**MINOR PROJECT REPORT**

**ON**

**“STOCK MARKET PREDICTION USING MACHINE LEARNING”**

**SUBMITTED IN PARTIAL FULFILLMENT OF THE DEGREE**

**OF**

**BACHELOR OF TECHNOLOGY**

**IN COMPUTER SCIENCE& ENGINEERING**



**SUBMITTED BY UNDER THE GUIDANCE OF**

PANWAR SHIVANI MS. SWATI VERMA

PRIYA JOSHI ASSISTANT PROFESSOR

SHIVANI SHARMA (CSE DEPT) BTKIT

YOGITA

(CSE 4th Year)

***CONTENTS***

1. Abstract
2. Introduction
3. Literary survey
4. What is Stock Market?
5. Machine Learning
6. Linear Regression
7. Support Vector Regression
8. Random Forest Regressor
9. Parameters Used For Comparing Algorithms
10. Project Code
11. Conclusion
12. References

**DECLARATION**

We hereby declare that the project work entitled **“STOCK MARKET PREDICTION USING MACHINE LEARNING** “submitted to the B.T.KIT, DWARAHAT, is a record of an original work done by us under the guidance of **(CSEDEPT),**  and this project work is submitted in the partial fulfillment of requirements for the award of the degree of Bachelor of Technology in Computer Science& Engineering. The results embodied in this report have not been submitted to any other University or Institute for the award of any degree or diploma.

***ACKNOWLEDGEMENT***

We have taken efforts in this project. However, it would not have been possible without the kind support and help of many individuals and organizations. We would like to extend our sincere thanks to all of them.

We are highly indebted to **Dr. Swati Verma (Project Guide) and Dr. RAKESH BISHT (Assistant Professor and Project Head)** for their guidance and constant supervision as well as for providing necessary information regarding the project & also for their support in completing the project.

We would like to express our gratitude towards our parents, friends & all the faculty members of B.T.K.I.T, Dwarahat, for their kind cooperation and encouragement which help us in the completion of this project.

We would like to express our special gratitude and thanks to all the people for giving us such attention and time.

Our thanks and appreciations also go to our colleagues in developing the project and the people who have willingly helped us out with their abilities.

Sincerely,

Name: Panwar Shivani (190180101038)

Priya Joshi (190180101042)

Shivani Sharma (190180101051)

Yogita (190180101063)

ABSTRACT

The prediction of a stock market direction may serve as an early recommendation system for short-term investors and as an early financial distress warning system for long-term shareholders. Forecasting accuracy is the most important factor in selecting any forecasting method. Research efforts in improving the accuracy of forecasting models are increasing since the last decade. The appropriate stock selections that are suitable for investment is a very difficult task. The key factor for each investor is to earn maximum profits on their investments.

In this paper, three supervised learning algorithms are used namely Linear Regression, Support Vector Regressor, and Random Forest Regressor. Supervised learning is an approach to creating artificial intelligence (AI), where a computer algorithm is trained on input data that has been labelled for a particular output. In this paper, we are comparing different supervised algorithms to get an efficient predicting model.

These methods are applied to 5 years of data retrieved from Yahoo Finance. The results will be used to analyze the stock prices and their prediction in depth in future research efforts.

INTRODUCTION

Investment firms, hedge funds and even individuals have been using financial models to better understand market behavior and make profitable investments and trades. A wealth of information is available in the form of historical stock prices and company performance data, suitable for machine learning algorithms to process. Can we actually predict stock prices with machine learning? Investors make educated guesses by analyzing data. They'll read the news, study the company history, industry trends, and other lots of data points that go into making a prediction. The prevailing theory is that stock prices are totally random and unpredictable but that raises the question why top firms like Morgan Stanley and Citigroup hire quantitative analysts to build predictive models. We have this idea of a trading floor being filled with adrenaline infuse men with loose ties running around yelling something into a phone but these days they're more likely to see rows of machine learning experts quietly sitting in front of computer screens. In fact, about 70% of all orders on Wall Street are now placed by software, we're now living in the age of the algorithm.

This project seeks to utilize Machine Learning models, and different algorithms to predict stock prices. We will use Machine Learning to build a supervised learning model to predict stock prices using historical closing price, trading volume, opening price, and other constraints of stock market. We visualize both the predicted price values over time and the optimal parameters for the model.

**OBJECTIVE**:

In the past decades, there is an increasing interest in predicting markets among economists, policymakers, academics and market makers. The objective of the proposed work is to study and improve the supervised learning algorithms to predict the stock price.

**Technical Objective**:

The technical objectives will be implemented in Python. The system must be able to access a list of historical prices. It must calculate the estimated price of stock based on the historical data. It must also provide an instantaneous visualization of the market index.

**Experimental Objective:**

Three versions of prediction system will be implemented; one using Linear Regression, one using Support Vector Regressor and one is using Random Forest. The experimental objective will be to compare the forecasting ability of these algorithms. We will test and evaluate both the systems with same test data to find their prediction accuracy.

**WHAT IS THE PROBLEM?**

Investors are familiar with the saying, “buy low, sell high” but this does not provide enough context to make proper investment decisions. Before an investor invests in any stock, he needs to be aware how the stock market behaves. Investing in a good stock but at a bad time can have disastrous results, while investment in a mediocre stock at the right time can bear profits. Financial investors of today are facing this problem of trading as they do not properly understand as to which stocks to buy or which stocks to sell in order to get optimum profits. Predicting long term value of the stock is relatively easy than predicting on day-to-day basis as the stocks fluctuate rapidly every hour based on world events.

**WHY THIS IS A PROJECT RELATED TO THIS CLASS?**

The solution to this problem demands the use of tools and technologies related to the field of data mining, machine learning and data prediction. The application will predict the stock prices for the next trading day. The requirements and the functionality of this application correlates it to the class.

**STATEMENT OF THE PROBLEM:**

Financial analysts investing in stock market usually are not aware of the stock market behavior. They are facing the problem of trading as they do not properly understand which stocks to buy or which stocks to sell in order to get more profits. In today’s world, all the information pertaining to stock market is available. Analyzing all this information individually or manually is tremendously difficult. As such, automation of the process is required. This is where Data mining techniques help.

Understanding that analysis of numerical time series gives close results, intelligent investors use machine learning techniques in predicting the stock market behavior. This will allow financial analysts to foresee the behavior of the stock that they are interested in and thus act accordingly.

The input to our system will be historical data from Yahoo Finance. Appropriate data would be applied to find the stock price trends. Hence the prediction model will notify the up or down of the stock price movement for the next trading day and investors can act upon it so as to maximize their chances of gaining a profit. The entire system would be implemented in very few Pages of Python language using open-source libraries. Hence it will effectively be a zero-cost system

***LITERATURE SURVEY***

Recently, a lot of interesting work has been done in the area of applying Machine Learning Algorithms for analyzing price patterns and predicting stock price. Most stock traders nowadays depend on Intelligent Trading Systems which help them in predicting prices based on various situations and conditions.

Recent researches use input data from various sources and multiple forms. Some systems use historical stock data, some use financial news articles, some use expert reviews while some use a hybrid system which takes multiple inputs to predict the market.

Also, a wide range of machine learning algorithms are available that can be used to design the system. These systems have different approaches to solve the problem. Some systems perform mathematical analysis on historic data for prediction while some perform sentiment analysis on financial news articles and expert reviews for prediction.

However, because of the volatility of the stock market, no system has a perfect or accurate prediction

What is Stock Market?

The term stock market refers to several exchanges in which shares of publicly held companies are bought and sold. Such financial activities are conducted through formal exchanges and via [over-the-counter](https://www.investopedia.com/terms/o/otc.asp) (OTC) marketplaces that operate under a defined set of regulations.

Both “stock market” and “stock exchange” are often used interchangeably. Traders in the stock market buy or sell shares on one or more of the stock exchanges that are part of the overall stock market.

The Data collected have following Features:

1.Open: It means the price at which a stock started trading at the start of day.

2. Close: It refers to the price of an individual stock when the stock exchange is closed.

3. High: This is the highest price at which stock is traded during the day.

4. Low: This indicates the lowest price of the period.

5.Volume: The number of shares traded in an entire market during given period of time.

**HOW TO COLLECT INPUT DATA?**

Input data is taken from Yahoo Finance using following steps:

1. For our project, we are considering dataset of the Coca-Cola company.
2. Use stock’s ticker symbol from step a to get data from Yahoo Finance.
3. System will take last 5 years’ stock data of the Coca-Cola company.
4. Further we divide the data into two parts, training data and testing data, where 80% of the data will be used for training and 20% of the data will be used for testing

Machine Learning:

*Machine learning enables a machine to automatically learn from data, improve performance from experiences, and predict things without being explicitly programmed.*

With the help of sample historical data, which is known as **training data**, machine learning algorithms build a **mathematical model** that helps in making predictions or decisions without being explicitly programmed. Machine learning brings computer science and statistics together for creating predictive models. Machine learning constructs or uses the algorithms that learn from historical data. The more we will provide the information, the higher will be the performance.



At a broad level, machine learning can be classified into three types:

1. **Supervised learning**
2. **Unsupervised learning**
3. **Reinforcement learning**

We are using supervised learning algorithms for our project.

**Supervised Learning**

Supervised learning is a type of machine learning method in which we provide sample labelled data to the machine learning system in order to train it, and on that basis, it predicts the output.

The system creates a model using labelled data to understand the datasets and learn about each data, once the training and processing are done then we test the model by providing a sample data to check whether it is predicting the exact output or not.

In this project we are applying three supervised learning algorithms:

1. Linear Regression
2. Support Vector Regressor
3. Random Forest Regressor

1.Linear Regression:

Linear regression algorithm shows a linear relationship between a dependent (y) and one or more independent (y) variables, hence called as linear regression. Since linear regression shows the linear relationship, which means it finds how the value of the dependent variable is changing according to the value of the independent variable.

The linear regression model provides a sloped straight line representing the relationship between the variables. Consider the below image:



Mathematically, we can represent a linear regression as:

**y= a0+a1x+ ε**

**Here,**

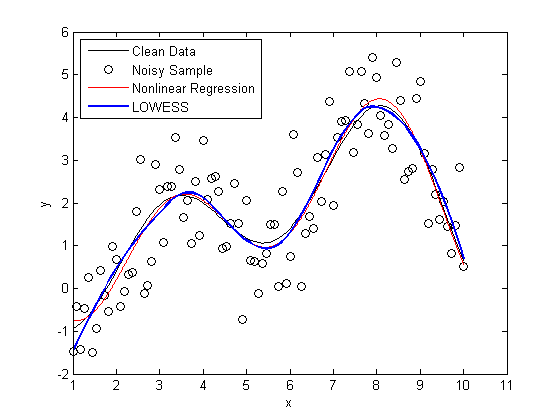
Y=Dependent Variable (Target Variable)  
X= Independent Variable (predictor Variable)  
a0= intercept of the line (Gives an additional degree of freedom)  
a1 = Linear regression coefficient (scale factor to each input value).  
ε = random error

The values for x and y variables are training datasets for Linear Regression model representation.

We are using the **LinearRegression()** function, imported from **sklearn Library.**

2. Support Vector:

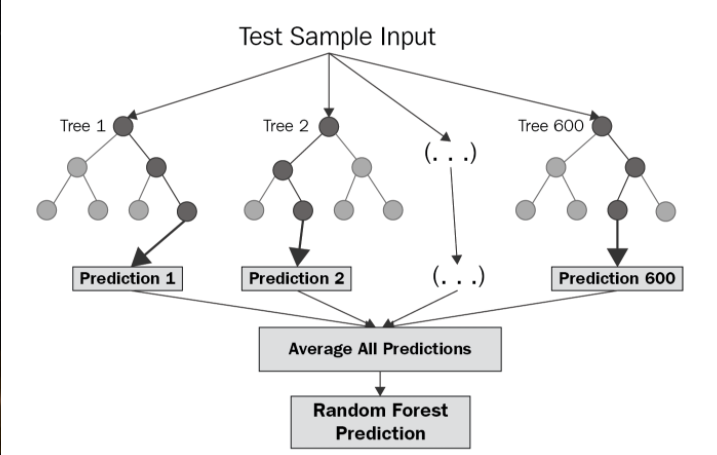
Support Vector Regression is a supervised learning algorithm that is used to predict discrete values. Support Vector Regression uses the same principle as the SVMs. The basic idea behind SVR is to find the best fit line. In SVR, the best fit line is the hyperplane that has the maximum number of points.



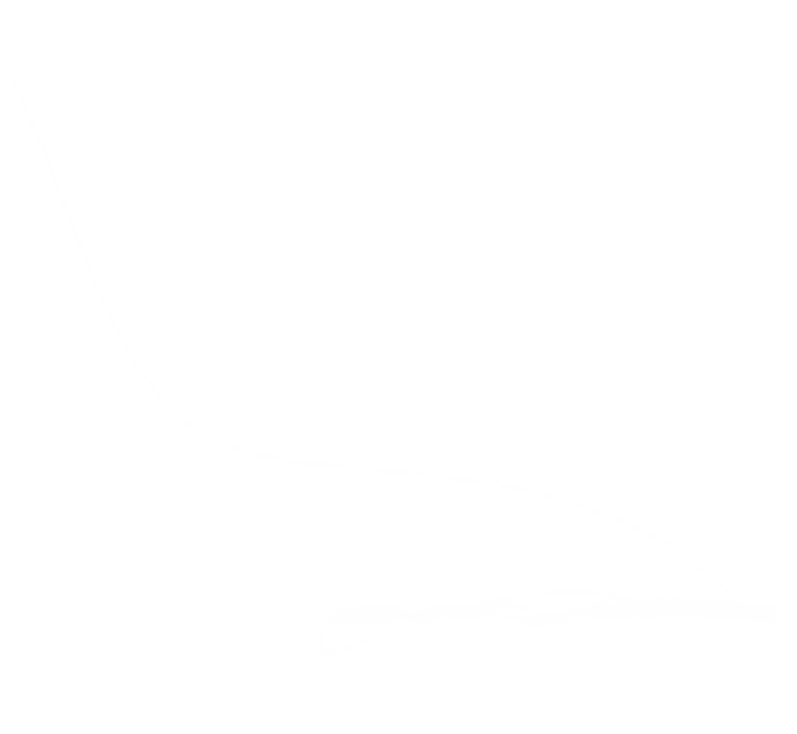
Unlike other Regression models that try to minimize the error between the real and predicted value, the SVR tries to fit the best line within a threshold value. The threshold value is the distance between the hyperplane and boundary line. The fit time complexity of SVR is more than quadratic with the number of samples which makes it hard to scale to datasets with more than a couple of 10000 samples

3.Random Forest Regression

**Random Forest Regression** is a supervised learning algorithm that uses **ensemble learning** method for regression. Ensemble learning method is a technique that combines predictions from multiple machine learning algorithms to make a more accurate prediction than a single model.

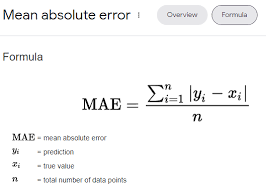


A Random Forest Regression model is powerful and accurate. It usually performs great on many problems, including features with non-linear relationships. Disadvantages, however, include the following: there is no interpretability, overfitting may easily occur, we must choose the number of trees to include in the model.

**Parameters used for comparing Algorithms:**

1. Mean Absolute Error:

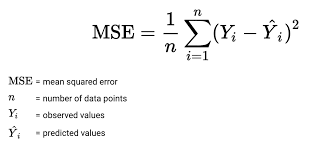
Mean absolute error gives the mean of absolute difference between model prediction and target value.





2. Mean Squared Error(MSE):

Mean squared error (MSE) measures the amount of error in statistical models. It assesses the average squared difference between the actual and predicted values.



3.Root Mean Squared Error(RMSE):

Root mean squared error (RMSE) is the square root of the mean of the square of all of the errors.

The use of RMSE is very common, and it is considered an excellent general-purpose error metric for numerical predictions.

RMSE is the square root of mean squared error or MSE.



Project Code:

#importing required libraries

import pandas as pd

import numpy as np

import matplotlib.pyplot as plt

import sklearn

from sklearn import preprocessing, svm

from sklearn.linear\_model import LinearRegression

from sklearn.ensemble import RandomForestRegressor

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

from sklearn import metrics

#importing dataset

data = pd.read\_csv('C:/Users/Dell/Downloads/KO (1).csv')

#information about the dataset

data.info()

#defining the target and independent variables

x = np.array((data[['Volume','Open','High','Low']]))

y = np.array((data['Close']))

# data preprocessing and exploration

x= preprocessing.scale(x)

data['Close'].describe()

# removing outliers

q1 = data['Close'].quantile(0.25)

q3 = data['Close'].quantile(0.75)

iqr = q3-q1

upper\_limit = q3 + iqr\*1.5

lower\_limit = q1 - iqr\*1.5

def limit\_imputer(value):

if value>upper\_limit:

return upper\_limit

if value<lower\_limit:

return lower\_limit

else:

return value

data['Close'] = data['Close'].apply(limit\_imputer)

# Filling missing values

from sklearn.impute import SimpleImputer

imputer = SimpleImputer(missing\_values = np.nan, strategy = 'median' )

data[num\_data] = imputer.fit\_transform(data[num\_data])

plt.figure(dpi = 100)

k = range(0, len(data)) #index of data at x-axis

plt.plot(k, data['Close'].sort\_values(), color = 'red', label = 'Close')

plt.xlabel("Datapoints")

plt.ylabel("Close")

plt.legend()

# Linear Regression

#splitting the training and testing data

x\_train,x\_test,y\_train,y\_test = train\_test\_split(x,y,test\_size = 0.3, random\_state = 0)

#Implementing linear regression

regressor = LinearRegression()

#fitting the model

regressor.fit(x\_train,y\_train)

# prediction

predictions = regressor.predict(x\_test)

plt.figure(dpi = 130, figsize = (7,5))

plt.plot(y\_test,color = 'b', label = 'Actuals')

plt.plot(predictions,color='k')

plt.title("Linear Regression")

plt.xlabel('Datapoints')

plt.ylabel('Values')

plt.show()

dframe = pd.DataFrame({"Actual":y\_test,"Predicted": predictions.flatten()})

dframe.head()

# Calculating Errors

#Using MAE

residuals = predictions - y\_test

from sklearn.metrics import mean\_absolute\_error

error\_mean = mean\_absolute\_error(predictions,y\_test)

error\_mean

# Using MSE

from sklearn.metrics import mean\_squared\_error

squared\_error\_mean = mean\_squared\_error(predictions, y\_test)

squared\_error\_mean

# Using RMSE

from sklearn.metrics import mean\_squared\_error

squared\_error\_mean = mean\_squared\_error(predictions, y\_test)\*\*0.5

squared\_error\_mean

# SVR

#defining the target and independent variables

x = np.array(data[['Volume','Open','High','Low']])

y = np.array(data['Close'])

#splitting the training and testing data

X\_train,X\_test,Y\_train,Y\_test = train\_test\_split(x,y,test\_size = 0.3, random\_state = 0)

h1 = svm.SVR()

h1.fit(X\_train,Y\_train)

Y\_predict = h1.predict(X\_test)

#Using MAE

residuals\_svm = Y\_predict - Y\_test

from sklearn.metrics import mean\_absolute\_error

error\_mean = mean\_absolute\_error(Y\_predict,Y\_test)

error\_mean

# Using MSE

from sklearn.metrics import mean\_squared\_error

squared\_error\_mean = mean\_squared\_error(Y\_predict, Y\_test)

squared\_error\_mean

# Using RMSE

from sklearn.metrics import mean\_squared\_error

squared\_error\_mean = mean\_squared\_error(Y\_predict, Y\_test)\*\*0.5

squared\_error\_mean

plt.figure(dpi = 130, figsize = (6,5))

plt.plot(Y\_test,color = 'b')

plt.plot(Y\_predict,color='k')

plt.title("Support Vector Regressor")

plt.xlabel('Datapoints')

plt.ylabel('Values')

plt.show()

# Random Forest

#defining the target and independent variables

x = np.array(data[['Volume','Open','High','Low']])

y = np.array(data['Close'])

#splitting the training and testing data

X\_train,X\_test,Y\_train,Y\_test = train\_test\_split(x,y,test\_size = 0.3, random\_state = 0)

model = RandomForestRegressor()

model.fit(X\_train, Y\_train)

# prediction

RFY\_predict= model.predict(X\_test)

dframe = pd.DataFrame({"Actual":Y\_test,"Predicted": RFY\_predict .flatten()})

#Using MAE

from sklearn.metrics import mean\_absolute\_error

error\_mean = mean\_absolute\_error(RFY\_predict ,Y\_test)

error\_mean

# Using MSE

from sklearn.metrics import mean\_squared\_error

squared\_error\_mean = mean\_squared\_error(RFY\_predict , Y\_test)

squared\_error\_mean

# Using RMSE

from sklearn.metrics import mean\_squared\_error

squared\_error\_mean = mean\_squared\_error(RFY\_predict , Y\_test)\*\*0.5

squared\_error\_mean

plt.figure(dpi = 130, figsize = (7,5))

plt.plot(Y\_test, color = 'blue', label = 'Actual')

plt.plot(RFY\_predict,color='black', label = 'Prediction')

plt.title("Random Forest Regressor")

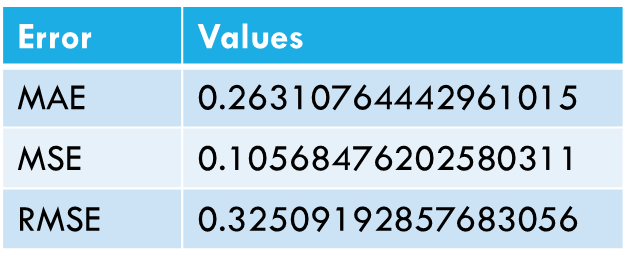
plt.xlabel('Datapoints')

plt.ylabel('Values')

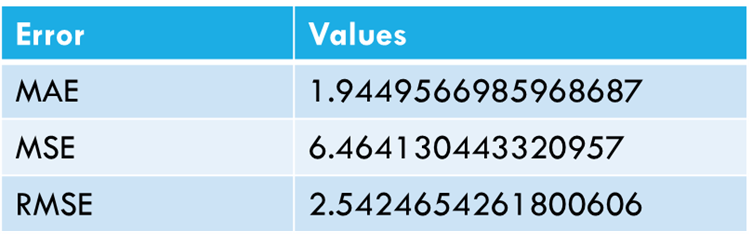
plt.show()

**Calculated Errors:**

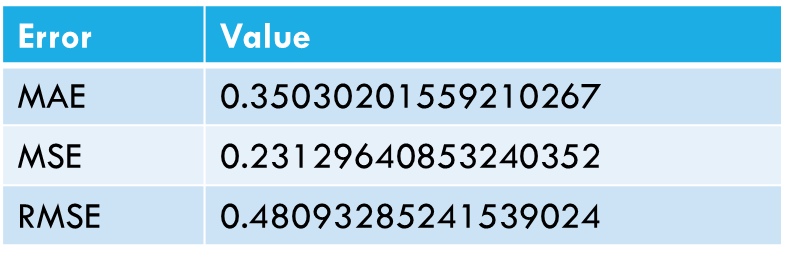
1. Linear Regression



1. Support Vector regressor



1. Random Forest Regressor:



Conclusion:

In this project we studied the use of three supervised learning algorithms namely linear regression, support vector regression and random forest regressor. On comparing all these algos we saw that **Linear Regression** gave us better results.

The random forest regressor’s error were also comparatively low , therefore it can also be used for prediction.

However each method has its own strengths and weaknesses. In this project, we have used mainly five features of the stock market data. Also a particular ML algorithm might be better suited to a particular type of stock and may give low accuracy for other types of data.

**References**

<https://finance.yahoo.com/quote/KO/history?period1=1514937600&period2=1672444800&interval=1d&filter=history&frequency=1d&includeAdjustedClose=true>

<https://ieeexplore.ieee.org/abstract/document/8703332/>

<https://www.analyticsvidhya.com/blog/2021/05/feature-scaling-techniques-in-python-a-complete-guide/>

<https://towardsdatascience.com/random-forest-regression-5f605132d19d>

<https://towardsdatascience.com/unlocking-the-true-power-of-support-vector-regression-847fd123a4a0#:~:text=Support%20Vector%20Regression%20is%20a,the%20maximum%20number%20of%20points>.

Stock Market Prediction using Machine Learning

<https://www.researchgate.net/publication/331279199_Stock_Market_Prediction_Using_Machine_Learning>

Predicting the direction of stock market prices using random forest Luckyson Khaidem Snehanshu Saha Sudeepa Roy Dey khaidem90@gmail.com snehanshusaha@pes.edu [sudeepar@pes.ed](mailto:sudeepar@pes.ed)