Stock Prediction using ML Algorithms

*Note: Comparision of various machine learning algorithms for stock prediction

1st Akshata Deo
Department of Computer Engineering
San Jose State University
San Jose, USA
akshata.deo@sjsu.edu

2nd Charlie Brayton
Department of Computer Engineering
San Jose State University
San Jose, USA
charles.brayton@sjsu.edu

3rd Dikshita Borkakati
Department of Computer Engineering
San Jose State University
San Jose, USA
dikshita.borkakati@sjsu.edu

4th Priyanshi Jajoo
Department of Software Engineering
San Jose State University
San Jose, USA
priyanshi.jajoo@sjsu.edu

***Abstract*—AIML techniques are being used with data mining to solve a plethora of real-world problems, and the prediction of the stock market is one of them. The stock market is one of the most complicated and lucrative businesses. Many other businesses depend on the stock market to generate revenue. Today, not only traders but Computer Engineers are also highly interested in predicting stock price. This can be done in two ways. First, by using previous stock data, and second, by analysing social media behaviour. However, this paper proposes to use various machine learning algorithms like Naive Bayes, ARIMA, neural network, and SVR to predict future stock prices using previous stock data. These ML algorithms use different open-source libraries and preexisting algorithms to make this highly unpredictable stock market, a little more predictable. In comparison to the traditional approach, this technique has proven to be more accurate, and it saves time and money also. The implementation of these algorithms shows some good results at the end. Further, this paper suggests some measures that can be taken to improve the accuracy of the models.**

***Index Terms*—stock market, share market, ML algorithms, data mining, Naive Bayes, ARIMA, neural network, SVR.**

I. Introduction

The stock market or financial market is a system where people buy or sell stocks. A stock is commonly known as shares and it represents ownership claims on a business by a particular person. Predicting stock trends is a cumbersome task, as the stock market is very dynamic and composed of various factors. The main factors involved are Physical, Psychological, Rational, Irrational behavior, etc. These factors can be grouped into fundamental factors, technical factors and market sentiments like [1]:

- Supply and demand. For example, if traders tend to buy a particular stock more than selling it, this will affect stock price.
- Stock prices can show unexpected outcomes because of an announcement which keeps a stock artificially high or low.
- Global economical trends also affect stock prices. The more liquidity in the market the more it will be dynamic.
- Global disasters, pandemic, and any emergency conditions affect stock prices. For an example, in recent days, pandemic COVID-19 is severely affecting stock market.
- Stock historical prices. Each stock value varies in between a particular range of data and this allows investors to understand the pattern of the stock price.
- Inflation and deflation are other technical factors that affect the global economy and indirectly affect stock prices.

Stock prediction needs to be accurate and efficient, in order to help people invest in the stock market. This paper employs different Machine Learning Algorithms on Stock Market data to analyze the trends of the stock market. The stock market prediction performs better when we consider it as a regression problem as well as a classification problem. Considering these facts, our goal is to design a model that gains insight from the market information utilizing machine learning strategies, then predict the future patterns of stock value. This paper discusses Support Vector Machines analysis [2] models which are linear, polynomial, and radial basic function; Tensor Flow that is a Neural Network computation framework [3]; Auto ARIMA (Auto-regressive integrated moving average) model [4] and Naive-Bayes clustering model [5], in which both Gaussian and Bernoulli versions are used. The rest of the sections of this paper will provide a detailed description of data-sets, methods, and results.

II. DATASET OVERVIEW

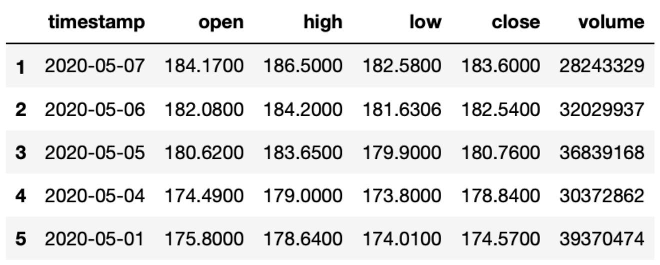
The data-set is downloaded from Alpha Vantage for Microsoft and Adobe Stocks for the last 20 years. All the required libraries like request, JSON, CSV, etc have been imported for fetching the data from Alpha Vantage.The historical stock data is fetched into a csv file using an API key and passing parameters such as the ticker symbol for the company stock, 'TIMESERIESDAILY' for the function parameter and 'csv' for the datatype parameter. The csv data is then converted into dataframe using pandas library. The dataframe has six features: timestamp, open, high, low, close and volume. Fig 1 shows the code snippet and the raw data after fetching.

Fig. 1. Dataset from Alpha Vantage

III. METHODS

A. Neural Network

a) Importing Dataset: Fig2 shows the raw microsoft dataset after fetching from the Alpha Vantage.



	timestamp	open	high	low	close	volume
1	2020-05-07	184.1700	186.5000	182.5800	183.6000	28243329
2	2020-05-06	182.0800	184.2000	181.6306	182.5400	32029937
3	2020-05-05	180.6200	183.6500	179.9000	180.7600	36839168
4	2020-05-04	174.4900	179.0000	173.8000	178.8400	30372862
5	2020-05-01	175.8000	178.6400	174.0100	174.5700	39370474

Fig. 2. Raw Dataset

b) Data Preprocessing: Dataset has been checked for duplicate, null and missing values. It gave false output, which means the dataset was already clean. One extra column has been added in the dataset with the average of high and low price of the stock. 'timestamp' column has been deleted, since it does not add any value while predicting stock price. Besides, 'timestamp' has dtype as a string, which might create errors while training the model, so it is better to drop this column. Now the data frame has been converted to a NumPy array to divide the whole dataset into training and testing dataset. The dataset has been divided sequentially like the first 80 percent of the data has been reserved for training purposes, and the rest of the 20 percent data has been reserved for testing purposes. Then, data scaling has been performed on the input and output data by importing the MinMaxScaler function from sklearn.preprocessing library. Data scaling should not be performed before categorizing training and testing dataset. Sequence should be like categorization of the data, then scaling of training data and then scaling of testing data. Doing so helps model to avoid over fitting.

c) Data Visualization: Fig3 shows the visualization of the microsoft dataset after preprocessing and before implemention of the model.

d) Training of Model using Tensorflow: First, TensorFlow has been installed, and all the required libraries have been imported. Now the TensorFlow session starts. Here, X, input data is a 2D matrix, and Y, output data is a 1D vector. At this point, Neural Network is not sure about the number of observations it will take. Therefore, it will consider 'None' value for unsure variables. Initialization of the variables is required before the training of the model starts. Tensorflow

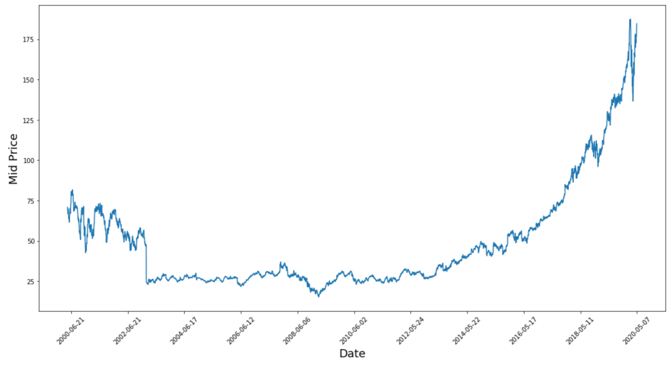


Fig. 3. Visualization of the Dataset

uses variance scaling initializer & zeros initializer methods to initialize the weight and bias variables, respectively. In the feedforward network, Neural Network has one input, four hidden, and one output layer. The input layer consists of five neurons, first hidden layer consists of 16 neurons, second hidden layer consists of 8 neurons, third hidden layer consists of 4 neurons, fourth hidden layer consists of 2 neurons, and finally output layer consists of 1 neuron. In each subsequent hidden layer number of neurons, get half of the previous layer to compress the information. Then Matrix Multiplication is performed between different layers of the neural network model by using the matmul method. Hidden layers are then transformed by using the relu activation function, which also introduces non-linearity to the model. Transpose of the output layer will be taken. MSE (mean squared error) function has been used as a cost function to measure the deviation between the predicted and the actual value. AdamOptimizer function, which is the combination of two other popular optimizers AdaGrad and RMSProp, has been used as an optimizer function to choose weight and bias variables during training to minimize the cost function. Finally, model training starts by running the neural network session.

e) Architecture: Fig4 shows the architecture of the feedforward neural network adopted to train the model. In this, the flow of data is unidirectional that is from left to right. The three main components of the architecture are the input layer, hidden layer, and output layer.

f) Interactive Plot: Fig5 shows the interactive plot

g) Results: To fit the neural network, 10 number of epochs and 50 batch size has been considered. With this measure, the neural network is giving final MSE for training data as 0.00010041545 and MSE for testing data as 0.00012404985

B. Time Series Analysis using ARIMA

Time series in simple terms means data collected over time and is dependent on it. Forecasting stock prices is a very difficult and challenging task in the financial market because the trends of stock prices are non-linear and non-stationary time-series data. ARIMA is a class of models that 'explains' a given time series based on its own past values, that is, its own lags and the lagged forecast errors, so that equation can be

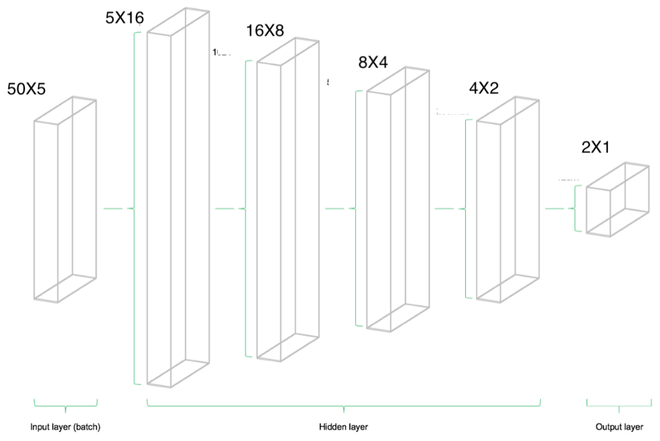


Fig. 4. Feedforward Neural Network Architecture

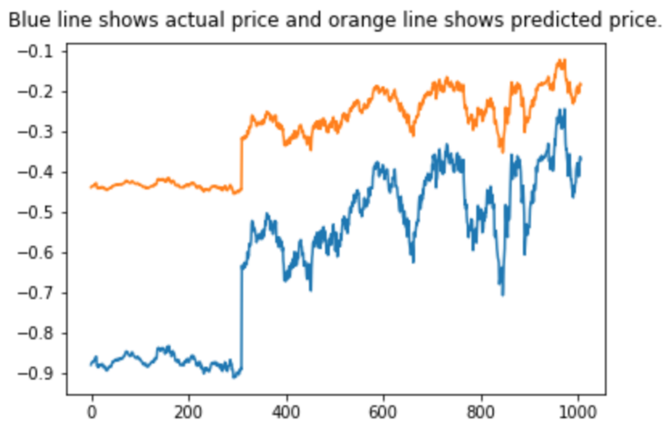


Fig. 5. Interactive Plot

used to forecast future values. The parameters of the ARIMA model are defined as

- p: the number of lag observations included in the model, also called the lag order.
- d: The number of times that the raw observations are differenced, also called the degree of differencing
- q: The size of the moving average window, also called the order of moving average.

a) Data Visualization: Time series data is composed of a combination of Trend, Cycles, Seasonality and Residuals. Statsmodels library has a function which decomposes time series into Trend, Seasonality and Residuals. Fig 6 shows the seasonal decomposition of Microsoft stocks from 2010 to 2020. There is an upward trend to the data but seasonal effect in insignificant. The Residuals component is left after other components have been calculated and removed from time series data. It is randomly, identically and independently distributed.

b) Make Data Stationary: When we have trend and/or seasonality in a time series data we call it non-stationary. Stationarity means the statistical properties of data, such as mean, variance and standard deviation remain constant over time.

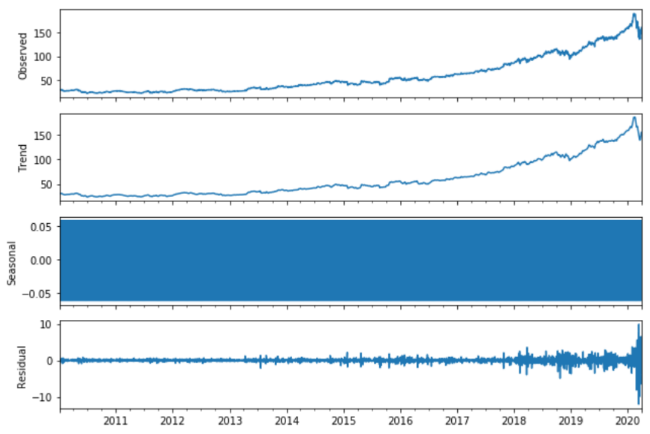


Fig. 6. Seasonal Decomposition of Microsoft Closing Price

There are two ways we can check the stationarity of a time series. By visualizing the data, it should be easy to identify a changing mean or variation in the data. For a more accurate assessment there is the Augmented Dickey-Fuller test, which provides test statistic and P value. P value greater than or equal to 0.05 means the data is not stationary, otherwise, we reject the null hypothesis and say data is stationary. Rolling mean and standard deviation is plotted for the data to check stationarity. From fig 7, we can see that the data is not stationary.

To make the data stationary, we take a first difference of the data and the number of times the differencing was performed to make the series stationary is the value of the 'd' parameter for the model. Fig 8 shows the stationary data after differencing was performed.

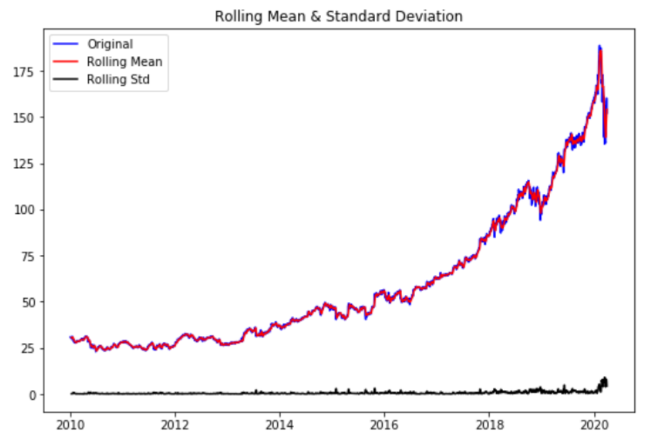


Fig. 7. Non Stationary Time Series

c) Plot ACF and PACF and Find the Optimal Parameters: The next step is to determine the tuning parameters of the model by looking at the auto-correlation and partial auto-correlation graphs. The big issue as with all models is that we do not want to over-fit our model to the data by using too many terms. Fig 9 shows the plot of ACF and PACF. The

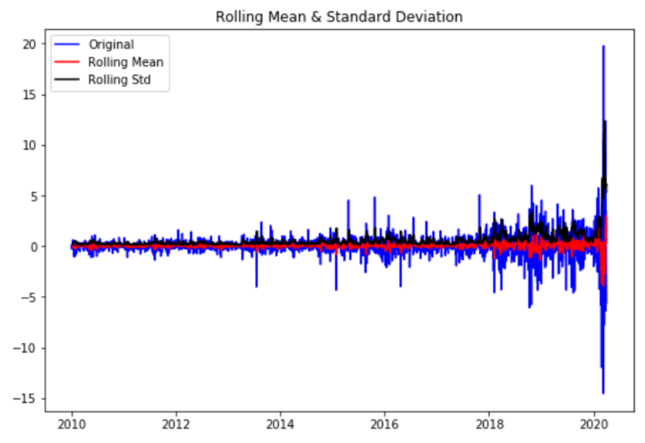


Fig. 8. Stationary Time Series

grey lines represent the confidence level. If we look at Fig 9 carefully, we will see that the upper confidence level line has been crossed at lag1. Therefore, the order of AR would be 1 and p=1. Similarly, in Fig 9 for PACF, the upper confidence level line has been crossed at lag1. Therefore, the order of MA would be 1 and q=1.

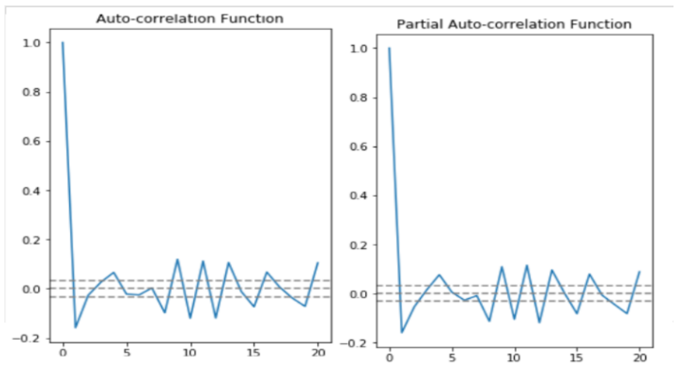


Fig. 9. ACF and PACF

d) Build Model and Predict: After determining the parameters p,d and q, now we are ready to build the model. Split the data-set for training and testing. Each time step of the test set is iterated. A prediction is made each iteration and stored in a list. This is so that at the end of the test set, all predictions can be compared to the list of expected values and an error score calculated. In this case, a mean squared error score is calculated and returned. Fig 10 shows the predicted price vs the actual price.

e) Model Tuning: A suite of parameters p,d and q are evaluated to find the best fit for the model. A grid of p,d and q parameters are specified to iterate. A model is created for each parameter and its performance evaluated by keeping track of the lowest error score observed and the configuration that caused it. Grid search for parameters were performed for Microsoft stocks where the parameters p,d and q were specified values in the range (0,2) and MSE calculated. After

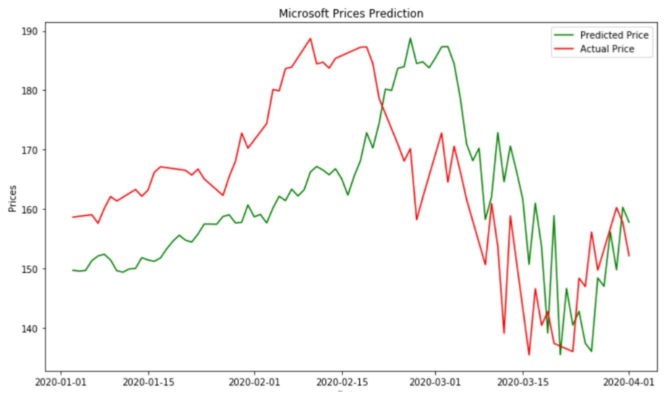
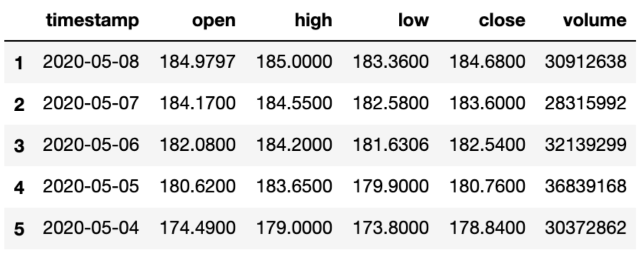


Fig. 10. Prediction vs Actual Price

performing grid search, the best best parameters for our model was (0,1,1) with MSE 3.694.

C. SVR: Support Vector Regression

For this method we used Microsoft's stock data and Fig11 shows Microsoft's data fetched from Alpha Vantage API till May 8th, 2020. In this project we analyse stock data of



	timestamp	open	high	low	close	volume
1	2020-05-08	184.9797	185.0000	183.3600	184.6800	30912638
2	2020-05-07	184.1700	184.5500	182.5800	183.6000	28315992
3	2020-05-06	182.0800	184.2000	181.6306	182.5400	32139299
4	2020-05-05	180.6200	183.6500	179.9000	180.7600	36839168
5	2020-05-04	174.4900	179.0000	173.8000	178.8400	30372862

Fig. 11. Raw Dataset for Microsoft stocks (May 8th, 2020)

a particular company using Support Vector Regression with SK learn library. For this all the required dependencies are imported.

a) Data Pre-processing: We employed same steps as we did earlier in method A (Neural Networks framework) for data pre-processing. We checked for duplicate values, data types of the columns, and for NaN or null values. Here, we also updated the data types of the columns from the object to float and timestamp as per the requirement.

b) Data Visualization: After data pre-processing (similar as mentioned in Neural Networks section), curve between timestamp and close value is plotted and it is shown in the Fig12. Since we were doing a time series prediction, so we sorted our data by date. We got our adjusted closing prices from our dataframe and we plot a rolling mean on our data. We also visualized the training stock data and plotted curves shown in the Fig13 and Fig 14.

c) SVR Algorithm: Support Vector Regression (SVR) uses the same principle as Support Vector Machine, but for regression problems. The problem of regression is to find a function that approximates mapping from an input domain to real numbers on the basis of a training sample. Support Vector

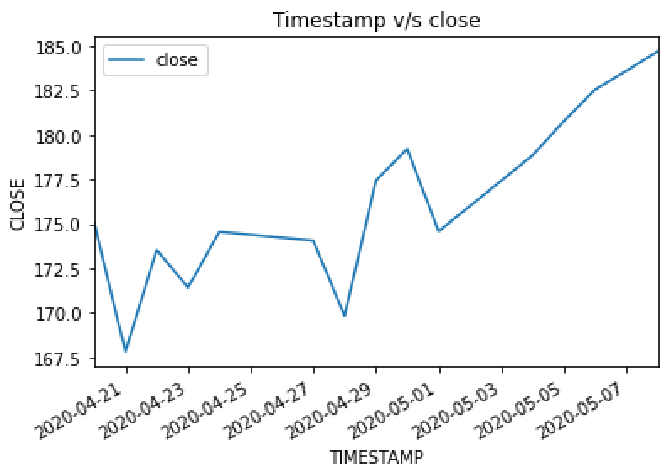


Fig. 12. Timestamp v/s Close Value curve

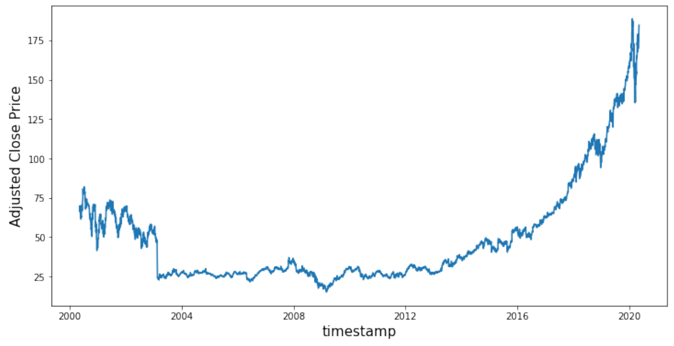


Fig. 13. Timestamp v/s Close Value curve

Regression (SVR) is the combination of a Support Vector Machines and Regression. With stock data, we are predicting the next value in a series. Using regression we try to minimize the cost function using something like gradient descent.

d) Model Implementation: We employed three models on default SVR (Support Vector Regression) kernel,

- Linear Regression. Linear Regression is a way to find the best linear relationship between two given variables. The goal of linear regression is to find the line of best fit for our data that will result in the predicted values to be as close to our actual values.

$$Y' = A + B * Xeq \tag{1}$$

  where, X is Predictor (Present in data); B is coefficient (Estimated by regression); A is intercept (estimated by regression); Y' is predicted value.
- Radial Basic Function. RBF stands for radial basic function. RBF transferred our 2D space into a higher dimension to help better fit our data. The function takes the squared euclidean distance between 2 samples and divides by some sigma value.
- Polynomial Regression. Polynomial Regression is a form of linear regression in which the relationship between the independent variable x and dependent variable y is modeled as an nth degree polynomial.

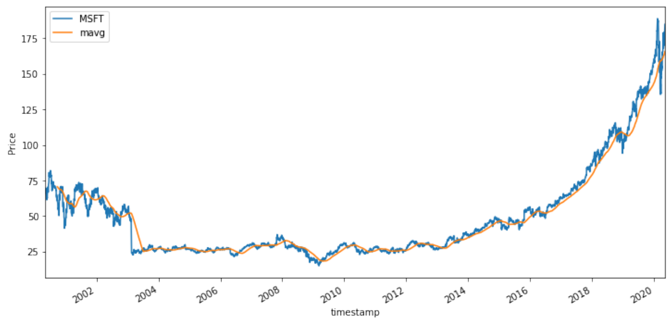


Fig. 14. Timestamp v/s Close Value curve

e) Results: After applying SVR models on the stock data of the prior 15 days, we find out the most accurate regression values. The Fig15 shows the Support Vector regression curve for the three different models. The results show accurate

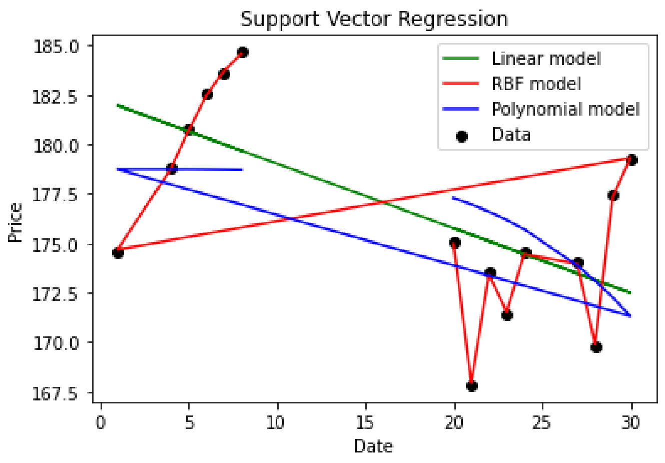
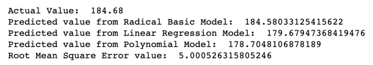


Fig. 15. Support Vector Regression

prediction of the stock price. Here, we observed, though linear regression and polynomial model show good results, the Radical Basic function model provided the best results. The Fig16 represents a snapshot of the output of the support vector regression models.



```
Actual Value:  184.68
Predicted value from Radical Basic Model:  184.58033125415622
Predicted value from Linear Regression Model:  179.67947368419476
Predicted value from Polynomial Model:  178.7048106878189
Root Mean Square Error value:  5.000526315805246
```

Fig. 16. Predicted values from SVR Models

D. Naive Bayes

a) Data Preprocessing: The initial data was processed to generate a daily change in the market, given by close-open. This daily change value was used for all future calculations. The market data for analysis with Naive Bayes was modified to associate each day's change in value with the previous n days change. This sliding window provides a consistent scope for each days' prediction, and the variable size allows for the model's scope to be adjusted.

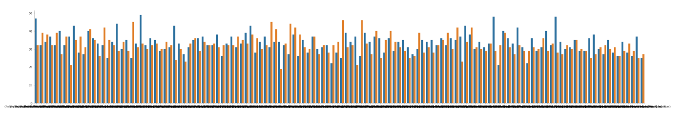


Fig. 17. 7 Day Combinations

b) Data Visualization: Data visualization (Fig 17) was also performed on this windowed data set to visualize how the windows could affect the predictions. This visualization was done with a window size of 7 and shows all 256 combinations of those 7 binary attributes. Each bar represents the number of occurrences of that combination that resulted in an increase (blue) or decrease (orange). By observing these bars in pairs, the influence of the past 7 days on the current day can be estimated. For example a tall blue bar and a short orange bar indicates that that particular sequence of days is generally followed by an uptick in stock value.

c) Training: The training of the model was done by splitting the windowed dataset into testing and training data based on a 50/50 split. The training data was then input in sklearn's naive bayes method, using both the Gaussian and Bernoulli options. For the Bernoulli option, the data is binarized into either a gain or loss for the day rather than the actual numeric change.

d) Algorithm: The Naive-Bayes algorithm is based off of Bayes Theorem:

$$P(C \mid X) = \frac{P(X \mid C)\, P(C)}{P(X)}$$

, where the C is the class assignments and X is the data observations. The denominator remains constant across all classes, so the numerator is the decider between which class is predicted. Using the naive assumption that all the attributes of X are conditionally independent, the attributes can be computed individually and multiplied together, so the numerator becomes:

$$= p(C_k) \prod_{i=1}^{n} p(x_i \mid C_k). \tag{2}$$

Training the model is done by determining the probabilities associated with each attribute and class. Once these relationships and values are determined, the class of new data is determined by simply picking the class with the highest probability given the associated attributes.

e) Results: The naive-Bayes method failed to successfully or accurately predict, whether a day would result in an increase or decrease in stock value based on the last 7, 30, or 365 days. For all of these window sizes the naive-Bayes algorithm correctly predicted about half of the time. Given that there are only two options to pick from, the naive-Bayes algorithm did as well as a random guess.

IV. Challenges Faced

Considering the inherent complexity of stock markets , and large number of factors that can affect its behaviour, it is not an obvious or routine data mining task to come up with a model that performs consistently well. The outcome is completely based on numbers and assumes a lot of axioms that may or may not follow in the real world so as the time of prediction. Employing traditional methods like fundamental and technical analysis may not ensure the reliability of the prediction. To make predictions regression analysis is used mostly. The team decided to focus on exploring the algorithms well suited for time series analysis and dive deeper on understanding how these algorithms/libraries work.

V. Future work

This project uses the dataset with six parameters: timestamp, open, high, low, close, and volume. We are predicting the price at time 't+1' using the data till the time 't.' However, this much consideration is not enough to create an efficient model. The prediction is completely depends on these six parameters only and it may not follow in the real world with that accuracy. To build a model that can forecast the future stock price, we need to consider sentiment analysis and various other aspects. Sentiment analysis of tweets using twitter API can be done as an enhancement of this project. Such a prediction would greatly help an investor in making informed decisions that would directly contribute to his profits.

VI. Conclusion

Overall the stock predictions performed surprisingly well. The Neural Network prediction especially performed incredibly well with a MSE of 0.00012404985. Given this level of accuracy, future extensions of the project could test this model with stocks from other companies to help further validate the model and ensure it isn't over fitting. The Support Vector Regression model also performed exceedingly well, predicting the stock price to within a dollar based off of the prior 15 days. Based off of these models and the ARIMA model, which also produced quality results with a MSE of 3.694, we were able to successfully and accurately predict stock values based on prior days stock events. The success of these models helps explain the success and rise of algo-trading, which is currently used to make numerous small profitable trades in the stock market. The primary difference between our models and algo-trader's is that our models used daily data, while algo-traders use a live stream of current data.

VII. Acknowledgement

We would like to thank Professor Carlos Rojas for his guidance and support in successfully executing this project. We would also like to thank our team members who worked long hours and on weekends to make this happen.

References

[1] M. Moukalled, W. El-Hajj, M. Jaber, ”Automated Stock Price Prediction Using Machine Learning”, Computer Science Department, American University of Beirut. Available [Online]: https://www.aclweb.org/anthology/W19-6403.pdf.

[2] K. Hiba Sadia, A. Sharma, A. Paul, SarmisthaPadhi, S. Sanyal., ”Stock Market Prediction Using Machine Learning Algorithms,” *International Journal of Engineering and Advanced Technology (IJEAT)*, Vol. 8 Sno. 4, pp. 25-31, April 2019. Available [Online]: https://www.ijeat.org/wp-content/uploads/papers/v8i4/D6321048419.pdf

[3] M. A. Hossain, R. Karim, R. Thulasiram, N. D. B. Bruce, Y. Wang, ”Hybrid Deep Learning Model for Stock Price Prediction,” *IEEE Symposium Series on Computational Intelligence (SSCI)*, Bangalore, India, pp. 1837-1844, January 2018. Available: 10.1109/SSCI.2018.8628641

[4] A. A. Ariyo, A. O. Adewumi, C. K. Ayo, ”Stock Price Prediction Using the ARIMA Model,” *UKSim-AMSS 16th International Conference on Computer Modelling and Simulation*, Cambridge, pp. 106-112, February 2015. Available [Online]: https://ieeexplore.ieee.org/document/7046047

[5] L. Ryll, S. Seidens. ”Evaluating the Performance of Machine Learning Algorithms in Financial Market Forecasting: A Comprehensive Survey,” July 2019. Available [Online]: https://arxiv.org/pdf/1906.07786.pdf