "A local and global event sentiment based efficient stock exchange forecasting using deep learning." International Journal of Information Management 50 (2020): 432-451.
    4. "Stock Price Prediction Using LSTM Recurrent Neural Network" (2023).

*International Journal of Financial Studies*.

* + 1. Baek, Yujin, and Ha Young Kim. "ModAugNet: A new forecasting framework for stock market index value with an overfitting prevention LSTM module and a prediction LSTM module." Expert Systems with Applications 113 (2018): 457-480.

### WEBSITES

[1] https://ieeexplore.ieee.org/stamp/stamp.jsp?tp=&arnumber=9165760

CMRTC