Project Report on

COMPARATIVE STUDY OF MACHINE LEARNING ALGORITHMS FOR STOCK PRICE PREDICTION

Submitted by,

Febin Basheer E(2K18/MBA/908)

Gishnu Raj (2K18/MBA/909)

Under the Guidance of

Dr. Gaganmeet Kaur Awal

UNIVERSITY SCHOOL OF MANAGEMENT

& ENTREPRENEURSHIP

Delhi Technological University

MAY 2020

CERTIFICATE

This is to certify that the dissertation report titled "**Comparative Study of Machine Learning Algorithms for Stock Price Prediction**" is a bonafide work carried out by **Mr. Febin Basheer E** and **Mr. Gishnu Raj** of **MBA 2018-2020** under the guidance of Dr. Gaganmeet Kaur Awal submitted to University School of Management and Entrepreneurship, Delhi Technological University, Bawana Road , Delhi-42 in partial fulfilment of the requirement for the award of the Degree of Masters of Business Administration.

Signature of Guide   Signature of HOD

Place   Seal of HOD

Date

DECLARATION

I hereby declare that the project titled "**COMPARATIVE STUDY OF MACHINE LEARNING ALGORITHMS FOR STOCK PRICE PREDICTION**" under the guidance of **Dr. Gaganmeet Kaur Awal**, faculty in charge, submitted in partial fulfillment of the requirements for the degree of Master of Business Administration is true and original to the best of my knowledge and belief.

I further declare that the information presented in this project has not been submitted for the award of any other degree/diploma/fellowship or other similar titles or prizes.

Date: th May 2020

Place: Delhi

Febin Basheer E

Roll No: 2K18/MBA/908

Gishnu Raj

Roll No: 2K18/MBA/909

MBA (2018-2020)

ACKNOWLEDGEMENT

I would like to express my gratitude and thanks to Dr. Gaganmeet Kaur Awal for all her support and guidance provided to me. I would like to thank her for the support and understanding my requests and with her valuable suggestions. I would also like to thank University School of Management and Entrepreneurship (DTU) for giving me this opportunity for making this Project meaningful and effective. I would also like to thank all my friends and seniors who have responded to my requests and who shared their experiences and helped me in understanding and approaching situations in a better way.

I would specially thank my family for their support, motivations and also for being there in my hardships.

Thanking You,

Febin Basheer E

2K18/MBA/908

Gishnu Raj

2K18/MBA/909

USME, DTU

ABSTRACT

The objective of the project is to do a comparative analysis of the machine learning algorithm to predict the price of a stock which is listed on the Indian stock markets which is the NSE and the BSE. We will be looking in to some of the machine learning algorithms that can be used to predict the stock price. The comparative study of these algorithms is to know which algorithm gives the best accurate result.

In this report we will be talking about number of machine learning techniques that can be used to predict the stock price of the selected stock. Regression model, ARIMA, Long-short term memory, SVM, Random forest are some of the methods we use in the analysis and prediction of the stock price. A comparative study of these algorithms is also drawn out to best understand the accuracy of each of the methods. We will be using the data of sample stock price with which we will be implementing the machine learning algorithms to predict the stock value. We will be giving the detailed analysis of algorithm that we are using to predict the stock market price. Moreover, we will be using plots and charts to identify the trends in the stock price. Each method we use is distinct in each of the aspects regarding to the machine learning algorithms. There are number of algorithms used for the predictions, we will be focusing on certain algorithms which is best suited and gives the more accurate results from the data. We will also compare different algorithm used for prediction.

We will be looking carefully in to the results that gives a comparative analysis of each of the method and a detailed inference on each of the algorithm and its results. Since each of the algorithms are different in their own ways a clear-cut interpretation and results will provide a good insight in to each of the algorithms also their limitation. The successful prediction of the stock price will be a great asset for the stock market institutions and this will also provide a solution to the real-life scenarios that the stock market investors will face.

CONTENTS

CHAPTER 1

INTRODUCTION

This chapter includes the introduction to the report such as the problem in hand, the motivation for doing this study, further details about similar studies, the purpose of the study and a detailed problem statement.

1.1 Introduction

Prediction of the stock price is one of the most difficult scenarios we have in hand. There are many factors that should be incorporated and taken into account while predicting the stock price. These factors include physical, psychological, rational, irrational behavior etc. All this factor makes the stock price prediction more volatile and very difficult to predict with a high degree of accuracy.

The main objective or the motivation behind this comparative analysis is to find out a clear-cut analysis of each of the algorithm which can be further implemented in the prediction of the stock price. This will in turn perform as a preliminary step to incorporate they dynamic changes in the market and also put it in to the algorithm which then gives a real time value. The stock market act as a third party who is responsible for successful completion a trade between the buyer and seller. They act as the intermediary to avoid default and cheating in the market, so the exchange will select the shares of trusted firms. Stock which are also called as shares generally represents ownership of a company to retail and institutional investors trading in the stock market. In this work we are trying to predict the future price of a stock and this prediction is expected to be robust, accurate and efficient. The proposed system is studied and planned such that the working is according to the real-life scenarios and should be well suited to the real-world settings. The system or the algorithm which we are using is expected to take all the technical factors and variables that might affect the price of a stock and its performance from the historical data.

It has been found that number of these techniques have been previously studied up on in the prediction of the stock price. There are various methods used for the predicting the future value of the stock price and various way of implementing prediction system like Fundamental analysis, Technical analysis, Machine learning, Market mimicry, time series analysis etc. with the advancement of technology and the upcoming of the digital era the prediction has moved up to another promising level. The most prominent and promising technologies include the application of Artificial Neural Network, Recurrent Neural Network, which actually is the implementation of machine learning algorithms. This technique of machine learning also involves the usage of artificial intelligence which have a great deal of impact in this field which in-turn teaches the system to learn and improve from the learning experiences without being programmed in time to time. There are also some traditional methods used in the prediction of stock price using machine learning algorithm such as the Backward propagation, which is also known as the Backward Propagation Errors. We will also be taking in to consideration of machine learning algorithms such as Linear Regression model, Arima model, The Prophet method, Long short-term memory etc. Researchers are now using many techniques in predicting the future stock price, these consist of many ensemble learning techniques. The highlight of these techniques is that they are very less time consuming and low price in predicting the future value price.

Stock market prediction for short time window seems to be a random process. The stock price movement over a long time will usually develop a linear curve. The general tendency of the people is to buy the stock as the price may go high or rise in the near future. The main problem with people not investing in stock market is due to the uncertainty associated with the market value. These refrain people from investing in the stock. So there should be a proper mechanism that will help the people in predicting the price in real life scenarios. The methods used to predict the future price of the stock includes some time-series analysis using regression and other models. Also, some technical analysis and some machine learning algorithms are used for the price prediction. The dataset of the stock market prediction model includes details like the Opening price, High price, Low price, Closing price(OHLC), volume data etc… that are needed for the prediction of the price of the stock. We are also incorporating a classification technique that will also be a first hand in predicting the stock values. The aim is to design a model that gains from the market information utilizing machine learning strategies and gauge the future patterns in stock value development

The study will give an overview of each of the algorithms that are used and helps us to understand which will give a best result. Further this method can be used to incorporate the real time dynamics of the stock market along with artificial intelligence and deep learning which will help to predict the stock value in real time changes.

1.2 Problem Statement

The objective of this project is to identify the most accurate Machine Learning algorithms to predict the future price of an Indian stock (HDFC) in a volatile market. Stock price prediction is basically defined as the process of forecasting the future stock price and to offer a robust idea for the people to know and to help in predicting the market price of the stock in the near future. This is usually presented using the quarterly financial ratio using the dataset. Thus, it is not appropriate in relying on a single dataset for the prediction and can give a result which is inaccurate. We are contemplating towards the study of machine learning algorithms with various datasets integration to predict the future value and the stock trends movements. The global panic situation due to the pandemic has turned the bears in the market on and is highly volatile making the common machine learning algorithms fail in predicting the price movement with higher accuracy.

The lack of a better and accurate stock-price prediction algorithm makes the price prediction harder and will remain as such until an accurate system is formulated. Predicting the price movements in the market is very difficult due to the various direct and indirect factors which contributes to the price movement. Especially in a highly volatile times like this, it is even difficult to predict with higher accuracy. The movement in the stock market is usually depends on the sentiments of the investor as a reaction to daily news related to the stocks, global/local economical situations, inflation, political events etc…. The current global market reaction towards the Covid-19 pandemic is such an event where the stock markets round the globe were affected adversely. This can also be an international event like sharp movements in oil prices, political tension between countries. All these events affect the businesses around the globe and supply-demand cycle, which in turn affects the sentiment of investors which is reflected on the price of the stock.

Since almost all micro and macro-economic factors affect the stock movement it is beyond the scope of the investor to predict the future price accurately. Moreover, the investor reaction changes from person to person when it comes to investing. So the way investors react to a particular event varies from person to person. These factors make stock price prediction very difficult. Proper data identification classification and usage of proper algorithm can help us in predicting the stock value.

CHAPTER 2

LITERATURE REVIEW

This chapter gives the overview of the literature study. It has been carried out to understand in detail about each of the algorithms and the theory behind the same. There is also a detailed review about the stock markets and the machine learning which is the first steps in the study.

2.1 Stock price prediction using Machine Learning

The prediction of the price of the stock in the market has become so important in these days and it has become on of the key issues of the present time. Currently are plenty of methods used to predict the price of the stock and one of the widely used method is technical analysis, but the accuracy level of these kind of methods are less than 70%. Even then it is utilized heavily globally due to the higher accuracy compared to other methods. Hence it has become more and more important that we should develop a method that will give us more accuracy in predicting the price of the stock. Generally, investments are made using prediction from technical indicators and price actions with an optimism that the price will move accordingly. There are several techniques that are used in the prediction such as the regression classification and of course the machine learning algorithms. Each of the techniques we use have its own pros and cons which will show up in the results. Due to the nonstationary, nonlinear, high-noise characteristics of financial data, traditional statistical models are finding it difficult to predict the price of the stock with higher accuracy which is necessary. Although there are still some difficulties and problems in financial predictions using deep learning, researchers are working to establish a reliable stock market forecasting model. Increased attempts are being made to apply machine learning and deep learning to stock market prediction with higher accuracy. So it is very important that the methods developed with the help of machine learning algorithms can produce more accurate results when compared to the traditional methods. Since stock markets generate0enormous amounts of data at any given time a huge volume of data needs to go through analysis before a prediction can be made. Each of the techniques listed under regression has its own advantages and limitations over the other counterparts.

The most commonly used and easiest technique mentioned is Linear Regression. The way in which the linear regression models was best used was using the least square approach, but they can alternatively be fitted in many other ways also, such as by diminishing the "lack of fit" in some other norm, or by diminishing a handicapped version of the least square method of loss function. It can be further said that the least square method can be used to fit nonlinear models.

The application of machine learning techniques and artificial intelligence to predict the price of a stock has become a notable trend in the past years. More and more research money technology has been invested in this case to bring out a model which can best predict the score. More research has been carried to have a better accuracy on the results too. Due to the large number of available prediction methods, there exists a number of ways to how to predict the price of a stock, but it doesn't mean that all these methods work in the same way as per the given scenario. The output of each method varies from one another. Random forest algorithm is the method that is used to predict the price of the stock using0financial ratios form the previous quarter's report of the company. This is just another way of looking at the case by approaching it using a pred -ictive model, using the technique of random forest to predict the future price of the sock from historical data. However, there are many other factors that influence the value of the stock, such as volatility of the market, public's perspective about the company, daily news regarding stocks, acquisitions and mergers, and even local events that cause the entire stock market to fluctuate due to the social reach achieved in the present world. By using the financial ratio along with a model that can effectively analyze sentiments the accuracy of the prediction model of stock price can be increased.

Predicting the price of stocks in the market accurately is an impossible task, but the modern-day technology will make it a lot easier in predicting the price of the stock with higher accuracy. Due to the interconnected and huge amount format of data, it is easy to extract certain kind of sentiments thus making it easier to make a relationship between different variable and roughly bring out a pattern of investment. Investment pattern from different firms show sign of similarity, and the key behind successfully predicting the stock market is to exploit these same similarities between the data sets. The method by which the price of a stock can be predicted successfully is not just by using the available technical data, it also does consider using other methods like theuse of sentiment analyzer to derive an important connection between people's emotions and how they are influenced by investment in specific stocks, which in-turn increases the price of the stock.

Till now many attempts have been made at predicting the price of the stock and/or direction of a stock (Bullish, Sideways, Bearish) at a given time, using different machine learning models. The models used for the prediction includes simple linear regression to latest novel attempts at hybridization of models such as Neural Networks and Support Vector Machines. One of the important and widely used techniques that were mentioned was linear regression. This project considers the applications of standard (non-hybrid) models alongside alternative trading and prediction methods, to predict the future price movements from the historical data available. Many standard models have been used till date in predicting the stock prices, some of the most common methods used were reviewed while doing our project.

We will discuss some of the most commonly and suited methods used for price prediction here in this chapter.

2.2 Technical Analysis

Technical Analysis, is a research technique for data backed making trading decisions in the market based on the collective actions of traders in the market which is based on indicators, volumes and price actions. The collective actions of the traders will be visualized by means of a stock chart. Over time, certain patterns are observed, which can be spotted historically also, within these charts and each pattern is a trading signal on which trading decisions can be made. It not only helps in identifying the market direction but also helps in defining the trade variables like entry, exit, stop loss and risk.

Assumptions:

- Market Discounts everything: All known and unknown information in the public domain is reflected in the stock price.

- The ‘how’ is more important than ‘why’: How the price got affected more important than why.
- Price moves in the trend: All major moves in the market is an outcome of a trend.
- History tends to repeat itself.
- Round the globe almost every foreign exchange professional trader uses technical analysis for decision making over fundamental analysis to an extent. Indeed, technical analysis is a very important tool in identifying entry and exit points(trade) as it can be applied to each and every financial instrument. In Kwon and Kish document, published in 2002, it is mentioned that technical trading indicators achieve better profitability than the buy and hold strategy which was widely used in the NYSE (NewYork Stock Exchange), while Chong and NG recommended the use of technical trading signals using the Relative Strength Index(RSI) & Moving Average Convergence Divergence (MACD) indicators for the Future 30 index and they show that these oscillators generates a greater profitability than the buy and hold strategy.
- It is mainly due to the wide acceptance and high success rate of technical analysis and stochastic indicators in order to forecast stock markets compared to other low yielding indicators, why Relative Strength Index(RSI) & Moving Average Convergence Divergence (MACD) have been used as inputs of the SVM.

Technical Indicators

Charts: Japanese candle stick charts are the most common charts used in technical analysis for analysing the historical movements in the market as well as the stocks. From the candle stick chart previous open, close, high and low(OHLC) at a particular time frame can be obtained from a single candle.

Using the candles in combination with other indicators (will be explained) trading signals can be derived for crucial decision making. The most commonly used candle patterns are:

- Single candle sticks patterns: Doji, Hammer, Hanging man, Paper Umbrella etc…

- Multiple candle sticks patterns: Bullish Harami, Bearish Harami, Bullish and Bearish engulfing, Morning Star, Evening star etc…
- Volume and Momentum indicators: MACD, RSI, Stochastic Oscillators, Moving averages etc…

Moving Averages

Average volume of trades is plotted through moving average lines. Simple moving average and exponential moving average(EMA) are the most commonly used moving average systems. Simple Moving average or moving average plots data giving equal weightage to every data point, whereas exponential moving average gives more weightage to the most recent data point and hence is more reactive to the current market sentiment.

MACD

Moving Average Convergence & Divergence - MACD

MACD Line: 12-Day EMA 26-Day EMA

Signal Line: 9-Day EMA of MACD line

MACD Histogram: MACD Line – Signal Line

Histogram indicates the market momentum and from the moment trading signals are derived.

RSI

Relative Strength Index – RSI

RSI helps the trader to identify overbought and oversold price ranges indicating an impending reversal.

RSI 0-20: The security is oversold and is ready for bullish move.

RSI 80-100: The security is overbought and a bearish trend is imminent.

2.3 Machine Learning

Machine Learning can be said as a branch of computer science that is evolved from studying pattern recognition and computational learning theory of artificial intelligence. It is the process of learning and building algorithms that can learn from and make predictions on data sets. What make machine learning the talk of the hour is the ability of the machine to predict from the learnings made by machine from a given data set. These processes involve the developing of a model from the sample inputs in order to make data-driven predictions or choices rather than following firm static programming instructions.

"*A computer program is said to learn from experience E with respect to some task T and some performance measure P, if its performance on T, as measured by P, improves with*

experience E."

So, if we want our algorithm to forecast, for example, stock prices from NSE (task T), we can run it through a machine learning process with historical price data (experience E) and, if it has successfully "learned", it will then do better at predicting future prices from the learnings made (performance measure P).

Machine learning involves two types of tasks:

Supervised machine learning:

Supervised-learning is a machine learning, where we are having the input variables (X) and an output variable (Y), and we use an algorithm that is used for the mapping function from the input variables to the output variable.

$$Y = f(X)$$

The aim is to increase the accuracy, by which the mapping of the function so well that when we have the new input data (x) the program can predict the output variables (Y) for that given data.

This process is called supervised learning because the process of an algorithm learning from the training dataset can be related to that of as a teacher supervising the learning process. We already know the output; the algorithm iteratively makes predictions on the training data and is directed by the teacher to get the desired output. Learning halts when the algorithm achieves an acceptable/desirable level of performance.

Supervised learning can be further grouped into:

- Classification: Here the output variable will be a category. Eg: Disease/No disease and Male/Female etc…
- Regression: Here the output variable will be a real continuous vale. Eg: Price, height, weight etc…

Unsupervised Learning:

Unsupervised learning is where we are having the input data (X) and no output variables.

The main goal for unsupervised learning is to model the underlying structure or distribution in the data which in turn is a process to learn more about the data.

It is called unsupervised learning because unlike supervised learning there is no correct answers. Algorithms are left to their own devises to discover and interpret the interesting structure from the input data.

Unsupervised learning can be further grouped into:

- Clustering: A clustering problem is where we want to figure out the inherent groupings in the data given.
- Association: An association rule learning problem is where we want to find rules that can describe large portions of your data, such as people that buy milk also tend to buy bread.

Semi-supervised Learning:

It is the kind of problems where we have a large sum of input data (X) and only some of the data is labelled (Y) are called semi-supervised learning problems.

These problems come in between both supervised and unsupervised learning.

2.3.1 Linear/Polynomial/Logistics Regression

As mentioned above regression method is a supervised learning method which is widely used in predicting the stock price. It is one of the simplest method used in predicting the future price where the output variable will be predicted based on one/two input variables. As stock price depends on many factors multiple regression is usually needed in predicting stock prices. Regression techniques are widely used in algorithmic trading it is not widely accepted in predicting the stock price for research purpose

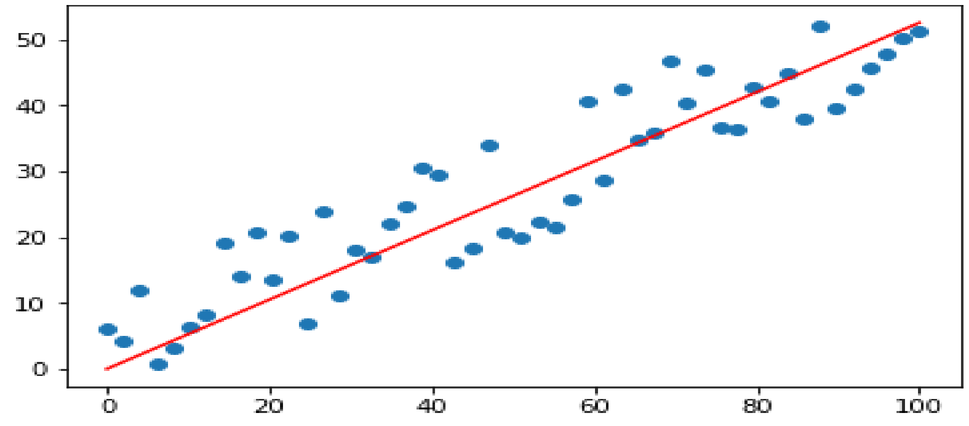
Linear Regression:

The representation of linear regression is a linear equation that combines a set of input values (x) and the solution to which is the predicted output for that set of input values (y). Given that, both the input values (x) and the output value are numeric.

The linear equation assigns one scale factor to each input value, called a coefficient and is represented by the capital Greek letter Beta (B). One additional coefficient is also added, in-order to give the line an additional degree of freedom and is often called the intercept.

For example, a simple regression problem (a single x and a single y) can be of the form as given below:

$$y = B_0 + B_1 \times x$$



(Fig 2.1 Graph shown a linear equation problem [Source: https://www.ablebits.com/linearregression])

Multiple Regression:

Multiple regression explains the relation between multiple independent input variables and the predicted output variable. The dependent output variable is modelled as a function of several independent input variables with corresponding coefficients. Multiple regression has two or more independent input variable and hence it is called as multiple regression.

The multiple regression equation explained above takes the following form:

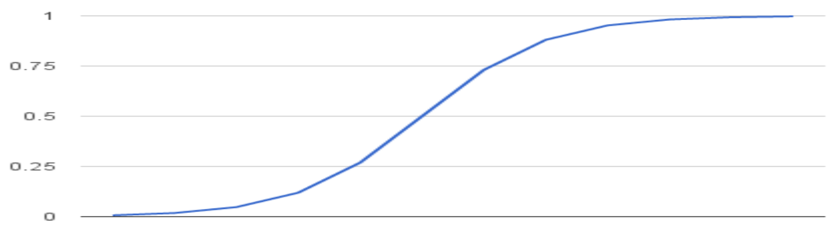
$$y = b_1x_1 + b_2x_2 + \cdots + b_nx_n + c \qquad (2.1)$$

Logistic Regression

Logistic regression is regression method where a logistic function will be used to model when the dependent variable is binary. In regression analysis, logistic regression is widely used to examine and describe the relationship between the binary variable and the predictor variable.

An example logistic regression equation:

$$y = e^{\frac{b_0+b_1*x}{(1+e^{(b_0+b_1*x)})}} \qquad (2.2)$$



(Fig 2.2 Plot of the numbers between -5 and 5 transformed into the range 0 and 1 using the logistic function. [Source: https://www.analyticsvidhya.com/blog/2018/10)])

2.3.2 Support Vector Machine (SVM)

The aim of the support vector machine (SVR) algorithm is to find a hyperplane in an N-dimensional space (N - the number of features) that distinctly classifies the given data points. The main task associated with support vector machine is to identify N-dimensional space that separates the data points individually. Here, N stands for a number of characteristics. Between the two classes of data points 0there can be multiple hyperplanes that can be chosen. The aim of this algorithm is to find a plane that has maximum margin. Maximizing margin refers to the distance between data points of both classes. The benefit associated with maximizing the margin is that it provides is that it provides some reinforcement so that future data points can be more easily classified.   Decision boundaries that will help to differentiate data points are said to be hyperplanes. Based on the position of data points relative to hyperplane they are assigned to different classes. The dimension of the hyperplane depends on the number of attributes, if the number of hyperplanes is two then is just a line, if the number of attributes are three the hyperplane is two dimensional.

They are characterized by the capacity control of decision function and the use of kernel functions. The selection of the correct kernel function is very important for higher accuracy outputs. The method based on kernel function suggests that instead of attaching a algebraic part to each part of the input domain which is given by,

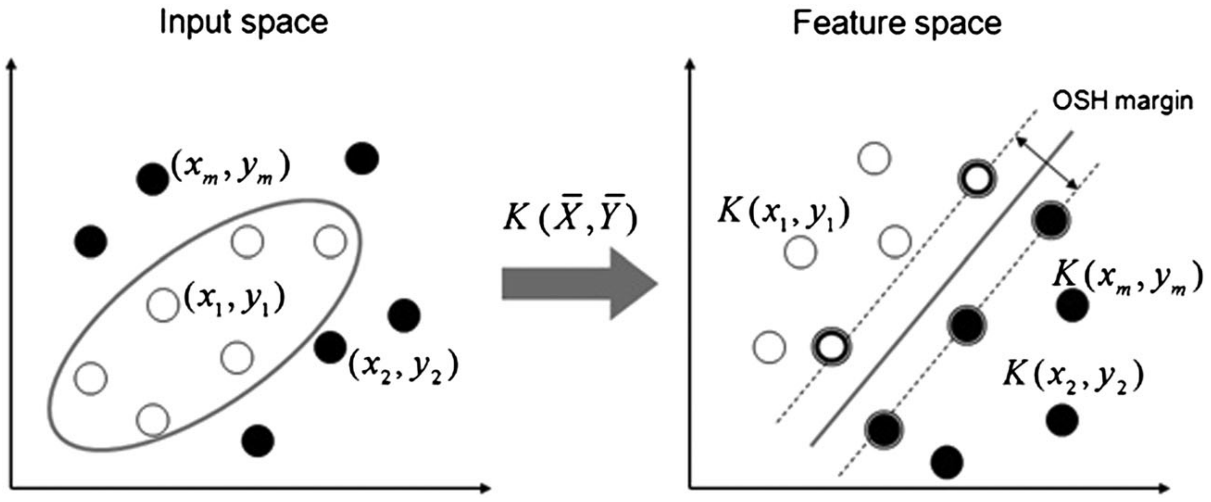
$$\Phi: X \rightarrow F \qquad (2.3.1)$$

a kernel functions

$$K: X\ X \rightarrow R \qquad (2.3.2)$$

is used to determine the similarity of each pair of objects in the given input set. An example is illustrated in the given Figure.

What makes SVM different from other traditional methods is that SVMs do not focus on optimisation protocol that make minimal errors like other technique. Traditionally, most learning algorithms have focused on minimizing the errors generated by the respective models. They are based on the principle of empirical risk minimization (ERM). The aim of SVM id different, its intention is not to reduce the empirical risk of making just few mistakes, but intend to build reliable models. This principle is known as structural risk minimization. The SVM searches a structural model that has little risk of making mistakes with future data. Hence it is one of the best available algorithms in predicting the future stock prices.



(Fig 2.3 Support Vector Machine example [Source: Stock Market Simulation Using Support Vector Machines
Rafael Rosillo, Javier Ginger1 and David De La Fuente])

The main idea of SVM is to construct the hyperplane as decision surface so that the margin separation between positive and negative samples is maximised. It is known as the optimum separation hyperplane (OSH), as shown above in the Figure.

Based on the purpose SVM can be used for regression and classification models. Two applications on SVM financial time series forecasting were developed in 2003: in Cao and Tay (2003), SVMs are applied to the problem of forecasting several future contracts from Chicago Mercantile Market, showing the superiority of SVMs over back propagation and regularised radial basis function neural networks in; Kim. SVMs are used to predict the direction of change in the daily Korean composite stock index and they are bench marked against back propagation, neural network and case-based reasoning. The experimental results show that SVMs outperform other machine learning methods and that they should be considered as a promising method for financial forecasting over others.

The majority of the stockbrokers while making the prediction are relying on the specialized, fundamental or the time series analysis. Overall, these techniques couldn't be trusted completely as their accuracy level is less than 70%, so there emerged the need to give a strong strategy to financial exchange prediction with more accuracy. To get the most accurate result, the methodology chose to be implemented as machine learning and AI along with supervised classifier.

SVM classifier

SVM classifier is a type of discriminative kind of classifier. SVM uses supervised learning i.e. a labelled training data set. The output of such hyperplane will categorize the new dataset. These are the supervised learning algorithm that uses associated learning algorithm for classification and as well as regression.

Parameters

The main parameters of SVM classifier are kernel, gamma and regularization parameters.

- Kernel parameter can be divided as linear & polynomial kernels which gives the prediction line. In the process of predicting the kernel, a new input is calculated by calculating the dot product of the vectors of input and support.
- C parameter which is also known as the regularization parameter; it determines the accuracy of the model is increased or decreased. The default value of C = 10. A value lower that this leads to misclassification.
- Gamma parameters look into the influence of single training on the model. Lower value signifies far from the acceptable margin and high value points to acceptable value limits.

2.3.3 Long short-term memory (LSTM)

LSTM uses the most common of RNN. This time recurrent neural network is meant to avoid long-term dependence problems and is suitable for processing and predicting time series. Proposed by Sepp Hochreiter and Jurgen Schmidhuber in 1997, the LSTM model consists of a unique set of memory cells that replace the hidden layer neurons of the RNN, and its main aspect is the state of memory cell. LSTM model filters information through the gate structure to maintain and update the levels of memory cells. Its door structure includes 3 gates: input, forgotten, and output gates. Each memory cell has 3 layer of sigmoid layer with tan(h) layer. Figure 2.4 displays the structure of LSTM memory cells.

The forgotten gate in LSTMN determines which cell state information is removed from the model. As shown in figure 4 below, the memory cell accepts the output h_{t-1} of the previous moment and the external information x_t of the current moment as inputs and mixes them in long vector $[h_{t-1}, x_t]$ through sigma transformation to become

$$f_t = \sigma(W_f \times (h_{t-1}, x_t) + b_f) \qquad (2.4.1)$$

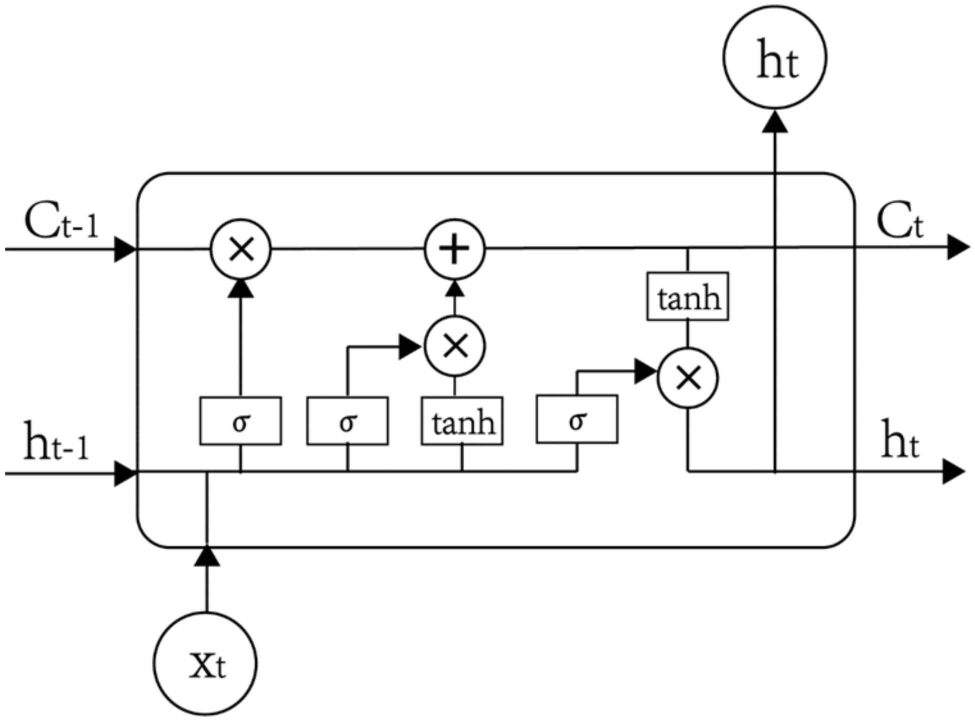
where W_f and b_f are the weight matrix and the bias for the forgotten gate, and σ is the sigmoid function. The forgotten gates main function is recording how much the cell condition C_{t-1} of the later time is reversed to the cell state C_t of the present time. The output gate value between 0 and 1 based on h_{t-1} and x_t, where the value 1 indicate complete reservation and 0 indicates complete disbarment.

The input gate determines how much of the current time network input x_t is reserved into the cell state C_t, which prevents insignificant content from entering the memory cells. It has two functions. One is to find the state of the cell that must be updated; the value to be updated is selected by the sigmoid layer, as in equation (2.4.2). The other method is to update the information to be updated to the cell. A new candidate vector $\hat{C}_t$ is created through the tan(h) layer to control how much new information is added, as in equation (2.4.3). Finally, equation (2.4.4) is used to update the cell state of the memory cells:

$$i_t = \sigma(W_t \times [h_{t-1}, x_t] + b_i) \qquad (2.4.2)$$

The forgotten gate is the gate which determines the where the information is removed from the model. As shown in figure 4, the memory cell accepts the output ht-1 of the previous moment and the external information xt of the current moment as inputs and combines them in a long vector [ht-1, xt] through σ transformation to become where Wf and bf are, respectively, the weight matrix and bias of the forgotten gate, and σ is the sigmoid function.

The input gate determines how much of the current time network input xt is reserved into the cell state Ct, which prevents insignificant content from entering the memory cells. It has two functions. One is to find the state of the cell that must be updated; the value to be updated is selected by the sigmoid layer, as in equation (2.4.2). The other is to update the information to be updated to the cell state. A new candidate vector is created through the tanh layer to control how much new information is added, as in equation (2.4.3). Finally, equation (2.4.4) is used to update the cell state of the memory cells:



(Figure 2.4 LSTM Model [Source: https://www.analyticsvidhya.com/blog/LSTM])

The output gate controls what proportion of the current cell state is discarded. The output information is first determined by a sigmoid layer, and then the cell state is processed by tanh and multiplied by the output of the sigmoid layer to obtain the final output portion:

The final output value of the cell is defined as:

$$C_t = \tanh(W_c \times [h_{t-1}, x_t] + b_c) \qquad (2.4.3)$$

$$C_t = f_t \times C_{t-1} + i_t \times C_t \qquad (2.4.4)$$

The output gate controls what proportion of the current cell state is discarded. The output information is first determined by a sigmoid layer, and then the cell state is processed by tan(h) and multiplied by the output of the sigmoid layer to obtain the final output portion:

$$O_t = \sigma(W_\sigma \times [h_{t-1}, x_t] + b_o) \qquad (2.4.5)$$

The final output value of the cell is defined as:

$$h_t = O_t \times \tanh(C_t) \qquad (2.4.6)$$

2.3.4 Autoregressive integrated moving average (ARIMA)

The ARIMA model is a statistical analysis model. It incorporates time series data in better understand of the data set which in turn helps in predicting the future price or value by doing detailed analysis on that. This model also does regression which helps to understand about the strength of the dependent variable which also relative to other data set variables. The main objective is to predict future securities price and financial market trends.

ARIMA model composed of the following parameters:

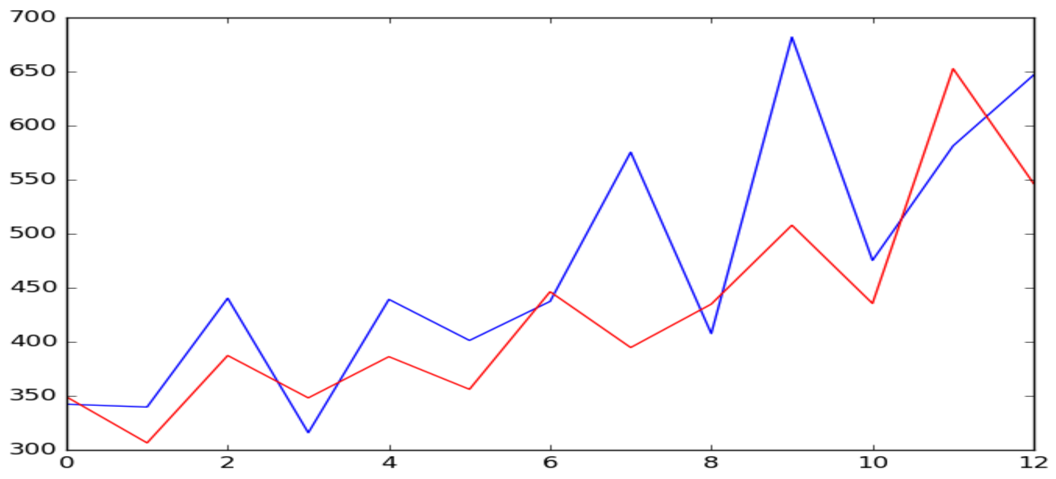
- *Autoregression (AR)* which refers to a model that depicts a changing variable that regresses on its own lagged values.
- *Integrated (I)* points to the differencing of raw observations which in-turn allow for the time series to become stationary, i.e., data values are replaced by the difference between the data values and the previous data values.

- *Moving average (MA)* tells us about the dependency between an observation and a residual error from a moving average model which is applied to lagged observations.

Each of the component functions as a parameter with a standard notation. The standard notation is p, d, and q, where integer values are substituted for the parameters which indicate the ARIMA model being used. It is given as:

- *p*: the number of lag observations in the model; known as the lag order.
- *d*: the number of times that the raw observations are differenced; known as the degree of differencing.
- q: the size of the moving average window; known as the order of the moving average.

In a linear regression model the number and type of terms are included. A 0 value can be used as a parameter, which mean that particular component should not be used in the model. The ARIMA model can be made to perform the function of an ARMA model, or in simpler terms AR, I, or MA models.



(Fig 2.5 ARIMA model example [Source: https://www.machinelearningmastery.com/arima-for-time-series])

In case of the ARIMA model the data is differentiated to make the model more stationary. We need a data which shows consistency over the period of time for analysis. The main objective associated with differentiating is that to remove the trends and seasonality related with the data we have taken for analysis

Seasonality or trends that which are associated with the data can negatively impart an effect when analyzing the data. If the trends and the stationarity is there in the data, then it may adversely affect the computation part of the model and in turn the predicting part too.

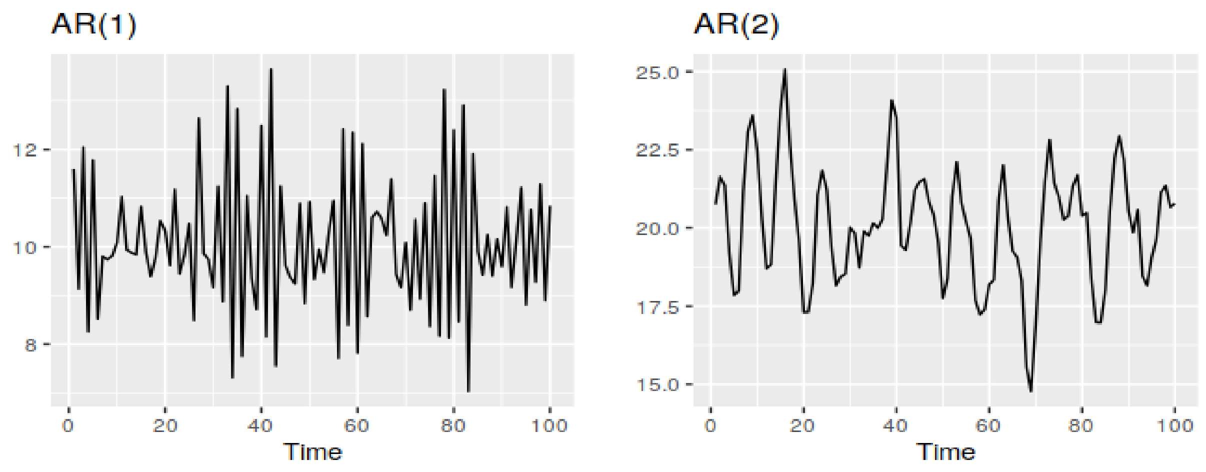
When considering the of multiple regression model, we forecast the variable we want using the help of the predictors. In the case of autoregression model, we forecast the variable we want using a linear combination of past data. Autoregression is in other terms altogether points to regression of same variable again.

Autoregressive model of the order p can be written as:

$$\boldsymbol{y_t = c + \phi_1 y_{t-1} + \phi_2 y_{t-2} + \cdots + \phi_p y_{t-p} + \varepsilon_t} \qquad \textbf{(}2.5)$$

where $\boldsymbol{\varepsilon_t}$ is called the white noise. This is similar to multiple regression but with *lagged values* of y_t as predictors. We say this as an **AR(**p**) model**, an autoregressive model of order p.

Autoregressive model is so flexible in handling wide range of data which have different parameters. The two series in Figure show series from an AR (1) model and an AR (2) model. Changing the parameters $\phi_1,\ldots,\phi_p\phi_1,\ldots,\phi_p$ results in different time series patterns. The variance of the error term ε_t will only change the scale of the series, not the patterns.



(Figure 2.6: Two examples of data from autoregressive models with different parameters. [Source: https://www.machinelearningmastery.com/arima-for-time-series])

For an AR (1) model:

- when $\phi_1=0$ $\phi_1=0$, y_t is equivalent to white noise;
- when $\phi_1=1$ $\phi_1=1$ and $c=0$, y_t is equivalent to a random walk;
- when $\phi_1=1$ $\phi_1=1$ and $c\neq0$, y_t is equivalent to a random walk with drift;
- when $\phi_1<0$ $\phi_1<0$, y_t tends to oscillate around the mean.

We normally restrict autoregressive models to stationary data, in which case some constraints on the values of the parameters are required.

- For an AR (1) model: $-1<\phi_1<1$.
- For an AR (2) model: $-1<\phi_2<1$, $\phi_1+\phi_2<1$, $\phi_2-\phi_1<1$.

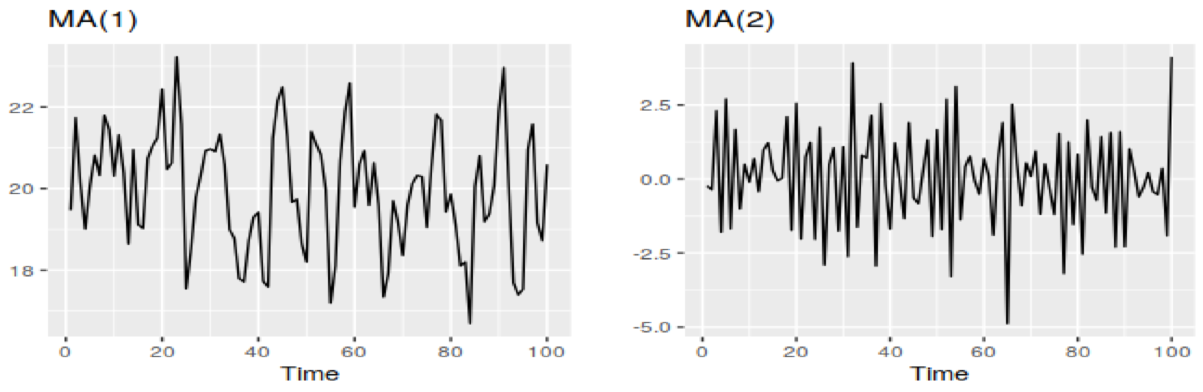
When $p\geq3$, the restrictions are much more complicated. R takes care of these restrictions when estimating a model.

Moving Average Model

Instead of using the past values to predict the variable using regression, the moving average model takes in to account past error in a regression-based model.

$$y_t = c + \varepsilon_t + \theta_1 \varepsilon_{t-1} + \theta_2 \varepsilon_{t-2} + \cdots + \theta_q \varepsilon_{t-q} \qquad (2.6)$$

where **ε_t** is called White Noise. This is called as **MA (q) model**, moving average model of the order q.



(Fig 2.7 Two examples of data from moving average models with different parameters [Source: https://www.machinelearningmastery.com/arima-for-time-series])

It is to be noted each value of y_t can considered as a weighted moving average of the past few forecast errors. However, moving average models should not be confused with the moving average smoothing. A moving average model is used for forecasting future values, while moving average smoothing is used for estimating the trend-cycle of past values.

Changing the parameters θ1,…, θqθ1,…,θq results in different time series patterns. As with autoregressive models, the variance of the error term εt will only change the scale of the series, not the patterns.

It is possible to write any stationary AR (p) model as an MA (∞) model. For example, using repeated substitution, we can demonstrate this for an AR (1) model:

$$y_t = \phi_1 y_{t-1} + \varepsilon_t$$

$$y_t = \phi_1(\phi_1 y_{t-2} + \varepsilon_{t-1}) + \varepsilon_t$$

$$y_t = \phi_1^2 y_{t-2} + \phi_1 \varepsilon_{t-1} + \varepsilon_t$$

$$y_t = \phi_1^3 y_{t-3} + \phi_1^2 \varepsilon_{t-2} + \phi_1 \varepsilon_{t-1} + \varepsilon_t \qquad (2.7)$$

Provided $-1<\phi_1<1$, the value of ϕ_1k will get smaller as 'k' gets larger. So eventually we obtain

$$y_t = \phi_1^3 y_{t-3} + \phi_1^2 \varepsilon_{t-2} + \phi_1 \varepsilon_{t-1} + \varepsilon_t$$

an MA (∞) process. The reverse result holds if we force some constraints on the MA parameters. Then the MA model is then called **invertible**. So, we can write any invertible MA (q) process as an AR (∞∞) process. These Invertible models are not simply introduced to enable us to convert from MA models to AR models. These models also have some desirable mathematical properties.

For e.g. consider the MA (1) process, $y_t = \phi_1 y_{t-1} + \varepsilon_t$. In its AR (∞) representation, we can write the most recent error as a linear function of current and past observations:

$$\varepsilon_t = \sum_{j=0}^{\infty} (-\theta)^j \, y_{t-j} \qquad (2.8)$$

When $|\theta|>1$, the weights increase as lags increase, so the more distant the observations the greater their influence on the current error. When $|\theta|=1$, the weights are constant in size, and the distant observations have the same influence as the recent observations. As neither of these situations make much sense, we require $|\theta|<1$, so the most recent observations have higher weight than observations from the more distant past. Thus, the process is invertible when $|\theta|<1$.

The invertibility constraints for other models are similar to the stationarity constraints.

- "For an MA (1) model: $-1<\theta_1<1$."
- "For an MA (2) model: $-1<\theta_2<1$, $\theta_2+\theta_1>-1$, $\theta_1-\theta_2<1$."

More complicated conditions hold for $q\geq3$. Again, R will take care of these constraints when estimating the models.

2.3.5 Random Forest Method.

Random forest or random decision forest are an ensemble learning method that we use for classification, regression and other tasks that operate by constructing a multitude of decision trees at training time and outputting the class that id the mode of the classification or mean prediction of individual tress. Random decision forest corrects for decision trees habit of over fitting to their training set.

This method can be a good choice for the market prediction. The things that make this algorithm different is because of its flexibility and the accuracy in prediction. This method is widely used for the classification as this method have as associated volatility. In the case of random forest, we are using the decision tree analysis as a base which give a better analyzing technique, which also look in tot different features for the prediction. In random forest method these features are incorporated in to the decision tress to identify the best node for which can be used as a tool for the prediction. Here the data is split in such a way that the 80% data is considered as a training data set and a 20% data can be taken as a test data set. This will in turn gives a best output for the prediction. It will also help to build up a good relationship too.

Random Forest Classifier

It can be called as a type of classifier as well as a supervised learning algorithm. This classifier in turn makes some decision tree and try to bring out some results from that. The basic idea behind this random forest classifier is to take a decision value from a set of decision trees and then to give a final result from this .

Parameters

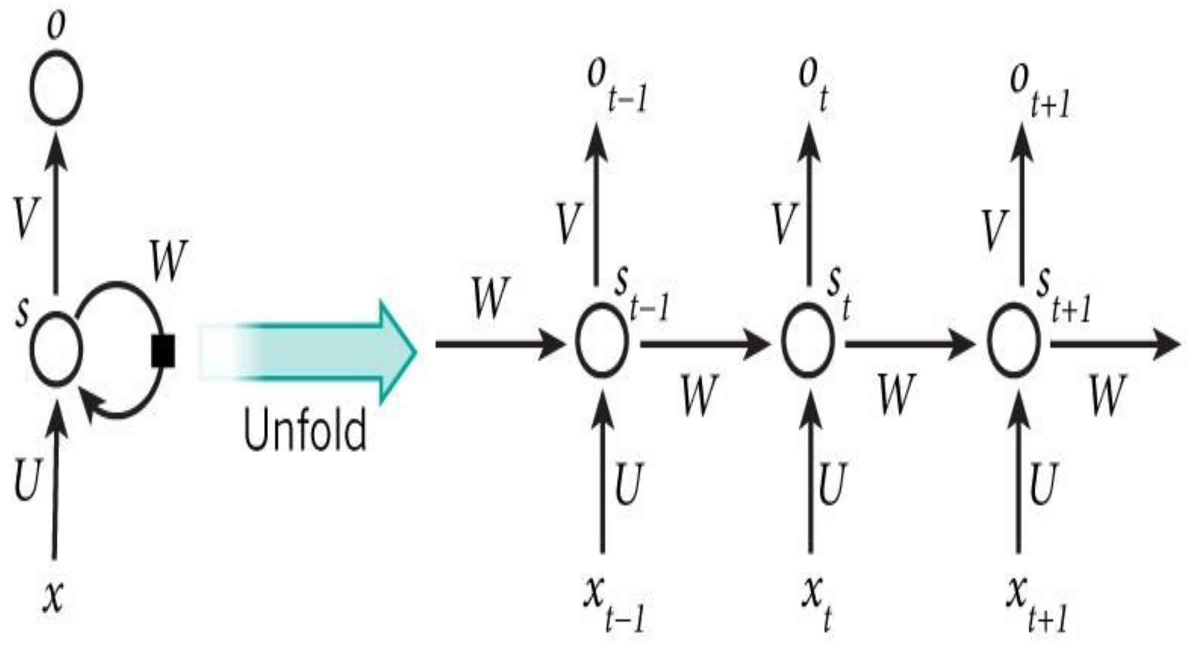
There are many parameters incorporated for the analysis in the case of random forest model. The n-estimator which gives the total number of decision trees used and b-score used for identification of the accuracy of the model and also to identify the best split in each case. Minimum weight fraction gives the total minimum sum of weight required to be as the leaf node. We can assume that the sample is with equal weight when the weight is not provided.

2.4 Deep Learning

Deep learning is a form of artificial intelligence that imitates the working of the human brain in processing the data and identifying patterns which can be used for decision making. Deep learning which comes under the machine learning, which has the capability of learning from unsupervised data.

2.4.1 Recurrent Neural Networks

Recurrent Neural Networks or RNN, is a kind of neural networks extensively used in Natural Language Processing (NLP). In the case of a neural network the input is processed through a different number of steps and output is given, we are taking in to the assumption that the two adjacent input are not dependent to each other. There is also a contradiction that this may not be the same in the case of real time cases too. This can be explained the price of a stock has a dependence on the previous data and another example can be said that of the lines in a paragraph which has relation the previous paragraph.



(Figure 2.8 Recurrent Neural Network [Source: https://www.analyticsvidhya.com/blog/RNN])

In the case of the RNN which can be called as a recurrent since the output has a dependence to the previous methods used. There is also another way of explaning about RNN which has a memory system which will facilitate the prediction more accurate. But we are not looking back in to all the memory byut instead we will be looking in to a very few.

RNN can be represented as shown in the figure 2.7.1it can be said as neural network with multilayer with each layer pointing to the observations at a particular point of time which can be denoted as ‘t

CHAPTER 3

RESEARCH METHODOLOGY

The following chapter details about the step by step process of understanding the data, analyzing the data and also implementing the machine learning algorithms in the data of HDFC Bank historical stock data.

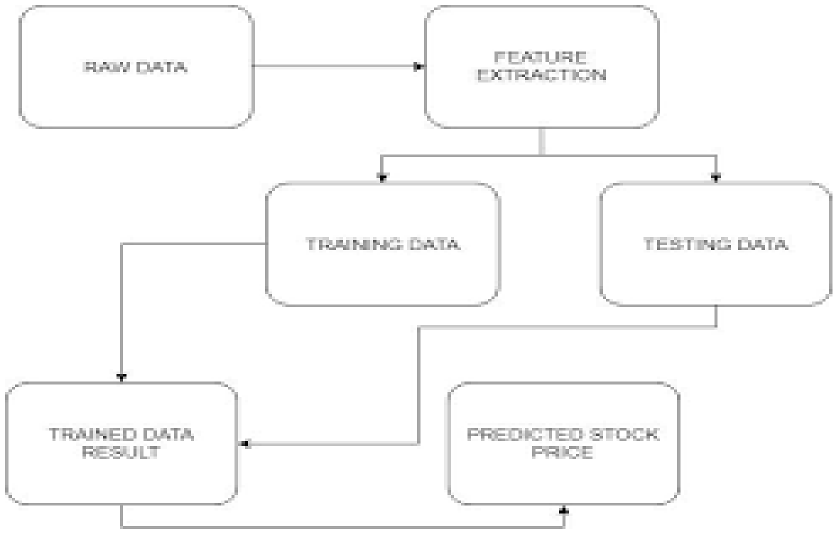
3.1 Proposed system

In the following system, we look in to the prediction of the price of the stock with the help of ML algorithms such as SVM, random forest method, linear regression, ARIMA, long short-term memory and do a comparative analysis on the same. We took data from previous year stock for the prediction and in-turn train the model too. We use libraries in ML like that of the numpy, for cleaning isolating and manipulating to make it in to a form which was ready for analysis. Other package used is scikit, which can be used for real time prediction and analysis. Here we are using the data set from the previous year data of stock market and from this data we are making 80% as the training data and the remaining data as the test data. We know that the basic idea behind the supervised learning method is to bring out the patterns and relationships in the data and to make a clear-cut analysis from that. We are also using other packages such as the pandas for the analysis too. Now processed on analyzed data frame can be further used for the feature extraction. The main features considered was the closing price of a day, opening price of a day.

3.2 System Architecture

The first step is the conversion of this raw data into processed data. This is done using feature extraction, since in the raw data collected there are multiple attributes but only a few of those attributes are useful for the purpose of prediction. So, the first step is feature extraction, where the key attributes are extracted from the whole list of attributes available in the raw dataset. Feature extraction starts from an initial state of measured data and builds derived values or features. These features are intended to be informative and non-redundant, facilitating the subsequent learning and generalization steps. Feature extraction is a dimensionality reduction process, where the initial set of raw variables is diminished to progressively reasonable features for ease of management, while still precisely and totally depicting the first informational collection.

The feature extraction process is then followed by a classification process where in which the data that was obtained after feature extraction is then divided into two different segments. Classification is the process of recognizing to which set of categories a new observation belongs. The use of training data set is to train the model, whereas the test data is to predict the accuracy of the model we use. The splitting was done on the basis that the training data has the more proportion than the test data.



(Fig 3.1 Proposed system model [Source: Stock Market Simulation Using Support Vector Machines
Rafael Rosillo, Javier Ginger1 and David De La Fuente])

The random forest algorithm uses a collection of different decision trees to understand the data. In layman terms, from the total number of decision trees in the forest, look for specific attributes in the data. This process is known as data splitting. Here in our case, since the end goal of our proposed system is to predict the price of the stock by analyzing its historical date.

3.3 Data Sourcing

Historic data dating back to the Initial Public Offering (IPO) of each stock can sourced from Yahoo! Finance, which provides daily Open, High, Low and Close prices and trade Volume

(OHLCV) data. Same data with higher accuracy can be sourced from Quandl. We carried out our project using the data from yahoo finance as it was an open source.

3.4 Data Pre-processing

The following parts details about data pre processing part which includes data cleaning, technical indicator extraction and finally the step by step of the proposed system.

3.4.1 Data Cleaning

Data is cleaned before use to prevent over-fitting to outliers or noise. Removal of useless values and missing or null values is a part of cleaning process. Then the series is smoothed exponentially, to prevent the impact of noise or outliers and make any clear trends that are easily identifiable. Based on research by *Dzikevicius*, the formula used for exponential smoothing in the application is as follows:

$$X_0' = X_0 \qquad (3.4.1)$$

$$X_i = \alpha \times X_i + (1 - \alpha) \times X_{i-1}' \qquad \forall\ i > 0 \qquad (3.4.2)$$

where X is the set of close prices for a stock under consideration, X' is the smoothed price set, i is the day under consideration and the α parameter determines the extent of smoothing, taking values between 0 and 1.

3.4.2 Technical Indicator Extraction

Using exponentially smoothed stock prices, some of the most commonly used technical indicators like MACD, RSI etc… are to be extracted, for use within feature vectors used for training and testing the models.

SIMPLE MOVING AVERAGE.

$$\text{SMA} = \frac{\sum_{i=d-n}^{d} x_i}{n} \qquad (3.4.3)$$

Where;
x = Closing Prices
i = Current Day

n = Number of Days
d = Day to calculate

Exponential Moving Average (EMA)

EMA0 = SMA

$$\mathrm{EMA}_i = (X_{i-1} - \mathrm{EMA}_{i-1})\frac{2}{n+1} + \mathrm{EMA}_{i-1} \qquad (3.4.4)$$

Where;
X = Closing Prices
i = Current Day
n = Number of Days

MACD

$$\mathrm{MACD}_d = \mathrm{EMA12}_d - \mathrm{EMA26}_d \qquad (3.4.5)$$

Where;
EMA12 = 12 Day EMA
EMA26 = 26 Day EMA
d = Day to calculate

Stochastic Oscillator (StoOsc)

$$\mathrm{StoOsc} = 100[(x - L)\ /\ (H - L)] \qquad (3.4.6)$$

Where;
x = Current closing price
Low14 = Lowest price of last 14 days
High14 = Highest price of last 14 days

Relative Strength Index (RSI)

$$\mathrm{RSI} = 100 - [(100/(1 + \mathrm{RS})] \qquad (3.4.7)$$

Where;
RS = (AVG ($\mathrm{Gains}_{i-14\ldots i}$))/(AV G($\mathrm{Losses}_{i-14\ldots i}$))
Gains = Price increase sum over last 14 days
Losses = Price decrease sum over last 14 days

Other indicators like Williams % R, Accumulation Distribution Line(AD), On-Balance Volume(OBV), Commodity Channel Index and Average Direction Index(ADI) are also used in the model vector for the price prediction. But in the present market condition due to the pandemic Covid-19, the financial markets are overrun by sentiments and hence these technical factors do not have much role in the price determination. So in our project we are mainly focusing on historical data to predict the future price and all other technical factors are excluded.

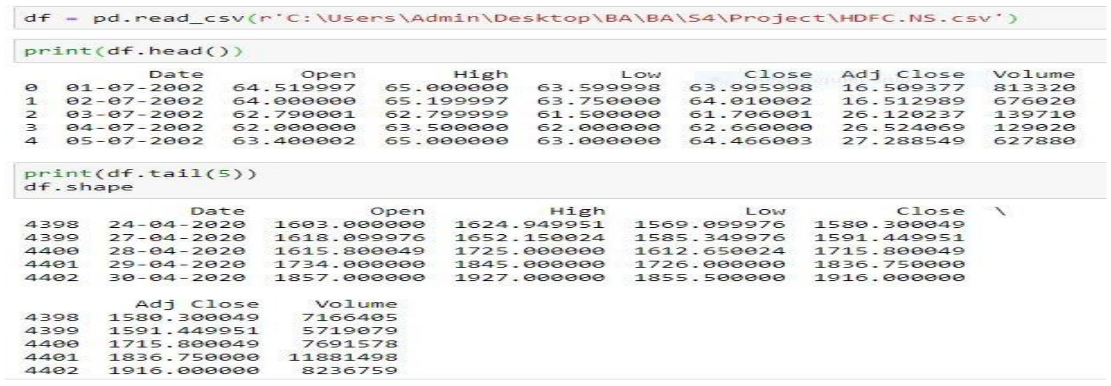
3.5 Step by step implementation of the proposed system

The following methods give in detail about each of the machine learning algorithms used in the prediction of stock price.

3.5.1 Prediction using Linear Regression Model

- Import (Install the libraries if not installed already) the following libraries needed for the program:
    - Import quandl
    - Import numpy as np
    - Import panads as pd
    - From sklearn.linear_model import LinearRegression
    - From sklearn.model_selection import train_test_split
- Importing the stock data from quandl and pandas(pd)
    - df = quandl.get("WIKI/GOOGL", start_date="1970-01-01", end_date="2020-03-31")
      or
    - df = quandl.get("WIKI/GOOGL")
        - Stock data of Facebook("WIKI/FB"), Amazon("WIKI/AMZN") and Tesla("WIKI/TSLA") were also taken to analyze the program.
- For the project Indian stock HDFC data was used. The data downloaded from yahoo finance website was cleaned and saved in csv format as is imported as:
    - df = pd.read_csv(r'C:\Users\Admin\Desktop\BA\BA\S4\Project\HDFC.NS.csv')

- print(df.head(5))
- df.shape – for getting the number of rows and columns in the data-frame.



```
df = pd.read_csv(r'C:\Users\Admin\Desktop\BA\BA\S4\Project\HDFC.NS.csv')
```

```
print(df.head())
```

```
         Date       Open       High        Low      Close  Adj Close  Volume
0  01-07-2002  64.519997  65.000000  63.599998  63.995998  16.509377  813320
1  02-07-2002  64.000000  65.199997  63.750000  64.010002  16.512989  676020
2  03-07-2002  62.790001  62.799999  61.500000  61.706001  26.120237  139710
3  04-07-2002  62.000000  63.500000  62.000000  62.660000  26.524069  129020
4  05-07-2002  63.400002  65.000000  63.000000  64.466003  27.288549  627880
```

```
print(df.tail(5))
df.shape
```

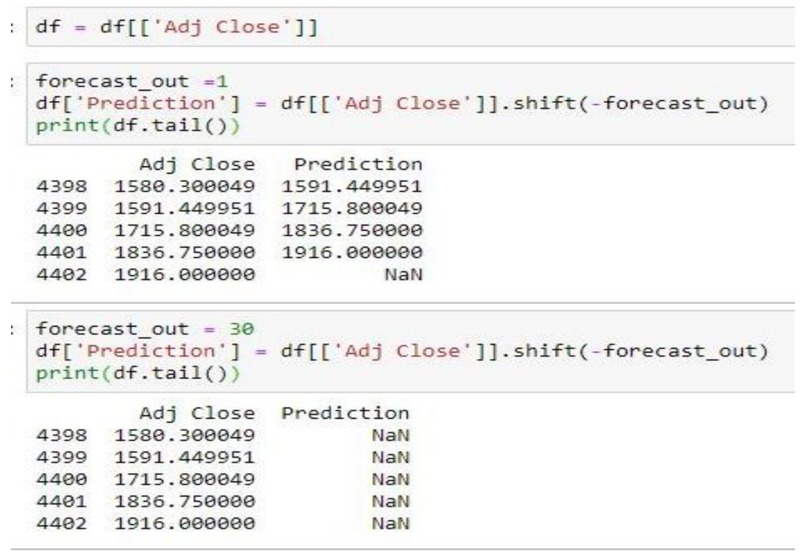
```
            Date         Open         High          Low        Close  \
4398  24-04-2020  1603.000000  1624.949951  1569.099976  1580.300049
4399  27-04-2020  1618.099976  1652.150024  1585.349976  1591.449951
4400  28-04-2020  1615.800049  1725.000000  1612.650024  1715.800049
4401  29-04-2020  1734.000000  1845.000000  1726.000000  1836.750000
4402  30-04-2020  1857.000000  1927.000000  1855.500000  1916.000000

        Adj Close    Volume
4398  1580.300049   7166405
4399  1591.449951   5719079
4400  1715.800049   7691578
4401  1836.750000  11881498
4402  1916.000000   8236759
```

(Fig 3.2 view of the data used for prediction)

- Creating the new data set with the adjusted close price as we are predicting the future price only using the historical data.
  - df = df[['Adj Close']]
- Creating a variable for predicting the future stock price called forecast_out for 'n' coming days. forecast_out is an independent variable.
  - forecast_out = 1: - predicts the next value, i.e. if stock price data till date is given then the stock price of the next day will be predicted.

- A new column, dependent variable, predicted value is also created. This will be shifted 'n' units up.
  - df ['Prediction'] = df[['Adj Close']]. shift(-forecast_out)
  - print(df.head())

  By changing the value of forecast_out value we will get the forecasted value for a certain time period.



```
df = df[['Adj Close']]
```

```
forecast_out =1
df['Prediction'] = df[['Adj Close']].shift(-forecast_out)
print(df.tail())
```

```
        Adj Close   Prediction
4398  1580.300049  1591.449951
4399  1591.449951  1715.800049
4400  1715.800049  1836.750000
4401  1836.750000  1916.000000
4402  1916.000000          NaN
```

```
forecast_out = 30
df['Prediction'] = df[['Adj Close']].shift(-forecast_out)
print(df.tail())
```

```
        Adj Close  Prediction
4398  1580.300049         NaN
4399  1591.449951         NaN
4400  1715.800049         NaN
4401  1836.750000         NaN
4402  1916.000000         NaN
```

(Fig 3.3 Table showing the forecast for a time period)

- The adjusted close price of succeeding day is predicted in the prediction column. But for the last day there is no prediction. Which is what we will have to forecast using linear regression algorithm.
- For this the data-frame is converted to a numpy array to create a separate data set. For this the independent and dependent variables x and y respectively are created. Here the last 'n' rows will be removed(forecate_out) from the numpy array.
    - 'x = np.array(df(['Adj Close']))'

      'x = x[:-forecast_out]'

      'print(x)'
    - output will be:

```
[[  16.509377]
 [  16.512989]
 [  26.120237]
 ...
 [2036.25   ]
[1875.699951]
[2066.800049]]
```

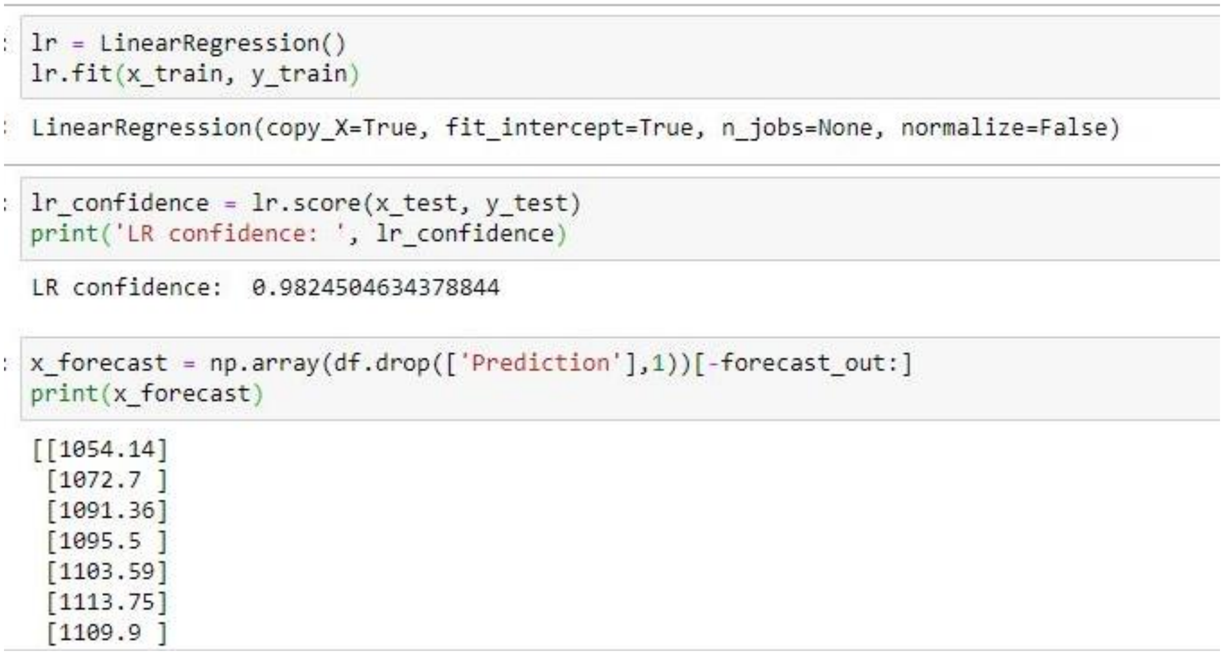
- “y = np.array(df['Prediction'])”

  “y = y[:-forecast_out]”

  “print(y)”

  [26.477501   26.200666   25.829853 ... 1715.800049 1836.75, 1916.]

- Now the data will be split to training and test data. Splitting can be done using the package installed. Here the data was split in the ratio 8:2(Training(80%):Testing(20%))(the ratio whic h yields best outcome should be used).
  - “x_train, x_test, y_train, y_test = train_test_split(x,y,test_size=0.20)”
- Then the linear regression model is created and trained.
  - “lr = LinearRegression ()”
    “lr.fit(x_train, y_train)”
- The efficiency of the model can be checkd from the coefficient of determination R^2, with the highest value of 1.
  - “lr_confidence = lr.score(x_test, y_test)”
    “print('LR confidence: ', lr_confidence)”
- A new variable(x_forecast) is created, which is the values which we wanted to predict.
  - “x_forecast = np.array(df.drop(['Prediction'],1))[-forecast_out:]”
    “print(x_forecast)”



```
lr = LinearRegression()
lr.fit(x_train, y_train)
```

```
LinearRegression(copy_X=True, fit_intercept=True, n_jobs=None, normalize=False)
```

```
lr_confidence = lr.score(x_test, y_test)
print('LR confidence: ', lr_confidence)
```

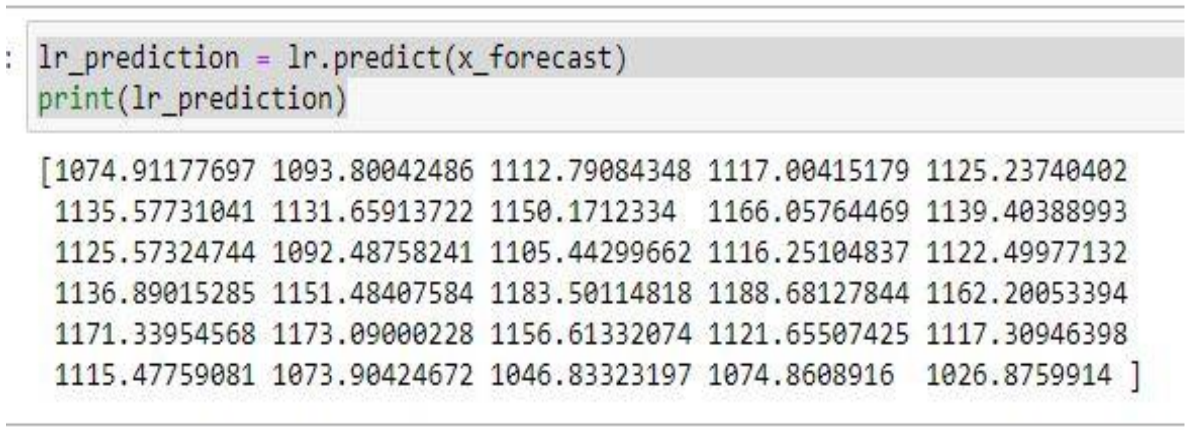
```
LR confidence:  0.9824504634378844
```

```
x_forecast = np.array(df.drop(['Prediction'],1))[-forecast_out:]
print(x_forecast)
```

```
[[1054.14]
 [1072.7 ]
 [1091.36]
 [1095.5 ]
 [1103.59]
 [1113.75]
 [1109.9 ]
```

(Fig 3.4 Implementing of linear regression model)

- Finally the regression model predicts the value of the stock for the coming 30 days.
    - “lr_prediction = lr.predict(x_forecast)”
      “print(lr_prediction)”



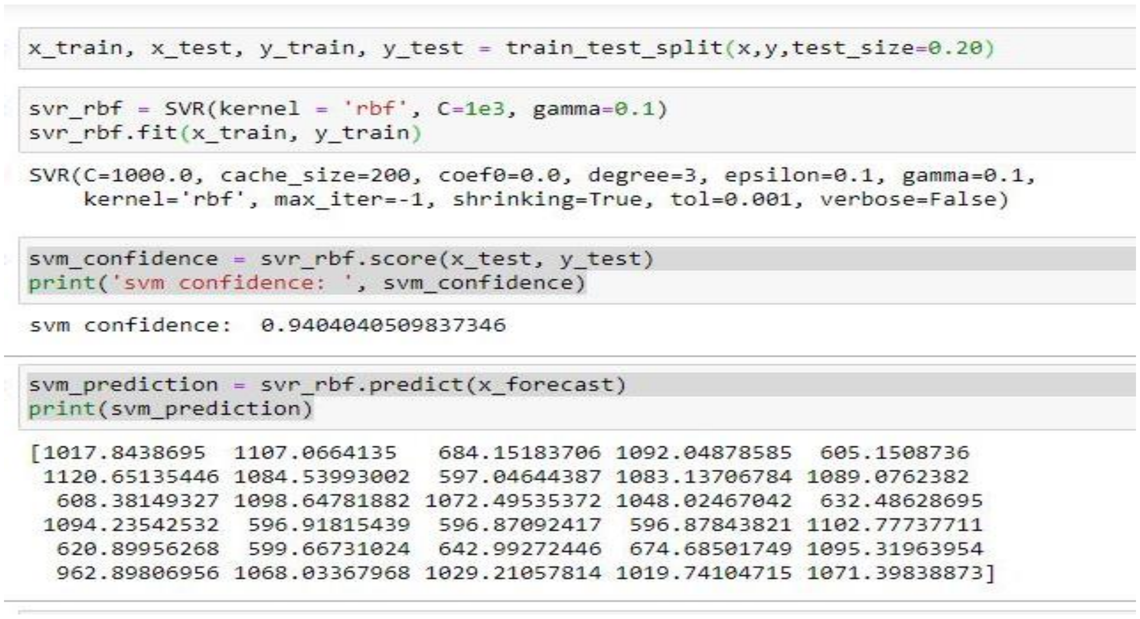
```
: lr_prediction = lr.predict(x_forecast)
  print(lr_prediction)

  [1074.91177697 1093.80042486 1112.79084348 1117.00415179 1125.23740402
   1135.57731041 1131.65913722 1150.1712334  1166.05764469 1139.40388993
   1125.57324744 1092.48758241 1105.44299662 1116.25104837 1122.49977132
   1136.89015285 1151.48407584 1183.50114818 1188.68127844 1162.20053394
   1171.33954568 1173.09000228 1156.61332074 1121.65507425 1117.30946398
   1115.47759081 1073.90424672 1046.83323197 1074.8608916  1026.8759914 ]
```

(Fig 3.5 Values of prediction from Linear regression Model)

3.5.2 Prediction Using Support Vector Machine (SVM)

The steps till the training and testing is same for both linear regression model and SVR regression.

- Training the SVR regressor:
    - “x_train, x_test, y_train, y_test = train_test_split(x,y,test_size=0.20)”
    - “svr_rbf = SVR(kernel = 'rbf', C=1e3, gamma=0.1)”
      “svr_rbf.fit(x_train, y_train)”
    - “svm_confidence = svr_rbf.score(x_test, y_test)”
      “print('svm confidence: ', svm_confidence)”



```
x_train, x_test, y_train, y_test = train_test_split(x,y,test_size=0.20)

svr_rbf = SVR(kernel = 'rbf', C=1e3, gamma=0.1)
svr_rbf.fit(x_train, y_train)

SVR(C=1000.0, cache_size=200, coef0=0.0, degree=3, epsilon=0.1, gamma=0.1,
    kernel='rbf', max_iter=-1, shrinking=True, tol=0.001, verbose=False)

svm_confidence = svr_rbf.score(x_test, y_test)
print('svm confidence: ', svm_confidence)

svm confidence:  0.9404040509837346

svm_prediction = svr_rbf.predict(x_forecast)
print(svm_prediction)

[1017.8438695  1107.0664135   684.15183706 1092.04878585  605.1508736
 1120.65135446 1084.53993002  597.04644387 1083.13706784 1089.0762382
  608.38149327 1098.64781882 1072.49535372 1048.02467042  632.48628695
 1094.23542532  596.91815439  596.87092417  596.87843821 1102.77737711
  620.89956268  599.66731024  642.99272446  674.68501749 1095.31963954
  962.89806956 1068.03367968 1029.21057814 1019.74104715 1071.39838873]
```

(Fig 3.6 SVM prediction steps and results)

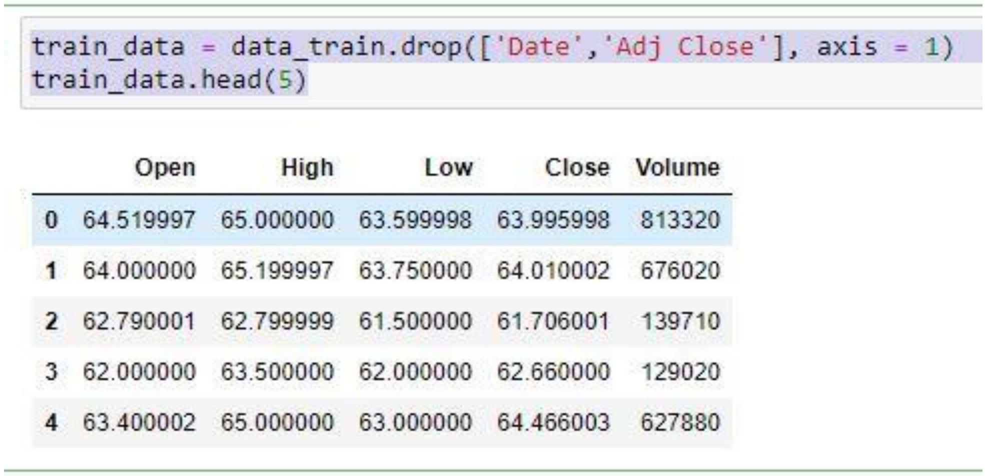
Usually SVM model is one of the most opted machine learning algorithm used for predicting the stock prices, but here linear regression model is giving better accuracy than the SVR model. This might be due to the volatility existing in the market due to the corona virus pandemic. As the investor and traders round the globe are in panic, the stock prices are reflecting the sentiments making it hard to predict using technical indicators.

3.5.3 Prediction using RNN-LSTM

- Packages:
    - ‘import numpy as np’
      ‘import matplotlib.pyplot as plt’
      ‘import pandas as pd’
      ‘from sklearn.preprocessing import MinMaxScaler’
      ‘from tensorflow.keras import Sequential’
      ‘from tensorflow.keras.layers import Dense, LSTM, Dropout’
- Dat-apreprocessing:
        - “data=pd.read_csv (r'C:\Users\Admin\Desktop\BA\BA\S4\Project\HDFC.NS.csv')”
          “data.tail()”
    - Splitting the data to training and testing data sets.

- “data_train = data[data['Date']<'2020-01-01'].copy()”
  “data_train”
- “data_test = data[data['Date']>='2020-01-01'].copy()”
  “data_test”

- Dropping Date and Adj Close from the training data(cleaning)
  - “train_data = data_train.drop(['Date','Adj Close'], axis = 1)”
    “train_data.head(5)”



```
train_data = data_train.drop(['Date','Adj Close'], axis = 1)
train_data.head(5)
```

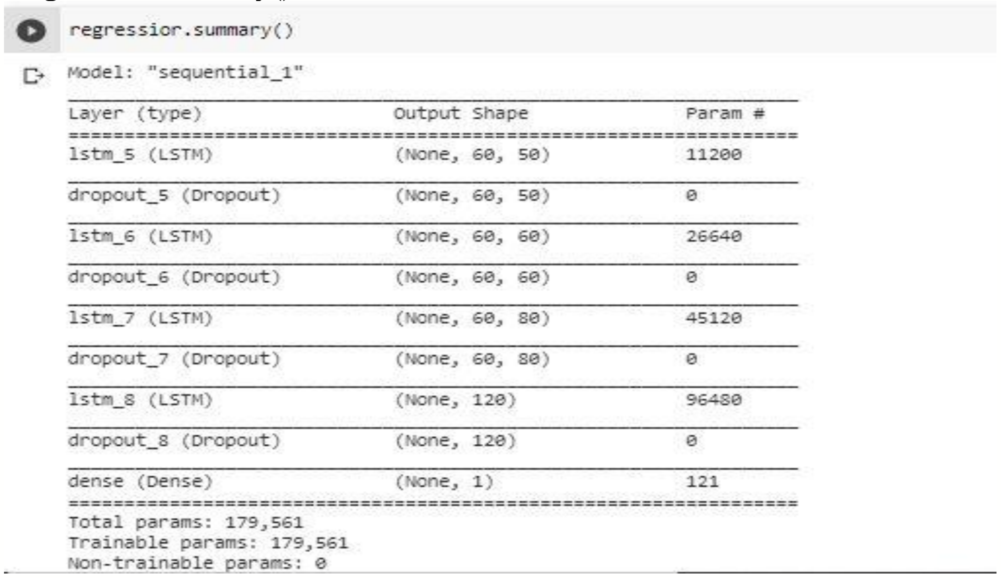
	Open	High	Low	Close	Volume
0	64.519997	65.000000	63.599998	63.995998	813320
1	64.000000	65.199997	63.750000	64.010002	676020
2	62.790001	62.799999	61.500000	61.706001	139710
3	62.000000	63.500000	62.000000	62.660000	129020
4	63.400002	65.000000	63.000000	64.466003	627880

(Fig 3.7 Tabular form of the training data used for LSTM RNN)

- The model will predict the open price of 61st day using previous 60 days’ open price and 62nd days’ price using previous 60 days’ open price and so on.
- Scaling – the training data is scaled to the default min. of zero and max. of one using MinMaxScaler before training the set.
  - “scaler = MinMaxScaler()”
    “train_data = scaler.fit_transform(train_data)”
    “train_data”
- Creating the training set:
  - “x_train = []”
  - “y_train = []”
  - “for i in range(60, train_data.shape[0]):”
    “x_train.append(train_data[(i-60):i])”
    “y_train.append(train_data[i , 0])”
- changing the training sets to numpy array:
  - “x_train, y_train = np.array(x_train), np.array(y_train)”

- Building the LSTM Model:
  - Keras model is imported from tensorflow in-order to build themodel.
    - “from tensorflow.keras import Sequential”

      “from tensorflow.keras.layers import Dense, LSTM, Dropout”
  - A regressior function is created using Sequential, as we are predicting a continuous value.
    - “regressior = Sequential()”
  - Then LSTM of 4 layers is added to the regressior. And a dropout is added to generalize our model.
    - “regressior.add(LSTM(units=50, activation='relu',return_sequences=True, input_shape=(x_train.shape[1],5)))”
    - “regressior.add(Dropout(0.2))”
    - “regressior.add(LSTM(units=60, activation='relu',return_sequences=True))”
      “regressior.add(Dropout(0.3))”
    - “regressior.add (LSTM (units=80, activation='relu',return_sequences=True))”
      “regressior.add (Dropout(0.4))”
  - The sequence return will not be present in the final layer.
    - “regressior.add(LSTM(units=120, activation='relu'))”
      “regressior.add(Dropout(0.5))”
  - To get output of our CNN the Dense layer is added.
    - “regressior.add(Dense(units = 1))”
  - “regressior.summary()”



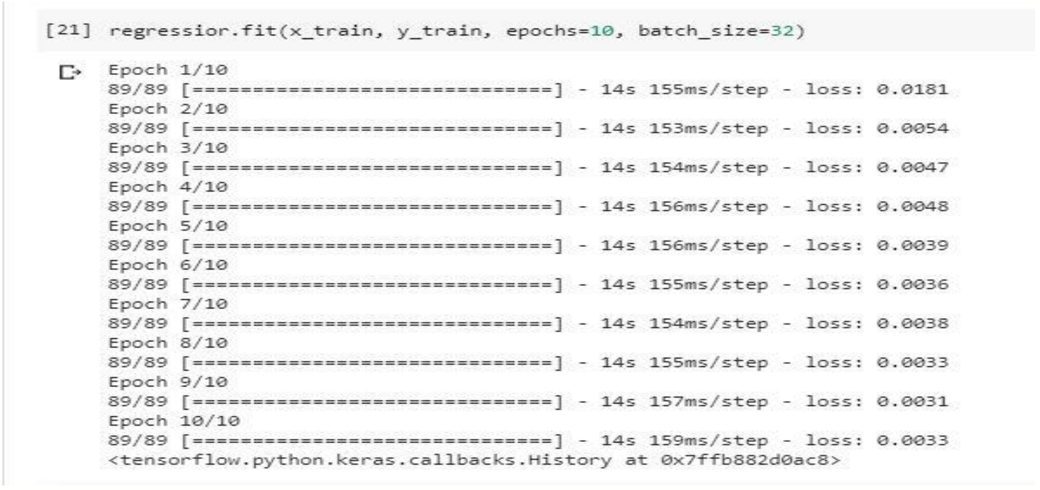
```
regressior.summary()

Model: "sequential_1"
_________________________________________________________________
Layer (type)                 Output Shape              Param #
=================================================================
lstm_5 (LSTM)                (None, 60, 50)            11200
_________________________________________________________________
dropout_5 (Dropout)          (None, 60, 50)            0
_________________________________________________________________
lstm_6 (LSTM)                (None, 60, 60)            26640
_________________________________________________________________
dropout_6 (Dropout)          (None, 60, 60)            0
_________________________________________________________________
lstm_7 (LSTM)                (None, 60, 80)            45120
_________________________________________________________________
dropout_7 (Dropout)          (None, 60, 80)            0
_________________________________________________________________
lstm_8 (LSTM)                (None, 120)               96480
_________________________________________________________________
dropout_8 (Dropout)          (None, 120)               0
_________________________________________________________________
dense (Dense)                (None, 1)                 121
=================================================================
Total params: 179,561
Trainable params: 179,561
Non-trainable params: 0
```

(fig 3.8 Regression summary of the model)

Here we have 179,561 parameters in our LSTM model for predicting the future stock price of HDFC. Now we have to compile the model for further processes.

- Compiling the model:
    - “regressior.compile(optimizer='adam', loss='mean_squared_error')”

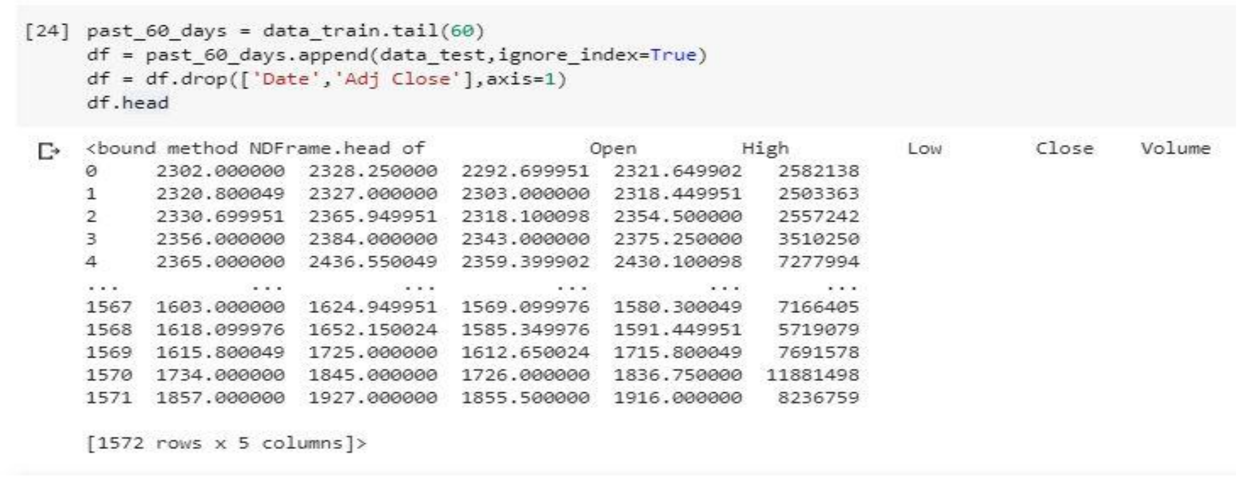
- Training the model:
    - “regressior.fit(x_train, y_train, epochs=10, batch_size=32)”
      [Loss can be minimized by changing the number of units and dropouts in the training data.



```
[21] regressior.fit(x_train, y_train, epochs=10, batch_size=32)

     Epoch 1/10
     89/89 [==============================] - 14s 155ms/step - loss: 0.0181
     Epoch 2/10
     89/89 [==============================] - 14s 153ms/step - loss: 0.0054
     Epoch 3/10
     89/89 [==============================] - 14s 154ms/step - loss: 0.0047
     Epoch 4/10
     89/89 [==============================] - 14s 156ms/step - loss: 0.0048
     Epoch 5/10
     89/89 [==============================] - 14s 156ms/step - loss: 0.0039
     Epoch 6/10
     89/89 [==============================] - 14s 155ms/step - loss: 0.0036
     Epoch 7/10
     89/89 [==============================] - 14s 154ms/step - loss: 0.0038
     Epoch 8/10
     89/89 [==============================] - 14s 155ms/step - loss: 0.0033
     Epoch 9/10
     89/89 [==============================] - 14s 157ms/step - loss: 0.0031
     Epoch 10/10
     89/89 [==============================] - 14s 159ms/step - loss: 0.0033
     <tensorflow.python.keras.callbacks.History at 0x7ffb882d0ac8>
```

(Fig 3.9 Training data set for LSTM model)

- Testing the data:
    - Now we have to test the model and predict the future stock price of HDFC.
    - For predicting 61st days’ stock price we need the open price of past 60 days, so we have get this 60 days’ data from the training set and will add this to our test data. Then the ‘Date’ and ‘Adj Close’ will be dropped from the test data.
        - “Past_60_days = data_train.tail(60)”
          “df = past_60_days.append(data_test,ignore_index=True)”
          “df = df. drop(['Date','Adj Close'], axis=1)”
          “df. Head”



```
[24] past_60_days = data_train.tail(60)
     df = past_60_days.append(data_test,ignore_index=True)
     df = df.drop(['Date','Adj Close'],axis=1)
     df.head
```

```
<bound method NDFrame.head of                Open         High          Low        Close     Volume
0     2302.000000  2328.250000  2292.699951  2321.649902   2582138
1     2320.800049  2327.000000  2303.000000  2318.449951   2503363
2     2330.699951  2365.949951  2318.100098  2354.500000   2557242
3     2356.000000  2384.000000  2343.000000  2375.250000   3510250
4     2365.000000  2436.550049  2359.399902  2430.100098   7277994
...           ...          ...          ...          ...       ...
1567  1603.000000  1624.949951  1569.099976  1580.300049   7166405
1568  1618.099976  1652.150024  1585.349976  1591.449951   5719079
1569  1615.800049  1725.000000  1612.650024  1715.800049   7691578
1570  1734.000000  1845.000000  1726.000000  1836.750000  11881498
1571  1857.000000  1927.000000  1855.500000  1916.000000   8236759

[1572 rows x 5 columns]>
```

(Fig 3.10 Testing data set set for LSTM model)

- The test data is also scaled.
    - “inputs = scaler.transform(df)”

- Preparing the test set
    - “x_test = []”

      “y_test = []”

      “for i in range(60, inputs.shape[0]):”

        “x_test.append(inputs[(i-60):i])”

        “ y_test.append(inputs[i,0])”

    - “x_test, y_test = np.array(x_test), np.array(y_test)”
      “x_test.shape, y_test.shape”
        - ((1512, 60, 5), (1512,))

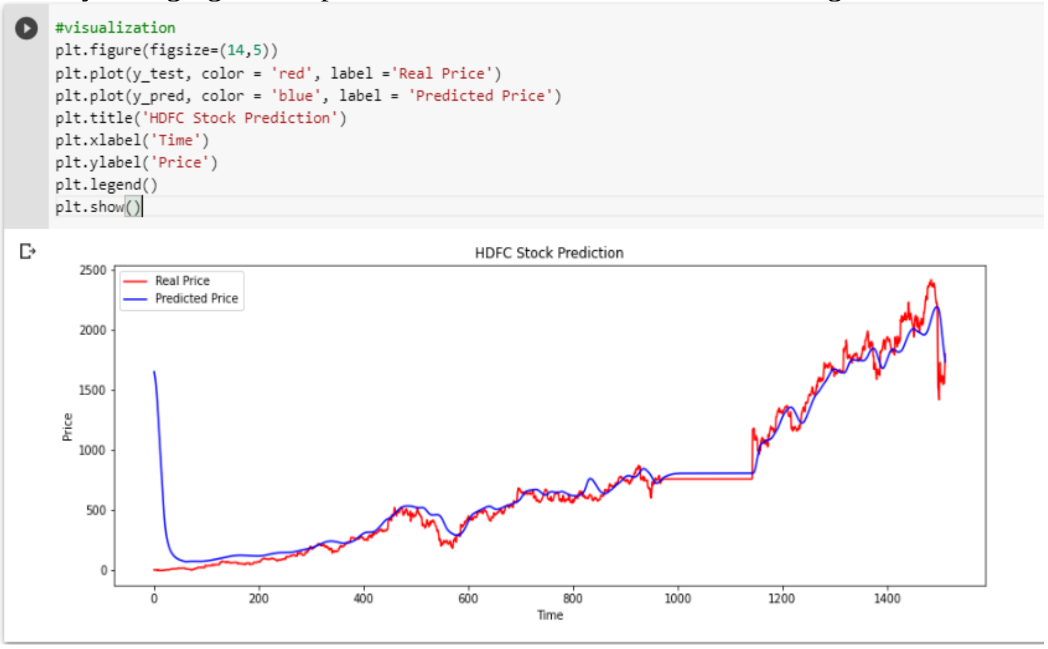
- The Prediction:
    - Finally, the model will be predicting the stock prices using the test data. But the predicted price will be a scaled value between 0 and 1.

    - “y_pred = regressior.predict(x_test)”
      “scaler.scale”
      “scale = 1/4.11810732e-04”
      “y_pred = y_pred*scale”

“y_predict”

```
[[1651.4244]
 [1627.955 ]
 [1599.7715]
 ...
 [1799.9218]
 [1765.2004]
 [1733.847 ]]
```

- Visualization
  The predicted value and the real price are almost same in this model. More accuracy can be obtained by changing the dropouts and number of units in the training data set.



```
#visualization
plt.figure(figsize=(14,5))
plt.plot(y_test, color = 'red', label ='Real Price')
plt.plot(y_pred, color = 'blue', label = 'Predicted Price')
plt.title('HDFC Stock Prediction')
plt.xlabel('Time')
plt.ylabel('Price')
plt.legend()
plt.show()
```

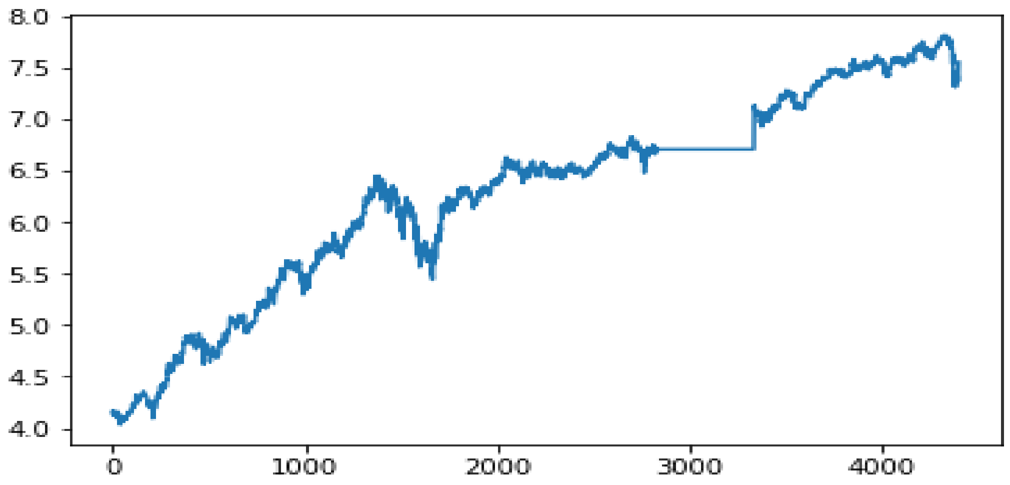
(Fig 3.11 Plot of real price vs predicted price)

3.5.4 ARIMA model

- Installing the packages required:
  - “import numpy as np”
    “import matplotlib.pyplot as plt”
    “import matplotlib.mlab as mlab”
    “import pandas as pd”
    “import math”
    “from statsmodels.tsa.stattools import acf, pacf”
    “import statsmodels.tsa.stattools as ts”

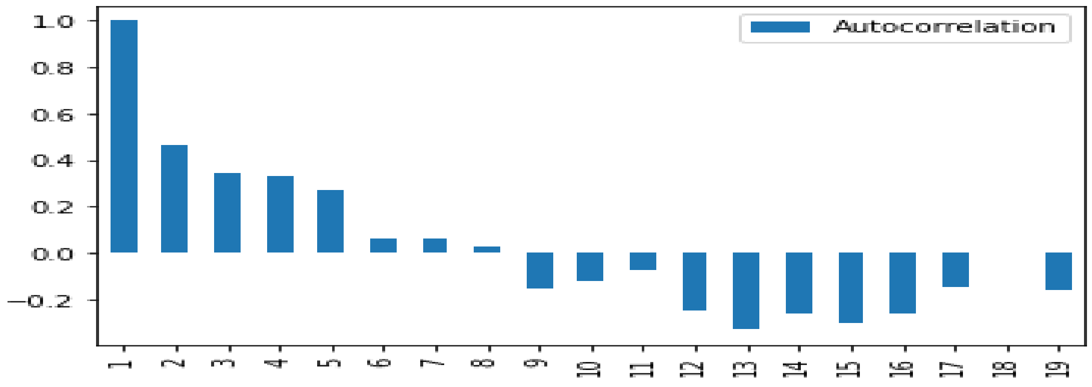
“from statsmodels.tsa.arima_model import ARIMA”

- Data is loaded, removed all columns except close price and converted to log format and is plotted.
    - “df = pd.read_csv(r’/content/HDFC.NS.csv’)”
      “df.head(5)”
      “price = df[‘Close’]”
      “df”
      “lnprice = np.log(price)”
      “lnprice”
      “plt.plot(lnprice)”
      “plt.show()”



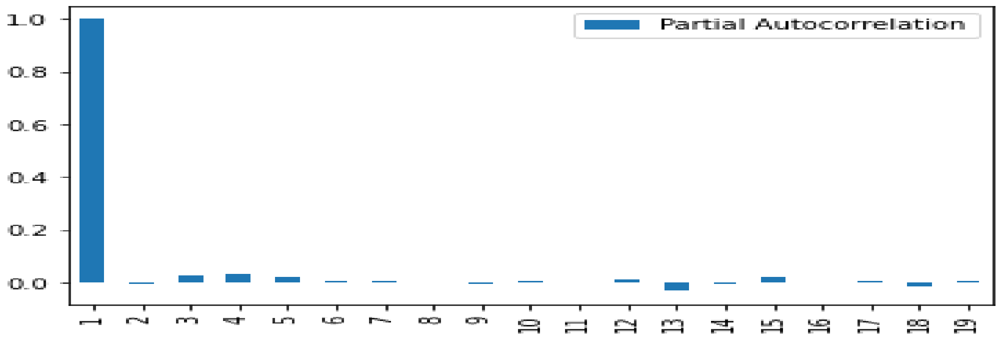
(Fig 3.12 Plot of the HDFC data used for prediction)

- In an ARIMA model the data must be stationary to get an accurate prediction. So we have to check the autocorrelation and partial autocorrelation of the data set taken for the model.
    - “acf_1 = acf(lnprice[1:20])”
      “test_df = pd.DataFrame([acf_1]).T”
      “test_df.columns = ['Autocorrelation']”
      “test_df. index += 1”
      “test_df. Plot(kind='bar')”
      “plt. show ()”



(Fig 3.13 Graph showing auto correlation)

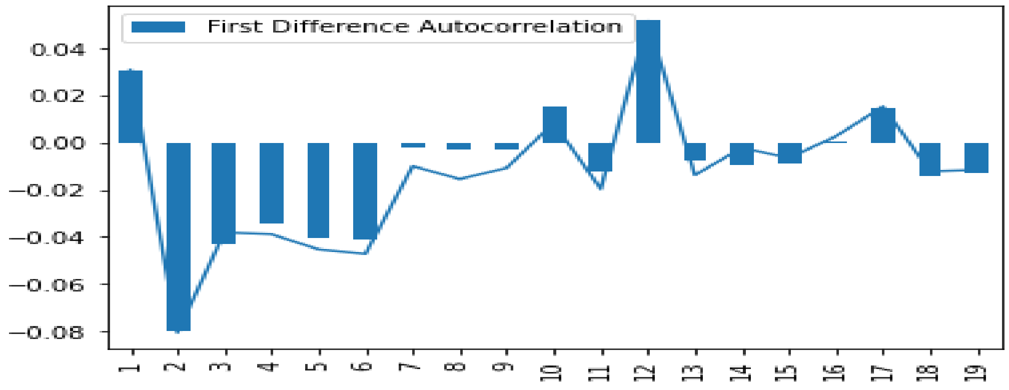
- Partial Autocorrelation.
    - "pacf_1 = pacf(lnprice)[1:20]"
      "test_df = pd.DataFrame([pacf_1]).T"
      "test_df. columns = ['Partial Autocorrelation']"
      "test_df. index += 1"
      "test_df. plot(kind='bar')"
      "plt.show()"



(fig 3.14 Graph showing partial correlation)

- From the autocorrelation and partial auto-correlation we can conclude that our data is essentially of the AR1 format. So we have to get the first difference of the data to make it stationary before training the data. Now we will carry out Dicky Fuller test on the data.
    - "result = ts.adfuller(lnprice, 1)"
      "result"
      "lnprice_diff = lnprice-lnprice.shift()"
      "diff = lnprice_diff.dropna()"

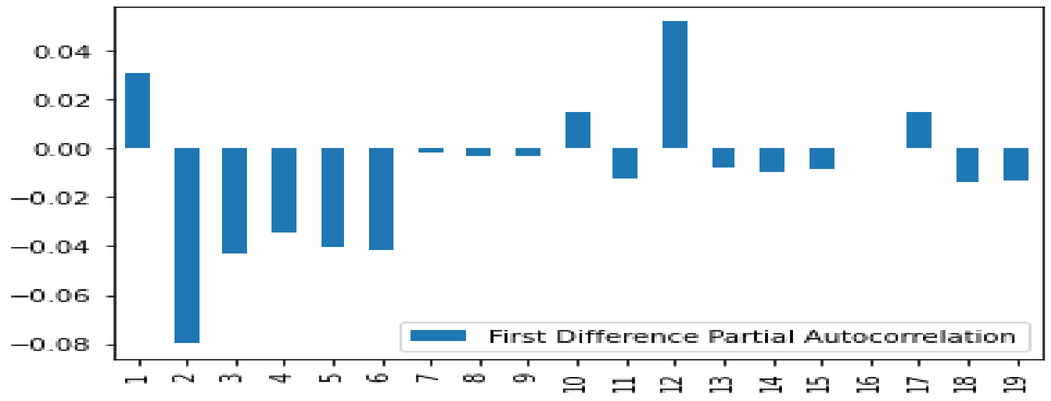
“acf_1_diff = acf(diff)[1:20]”
“test_df = pd.DataFrame([acf_1_diff]).T”
“test_df.columns = ['First Difference Autocorrelation']”
“test_df.index += 1”
“test_df.plot(kind='bar')”
“pacf_1_diff = pacf(diff)[1:20]”
“plt.plot(pacf_1_diff)”
“plt.show()”



(Fig 3.15 Graph showing first difference autocorrelation)

The zig-zag movement is of the stationary data which we want our data to be.

- Dickey Fuller test on partial autocorrelation.
    - “test_df = pd.DataFrame([acf_1_diff]).T”
      “test_df.columns = ['First Difference Partial Autocorrelation']”
      “test_df.index += 1”
      “test_df.plot(kind='bar')”
      “plt.show()”



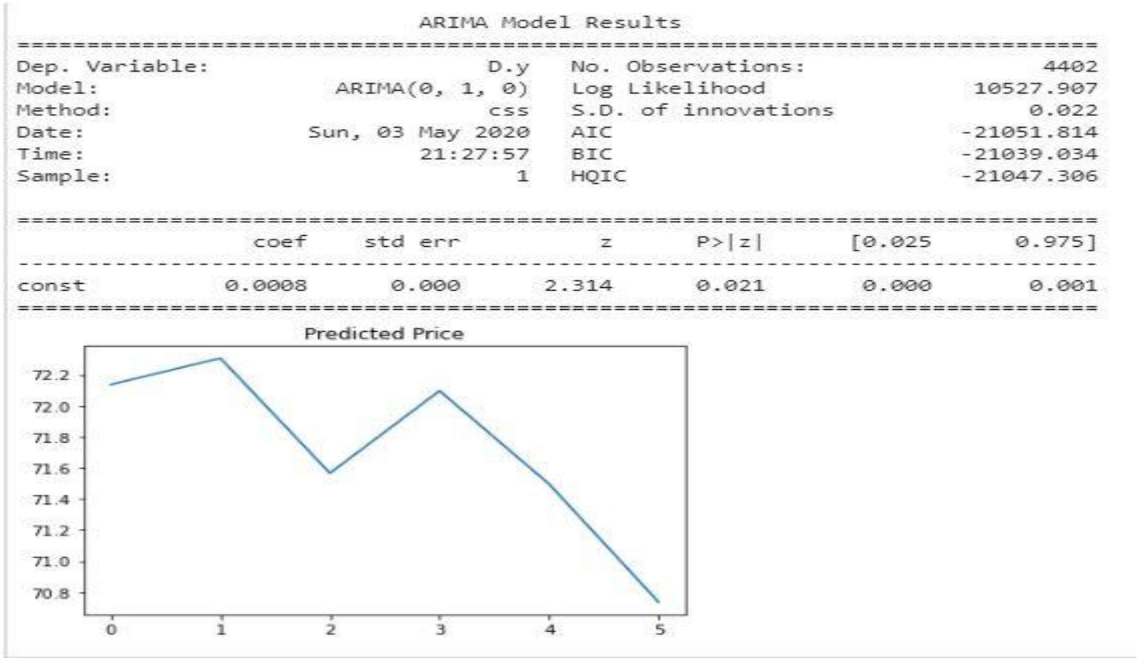
(Fig 3.16 Graph showing first difference partial autocorrelation)

The zig-zag movement confirms stationarity of the data.

The Prediction:

- “price_matrix = lnprice.to_numpy()”
  “model = ARIMA(price_matrix, order=(0,1,0))”
  “model_fit = model.fit(disp=0)”
  “print(model_fit.summary())”
  “predictions = model_fit.predict(122, 127, typ =’levels’)”
  “predictions”
  “predictionsadjusted=np.exp(predictions)”
  “predictionsadjusted”
  “plt.plot(predictionsadjusted)”
  “plt.title(‘Predicted Price’)”
  “plt.show()”

- The prediction for the next five trading sessions and the ARIMA results:



```
                              ARIMA Model Results
==============================================================================
Dep. Variable:                    D.y   No. Observations:                 4402
Model:                 ARIMA(0, 1, 0)   Log Likelihood               10527.907
Method:                           css   S.D. of innovations              0.022
Date:                Sun, 03 May 2020   AIC                         -21051.814
Time:                        21:27:57   BIC                         -21039.034
Sample:                             1   HQIC                        -21047.306

==============================================================================
                 coef    std err          z      P>|z|      [0.025      0.975]
------------------------------------------------------------------------------
const          0.0008      0.000      2.314      0.021       0.000       0.001
==============================================================================
```

(Fig 3.17 ARIMA model results and graphs)

The ARIMA model gives an accuracy of 97.5%.

CHAPTER 4

Results

A detailed comparative analysis of each of the machine learning algorithm was done and results are then compared to find the best fitted algorithm for the prediction was identified

It was evident that all the methods that we used for predicting the stock value was successful. All the methods gave similar result. From all the methods that we have used for prediction Long short-term method and ARIMA gives the best output.

The algorithms will be a great asset to investors for investing in stock market since it's is trained on huge collection of historical data and has been chosen after being tested on sample data. The project demonstrates machine learning algorithm to predict the stock value with more accuracy and efficiency as compared to previously used machine learning algorithms. Due to the volatility in the market as a result of the on-going global health emergency the prediction is not accurate as expected. This is mainly because of the high volatility globally due to the panic situation prevailing among the investors.

After the detailed comparative study on each of the algorithm it was found that ARIMA and LSTM are the most opted methods for stock market predictions, but the level of uncertainty existing in the market makes LR model better prediction with an accuracy of 98.2%.

(Table:1 algorithm prediction accuracy)

Model	Accuracy
Linear Regression	98.26%
SVM	94.04%
LSTM	96.67%
ARIMA	97.50%

CHAPTER 5

Findings and Recommendations

Machine learning algorithms used in predicting the price of the stock have a great impact in this emerging economy. In the project we are using different machine learning algorithms such as Support vector machine, Random forest algorithm, Regression, ARIMA models, Long short-term memory to draw out a comparative analysis on each of the algorithm. Each algorithm is different in their own way in comparing to the analyzing part. But each algorithm is very much reliable in giving the results.

Regression model is quite simple and easy to implement. One main problem associated with regression algorithm is that the model overfits to the date and month column. Instead of taking into account the previous values from the prediction, the model will consider the value from the same date month ago or may be a year ago. In the case of ARIMA model we use past data to understand the pattern in the time series. This model shows an increasing trend in the series. Although this prediction method is far much better compared to other. As per the case of SVM and random forest also shows much confidence in predicting the stock market price as these methods also do take into account of the historical data. The LSTM model can be tuned for various parameters such as changing the number of LSTM layers, adding dropout values or increasing the number of epochs. As far as the implementation LSTM shows better accuracy and reliability in the results.

Based on the above findings it is evident have each method is suitable for prediction with each having its on specific features. All this prediction is based on the previous data or the historical data which in turn is a time series analysis of the data. Since these are based on the historical data we are not in cooperating the fact of dynamic changes in price value of share based on the real time news and economic factors.

Further future scopes include more parameter, ratios, financial aspects etc. More the number of parameters considers more the accuracy in the prediction. We can also incorporate public comments and analysis from the public and corporate employees or business news.

CHAPTER 6

Limitation of Study

After thoroughly going through each of the machine learning algorithm for predicting the stock price we know that we use historical data for the prediction. There are not only plus points but also some negative points associated with each of the algorithm. These are the ones we are trying to overcome in our future studies on the project. One of the main drawbacks includes the usage of historical data and not incorporating the real time data. These predictions are all based of the data we have already collected so the results will be based on that only. The dynamics of real time such that of news regarding the share prices when there is merging of company, any news associated with company which is listed in the stock market or any other economic and business news that has impact on stock market value.

Along with these there are also many other limitations for the study. The existing system fails when there are rare outcomes or predictors, as the algorithm follows a bootstrap sampling. It was identified that the stock price is unpredictable when the traditional classifier is used. The existing system reported high predictive values, by selecting appropriate time period for their experiment to obtain high predictive scores. The existing system does not perform well when there is change in the operating environment. The current system does not give focus on external events in the environment, like news events or social media. Its exploits only data source, thus we can say its highly biased. The existing system need some form of input interpretation, thus need scaling. It doesn’t exploit data pre-processing technique to remove inconsistency and incompleteness of the data.

REFERENCE

1. Data source for the comparative study https://in.finance.yahoo.com/quote/HDFC.NS/
2. Journal by S Prasanna School of Information Technology, VIT University, Vellore and Dr. D Ezhilmaran School of Advanced Science, VIT University, Vellore.An analysis of stock market prediction using Data Mining Techniques.
3. Loke K.S. Faculty of Engineering, Computing and Science. Swinburne University of Technology Sarawak Campus, Sarawak, Malaysia.
4. An Overview of Machine Learning and its Application. Annina Simon Department of Computer Science and Engineering, Mahima Singh Deo, Department of Computer Science and Engineering, D.R Ramesh Babu, Professor, Department of Computer Science and Engineering, Dayananda Sagar college of Engineering, Bangalore.
5. Murtaza Roondiwala, Harshal Patel, Shraddha Varma, Undergraduate Engineering Students, Department of Information Technology, Mumbai University. Predicting Stock Prices Using LSTM.
6. Ashish Sharma, Dinesh Bhuriya, Upendra Singh. "Survey of Stock Market Prediction Using Machine Learning Approach", ICECA 2017.
7. Loke. K.S “Impact of Financial Ratios and Technical Analysis On Stock Price Prediction Using Random Forests”, IEEE, 2017.
8. Xi Zhang1, Siyu Qu1, Jieyun Huang1, Binxing Fang1, Philip Yu2, “Stock Market Prediction via Multi-Source Multiple Instance Learning.” IEEE 2018.
9. Vivek Kanade, Bhausaheb Devikar, Sayali Phadatare, Pranali Munde, Shubhangi Sonone. “Stock Market Prediction: Using Historical Data Analysis”, IJARCSSE 2017.
10. Sachin Sampa t Patil, Prof. Kailash Patidar, Asst. Prof. Megha Jain, “A Survey on Stock Market Prediction Using SVM”, IJCTET 2016.
11. Hakob GRIGORYAN, “A Stock Market Prediction Method Based on Support Vector Machines (SVM) and Independent Component Analysis (ICA)”, DSJ 2016.
12. RautSushrut Deepak, ShindeIshaUday, Dr. D. Malathi, “Machine Learning Approach In Stock Market Prediction”, IJPAM 2017.
13. Pei-Yuan Zhou , Keith C.C. Chan, Member, IEEE, and Carol XiaojuanOu, “Corporate Communication Network and Stock Price Movements: Insights From Data Mining”, IEEE 2018.
14. Xi Zhang1, Siyu Qu1, Jieyun Huang1, Binxing Fang1, Philip Yu2, “Stock Market Prediction via Multi-Source Multiple Instance Learning.” IEEE 2018

PLAGARISM REPORT

ORIGINALITY REPORT

14%	12%	3%	7%
SIMILARITY INDEX	INTERNET SOURCES	PUBLICATIONS	STUDENT PAPERS

PRIMARY SOURCES

1	journals.plos.org Internet Source	3%
2	otexts.org Internet Source	2%
3	www.ijeat.org Internet Source	2%
4	Rafael Rosillo, Javier Giner, David De la Fuente. "Stock Market Simulation Using Support Vector Machines", Journal of Forecasting, 2014 Publication	1%
5	www.investopedia.com Internet Source	1%

6	www.analyticsvidhya.com Internet Source	1%
7	Submitted to Cork Institute of Technology Student Paper	1%
8	otexts.com Internet Source	1%
9	Submitted to University of Sydney Student Paper	1%
10	Submitted to Uczelnia Łazarskiego Student Paper	<1%
11	Submitted to Kwame Nkrumah University of Science and Technology Student Paper	<1%

Exclude quotes	On	Exclude bibliography	On
Exclude matches	< 1%		