UNIVERSITY OF ESSEX

DEPARTMENT OF MATHEMATICAL SCIENCES

MA981-7-FY: MSC DISSERTATION

Predicting Short-Term Stock Movements with Quantitative Finance and Machine Learning in Python.

REGISTRATION NUMBER: 1901423

SUPERVISOR: DR. ANDREW HARRISON

DATE OF SUBMISSION (4TH SEPTEMBER 2020)

WORD COUNT: 10521

Contents

Abstract

The key purpose and the major concern pertaining to the study is towards determining the best/ suitable model that facilitates in predicting the value of stock market. From the assessing investigating process, there exists several techniques as well as pertinent variables which are essential for consideration, as it appeared that the these techniques that are discovered by scientists over the years, like random forest as well as SVMs being comparatively been underutilized. The study presents us and further emphasises and explores over appropriate approaches that has more accurately inferred with stock's movements. The primary consideration concerning with the data set concerning with stock price on prior year. The dataset achieved was then pre-processed and furthermore developed to its actual analysis. Therefore, this study also focuses on the data pre-processing for raw form of dataset. Then, follows pre-processing, as we further discussed how to employ random forests, vector machine support in dataset, and outcomes that the data produces. In addition, the proposed study will examine the issues which are in association with that of predictive systems usage under real-world contexts following with determining accuracy over the given overall ranges. The study also introduces a ML model for predicting stock for highly competitive range of market. With implementation of successful stock forecasting could pose as an important asset for stock exchanges and offers concrete solutions for counteracting problems faced by investors in majority cases.

Keywords: ML, Data Pre-processing, Data Mining, Stock, Prediction, SVM Clas-sifier, RNN, Python.

1 Introduction

For the scholars & researchers who are from various academic fields, the topic which they feel attractive is Modelling & Forecasting of the financial market. The concept of transactions which happen between buyers & sellers of the financial group/ sets of commodities like that of bonds, stocks, & other forms of precious metals is categorized as abstract of the financial market. The most attractive issue to be investigated especially in the stock market is the use of machine learning techniques & artificial neural networks to forecast the trend or the price of stocks in the present scenario of the financial market world. As a result of greater noise level, minimalized sample size, non-stationary & non-linearity, it is difficult for signal processing problem, an instance of financial forecasting as explained by Giles. The inadequate data gap between the past stock forms of trading price as well as volume with its future price has meant as noisy characteristics [1]

The political & macroeconomic environment is the reason behind the sensitive stock market. However, gathering of these two forms of information are too complex as well as unstable. Noise is considered as the information that cannot be included in features. The records of the real-world transaction determines the sample size of financial data. A longer period of transaction records is referred to a larger sample size & the uncertainty of financial environment is increased by it. In order for reducing probability over uncertain noise as well as relatively improve sample size over certain time period, stock data is used instead of daily data in this project. In 1991, Burton G. Malkiel developed Efficient Market Hypothesis. [1]

As the price changes in the real world are unpredictable, Burton represented predicting/ forecasting in case of financial market appeared unrealistic in his hypothesis. The immediate economic events or news is the base for the sudden changes in the prices of the financial market. In accordance for most of its recent events that are regardless from the previous analysis/ plans, decisions are made for their buying or selling as Investors are profit-oriented.

There is no end for this argument about Efficient Market Hypothesis.[1] For verification whether the efficient market hypothesis is proper or not, there is no strong proof so far. However to a certain extent, financial markets are predictable as claimed by Yaser. The two main pieces of

evidence which opposes the Efficient Market Hypothesis are the future financial market affected by the undiscounted serial correlations among vital economic events & within a certain time frame emphasising on past experience of many price changes in the financial market.[2]

For forecasting & prediction of the financial market, there has been extensive research for the potential of machine learning in recent years. In the field of financial market, there are many approaches in which the hottest topics are prediction neural networks (Multi-layer feed forward), reinforcement learning, SVM, recurrent neural networks as well as relevance vector machines. For forecasting & modelling financial, market neural networks (NN) that are utilized successfully & are well studied among all the machine learning methods. When compared to traditional statistical models, NNs appear quite flexible are comparatively noise tolerant as Tay & Cao explained in their studies.[2]The ability to be trained by incomplete overlapped data is with the neural networks that is why it is meant as noise tolerance. Using new data patterns and through a process of retraining, the capability to learn dynamic systems is possessed by the neural networks as referred by Flexibility. In 1997, Sepp Hochreite & Jurgen Schmidhuber introduced RNN which has short-term (long) memory. From our deep insight, it could be determined that the project's major attribute as well as solution is generalization of two major attributes or parts. By testing the model with various configurations, examination of the feasibility of SSTM in stock market forecasting is done.[3]

2 Literature Review

Efficient Market Hypothesis otherwise known has EMH introduced by Fama, E. F. (1970) during the early research related to predict the stock market, alongside with efforts from Horne, [4] & Parker, (1967) [5] as they stated the efficiency governing with Random Walk theory. The market price cannot be predicted as the prices are influenced by data and facts apart from historical prices that are framed with the help of various theories. The market information decides the price of a stock & as a response of any new information that will bring about a change in prices was suggested by the EMH theory. The three different variations which affect the market price are discussed by EMH:

Weak Form integrates existing public data along with historical data in which historical data is taken as semi-strong Form while Strong Form integrates private data which goes a little farther. Prediction models from success would be prevented if there is any price movement is moreover an outcome of novel information as stated by EMH. The stock prices randomly changes as stated in the RWH by Horne and Parker (1967) and they have stated the former movements pertaining with price and are independent from that of current set of movements. Short-term pattern of stock market is focused by it as it is slightly different from EMH. Numerous latest researches have revealed that the movement in the stock market price could be easily forecasted to certain degrees which oppose such theories.

As the ML uses the techniques to analyse data in drawing generalized pattern was established to be a noble tool which is employed for price predictions tasks. The following proposed theories on the other hand represents on how market prices appeared to have affected on the basis of notifications which are other than that of historical representation over prices; thereby describing on the fact of difficulty concerning with prediction of market price. As EMH oriented theory provides a suggestive statement on the fact that the price governing with the stock of interest basically relies completely over the provided market based data and thus with any sort of newer forms of information tend to eventually lead to a price modification since the reaction concerning with newly released data/ notification. Such kind of theory appears to even claim over the fact that these stocks are traded always on the basis of its fair value, as the designated traders

appear to have neither buy or otherwise take actions for selling stocks under a specialized price either undervalued/ inflated and thereby it is the only approach for the trader could eventually raise their profits (i.e.) via creeping up with the degree of risk associated with it.

Efficient Market Hypothesis (EMH) was proposed by Fama, E. F. (1970) during the early research related to predict the stock market.[6] The Random Walk theory was proposed by the aforementioned investigators. These market value pertaining to the price cannot be predicted as market range in the prices tend to be impacted as a result of achieved notifications apart from the historical trend over its prices as stressed from these defined theoretical statements. The market info on the other hand makes some decision over the price of a stock as a resultant outcome newly released information any new information will result to price modifications as suggested from EMH theory. Whilst, the only option for trader to control the range is to increase their profits is via increasing the overall risk involved as theory appear to claim over its fact as stocks happened to always have been traded on the basis of fair amount of value involved, wherein traders couldnot seldom buy/ sell stocks under the specialized range of price either undervalued/ being inflated.[4, 5]

EMH defines the following three aspects with varied impact over the designated market range in price and its value. These are: Weak Form, represents only historical representation of data being considered, followed with semi- Strong Form, denoting incorporation of present public info apart from that of historically represented data, lastly involving with Strong Form, which involves with incorporation of privatized data that happens to be little farther. As these models used for the purpose of prediction would also be prevented from gaining their necessary success when there lies any sort of flux in the price which in particular could be attributed as outcome for result pertaining with newly released forms of information/ random change or flux as denoted from EMH. With regards to the aforementioned Hypothesis by authors stating the randomized changes pertaining to the stock values/ prices appeared to be really argue as past historical predicament of price fluctuations appeared to be of independent in nature from that of the present mode of fluctuations.[5] This appears to have slightly been varied from that of EMH since it emphasises more on pattern of short-term in overall stock market. On the basis of above mentioned hypotheses, stock market tend to follow randomized movement

and the accuracy towards predicting on such movement happen to have not exceeded 50%.[4, 5] Unlike these hypotheses, several recent experiments have demonstrated that fluctuations in stock market values can be forecast to some degree.

Such sort of literature tend to depend on two major attributes/ types falling under the financial analysis for prediction over the market prices concerning with the existing stocks, namely:

• **Fundamental Analysis:** This could be attributed on the basis of company's company and which comprises of qualitative followed with quantitative attributes in the form of interest rate, ROA, expenses, revenues involved as well as prices. In case of check on long-term effects over the strength/ sustainability for an organization under the prolonged phase of investment serves as the prime aim in this analysis.

• **Technical analysis:** In this approach is primarily attributed on the basis of data from time series. Traders tend to analyze historical aspects of price fluctuations as well as with chart patterns which consider instances in the form of crucial parameter in the forms of prediction. Economic research can rely on three main keys: movement of stock prices as the movement seems to be unpredictable at times, past patterns that are expected to be repeated as time goes by, and other related stock statistics. Different machine learning methods have been used in recent experiments for forecasting asset prices.

As the ML uses the techniques for analysis of data in drawing generalized pattern was established to be a noble tool which is employed for price predictions tasks. Various ML models as well as risk oriented strategies were applied for prediction task concerning with stock market for estimating/ predicting primly on direction of trend with regards to price on varied time frames and utilizing varied features which tend to impact on prices of market levels. As an input to a Deep Neural Network (DNN) model Arévalo, et al. (2016).[7] The author employed 4 key features. As they are based on mathematical calculations, these sorts happened to be measured in the form of technical analysis over stocks in market as defined below:

These features can be viewed as stock market technical analysis instruments since they are based on computational formulas listed below:

Log return: the logarithmic disparity that exists between nearby price under time range of t and with nearby price under the time interval of t-1 as stated from finance term.

Pseudo-log-return: This represents logarithmic disparity that lies amid average level of prices over the consecutive time in ranges of minutes.

Trend Indicator: This involves with consideration of linear trend within 1 minute as data is used for create a specific fit and a specific slope. A slope that is negative can be considered as decrease in price while a slope that positive can be considered as increase. While a slope closer to 0 means that the price is nearly constant.

Arevalo and others (2016) verifies the input data as provides: variable window size (n) employed for time factor and various other inputs such as input and minute parameters. Therefore, the input file includes the final n pseudo-log-return, as finalized n normal deviation followed with final symptoms with n trend. Sample outcome involving "One minute next pseudo-recording-retrieval. After preparing the input data, it was specified DNN followed with input layer, 5 hidden forms/ layers and 1 output forms/layer. It was divided separately into training alongside with testing. The data further subjected to train for about 50 times. Numerous window sizes and the outcomes show that the size of window 3 can show excellent sample performance with 66% accuracy & MSE of 0.07. [7]

Weng, B. Forever. (2017) tried estimating further price movement one day using unequal data sources, where merging data from online sources with indicators and prices would increase stock market status forecasting. 20 features been selected to be its input. In case of general observation is drawn over, as any purpose with all sorts of datasets are rendered with minimal of being just 1 attribute. Different AI methods: ANNs, SVMs and DTs have been implemented to estimate and compare stock price movements. After evaluating on 3 varied forms of models as rendered above, we compare open price over its day with the open price of i + 1, using approximately 85% i SVM model achieves the best estimated accuracy.[8]

A general observation is drawn from as with any sort of purpose involving datasets are indicated by a minimum of 1 attribute. With several AI methods: ANNs, SVMs & DTs were implemented for estimation followed with comparison over stock price fluctuations. Post-evaluating over 3

different models that are being listed, as open price with regards to a day were further subjected to comparison with open price of i + 1, using approximately 85% i SVM model achieves the best estimated accuracy.[8]

Shoemaker, r. P. And others. (2009) attempted to estimate the direction over price fluctuation on basis of economic notifications. This study further conducted by 2009 since market predictor which appears to have been facing criticalities due to its mis-defined sets of parameters. For utilization of financial notifications as articles from the desired estimation sets of model, as news tend to have been indicated via in forms of numeric value. Many methods employed to label these methods emotionally or to analyze articles related to certain methods for use as vectors for input properties. Proper noun based approach involves combination over noun based phrases with named designated entities. From proposed technology which appeared to have improved other methods based on comparative studies [9]

AZFin Text is built by another system (Schumacher, et al 2009) which on the otherhand predicts price fluxes with 20 minutes post notifications release. Integral portion for the system involves with financial based notifications and articles that are rendered via Yoh Finance which is referred as a form of noun phrase; which involves with collected forms of noun phrases that are being denoted in the forms of vector underlying with binary forms of values representing on presence/ absence over the phrase pertaining to the article. While considering 2nd major forms/ component regarding the system is via data pertaining to stock price collected under 1-minute of period. Followed with final task which is major since it involve creation followed with AI model training post its data which is further being collected and which is followed with formalized input. For finalization of input governing with the model, there exists the quotation over the stock price release under the same instance, as notifications are attached with its input matrix, followed by the +20 minute price which acts a output over the system. Rendered form of data were then further being given for various models. The SVR model which is being created in estimating price for 20 minutes post release. Market time of the data being included only with 1 hour being left for open in market for showcasing effect over the notifications released in market hours. Additionally, there lies new form of barrier which is being included for the model as it can only utilize one article within the range of 20 minutes. Upon the usage

of two articles and its being released under the time frame of 20 minutes, as both appeared to have deleted. Outcomes achieved happen to showcase that established average involving with directional accuracy estimated to be 71.18%.[9]

Other studies have been intended to estimate text-report stock prices from financial news. [10]For example, Akita et al. In 2016, newspaper articles were transformed into representations distributed by para vectors and simulated the tentative effects of past events with LSTM to estimate the starting prices of shares on the Tokyo Stock Exchange.[11] Bolan et al. discovered in 2010 that investor sentiment gained from Twitter could have an impact on stock indices. He first introduced a novel word representation method to extract important features from textual data. Then, they used an SVM model to predict the direction of stock price movements the next day. Those studies demonstrated that textual information could be used in financial time series assessment tasks. However, their goal is to find effective feature extraction methods to help improve assessment performance.[12]

Unlike previous forms/ approaches as representation showcases relationship that lies between events as phrase is indicated because the vector/ bag of words does not indicate actor/action involved or whose responsible/ applied as action, thereby making a minor representation and may not show a relationship between events and stocks. To assess the performance pertaining to such newer representation, as data concerned with news article being collected via Reuters as well as from Bloomberg, alongside with daily basis of close prices from the defined S & P indices. 2 diverse models being created for testing representation: a linear SVM model with inputs and newsletters in the form of +1 or 1 at various intervals (1 d, 1 w & 1 m). Indicate value increase or decrease. [13]

The non-linear DNN being applied for discovering underlying relationships between the existing events. The input attributes being same in linear/ non-linear forms/ models: bag-of-words features, separating minor combinations (by1, p, 2, 1 + p, p +) after removing the stopword and event features. Representations use trivial TFIDF representations. 2, 1 + P + where2) where is1 represent 1st object on left side for sentence collected at top and 2 represent proximal object on right side, with P denoting verb. Such feature being used additional over verb classes

for minimizing representation. To evaluate pattern, pertaining with varied scenarios were employed. As model outcomes were compared with representation of bag's wording article, the structural events showed better performance. From another angle, DNN surpassed SVM on the basis of ability in determining underlying relationships, compared to models. In addition, were used from varied times (1 d, 1 w, 1 m); Outcome is less of a good frame. Therefore, the best approach is DNN involving a structured characteristics of event for regular form of estimation involving 60% accuracy. Recent studies based on machine practice have shown that stock price movements can be estimated with more than 50% accuracy, as contrasting to EMH and RWT using diverse timeframes, models and features.[14]

2.1 Types of Financial Analysis

To help with evaluating stocks & predicting future price movement, investors use various established techniques traditionally. These sort of techniques/ approaches could be categorized on the basis of the following three attributes namely: technical, fundamental as well as sentiment analysis. The former type intends to evaluate the investments involved. This is followed with identifying, followed by the status of the product either as buying/ selling of the opportunities. In case of fundamental analysis works on the contrary to the former type, as it attempted to evaluate the intrinsic attribute of the stocks via utilization of publicly framed information, and the case of technical analysts often tend to assume as the stock's price happen to already been reflected by these public sources that are disclosed. There exists 3 premises with regards to the technical analysis and these were framed as per the investigations of John, (1999).[15]

- 1. Market's action tend to discount on all
- 2. Prices tend to direct as per the form of trends
- 3. History repeating over itself

Brief representation to Fundamental Analysis: With regards to the industrial performance which is taken for consideration, as well as with the financial factors that ought to be considered from the company's side, there are certain financial factors in the forms of earnings, assets, profitability margin, liabilities & so forth, with a broader degree/ sets of factors being considered on an overall economy to be indicated as fundamental analysis, as the analytical approach intends to

measure net intrinsic value pertaining to the stock. In this case, the fundamental analysts tend to consider certain factors which appeared insignificant; this could be regarded as Price movement and its overall volume and history. Warren Buffet has been well known as the world renowned investor who is well recognized to be the practitioner to carry out fundamental analysis in an industrial scale.

Sentiment Analysis: This attributes with regards to extraction followed with identification of the subjective information in a much more systematic manner, as the analysis approach aim to utilize the natural language as the form of processing and which involves with the incorporation of text analysis for its operations.

Upon examination over the factor involving with predicting horizon alongside with fundamental & technical mediated analysis tend to exhibit greater significance. With prediction over time oriented horizon as quarter, an annum/ longer, usually preferable is Fundamental analysis. In case of prediction under short-term like days/ less, it is preferred to go with Technical analysis.

2.2 Challenges of Stock Prediction

When the market is open, the million investors as well as several traders in all across globe which is actively being involved under any given instances and in the case of trading of stocks are regarded as process pertaining with buying as well as selling of shares on publicly categorized lists of companies for stock exchange platform. As movement in price being affected as with many noises & factors, as it appeared extremely complicated & in difficult in terms of predict problems pertaining with stock market. As per our well-defined EMH [15],], as per stock's value under any given instance reflects all information available about it and is supported by several existing literatures attributing on prediction over stock market but appeared impossible over predicting it. [6] We further could compare over real forms of IBM historical ranging which is generated randomly via random walk outcomes. As it is noted as a person might easily being subjected to believe as both trends on price are being randomly generated.

When the market is open, the millions in the form of investors/ traders from across world actively involving under given instance and with stock trading being defined as process involving with buying-selling on the existing shares that publicly lists companies from the platform of

stock exchange. As, the fluctuation over price being impacted under several factors in the forms of noises, which appeared quite complex as the complexity on problem with regards to prediction of involved stock markets. As per well-known known EMH in accordance to Samuelson (1965), as price pertaining to stock under given time happens to reflect all the pertinent data that are available concerning with it. [15]which furthermore being supported by several of the existing literature attributed to prediction on stock market, however it appeared quite impossible for prediction as noted by Fama, 1965.[6] One may easily be led to believe that all price patterns are generated randomly.

Three forms of EMH exist, depending on the degree of stock market efficiency.If all the past market information is efficiently reflected by market then it is implied as weak form EMH. It is assumed by the hypothesis that there is no effect on future rates by the past rates of return. If all publicly available information is reflected by the market then it is implied as semi-strong form EMH.This hypothesis implies that no investor would be able to produce above-average returns even if new insight has been given that is not available to the public. Weak form EMH is incorporated by the semi-strong form.

All information of both the public & private is reflected efficiently by the market which is implied by strong form EMH. As the achievement of above average returns which would not be possible by any investor was assumed by this hypothesis. Weak form & semi-strong form from the EMH that was incorporated by Strong EMH form.

The weak as well as from semi strong form on EMH are challenged by the results achieved using ML & soft computational techniques towards stock prediction which was explored by the recent studies.[16, 17, 18].However, most of the studies use historical speed, technological indicators or investor attitudes as independent variables for model training and prediction.. On the basis of stock's fundamental attribute on financial ratios, simulating the decision-making based process for investment experts via developing machine learning models is the main motive of this research.

2.3 Machine Learning Methods

The background info concerning with ML approaches employed for study being denoted in this section which is the Random Forest. Supervised learning method is used here, since due to nature governing with problem as well as with dataset.

2.3.1 Random Forest

A RF denotes a supervised forms of classification algorithm. This involve with ordering nodes as well as splits randomly by creating a forest (many decision trees). Better results can be produced, if there are more trees in the forest. [19] To perform predictions, set of rules is used which is formulated while providing input over the training dataset via targets followed with features in DT. RF comprised several DT. These act as ensembles for DT in which each of it was created via using subset for the attributes being used for classifying provided population. Especially with regards to how it could be classified under given instance over input levels of data is voted by those decision trees & to choose on suitable prediction was voted by RF bootstraps. Over fitting, a common flaw of decision trees is prevented by doing this.The figure below shows that the Random Forest algorithm requires many decision trees to perform its tasks.

Out-of-bag (oob) error.

While considering the estimate on test of set error which is gained without the requirement on cross-validation/ separate test in RF. During the run it is internally estimated as follows:

Unbiased calculation of the test set error is done in the random forests without the need for cross-validation or a separate test. It is calculated internally during the run as follows:

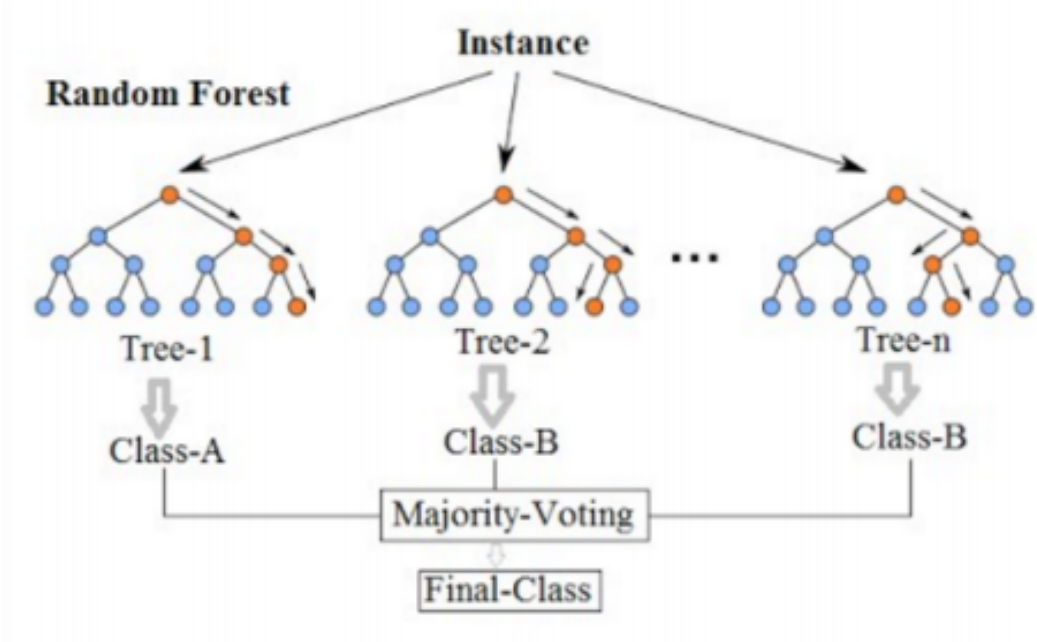


Figure 1: Action of RF(adapted from article**"Random Forest Classification"** [Source])

From the original data, each tree was designed using various bootstrap mediated approaches. In the construction of the kth tree showcasing 1/3rd of cases remains unused and being restricted off from bootstrap sample.

Every tree was created using a single bootstrap comparison of the initial data. Approximately one-third of cases are left out of the analysis on bootstrap and not included in the development of the k^{th} line. To get classification, each case which was devoid out off from construction on k^{th} tree should be declined from k^{th} tree. Thus a definition of the test set for each case is obtained in around one-third of the trees. Take j to be the class who received most of the votes at the end of the run every time if n was oob. The measure of the OOB error is the proportion of instances j which is not equal to the true class of n compounded over all instances. That has proved to be unbiased in several experiments.

Classification & regression tasks are done by the flexible supervised learning algorithm which is Random Forest (RF). During the process of data fitting, it builds multiple decision trees. The smallest forms of the sample with regards to the decision tree been illustrated from the Figure 1. In regression problem the mean value of all decision tree's output is taken by RF for generating results.

From the original data, each tree was designed using various bootstrap mediated approaches. In the construction of the kth tree showcasing 1/3rd of cases remains unused and being restricted off from bootstrap sample.

To get classification, each case which was devoid out off from construction on kth tree should be declined from kth tree. Test set classification obtained by such way for each case in about one-third trees. Every instances involved for case n (out of bag error) was oob, consider j to be the class which got most of the votes at the end of the run. The proportion of times that j is not equal to the true class of n averaged over all cases is the oob(out of bag) error estimate. In many tests, there are proofs that it is unbiased.

Classification & regression tasks are done by the flexible supervised learning algorithm denoting RF. During process involving with data fitting, it builds multiple decision trees. The smallest forms of the sample with regards to the decision tree been illustrated from the Figure 1. In regression problem the mean value of all decision tree's output is taken by RF for generating results. The majority forms of voting for decision trees employed as outcome for classification problems. To increase the performance of RF, many hyper parameters can be tuned. A few among them being most significant being hyper parameters under RF being listed below:

Number on the overall Estimators: Before considering with maximum degree of voting/ with averaging on predictions, with numbers concerning with DTs built via algorithm appeared to be denoted as Estimators numbers. In general, under the cost involving slower degree of computation as algorithm's performance appeared to have increased by a larger number involving DTs.

Minimum Sample Split: In case of undergoing split under an internal node, involving with minimal sample ranges which are required as Minimum sample split. This indicates with dataset size.

Maximum set of attributes/Features: When looking for best split, the feature numbers which are considered is the Maximum features. When tuning this hyper parameter, the dataset dimension must be taken for consideration.

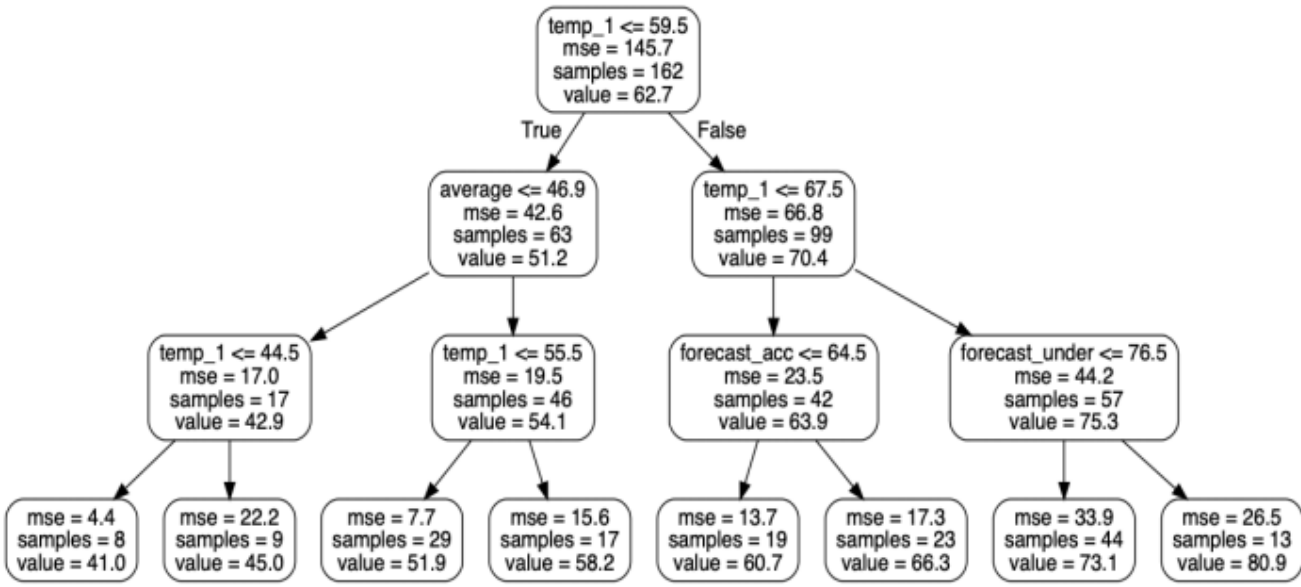


Figure 2: A minor decision tree [Source])

Variable Significance

Put down the OOB cases in every tree grown in the forest, and count the total number of votes for the right class. Now, in the oob cases, randomly permute the values of variable m and place those cases down the tree. [20] The voting number over corrected class in oob data (variable-m-permuted) is subtracted in the numbering over the votes in case of correct class with regards to untouched oob data. Averaging over number for all trees from forest appears to be of raw importance in terms of score over m as variable. In case of values pertaining to this score from tree-tree remain to be independent, as SE could be computed via standardized computational approach. The underlying correlations over such scores are between trees being computed under several datasets being proved as quite minimal, thus classically it could compute on SE, dividing raw score on SE for computing z-score, as it assigns significance range over z-score assumption on its normality. Forest will further be subjected to execution under existing variable as it involves with having greater levels in the variables and its number, then further being subjected to running again which involves with utilizing only most crucial variables from its initial run. As each of its case, tend to have considered all trees to which it is being denoted as oob.

2.3.2 Support Vector Machines(SVM)

For time series prediction, SVM regarded to be of most suitable. For both classification & regression task, it can be used. The SVM on the basis of minimization principle governing with structural risk. By incorporating concept of capacity control, the over-fitting problem is prevented in this principle. The two key elements in SVM implementation are Mathematical programming as well as Kernel Functions. SVM comes under supervised learning. As it measures well over high degree on dimensional attribute, as they are considered as the advantages of SVM. As the constructed model which are dependence over support vectors, it also minimizes computational range in terms of cost. For predictions from stock market in financial market, the SVM being defined as effective predictive forms of approach/ tool.[21]

Thus, the efficiency to handle the high-dimensional data is with the Support vector machines. It could function as more classes via developing multiple forms of binary classifications (1-versus 1 under every classes' pair) as it was originally designed as a two-class classifier. On basis of linear attribute on features, the classifying instances are being worked on by the algorithm. Using a kernel, the performance of non-linear classification can be done additionally. The classifier being fed on instances (pre-labelled) & in selection on points as SV with SVM attributes on hyper plane which on the other hand maximizes over margins. More information can be found in. [22]

3 Research Motivation

3.1 Problem Statement

In every day news, stock market appears. When it reaches a new high or a new low, everyone hears. If an efficient algorithm is devised to predict the short term price of an individual stock then the rate of investment business opportunities in the Stock market can increase.

Try to determine the stock value is basically called as Stock market prediction and it offers set of robust ideology to people for knowing, further with predicting market as well as with details pertaining with stock prices. Using the quarterly financial ratio along with the dataset, it is generally presented. It can give an inaccurate result when relied upon as single forms of dataset as it appeared insufficient with regards to prediction related approaches. Hence, with various datasets integration, contemplation towards the study of machine learning is analysed for determining/ prediction of market/ stock trends. Whilst considering sentiment of investors is affected by all these events which affect the corporate earnings. To predict these hyper parameters correctly & consistently is beyond the scope of almost all investors. Prediction of stock price is made very difficult because of all these factors. It can be employed for training machine & further towards generation of predictive outcomes, once the right data is collected.

3.2 Purpose

The key motivation for predicting stock-market movements is future monetary gains. A considerable amount of research has been carried out in the field of stock performance estimation since the introduction of this investment method, as investors generally want to invest in stocks that they thought would outperform others to gain profit by selling them later on. Over the years a vast variety of stock prediction techniques has been built up but the precision of the real forecast performance of each of these techniques is still debatable.[23]

The approaches concerning with stock prediction being classified as small degree on its categories:

- 1. Fundamental assessment, involved with studying on the underlying organization via their published financial statements.

- 2. Technical analysis, analysing the historical data, the predictions are made.
- 3. Sentiment analysis, by analysing the published manuscripts, insights from the gained reports followed with commentaries that pertain with such stocks, the predictions are made.

Potential monetary returns are the main motivation for the prediction of changes in stock price. To generate profit, investors would naturally tend to invest on stocks that were have predicted and which will eventually outperform than others, as greater degree of research were conducted within the field pertaining with performance prediction on stock denoting the birth over investment instrument. However with regards to consistency over the actual prediction and its performance for most of these techniques is still debatable, a greater inventory over prediction techniques over stock were developed with growing years[23]

The techniques pertaining to stock prediction could be effectively classified as smaller sets of categories: Fundamental analysis involves with studying on underlying companies through their published financial statements, as predictions being made. Technical analysis, involve with analysing only on historical volumes & prices, the predictions being made. Sentiment analysis, involve analysing published literatures emphasising on reports as well as commentaries pertaining over certain range of stocks, as these predictions are eventually made.

As the last category is attributed to be possible by Internet invention & with the utilization of numerous online domains providing up-to-date notifications, it is much newer than the other two. Technical as well as sentiment analysis tend to pose as primarily utilized approach towards gaining prediction on short-term with days scale/ less of 3 general categories of stock prediction techniques.

In case of prediction (mid & long-term) on quarters scale and with years, fundamental analysis is used. There has been growth in recent years in the field of stock prediction with the popularity towards applying several ML & mining approaches. From technical & sentiment analysis, the prominent studies existing tend to be focused on creating prediction models which using machine learning & data mining. Results from both of these experiments showed that models derived from past price and volume data can be successfully used for short-term forecasting.[24]

However, there is one major drawback for short-term forecasting and high-frequency trading, which is a frictional or contract trading price. Any acquisition and sale by a buyer selling stocks through a broker is typically a charge owed to the dealer.The commission rate varies from broker to broker but the profit of the future can be eaten as the trading tends to grow in volume, including with discount brokers. Since the short-term prediction models from many of the studies do not include frictional costs in the calculation, it will impact the research conclusion.[15]

3.3 Contributions

Based on fundamental analysis, evaluation of the advanced ML approaches for short-term prediction over stock performed from the research thesis.

The key contributions achieved from the thesis was then summarized below:

- 1. Based on fundamental analysis: Random Forest (RF), we have examined the important machine learning algorithms for stock prediction.
- 2. To construct portfolios based on predictions, ranking oriented portfolio based selection approach was used. With respect to benchmark index, the relative returns of constructed portfolios were evaluated.
- 3. For identification of significant features followed with improvement in performance over model, we have used RF based feature selection method.
- 4. With regards to assembly over prediction outcomes from ML algorithm, we used bootstrap aggregating for the sake of improving return over constructed portfolios.

4 Implementation

Technical indicators were selected to be used in stock market. The difference between close prices of market in successive days used for taking resolution in the beginning. Stock initial values as well as technical indicators are considered as featured variables.Target variables gives the outcome in the form of buy,hold or sell. To train ml models, discussed features and target variables have been used.

- **Stock Market Analysis:** This comprised of leveraging on technical indicators accurately for carrying out analysis over stocks. This appeared to be of primarily due to the technical indicators involved in it have been proved to be them-selves for accurately providing the desired values and gain stock market on the data trends not based on terms pertaining to stocks and their price but also based on the price's volume on an overall.
- **Data Preparation:** In case of the selected attribute/ indicator, involve with formula being applied over stock market in producing information that act as bed rock to our research. Every indicators held a value for stock observation. Columns are treated as technical indicator.column which corresponds to a different technical indicator data.
- **Feature Generation:** Together with the original stock data the technical indicator values were being used as a feature variable for the machine learning training. Since we employed under supervised learning, thus we require column (class label). These labels aid in calculating via cross-validation with that of the close price data.
- **Result Analysis:** After illustrating the models accuracy, deep neural network should give optimum outcomes.It must be depicting a staggering accuracy values which will make it the most appropriate choice.

4.1 Technologies

The language of choice for this research was Python. For multiple reasons, it was considered as an easy decision.

- 1. Python possess an enormous community behind it as a language. With a trip to Stack Overflow, any issues that appears to be encountered could be efficiently solved. The

prominent and most popular language on site is Python, making it more likely that there exists a direct answer for any kind of query.

- 2. For scientific computing, there are abundance of powerful tools which are ready with Python. Freely available well documented packages are such as NumPy, Pandas SciPy. The code needed to write a given program can be dramatically reduced & simplified using these Packages. Iteration can be made quickly by this.
- 3. As a language, Python forgives & facilitates programs that appears as pseudo code. As the pseudo code requires to be implemented & further then tested which is given in the academic papers this will come handy. Also the step appeared usually and reasonably trivial for utilizing the Python.

4.1.1 NumPy

Numpy is python modules which provide scientific and higher level mathematical abstractions wrapped in python. In most of the programming languages, we can't use mathematical abstractions such as f(x) as it would affect the semantics and the syntax of the code. But by using NumPy we can exploit such functions in our code. The tools for finding Eigenvectors, the basic numerical routines are also provided by Numpy.

4.1.2 Pandas

It is library known for doing manipulations,undergoing operations on structural data and for time series. It is used for model preprocessing and model deployment.

4.1.3 Scikit learn

For the Python programming language, Scikit-learn are freely available software ML library. Various classifications like regression as well as clustering based algorithms including the SVM, RF, gradient boosting, k-means etc are featured by it. For interoperation of the Python numerical & scientific libraries SciPy and NumPy, it is mainly designed.

4.1.4 TensorFlow

Using data flow graphs, an open source software library to perform numerical computation involving TensorFlow. Mathematically derived operations being performed by Nodes via graph,

whilst multidimensional arrays (tensors data) communicated with them has been illustrated in the form of graphical edges.Graph nodes contain mathematical operations while edges are tensors used to interact between nodes. The GUI allows you to spread computation over a desktop, computer or mobile device to one or more CPUs or GPUs with a single API.TensorFlow was originally developed by developers and engineers working on the Google Brain Team inside Google's Artificial Intelligence research agency to carry out machine learning and deep neural networks research, but the architecture is flexible enough to be applicable in a wide variety of other fields as well.

4.1.5 Keras

Keras is a high-level, Python-written neural network API that can run on top of TensorFlow, CNTK or Theano. This was developed primarily focussing towards enabling better experimentation. As the NN appeared to have gone from idea resulting to a least plausible way of delay as it pose as key to perform good investigation.

Keras facilitates towards easier & faster way towards prototyping (via user friendliness, extensibility & modularity) which supports not only convolutional or recurrent networks, however they facilitate for combinations too. It could be run quite smoothly using CPU as well as GPU.

4.1.6 Compiler Options

While considering data processing on a large-scale, predictive analytics alongside with scientific way of computing facilitates in simplifying management of the package as well as with deployment.Anaconda is a freemium open-source implementation for the programming languages Python and R. The package management system conda manages the package versions.

Anaconda is a freemium open-source implementation of the programming languages of Python and R for large-scale data analysis, predictive analytics, and scientific computing aimed at simplifying package management and deployment. Kit variants are administered by the conda product management framework.

4.2 Methodology

API has been used to fetch real time stock data. Different strategies has been tested by using python language to predict short term stock movements and comprehend with our best findings. We have used certain parameters measures and visualizations to determine the best among all techniques.

4.3 Data Collection

The data fetched from yahoo finance. It enables us to easily download the required data on any day. Ticker has been provided as input parameter and in return we got high,low,close,adjusted close values of stock.

Facebook:

Date	High	Low	Open	Close	Volume	Adj Close
2012-05-18	45.000000	38.000000	42.049999	38.230000	573576400	38.230000
2012-05-21	36.660000	33.000000	36.529999	34.029999	168192700	34.029999
2012-05-22	33.590000	30.940001	32.610001	31.000000	101786600	31.000000
2012-05-23	32.500000	31.360001	31.370001	32.000000	73600000	32.000000
2012-05-24	33.209999	31.770000	32.950001	33.029999	50237200	33.029999

Figure 3: Glimpse

	High	Low	Open	Close	Volume	Adj Close
count	2070.000000	2070.000000	2070.000000	2070.000000	2.070000e+03	2070.000000
mean	118.276879	115.581082	116.946850	116.996101	3.269486e+07	116.996101
std	61.761697	60.495594	61.112117	61.200772	2.909451e+07	61.200772
min	18.270000	17.549999	18.080000	17.730000	5.913100e+06	17.730000
25%	68.060001	66.645002	67.507500	67.417503	1.623050e+07	67.417503
50%	118.544998	116.689999	117.700001	117.615002	2.378540e+07	117.615002
75%	174.015003	170.337502	172.379997	172.149998	3.902600e+07	172.149998
max	278.890015	263.429993	268.040009	268.440002	5.735764e+08	268.440002

Figure 4: Statistics

S&P 500:

Date	High	Low	Open	Close	Volume	Adj Close
2010-01-04	113.389999	111.510002	112.370003	113.330002	118944600.0	91.841896
2010-01-05	113.680000	112.849998	113.260002	113.629997	111579900.0	92.084984
2010-01-06	113.989998	113.430000	113.519997	113.709999	116074400.0	92.149803
2010-01-07	114.330002	113.180000	113.500000	114.190002	131091100.0	92.538841
2010-01-08	114.620003	113.660004	113.889999	114.570000	126402800.0	92.846756

Figure 5: Glimpse

	High	Low	Open	Close	Volume	Adj Close
count	2669.000000	2669.000000	2669.000000	2669.000000	2.669000e+03	2669.000000
mean	203.380427	201.227381	202.351499	202.389131	1.264641e+08	185.974943
std	62.800393	62.212321	62.520265	62.521123	7.526137e+07	67.978021
min	103.419998	101.129997	103.110001	102.199997	2.027000e+07	83.558922
25%	141.380005	140.039993	140.639999	140.789993	7.454400e+07	119.803650
50%	203.839996	201.649994	202.770004	203.080002	1.064946e+08	183.382629
75%	258.390015	255.630005	256.859985	257.470001	1.568387e+08	245.335159
max	339.079987	337.480011	337.790009	338.339996	7.178287e+08	335.570007

Figure 6: Statistics

Exploration of stock data as well as S&P 500(a weighted index of the 500 largest public companies) has been done.

4.4 Import Relevant packages

In built functions of python has been used for serving our purpose well just like pandas and numpy for handling the data, "datetime" for data and time respectively, pandas for fetching the needed data , matplotlib for visualization, "math" for mathematical operations, ggplot for good plotting for interpretation, sklearn for undergoing machine learning, keras and tensorflow for advancement in ml in the form of RNN.

4.5 Fetching stock data

We have used 10 years historical data to predict fueture stock trends.for now I have extracted data from 2010 till now 2020.We can change the year as per our analysis.

Table 1: Python Code for fetching stock data,

Fetching Data for 10 years
start = dt.datetime(2010,1,1)
end = dt.datetime.now()

4.6 Exploratory Data Analysis

We use Facebook stock along with SP 500 data as an example and try predictive analytics on it. This analysis is applicable on any stock data fetched from Yahoo Finance.

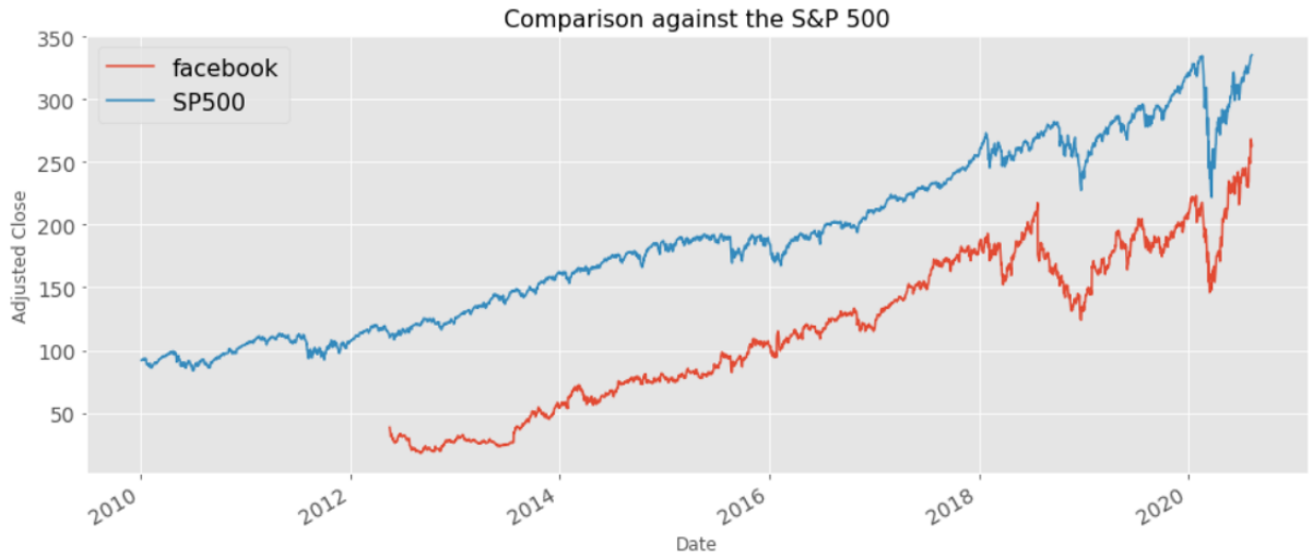


Figure 7: Facebook Stock Data Against S&P

The daily data is visualized in form of a line plot in Figure, giving us an overview of the whole stock prices since 2010. It can be observed that Facebook stock data starts from mid-2012 and before it was not listed on stocks. The minimum values for Facebook stock is around 18 and maximum is around 250. On the other can S&P 500 is minimum 83 and maximum 334. It could also be observed the both have a similar upward and downward trend.

4.7 Data Preparation

Post- data collection, it is essential for preparing it to further steps. This is followed with Data preparation which serves as step wherein we place our data at suitable location and further

prepare them for using it for ML training.

Table 2: Python Code for Splitting stock data into train and test set for applying ML algorithms.

Getting Train and Test Data
train=df[:'Dec 2017']
test=df['Dec 2017':]

Data was divided in December 2017 where data is used to train the model before the specified date, while it is used after that to undergo testing for constructing the models.

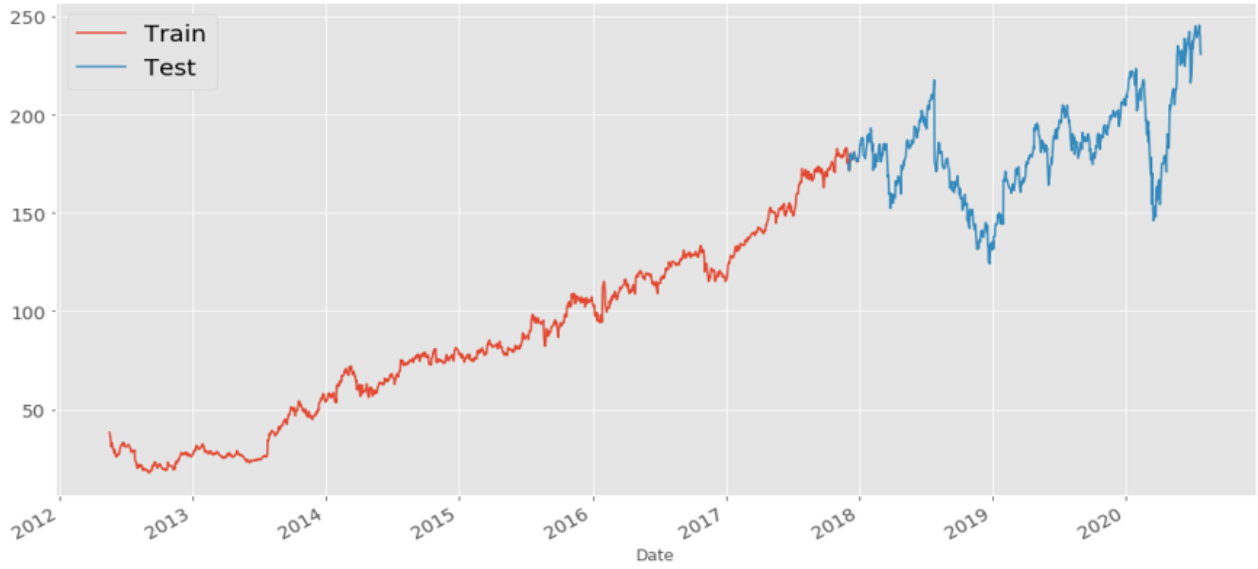


Figure 8: Split data(train set and test set.)

Splitting of data has been done for predicting test data after the model has been trained. It can be observed in Figure, how the split is made. We now proceed with the prediction of based on train data on the test data.

4.8 Applying Quantitative Finance

Crossover is one of the core concept essential to know while dealing with the stock data. We can understand it simply by knowing that it occurs when there is overlapping between short and long moving average. Trade can be done only if there is clash between two indicators.

Figure 9: Moving Average Crossover [Source])

When the short moving average goes from above the long moving average to below, that is a crossover indicating a sell (or short) opportunity, and vice-versa for a buy (or long) opportunity.

If long moving average indicator is going higher than short moving average there is a probability of buying a stock while if opposite happens then there is an opportunity to sell. Then, using a backtesting analysis, we can test our strategy with historical data and even plot the trades our algorithm made.

4.8.1 Naive Approach

In naïve approach there is no-complex prediction involved, we take the last stock value in train dataset and make a constant prediction for the test dataset. The results can be observed in figure below.

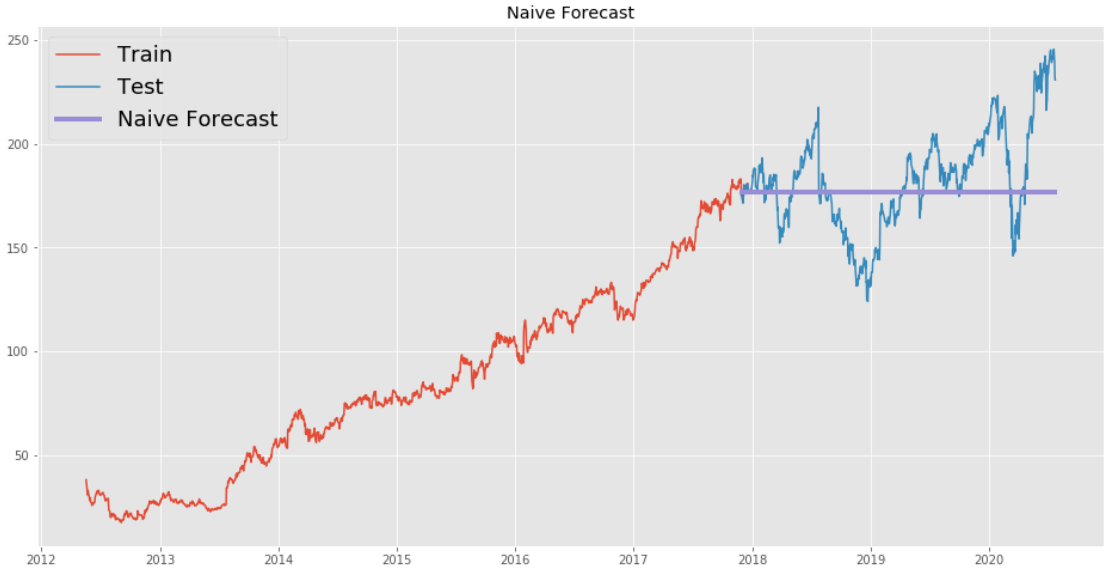


Figure 10: Quantitative Finance: Naive Approach

It is not preferred to be used. Larger the number, the worse it affects the company. Inaccurate forecasts in large range makes it inappropriate choice for order(buy or sell).

4.8.2 Average Approach

Average is one of the most simplest concept to deal with.Here in our financial analysis of data, we are computing mean of stock data values followed by using it to do the training and undergone prediction of test data on the basis of it.

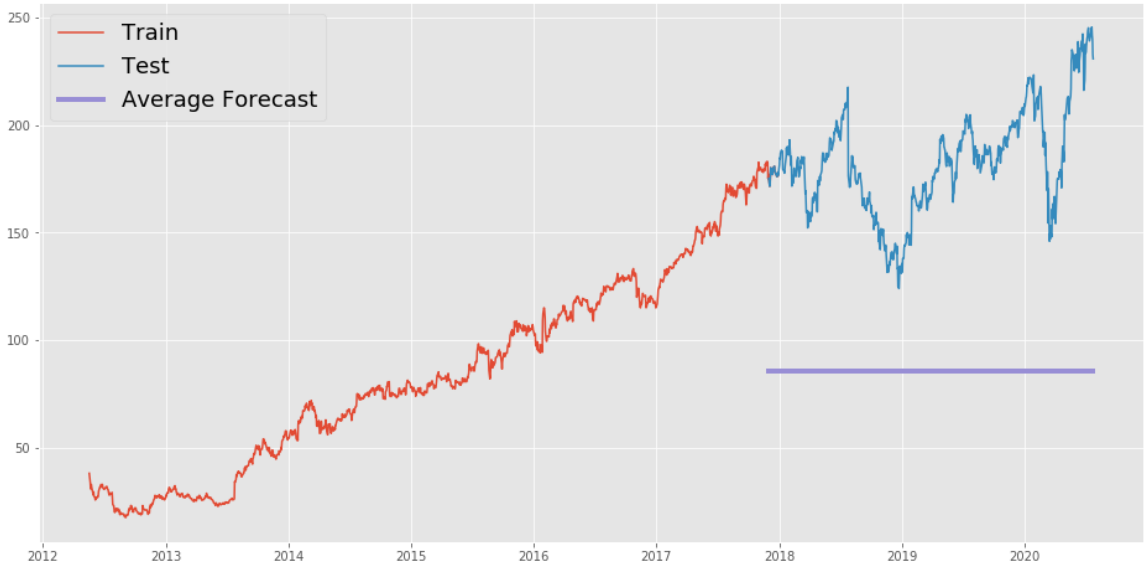


Figure 11: Quantitative Finance: Average Approach

We haven't got the desired output in this approach as well. This is however not a good approach to make the prediction ,the greater degree concerning with number, the worse tend to exhibit, indicating a negative affect on company since it acts as widely to be inaccurate forecast, thus making it quite impossible for planning orders. Therefore, we try out some other techniques.

4.8.3 Moving Approach

It is slightly different approach from the above two where we are using subset of the data and doing average of it for training. In our case, we have taken the size of the window as 30 and 50 respectively.

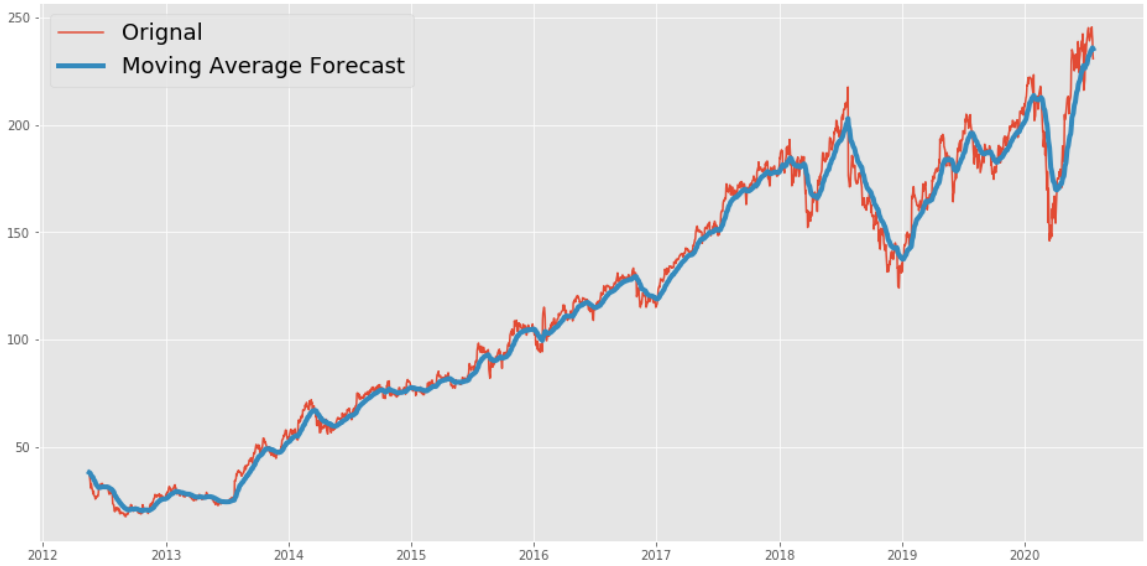


Figure 12:30 days window

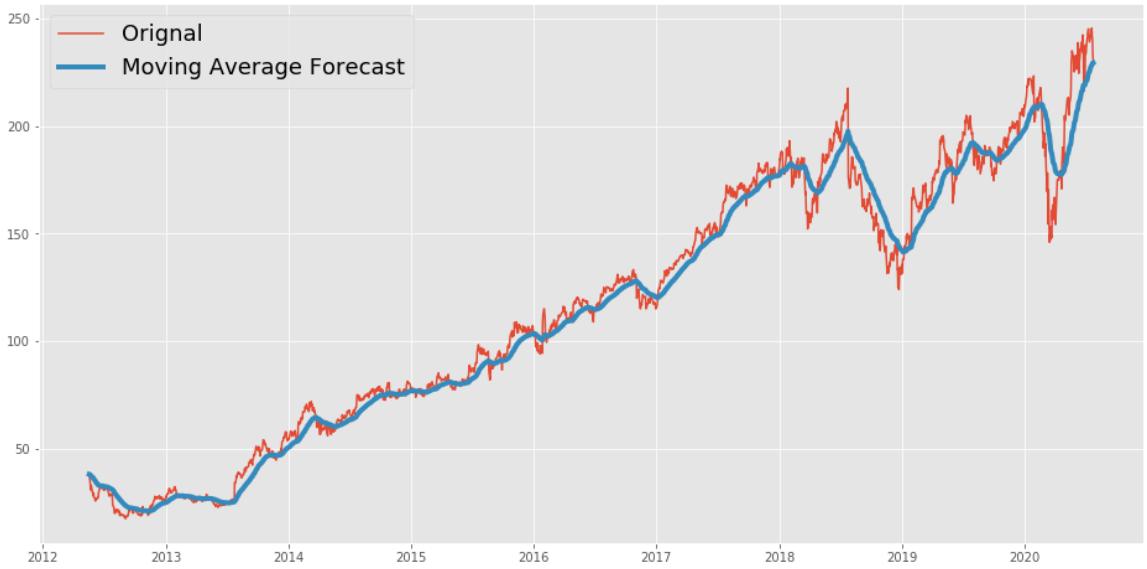


Figure 13:50 days window

As you can see just by doing this simple, yet effective, analysis our algorithm can often accurately predict trends. An algorithm could further capitalize on these swings by utilizing options trading rather than typical swing trading where the only opportunities are to buy and sell. Crossovers are one of the simplest and most often used technical indicators for traders, and now with the help of algorithmic trading, you can too.

Moving average is a very good technique to make time series predictions like for stock data. However, it can be observed that the correct prediction is slightly delayed based on the window size taken. The greater the window size the greater the delay. We now move to some advanced

techniques of machine learning and check our observations.

4.9 Conventional Machine Learning

4.9.1 Autoregressive Integrated Moving Average (ARIMA)

It is particularly used for forecasting in time series datasets. It consists of three parameters i.e., autoregression, integration and moving average. Autoregres-sion is the process of using past values to predict future values, integration is used to make static time series, moving average determines error by using prior terms of error.

The results are observed below, prediction confidence interval is take in green and the prediction itself in purple.

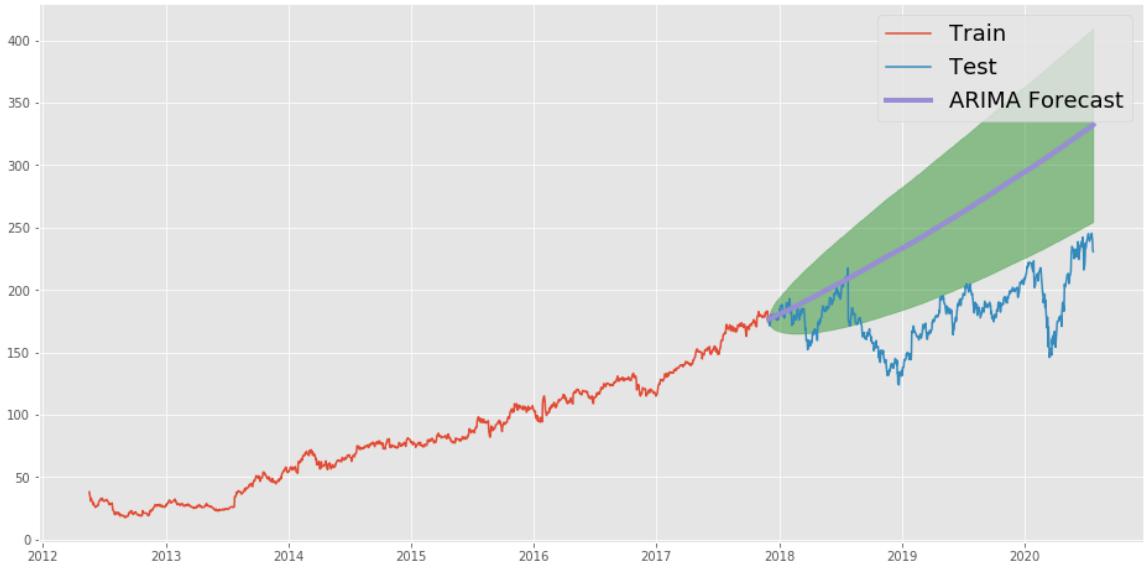


Figure 14:ARIMA with confidence interval on Facebook stock data

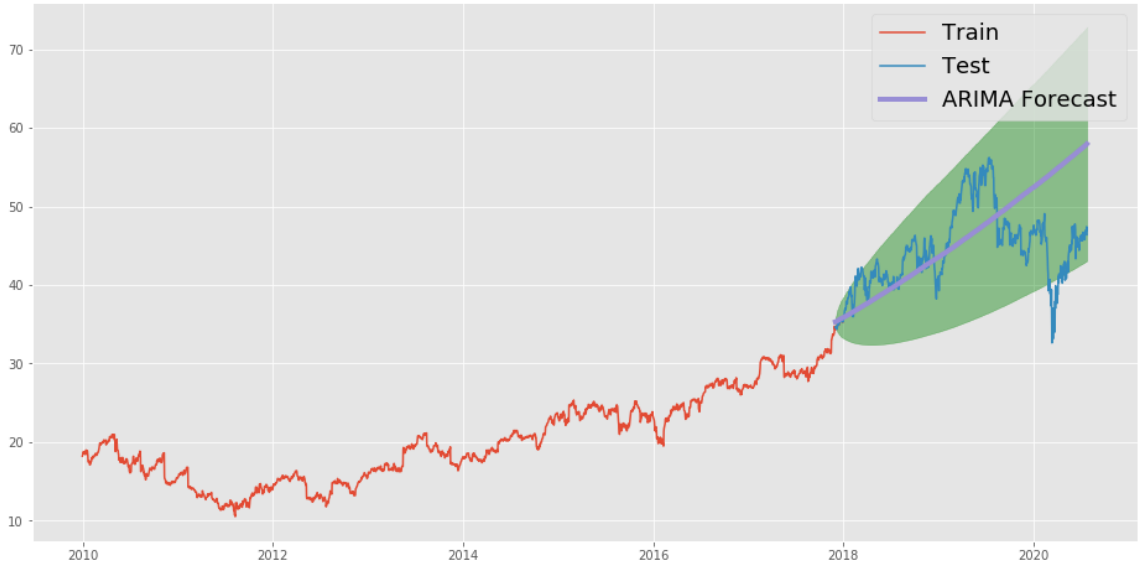


Figure 15: ARIMA with confidence interval on Cisco Stock data

It can be observed that ARIMA is good preforming with a similar trend is followed like in Cisco Stock data, the trend is upwards. The Facebook stocks fell due to Cambridge Analytica scandal in 2018 and Covid-19 in 2020. Therefore, the prediction is not so correct for Facebook. We will now be trying advance Machine Learning techniques including an RNN called LSTM.

4.9.2 Support Vector Machine

Support Vector Machine and support vector regression follows same algorithm. The reason we have used SVR here is that it will return continuous variable as an outcome and known for its application in determining values in stock. Line chart below depicts blue as existing stocks while Red is the predicted value of stock and blue is the original value of the stock. .With the difference in records in real and predicted stocks movements it can be computed that it is not showing good results as we are expecting. For stock data we have used support vector regressor as it contains continuous values.

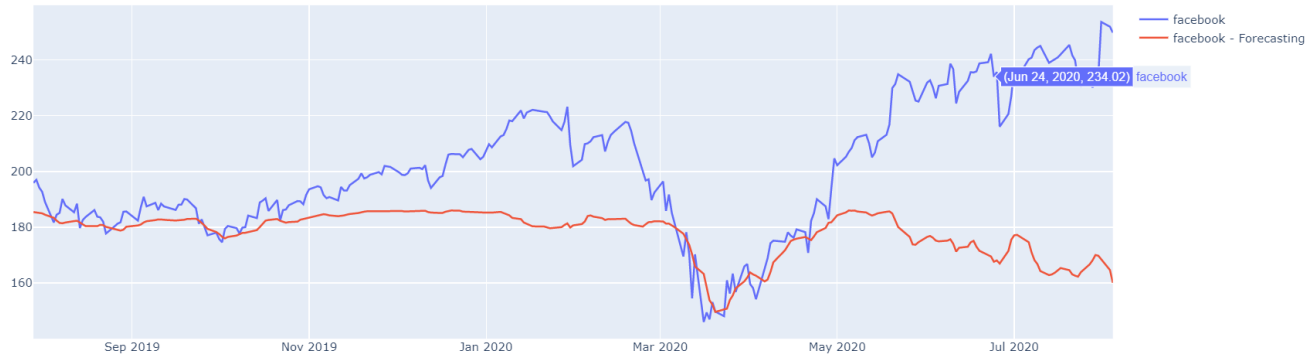


Figure 16: Support Vector Regression

4.9.3 Random Forest

Random forest regressor is utilized as it is capable enough to deal with continuous values. We have undergone selecting relevant features as a step for using it for training the model, followed by doing data pre-processing for fitting into the model appropriately. With the illustration we can compute that pre existing stock trends and future trends are almost similar. Therefore, showing that there is less error in estimation and producing good outcomes ultimately.

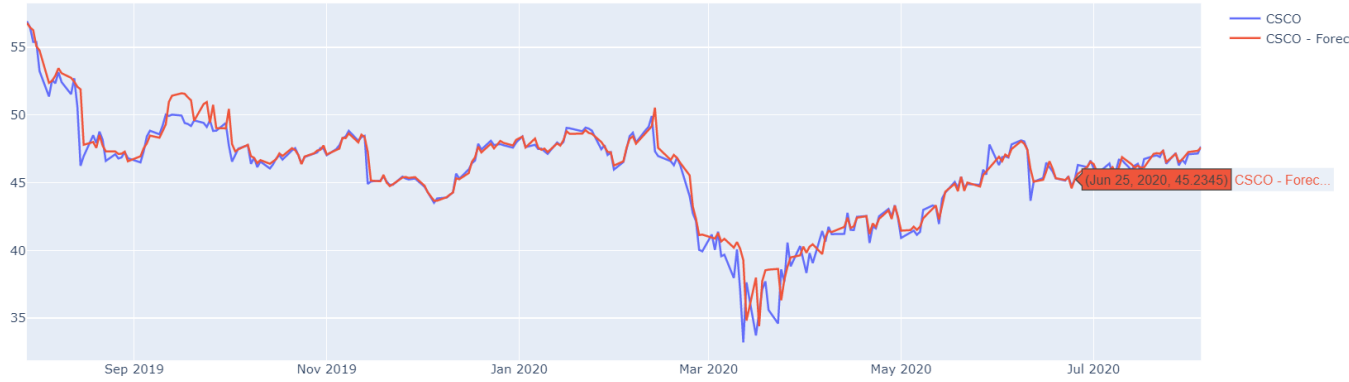


Figure 17: Random Forest Regressor

In random forest, error rate is also low as compared to SVM. We can see that the predicted and forecasted value of the stock are almost similar showing less error in predicting future stock prices.

4.10 Long-Short-Term Memory (LSTM) Recurrent Neural Network

RNN possess unique feature of having a memory which can store all information it has been processing. This is the most important capability of RNN known as hidden state. RNN is based on the concept where output of previous node taken as an input for current node. There is less complexity while using parameters in RNN because for performing the similar actions it uses similar parameters.

Input is taken in the form of singular time step, followed by computing current state with the help of previous step. We have the feature to use any number of time steps as per our requirements and collect data from previous nodes. After generating outputs, it is compared with target output. With the difference we compute the error which is fired back into the network. RNN is very helpful for doing predictions in time series data. Recalling prior inputs makes the RNN best fit for doing predictions in time series data which further breaks it into new category called Long Short-Term Memory (LSTM).

LSTM comes into picture due to its capability for holding the data for larger span of time. It memorizes all data encountered with in the network and ignore irrelevant information. It is a complex deep learning model that uses memory blocks, these blocks are recurrently connected. Each has one or more recurrently connected memory cells and three multiplicative units – the gates that are input, output and forget. Forget gate computes the probability of forgetting previous data, input gate indicates the information stored in inner states, input modulation gate to modify the information stored in internal nodes, output gate to interpret the outcome from current node.

Stack implementation has been used in our analysis where two days has been considered as time stamp for doing the prediction for upcoming or next day. Time stamp can be increased or decreased for doing modification in the predictions, there is delay in the predictions by increasing the time stamp.

LSTM is RNN ideal for making time series prediction like these. It is a complex deep learning model that uses memory blocks, these blocks are recurrently connected. We used a stack implementation of LSTM layers and made predictions. We took a timestamp size of 2 days, where

2 days are used for predicting the next day. We can increase the time stamp to make delayed predictions. We observed the result of multi-stacked LSTM below.

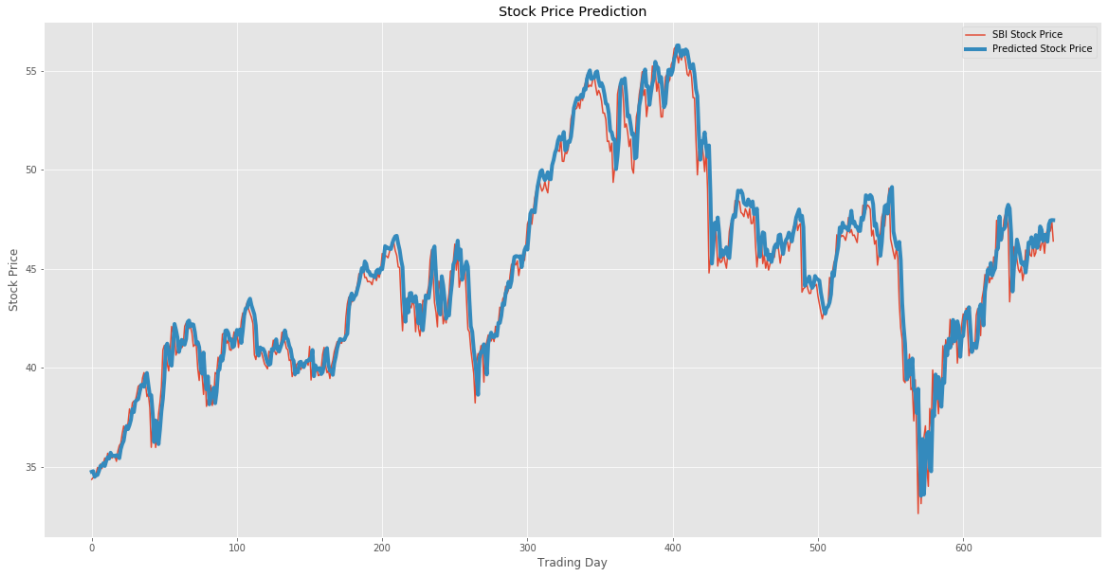


Figure 18:LSTM Prediction on test data.

It can be observed that the predictions are very good based on LSTM model, however if we increase the time stamp the result could get mis-calculated.

4.11 Evaluating Predictions

The prediction results are evaluated by computing the errors i.e., rmse (root mean sqaure error) and mae(mean absolute error). With lower value of rmse and mae, represent best model under consideration. In RMSE the mean error values squared and root been considered and in MAE as absolute mean considered as error. We take RMSE and MAE on test data prediction for each technique and find the best model in terms of RMSE and MAE. We got the absolute errors by subtracting the observed values from the true ones.By computing mean of all computed errors, we finally got our desired mean absolute errors required for analysis. Lower the value of rmse and mae, best the model will be.Forecasting is one of the common application of rmse where it has been used widely.

Comparison of Different Techniques		
Techniques	RMSE	MAE
LSTM(RNN)	5.22	1.91
SVM	1114.69	23.58
Random Forest	332.98	10.33
Moving Average	8.42	2.46
ARIMA	76.97	8.18
Naïve Approach	24.94	4.38
Moving Approach	8.42	2.46
Average Approach	100.23	9.86

It can be recapitulated that LSTM is performing very well , while SVM is worst performing.In our experiment, we have used a RNN, RF and SVMs because it was faster to run with less dam-age. From the above fig, we believe that these classifiers can accurately estimate the time range with the majority of the loss with respect to INX predictor. For providing better outcomes while maintaining high level of performance, the classification appeared to have fine-tuned.

5 Discussion

We used different time series methods to predict short-term stock movements. Separate observations were made for each technique. We were able to infer that LSTM (RNN) is the best working algorithm for predictions after calculating the root-mean-squared and the mean absolute error.

From reported outcomes achieved, the research presented the following set of contribution which are stated below:

- Emphasised on the effect pertaining to news/ notifications sentiment on stock price fluctuation/ movement.
- Determined/ identified effective time interval in estimation on stock price.
- Identified best news on scenario under each of the stock remaining affected varied as a result of notifications/ news.
- The designated model analysis tend to provide close price/ trend which is estimated via different AI models relative to yesterday's closing price.
- • Our achieved/ delivered model appeared to have been applied in variety of approaches.

Initially, the designated model could be employed by merchants who are devoid of any information program. Such traders could only assess the difference in price of our model and facilitate traders for carrying out one's analysis. Also this could be utilized by our automated approach of trading system devoid of any sort of monitoring, wherein the system could either open/ close trades on the basis of achieved predictions. Lastly, the developed code from the study could be effectively be deployed for performing of trading under shorter-terms.

A user-friendly interface must be provided in order to target the general audience. Despite their application, it is limited and cannot be applicable for all users to carry out/ run command-line based interfacing tool followed with communicating properly. Furthermore, ML classification and usage of in-depth learning are computationally heavy. After long-term users have fully implemented their CPU / GPU, there may be an obstacle in utilization of the desired solution. From the following section, it could be determined that the study will effectively rely on numerous steps for turning our project as a product with a proof-of-concept being released for purpose of general public and especially post-troubleshooting on several businesses as logic processors seemingly appeared quite Harsh. The research investigation observed for the study facilitates further towards designing POC to be a part of web service for providing greater benefits in the process.

6 Current Limitations

From the observed investigation so far, the current issues concerning with the product interface appears to be quite limited and concerning with the means governing with the proof-of-concept also. It appears to be not used as the finalized product to be rendered in public under any circumstances without prior investigation furthermore. In fact, it does not implement JavaScript-based frontend input sanitization-checking since they deliver an exponential time for ensuring on the fact as UI is well-defined and cleaned as input appeared to be correct. Additionally, option concerning with ticker indexing appeared hardcoded as HTML format, since there is currently no service that provides a complete list of ticker indicators that can be accessed via data tier.

For the sake of achieving such goal, the project could further then be extended under several other forms of etiquettes:

- Machine Implementation followed with additional ML classification.
- Expand the API over dynamically obtained return from the list comprised of taxonomic as well as index tickers.
- Speeding up process involving with a pre-setting of outcome for the popular index ticker as well as with caching achieved from user queries.
- JavaScript which involve Implementing (sanitization) JavaScript techniques
- Addition of authentication over the API backend.

7 Conclusion

Compared to the baseline, the forecast shows a useful trend trend with the actual stock trend. Through the application interface, the user can easily compare estimates and model scores from different machine learning models, and then select the one that best suits their choice. The model improves itself by exploring the improved model topology, structure and hyperemeters through the model development algorithm used in the application. Found the utility of the growth algorithm in reducing the average squared error when an estimating stock price, which helps retail investors improve trend forecasting. Therefore, with the results of the application and research, the project team achieved the goal of creating a user-friendly system for retail investors who had no previous technology.

By evaluating accuracy pertaining with different algorithms, it is observed that RF algorithm is prominent and is more appropriate form of algorithm for estimating market value on stock on the basis of different data points from historical data. The following algorithm would serve as great asset to brokers as well as investors for investing money over stock market since they trained over vast collection emphasising on historical data being selected after testing over sampling data. The research study demonstrated that ML models are effective towards estimation of stock price with a much greater accuracy compared with previously utilized ML models.

The resolutions of this project are to build on possible improvements. First, many approaches may be explored in the future to generate issues such as whether the stock price is up or down (binary classification) based on previous stock prices. Other features such as market news and sentiment may be included. Combined with the development of more sophisticated machine learning methods, the accuracy of the information given to retail investors can be greatly improved. Second, large-scale growth with larger population sizes and more recurrences can also be tested to get better results.

8 Future Enhancement

Addition of several attributes/ parameters/ factors such as financial ratios, followed with multiple instances, etc to enhance this project will be the future scope. The accurate are the results when more parameters are involved. For determining the overall patterns/relationships that lies among customer & corporate employee and also for analysing the contents of public comments various algorithms can be applied. On the whole, prediction of the corporation's performance structure can be done using the traditional algorithms & data mining techniques also.

References

[1] Burton G M. Returns from investing in equity mutual funds 1971 to 1991. The Journal of finance. 1995;50(2):549–572. Available from: https://doi.org/10.1111/j.1540-6261.1995.tb04795.x.

[2] Yaser S AM, Amir F A. Introduction to financial forecasting. Applied intelligence. 1996;6(3):205–213. Available from: https://doi.org/10.1007/BF00126626.

[3] Francis EH T, Lijuan C. Application of support vector machines in financial time series forecasting. omega. 2001;29(4):309–317. Available from: https://doi.org/10.1016/S0305-0483(01)00026-3.

[4] Eugene F F. Efficient capital markets: A review of theory and empirical work. The journal of Finance. 1970;25(2):383–417. Available from: https://doi.org/10.2307/2325486.

[5] D H, GG R. Is there a new neutral lepton? Physics Letters B. 1977;67(4):460–462. Available from: https://doi.org/10.1016/0370-2693(77)90444-0.

[6] Fama E, Marshall B. Filter Rules and Stock Market Trading. Journal of Business, Security Prices: A Supplement. 1966;39(1). Available from: https://doi.org/10.1086/294849.

[7] Arévalo A, Niño J, Hernández G, Sandoval J. High-frequency trading strategy based on deep neural networks. In: International conference on intelligent computing. Springer; 2016. p. 424–436. Available from: https://doi.org/10.1007/978-3-319-42297-8_40.

[8] Stephen F W, Jenna R, Joe K, Jonathan M G, Nadeem Q. Can machine-learning improve cardiovascular risk prediction using routine clinical data? PloS one. 2017;12(4):e0174944. Available from: https://doi.org/10.1371/journal.pone.0174944.

[9] Robert P S, Hsinchun C. Textual analysis of stock market prediction using breaking financial news: The AZFin text system. ACM Transactions on Information Systems (TOIS). 2009;27(2):1–19. Available from: https://doi.org/10.1145/1462198.1462204.

[10] Dariusz A, Janusz K, Hiroshi A, Mitsuo O. Identification of uniaxial tension tests of concrete based on machine learning technique. In: Brittle Matrix Composites 8. Elsevier; 2006. p. 195–204. Available from: https://doi.org/10.1533/9780857093080.195.

[11] Ryo A, Akira Y, Takashi M, Kuniaki U. Deep learning for stock prediction using numerical and textual information. In: 2016 IEEE/ACIS 15th International Conference on Computer and Information Science (ICIS). IEEE; 2016. p. 1–6. Available from: https://doi.org/10.1109/ICIS.2016.7550882.

[12] Johan B, Huina M, Alberto P. Determining the Public Mood State by Analysis of Microblogging Posts. In: ALIFE; 2010. p. 667–668. Available from: https://doi.org/10.1533/9780857093080.195.

[13] Xiao D, Yue Z, Ting L, Junwen D. Using structured events to predict stock price movement: An empirical investigation. In: Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP); 2014. p. 1415–1425. Available from: http://dx.doi.org/10.3115/v1/D14-1148.

[14] Sepp H, Jürgen S. Long short-term memory. Neural computation. 1997;9(8):1735–1780. Available from: https://doi.org/10.1162/neco.1997.9.8.1735.

[15] John T M, Carol C B, Kenneth B K. Benchmarking sales forecasting management. Business Horizons. 1999;42(3):48–57. Available from: https://doi.org/10.1016/S0007-6813(99)80021-4.

[16] Jigar P, Sahil S, Priyank T, Ketan K. Predicting stock and stock price index movement using trend deterministic data preparation and machine learning techniques. Expert systems with applications. 2015;42(1):259–268. Available from: https://doi.org/10.1016/j.eswa.2014.07.040.

[17] Teak-Wei C, Boon-Giin L. American sign language recognition using leap motion controller with machine learning approach. Sensors. 2018;18(10):3554. Available from: https://doi.org/10.3390/s18103554.

[18] George A, Kimon P V. Surveying stock market forecasting techniques–Part II: Soft computing methods. Expert Systems with applications. 2009;36(3):5932–5941. Available from: https://doi.org/10.1016/j.eswa.2008.07.006.

[19] Luckyson K, Snehanshu S, Sudeepa Roy D. Predicting the direction of stock market prices using random forest. arXiv preprint arXiv:160500003. 2016;.

[20] W M, et al. Bias of the Random Forest out-of-bag (OOB) error for certain input parameters. Open Journal of Statistics. 2011;2011. Available from: http://www.scirp.org/journal/PaperInformation.aspx?PaperID=8072.

[21] Hongwei L, Zheng B. Radar HRR profiles recognition based on SVM with power-transformed-correlation kernel. In: International Symposium on Neural Networks. Springer; 2004. p. 531–536. Available from: https://doi.org/10.1007/978-3-540-28647-9_88.

[22] Alex A F. Data mining and knowledge discovery with evolutionary algorithms. Springer Science & Business Media; 2013. Available from: https://doi.org/10.1007/978-3-662-04923-5.

[23] Paul A S. Proof that properly anticipated prices fluctuate randomly. In: The world scientific handbook of futures markets. World Scientific; 2016. p. 25–38. Available from: https://doi.org/10.1142/9789814566926_0002.

[24] Nikola G, Ramazan G. Fuzzy logic, trading uncertainty and technical trading. Journal of Banking & Finance. 2013;37(2):578–586. Available from: https://doi.org/10.1016/j.jbankfin.2012.09.012.

Appendix

Code

▾ Importing Relevant Packages

```
#Data Handling packages
import pandas as pd
import numpy as np

#for dates
import datetime as dt

#for fetching data from yahoo finances
import pandas_datareader.data as web

#for visualizations
import matplotlib.pyplot as plt
from matplotlib import style
import plotly.graph_objects as go

#maths packages
from math import sqrt

#ML packages
from sklearn.metrics import mean_squared_error
from sklearn.metrics import mean_absolute_error

#Presentable Plots
style.use('ggplot')
```

▾ Fetching Stock Data

```
[3] #Ten year period for fetching stock data, starting from 2010 you can change the period from here.
    start = dt.datetime(2010,1,1)
    end = dt.datetime.now()
```

```
#User adds any stock they want to query and Yahoo Finance goes out and get the stock data
mystock = input("Enter a Stock you are interested in: ")
print("Fetching Data for: " +mystock)

#Yahoo goest out and gets the stock data for the stock you entered plus the S&P 500 for the past 10 years
df = web.DataReader(mystock, 'yahoo', start,end)
sp = web.DataReader('SPY', 'yahoo', start,end)

#Enter the full name of the companies stock symbol you entered
mycompany = input("Enter the full name of the companies stock: ")

#Make company name and stock symbol equal to each other
mystock = mycompany
```

```
Enter a Stock you are interested in: CSCO
Fetching Data for: CSCO
Enter the full name of the companies stock: CSCO
```

```
[6] df.head()
```

Date	High	Low	Open	Close	Volume	Adj Close
2010-01-04	24.840000	24.010000	24.110001	24.690001	59853700.0	18.785395
2010-01-05	24.730000	24.379999	24.600000	24.580000	45124500.0	18.701702
2010-01-06	24.740000	24.340000	24.540001	24.420000	35715700.0	18.579962
2010-01-07	24.570000	24.170000	24.299999	24.530001	31531200.0	18.663656
2010-01-08	24.700001	24.250000	24.379999	24.660000	39115900.0	18.762569

```
[7] sp.head()
```

Date	High	Low	Open	Close	Volume	Adj Close
2010-01-04	113.389999	111.510002	112.370003	113.330002	118944600.0	91.841896
2010-01-05	113.680000	112.849998	113.260002	113.629997	111579900.0	92.084984
2010-01-06	113.989998	113.430000	113.519997	113.709999	116074400.0	92.149803
2010-01-07	114.330002	113.180000	113.500000	114.190002	131091100.0	92.538841
2010-01-08	114.620003	113.660004	113.889999	114.570000	126402800.0	92.846756

```
[8] df.tail()
```

Date	High	Low	Open	Close	Volume	Adj Close
2020-08-20	42.419998	41.730000	41.869999	42.310001	18443900.0	42.310001
2020-08-21	42.299999	41.810001	42.049999	42.250000	17389600.0	42.250000
2020-08-24	42.470001	42.000000	42.070000	42.180000	17742900.0	42.180000
2020-08-25	42.299999	41.779999	42.230000	41.959999	15407000.0	41.959999
2020-08-26	42.055000	41.450001	41.650002	41.895000	7904001.0	41.895000

```
[9] sp.tail()
```

Date	High	Low	Open	Close	Volume	Adj Close
2020-08-20	338.799988	335.220001	335.359985	338.279999	42207800.0	338.279999
2020-08-21	339.720001	337.549988	337.920013	339.480011	55106600.0	339.480011
2020-08-24	343.000000	339.450012	342.119995	342.920013	48588700.0	342.920013
2020-08-25	344.209991	342.269989	343.529999	344.119995	38369500.0	344.119995
2020-08-26	347.019989	344.170013	344.760010	346.584991	18432928.0	346.584991

```
print(df.shape)
print(sp.shape)
```

```
(2681, 6)
(2681, 6)
```

▾ Exploring Data

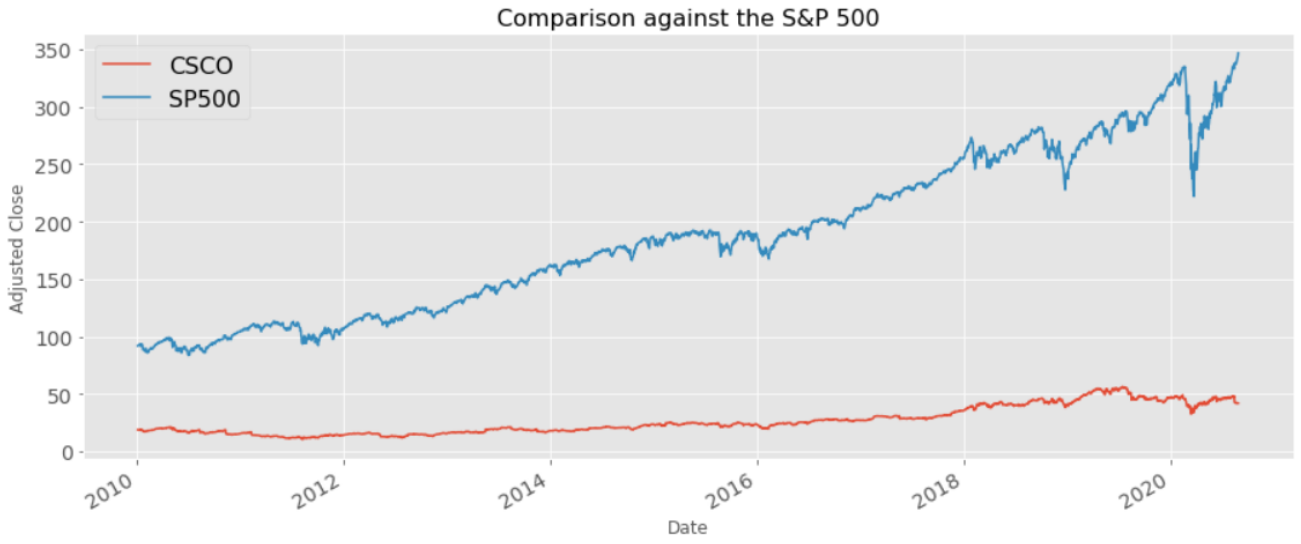
```
[12] df.describe()
```

	High	Low	Open	Close	Volume	Adj Close
count	2681.000000	2681.000000	2681.000000	2681.000000	2.681000e+03	2681.000000
mean	30.269286	29.719653	29.994793	30.000117	3.532616e+07	26.432846
std	10.971089	10.767250	10.875111	10.870868	2.740465e+07	11.893064
min	14.120000	13.300000	13.930000	13.730000	6.155800e+06	10.522875
25%	21.799999	21.400000	21.570000	21.590000	2.014400e+07	17.040558
50%	27.350000	26.820000	27.070000	27.059999	2.809750e+07	23.008736
75%	38.470001	37.419998	37.840000	37.910000	4.278470e+07	35.270294
max	58.259998	57.869999	58.130001	58.049999	5.600402e+08	56.213196

```
[13] sp.describe()
```

	High	Low	Open	Close	Volume	Adj Close
count	2681.000000	2681.000000	2681.000000	2681.000000	2.681000e+03	2681.000000
mean	203.992410	201.836438	202.961761	203.000994	1.261058e+08	186.660275
std	63.321391	62.734547	63.041138	63.045576	7.528718e+07	68.592140
min	103.419998	101.129997	103.110001	102.199997	1.843293e+07	83.558922
25%	141.419998	140.130005	140.649994	140.850006	7.420340e+07	119.956978
50%	204.139999	201.919998	203.169998	203.210007	1.062992e+08	183.631729
75%	259.040009	256.410004	257.679993	258.049988	1.561338e+08	246.587097
max	347.019989	344.170013	344.760010	346.584991	7.178287e+08	346.584991

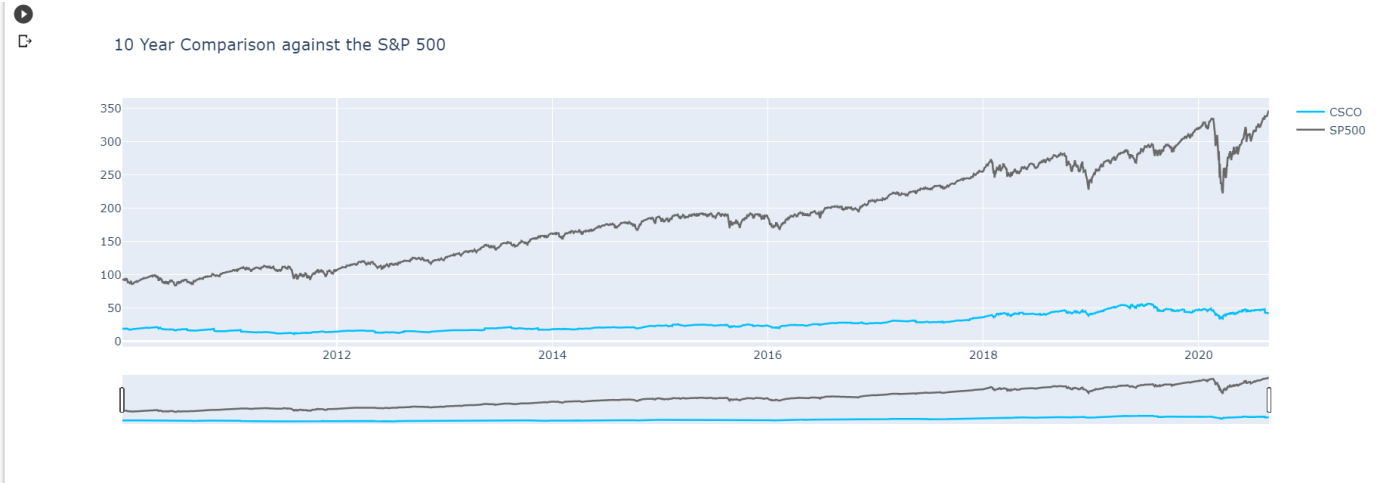
```
[14] df['Adj Close'].plot(figsize=(15,6), label=mystock, fontsize=14, grid=True)
     sp['Adj Close'].plot(figsize=(15,6), label="SP500", fontsize=14, grid=True)
     plt.legend(fontsize=16)
     plt.ylabel("Adjusted Close")
     plt.title("Comparison against the S&P 500",fontsize=16)
     plt.show()
```



```
fig = go.Figure()
fig.add_trace(go.Scatter(x=df.index, y=df['Adj Close'], name=mystock,
                         line_color='deepskyblue'))

fig.add_trace(go.Scatter(x=sp.index, y=sp['Adj Close'], name="SP500",
                         line_color='dimgray'))

fig.update_layout(title_text="10 Year Comparison against the S&P 500",
                  xaxis_rangeslider_visible=True)
fig.show()
```



Applying Predictive Techniques

```
[16] # Getting Train and Test Data
     train=df[:'Dec 2017']
     test=df['Dec 2017':]
```

```
[17] print(train.shape)
     print(test.shape)
```

```
(2013, 6)
(688, 6)
```

```
[20] train.head()
```

Date	High	Low	Open	Close	Volume	Adj Close
2010-01-04	24.840000	24.010000	24.110001	24.690001	59853700.0	18.785395
2010-01-05	24.730000	24.379999	24.600000	24.580000	45124500.0	18.701702
2010-01-06	24.740000	24.340000	24.540001	24.420000	35715700.0	18.579962
2010-01-07	24.570000	24.170000	24.299999	24.530001	31531200.0	18.663656
2010-01-08	24.700001	24.250000	24.379999	24.660000	39115900.0	18.762569

```
[21] train.tail()
```

Date	High	Low	Open	Close	Volume	Adj Close
2017-12-22	38.740002	38.470001	38.520000	38.549999	11441600.0	35.500523
2017-12-26	38.680000	38.360001	38.549999	38.480000	8186100.0	35.436054
2017-12-27	38.650002	38.450001	38.540001	38.560001	10543000.0	35.509731
2017-12-28	38.730000	38.450001	38.730000	38.590000	8807700.0	35.537357
2017-12-29	38.619999	38.299999	38.410000	38.299999	12583600.0	35.270294

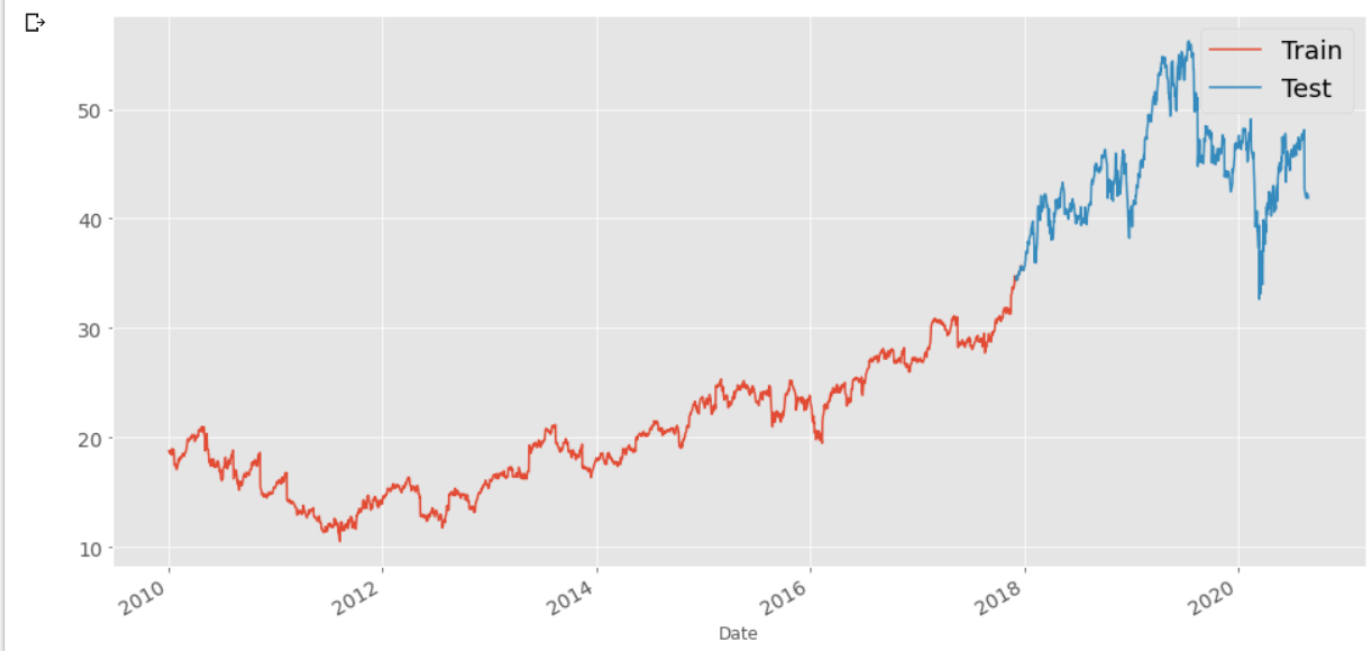
```
[22] test.head()
```

Date	High	Low	Open	Close	Volume	Adj Close
2017-12-01	37.660000	36.730000	37.090000	37.599998	26920900.0	34.625671
2017-12-04	37.990002	37.540001	37.810001	37.720001	29414400.0	34.736176
2017-12-05	37.820000	37.169998	37.740002	37.310001	23288200.0	34.358616
2017-12-06	37.720001	37.250000	37.369999	37.410000	16581800.0	34.450699
2017-12-07	37.779999	37.259998	37.259998	37.400002	17092000.0	34.441494

```
test.tail()
```

Date	High	Low	Open	Close	Volume	Adj Close
2020-08-20	42.419998	41.730000	41.869999	42.310001	18443900.0	42.310001
2020-08-21	42.299999	41.810001	42.049999	42.250000	17389600.0	42.250000
2020-08-24	42.470001	42.000000	42.070000	42.180000	17742900.0	42.180000
2020-08-25	42.299999	41.779999	42.230000	41.959999	15407000.0	41.959999
2020-08-26	42.055000	41.450001	41.650002	41.895000	7904001.0	41.895000

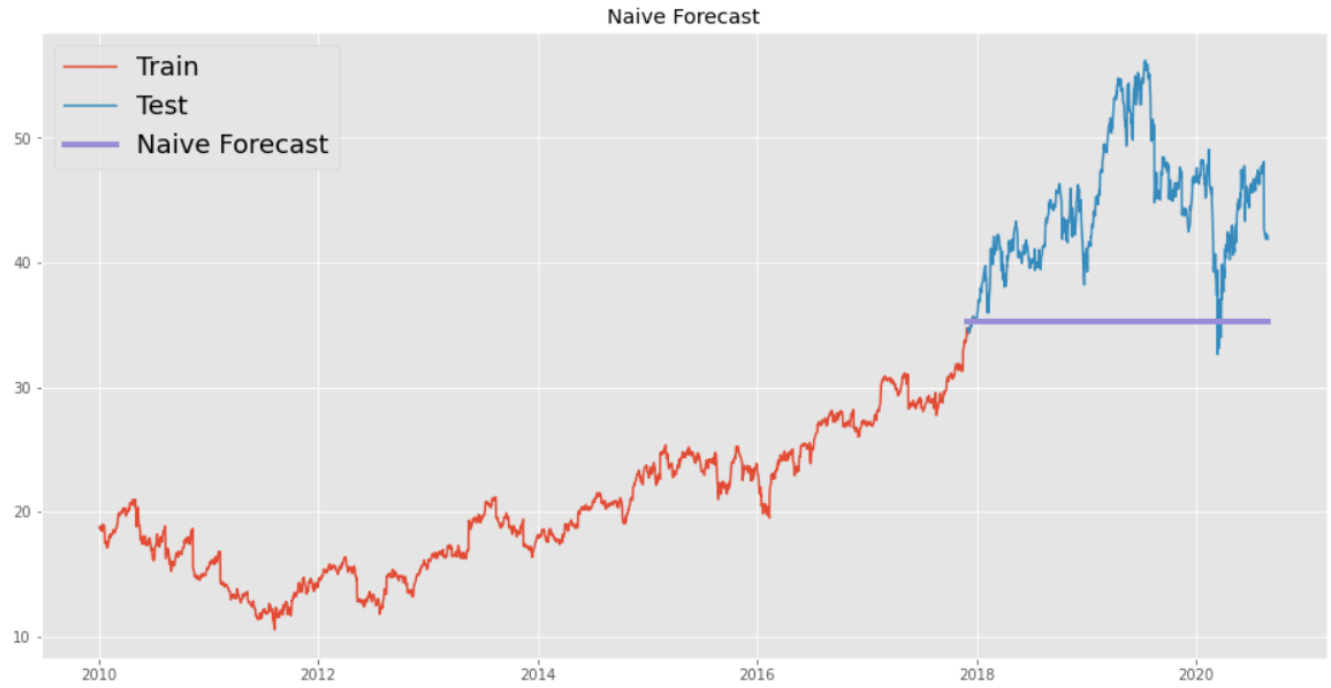
```
train['Adj Close'].plot(figsize=(16,8), label='Train',fontsize=14)
test['Adj Close'].plot(figsize=(16,8), label='Test',fontsize=14)
plt.legend(loc='best',fontsize=18)
plt.grid(True)
plt.show()
```



Naive Approach

```
dd= np.asarray(train['Adj Close'])
y_hat = test.copy()
y_hat['naive'] = dd[len(dd)-1]
plt.figure(figsize=(16,8))
plt.plot(train.index, train['Adj Close'], label='Train')
plt.plot(test.index,test['Adj Close'], label='Test')
plt.plot(y_hat.index,y_hat['naive'], label='Naive Forecast',linewidth=4)
plt.legend(loc='best',fontsize=18)
plt.title("Naive Forecast")
plt.grid(True)
plt.show()

rms_naive = sqrt(mean_squared_error(test['Adj Close'], y_hat.naive))
mae_naive = sqrt(mean_absolute_error(test['Adj Close'], y_hat.naive))
print('RMSE=',rms_naive)
print('MAE=',mae_naive)
```

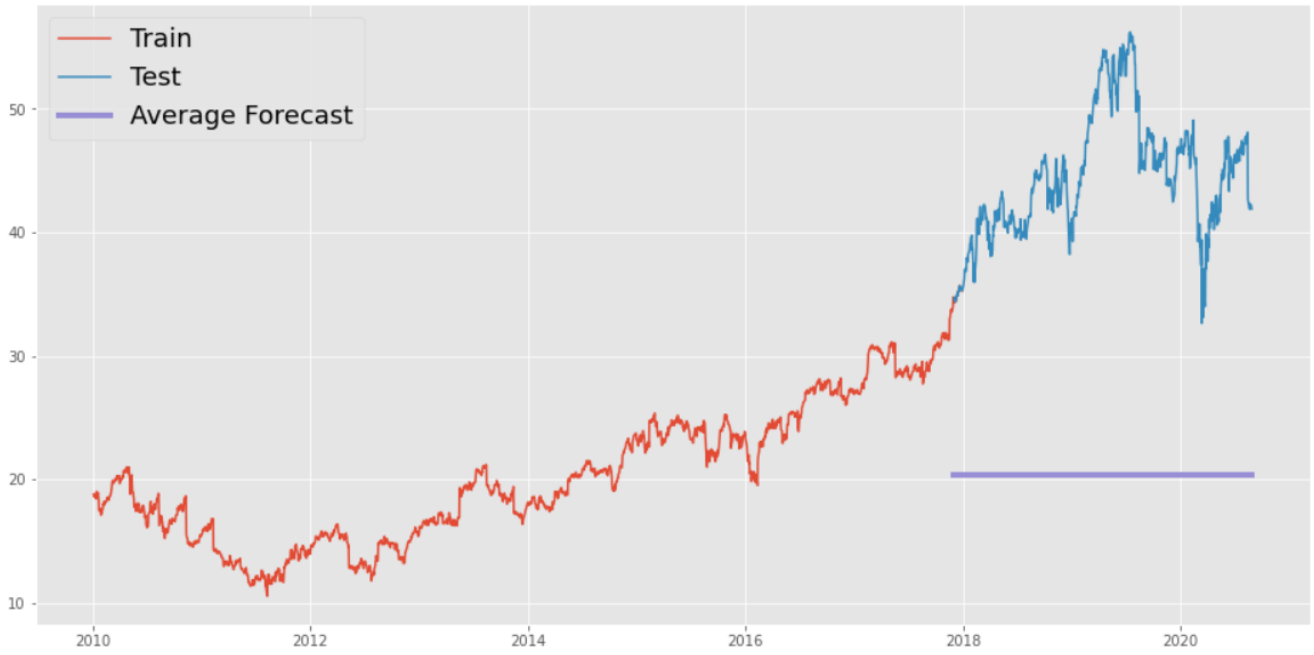



```
RMSE= 10.518390154496505
MAE= 3.0547951055573828
```

Average approach

```
[ ]  y_hat_avg = test.copy()
     y_hat_avg['avg_forecast'] = train['Adj Close'].mean()
     plt.figure(figsize=(16,8))
     plt.plot(train['Adj Close'], label='Train')
     plt.plot(test['Adj Close'], label='Test')
     plt.plot(y_hat_avg['avg_forecast'], label='Average Forecast',linewidth=4)
     plt.grid(True)
     plt.legend(loc='best',fontsize=18)
     plt.show()

     rms_avg = sqrt(mean_squared_error(test['Adj Close'], y_hat_avg.avg_forecast))
     mae_avg = sqrt(mean_absolute_error(test['Adj Close'], y_hat_avg.avg_forecast)
     print('RMS=',rms_avg)
     print('MAE=',mae_avg)
```

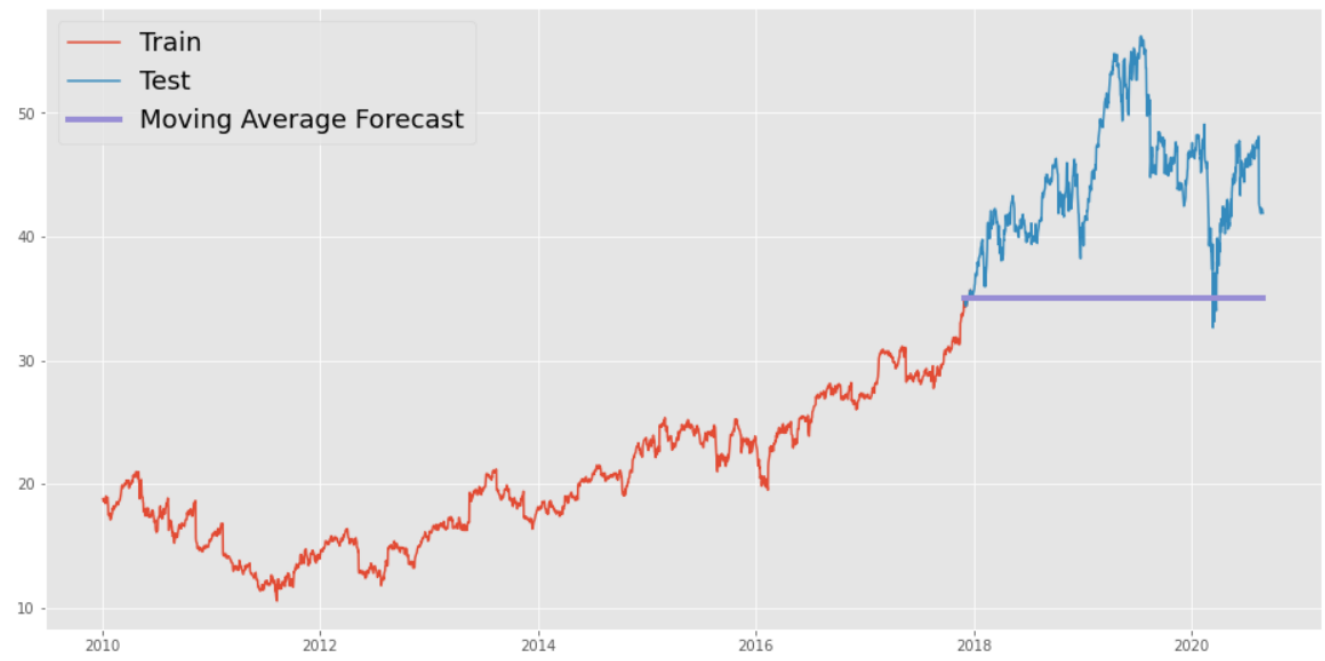



```
RMS= 24.73956474875596
MAE= 4.92392656884646
```

Moving Average on Test Data only

```
y_hat_avg = test.copy()
y_hat_avg['moving_avg_forecast'] = train['Adj Close'].rolling(20).mean().iloc[-1]
plt.figure(figsize=(16,8))
plt.plot(train['Adj Close'], label='Train')
plt.plot(test['Adj Close'], label='Test')
plt.plot(y_hat_avg['moving_avg_forecast'], label='Moving Average Forecast',linewidth=4
plt.grid(True)
plt.legend(loc='best',fontsize=18)
plt.show()

rms_ma = sqrt(mean_squared_error(test['Adj Close'], y_hat_avg.moving_avg_forecast))
mae_ma = sqrt(mean_absolute_error(test['Adj Close'], y_hat_avg.moving_avg_forecast))
print('RMS=',rms_ma)
print('MAE=',mae_ma)
```



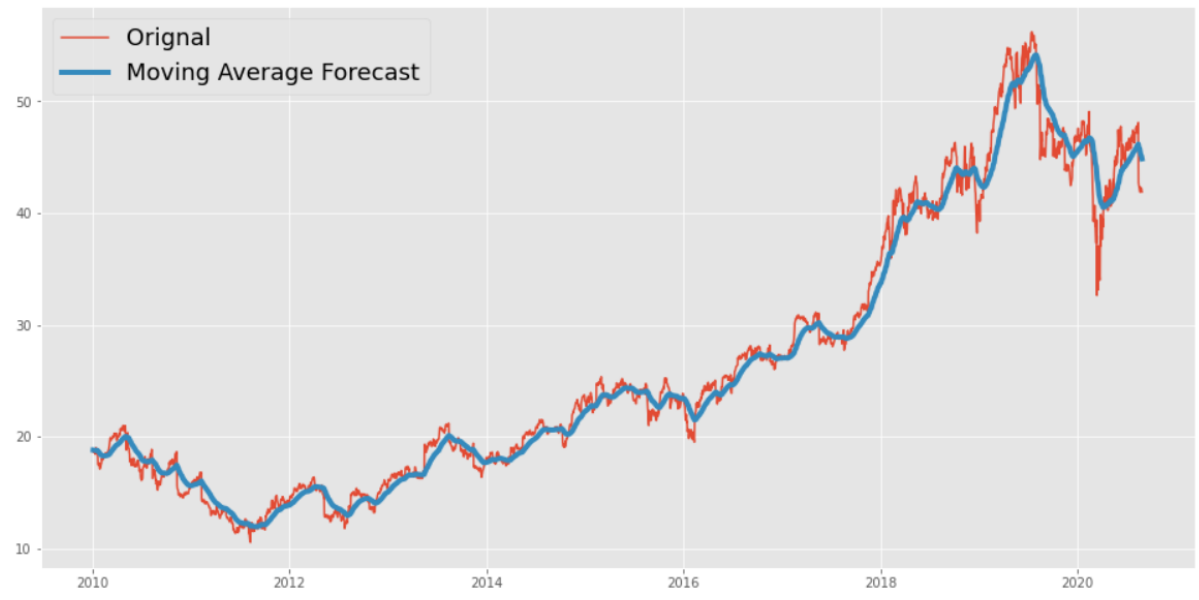

```
RMS= 10.694177669123592
MAE= 3.0856847552317146
```

Moving avarage on full Data

```
y_hat_avg = df.copy()
y_hat_avg['moving_avg_forecast']=df['Adj Close'].ewm(span=50, adjust=False).mean()
plt.figure(figsize=(16,8))
plt.plot(df['Adj Close'], label='Orignal')
plt.plot(y_hat_avg['moving_avg_forecast'], label='Moving Average Forecast',linewidth=4)

plt.grid(True)
plt.legend(loc='best',fontsize=18)
plt.show()

rms_ma = sqrt(mean_squared_error(df['Adj Close'], y_hat_avg.moving_avg_forecast))
print('RMS=',rms_ma)
mae_ma = sqrt(mean_absolute_error(df['Adj Close'], y_hat_avg.moving_avg_forecast))
print('MAE=',mae_ma)
```




```
RMS= 1.4773705458803825
MAE= 1.0315821050823324
```

Decomposing Time-Series Components

```
[29] df_weekly = df.resample('W').mean()
```

```
[31] import statsmodels.api as sm
     res=sm.tsa.seasonal_decompose(df_weekly['Adj Close'])
```

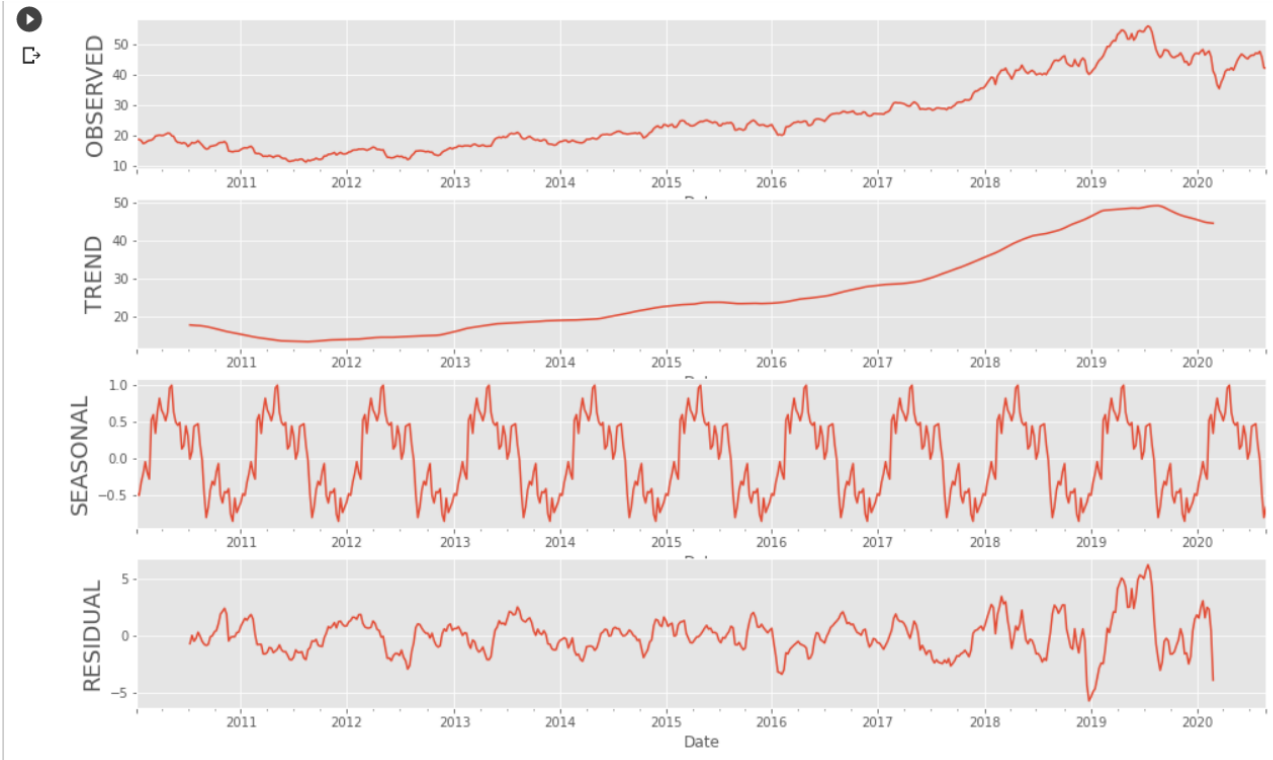
```
plt.figure(figsize=(16,10))
plt.subplot(4,1,1)
res.observed.plot()

plt.ylabel('OBSERVED',fontsize=18)
plt.subplot(4,1,2)
res.trend.plot()

plt.ylabel('TREND',fontsize=18)
plt.subplot(4,1,3)
res.seasonal.plot()

plt.ylabel('SEASONAL',fontsize=18)
plt.subplot(4,1,4)
res.resid.plot()

plt.ylabel('RESIDUAL',fontsize=18)
plt.show()
```



ARIMA

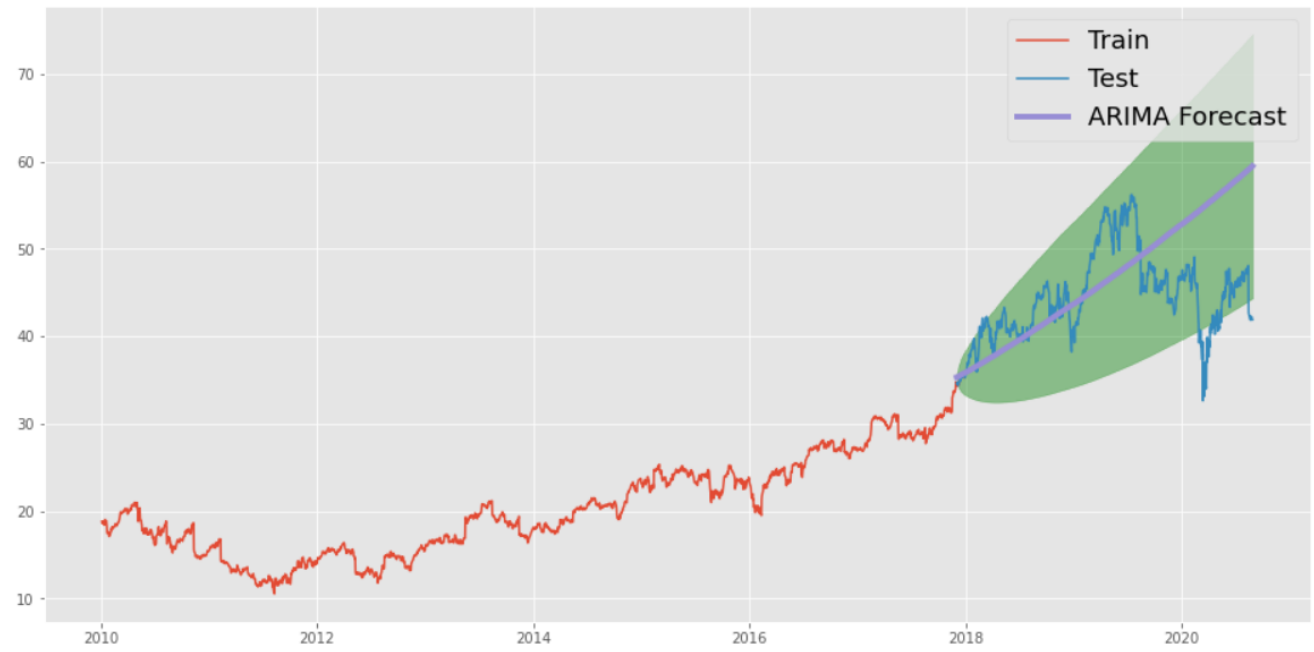
```
[33] from statsmodels.tsa.arima_model import ARIMA
     y_hat_avg = test.copy()
     model = ARIMA(train['Adj Close'], order=(1,2,1))
     model_fit=model.fit()
```

```
[34] #model_fit.predict(start='2013-11-01',end='2013-11-05',typ='levels')
     y_hat_avg['ARIMA']=model_fit.forecast(len(test))[0]
     y_hat_avg['2.5% CI']=model_fit.forecast(len(test))[2][:,0]
     y_hat_avg['97.5% CI']=model_fit.forecast(len(test))[2][:,1]
     y_hat_avg.head()
```

Date	High	Low	Open	Close	Volume	Adj Close	ARIMA	2.5% CI	97.5% CI
2017-12-01	37.660000	36.730000	37.090000	37.599998	26920900.0	34.625671	35.300127	34.723772	35.876481
2017-12-04	37.990002	37.540001	37.810001	37.720001	29414400.0	34.736176	35.328496	34.515436	36.141555
2017-12-05	37.820000	37.169998	37.740002	37.310001	23288200.0	34.358616	35.356892	34.361919	36.351865
2017-12-06	37.720001	37.250000	37.369999	37.410000	16581800.0	34.450699	35.385308	34.236880	36.533735
2017-12-07	37.779999	37.259998	37.259998	37.400002	17092000.0	34.441494	35.413744	34.130074	36.697414

```
plt.figure(figsize=(16,8))
plt.plot(train['Adj Close'], label='Train')
plt.plot(test['Adj Close'], label='Test')
plt.plot(y_hat_avg['ARIMA'], label='ARIMA Forecast',linewidth=4)
plt.fill_between(y_hat_avg.index,y1=y_hat_avg['2.5% CI'],y2=y_hat_avg['97.5% CI'],alph
plt.grid(True)
plt.legend(loc='best',fontsize=18)
plt.show()

rms_arima = sqrt(mean_squared_error(test['Adj Close'], y_hat_avg.ARIMA))
print("RMSE=",rms_arima)
mae_arima = sqrt(mean_absolute_error(test['Adj Close'], y_hat_avg.ARIMA))
print("MAE=",mae_arima)
```




```
RMSE= 7.249649915364026
MAE= 2.3485482519653567
```

Data Preprocessing

```
data=df

data=data.drop(['Volume'],axis=1)

columns=data.columns

df_stats = pd.DataFrame(columns=['P-value'], index=columns)
for col in columns:
  dftest = adfuller(data[col], autolag='AIC')
  df_stats.loc[col]=dftest[1]

data_diff = pd.DataFrame(columns=columns)
data_diff['Date']=data.index
for col in columns:
  data_diff[col] = pd.DataFrame(np.diff(data[col]))

data_diff=data_diff.dropna()
for col in columns:
  dftest = adfuller(data_diff[col], autolag='AIC')
  df_stats.loc[col]=dftest[1]
```

```
lags = pd.DataFrame()
for i in range(10,0,-1):
    lags['t-'+str(i)] = data_diff["Close"].shift(i)
    lags['t'] = data_diff['Close'].values
lags = lags[13:]

from sklearn.ensemble import RandomForestRegressor
array = lags.values
X = array[:,0:-1]
y = array[:,-1]
model = RandomForestRegressor(n_estimators=500, random_state=1)
model.fit(X, y)
```

```
RandomForestRegressor(bootstrap=True, ccp_alpha=0.0, criterion='mse',
                      max_depth=None, max_features='auto', max_leaf_nodes=None,
                      max_samples=None, min_impurity_decrease=0.0,
                      min_impurity_split=None, min_samples_leaf=1,
                      min_samples_split=2, min_weight_fraction_leaf=0.0,
                      n_estimators=500, n_jobs=None, oob_score=False,
                      random_state=1, verbose=0, warm_start=False)
```

```
#Feature Selection
from sklearn.feature_selection import RFE
rfe = RFE(RandomForestRegressor(n_estimators=500, random_state=1),
fit = rfe.fit(X, y)
names = lags.columns
columns=[]
for i in range(len(fit.support_)):
    if fit.support_[i]:
        columns.append(names[i])

print("Columns with predictive power:", columns )
```

```
Columns with predictive power: ['t-10', 't', 't-8', 't-2']
```

```
#pre-processing data for final models
df_forecasting=pd.DataFrame(data["Close"])
df_forecasting["Close_diff"] = df_forecasting["Close"].diff()
for i in range(4,0,-1):
    df_forecasting['t-'+str(i)] = df_forecasting["Close"].shift(i)
df_forecasting=df_forecasting.dropna()
df_forecasting["Close_rolling"] = df_forecasting["Close"].rolling(window = 80).mean()
df_forecasting= df_forecasting.dropna()
```

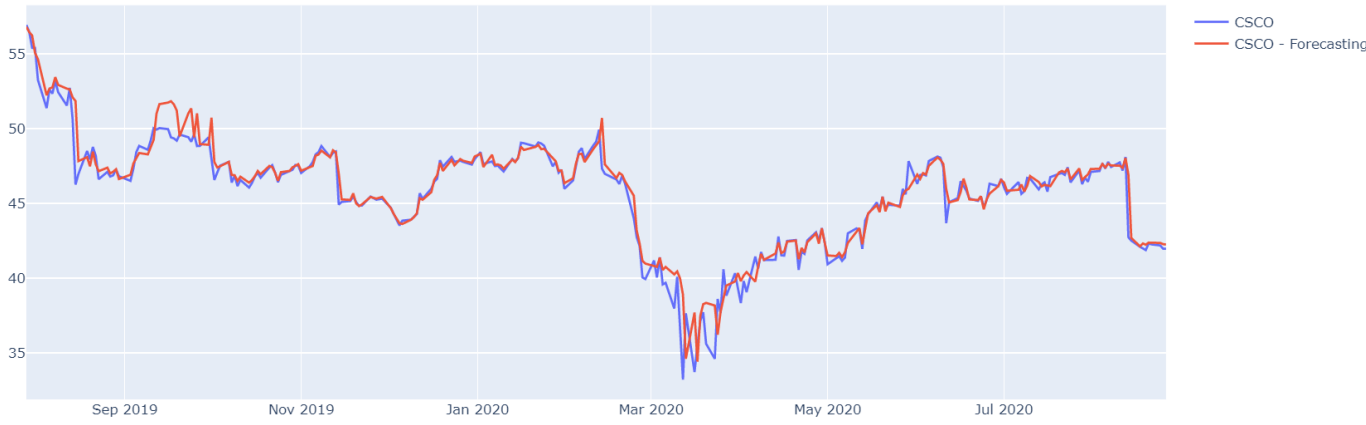
Random Forrest

```
[8] #model
    from sklearn.ensemble import RandomForestRegressor
    x=df_forecasting.iloc[:,1:]
    y=df_forecasting.iloc[:,0]
    x_train, x_valid = x.loc[x.index < '2019-07-27'], x.loc[x.index >= '2019-07-27']
    y_train, y_valid = y.loc[y.index < '2019-07-27'], y.loc[y.index >= '2019-07-27']
    mdl = RandomForestRegressor(n_estimators=100)
    mdl.fit(x_train, y_train)
    pred=mdl.predict(x_valid)
    pred=pd.Series(pred, index=y_valid.index)
```

```
[9] mae_rforest = print("MAE:",mean_absolute_error(y_valid, pred))
    rms_rforest = print("MSE:",mean_squared_error(y_valid, pred))
```

```
MAE: 0.5088806363961994
MSE: 0.9531082265735464
```

```
import plotly.graph_objects as go
fig = go.Figure()
fig.add_trace(go.Scatter(x=y_valid.index, y=y_valid.values, mode='lines', name=mycompany))
fig.add_trace(go.Scatter(x=pred.index, y=pred.values, mode='lines', name=mycompany+' - Forecasting'))
```



SVM

```
[11] from sklearn.svm import SVR
```

```
[12] mdl =SVR()
```

```
[13] mdl.fit(x_train, y_train)
```

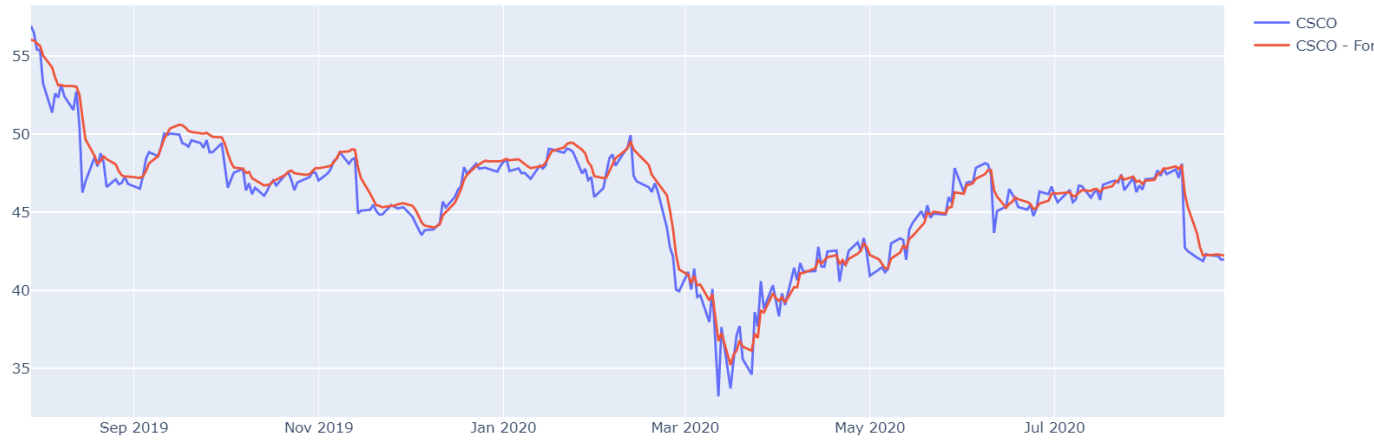
```
SVR(C=1.0, cache_size=200, coef0=0.0, degree=3, epsilon=0.1, gamma='scale',
    kernel='rbf', max_iter=-1, shrinking=True, tol=0.001, verbose=False)
```

```
[14] pred=mdl.predict(x_valid)
     pred=pd.Series(pred, index=y_valid.index)
```

```
[15] mae_svm=print("MAE:",mean_absolute_error(y_valid, pred))
     rms_svm=print("MSE:",mean_squared_error(y_valid, pred))
```

```
MAE: 0.6375741973576973
MSE: 0.8506133518928066
```

```
import plotly.graph_objects as go
fig = go.Figure()
fig.add_trace(go.Scatter(x=y_valid.index, y=y_valid.values, mode='lines', name=mycompany))
fig.add_trace(go.Scatter(x=pred.index, y=pred.values, mode='lines', name=mycompany+' - Forecasting'))
```



Long-Short-Term Memory (LSTM) Recurrent Neural Network

```
#Importing the packages for lstm implementation
from sklearn import model_selection
from sklearn.metrics import confusion_matrix
from sklearn.preprocessing import StandardScaler
from sklearn.model_selection import train_test_split
from sklearn.preprocessing import MinMaxScaler
from keras.models import Sequential
from keras.layers import Dense
from keras.layers import LSTM
from keras.layers import Dropout
```

```
[ ] print(train.shape)
    print(test.shape)
```

```
(1414, 6)
(665, 6)
```

```
[ ] #selecting adjusted close
    train_set = train.iloc[:,5:6].values
    test_set = test.iloc[:,5:6].values

    #scaling train and test set
    sc = MinMaxScaler(feature_range = (0, 1))
    training_set_scaled = sc.fit_transform(train_set)
    test_set_scaled = sc.fit_transform(test_set)
```

```python
#data preparation for lstm
from numpy import array

# split a univariate sequence into samples
def split_sequence(sequence, n_steps):
    X, y = list(), list()
    for i in range(len(sequence)):
        # find the end of this pattern
        end_ix = i + n_steps
        # check if we are beyond the sequence
        if end_ix > len(sequence)-1:
            break
        # gather input and output parts of the pattern
        seq_x, seq_y = sequence[i:end_ix], sequence[end_ix]
        X.append(seq_x)
        y.append(seq_y)
    return array(X), array(y)
```

```python
# define input sequence
raw_seq = training_set_scaled

# choose a number of time steps
time_steps = 2

# split into samples
X, y = split_sequence(raw_seq, time_steps)

# summarize the data
for i in range(len(X)):
    print(X[i], y[i])
```

```
 [0.83048501]] [0.8952886]
[[0.83048501]
 [0.8952886 ]] [0.89602112]
[[0.8952886 ]
 [0.89602112]] [0.91798815]
[[0.89602112]
 [0.91798815]] [0.92348014]
[[0.91798815]
 [0.92348014]] [0.91615775]
[[0.92348014]
 [0.91615775]] [0.91762235]
```

```
[44] # reshape from [samples, timesteps] into [samples, timesteps, features]
     n_features = 1
     X = X.reshape((X.shape[0], X.shape[1], n_features))
```

```
[45] #Defining Structure of a stacked LSTM Recurrent model
     regressor = Sequential()
     regressor.add(LSTM(units = 50, return_sequences = True, input_shape=(time_steps, n_features)))
     regressor.add(Dropout(0.2))
     regressor.add(LSTM(units = 50, return_sequences = True))
     regressor.add(Dropout(0.2))
     regressor.add(LSTM(units = 50, return_sequences = True))
     regressor.add(Dropout(0.2))
     regressor.add(LSTM(units = 50))
     regressor.add(Dropout(0.2))
     regressor.add(Dense(units = 1))
```

```
#Compiling and fitting the model
regressor.compile(optimizer = 'adam', loss = 'mean_squared_error')
regressor.fit(X, y, epochs = 15, batch_size = 32)
```

```
Epoch 1/15
63/63 [==============================] - 0s 7ms/step - loss: 0.0717
Epoch 2/15
63/63 [==============================] - 0s 7ms/step - loss: 0.0060
Epoch 3/15
63/63 [==============================] - 0s 7ms/step - loss: 0.0034
Epoch 4/15
63/63 [==============================] - 0s 7ms/step - loss: 0.0025
Epoch 5/15
63/63 [==============================] - 0s 7ms/step - loss: 0.0024
Epoch 6/15
63/63 [==============================] - 0s 7ms/step - loss: 0.0021
Epoch 7/15
63/63 [==============================] - 0s 7ms/step - loss: 0.0019
Epoch 8/15
63/63 [==============================] - 0s 7ms/step - loss: 0.0021
Epoch 9/15
63/63 [==============================] - 0s 7ms/step - loss: 0.0019
Epoch 10/15
63/63 [==============================] - 0s 7ms/step - loss: 0.0017
Epoch 11/15
63/63 [==============================] - 0s 7ms/step - loss: 0.0017
Epoch 12/15
63/63 [==============================] - 0s 7ms/step - loss: 0.0017
Epoch 13/15
63/63 [==============================] - 0s 7ms/step - loss: 0.0016
Epoch 14/15
63/63 [==============================] - 0s 7ms/step - loss: 0.0016
Epoch 15/15
63/63 [==============================] - 0s 7ms/step - loss: 0.0016
<tensorflow.python.keras.callbacks.History at 0x7fd0b3d08ba8>
```

```
# define input sequence
raw_seq = test_set_scaled

# choose a number of time steps
time_steps = 2

# split into samples
X_test, y_test = split_sequence(raw_seq, time_steps)

# summarize the data
for i in range(len(X_test)):
    print(X_test[i], y_test[i])
```

```
 [0.58387045]] [0.55757917]
[[0.58387045]
 [0.55757917]] [0.59871218]
[[0.55757917]
 [0.59871218]] [0.60804135]
[[0.59871218]
 [0.60804135]] [0.61016157]
[[0.60804135]
 [0.61016157]] [0.60507301]
[[0.61016157]
 [0.60507301]] [0.62669954]
[[0.60507301]
 [0.62669954]] [0.58387045]
[[0.62669954]
 [0.58387045]] [0.61737036]
[[0.58387045]
 [0.61737036]] [0.57878172]
```

```
[48] #Making predictions on the test data
     predicted_stock_price = regressor.predict(X_test)
     predicted_stock_price = sc.inverse_transform(predicted_stock_price)
```

```
[49] predicted_stock_price.shape
```

```
(686, 1)
```

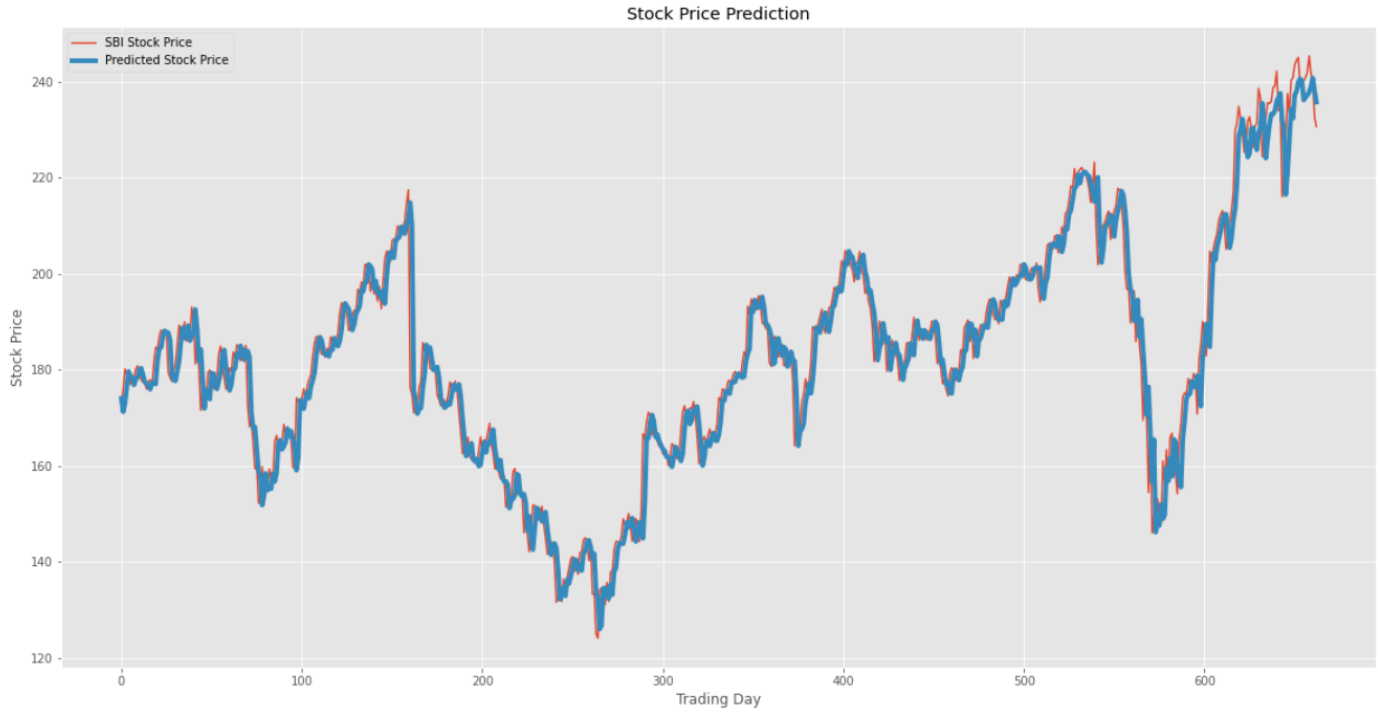
```
test_set.shape
```

```
(688, 1)
```

```
#Visualizing the prediction
plt.figure(figsize=(20,10))

plt.plot(sc.inverse_transform(y_test), label = 'SBI Stock Price')
plt.plot(predicted_stock_price, label = 'Predicted Stock Price',linewidth=4)
plt.title('Stock Price Prediction')
plt.xlabel('Trading Day')
plt.ylabel('Stock Price')
plt.legend()
plt.show()

rms_lstm = sqrt(mean_squared_error(sc.inverse_transform(y_test), predicted_stock_price))
print("RMSE=",rms_lstm)
mae_lstm = sqrt(mean_absolute_error(sc.inverse_transform(y_test), predicted_stock_price))
print("MAE=",mae_lstm)
```




```
RMSE= 5.244174820261117
MAE= 1.9239566676767033
```