Stock Price Prediction Using Data Analytics

Shashank Tiwari
Dept. Of Electronics,
K J Somaiya College of Engineering
Mumbai, India
shashank.tiwari@somaiya.edu

Akshay Bharadwaj
Dept. Of Electronics,
K J Somaiya College of Engineering
Mumbai, India
akshay.bharadwaj@somaiya.edu

Dr. Sudha Gupta
Associate Professor,
K J Somaiya College of Engineering
Mumbai, India
sudhagupta@somaiya.edu

Abstract— **Accurate financial prediction is of great interest for investors. This paper proposes use of Data analytics to be used in assist with investors for making right financial prediction so that right decision on investment can be taken by Investors. Two platforms are used for operation: Python and R. various techniques like Arima, Holt winters, Neural networks (Feed forward and Multi-layer perceptron), linear regression and time series are implemented to forecast the opening index price performance in R. While in python Multi-layer perceptron and support vector regression are implemented for forecasting Nifty 50 stock price and also sentiment analysis of the stock was done using recent tweets on Twitter. Nifty 50 (^NSEI) stock indices is considered as a data input for methods which are implemented. 9 years of data is used. The accuracy was calculated using 2-3 years of forecast results of R and 2 months of forecast results of Python after comparing with the actual price of the stocks. Mean squared error and other error parameters for every prediction system were calculated and it is found that feed forward network only produces 1.81598342% error when opening price of stock is forecasted using it.**

Keywords—Big Data Analytics, Data analytics, Predictive Analytics, Stock Index Prediction, Time Series Model, Moving Average, Auto Regression, Linear Regression, Artificial Neural Networks, ARIMA, Holt-Winters, Multi-Layer Perceptron, Radial Basis Function(RBF), Twitter sentiment analysis, Web Scrapping, Support Vector Regression

I. INTRODUCTION

Recently forecasting stock market indices is gaining more attention, because of the fact that if the direction of the market is successfully predicted the investors will be better guided. Here, an exhaustive study on various mathematical models and algorithms that can be used in stock index prediction is presented.

Author of Stock Price Prediction and Trend Prediction Using Neural Networks [17] performed stock price prediction using Neural networks and used 500 nodes in his work. He stated that accuracy of the forecast can be increased if the number of nodes are increased. So we increased the nodes for more accuracy. Authors of Stock Price Prediction Using the ARIMA Model [18] implemented Arima with different (p,q,r) values and calculated R squared error for every respective value. So to make process less time consuming we used Auto Arima available in R. Auto arima automatically tests data with all p, q and r options and chooses the best value that can provide best forecast.

We have used various models including Time Series Model, Artificial Neural Networks (ANNs), Regression, etc. For modelling the system we have used the stock data of "Nifty 50" for the last 9 years. This stock data is then used as an input to our various mathematical models' algorithms like Autoregressive Integrated Moving Average model (ARIMA), Polynomial model, and Linear Regression method, Time Series Model and Radial Basis Function Neural Network and Multi-Layer Perceptron Neural Network, of the Artificial Neural Network (ANN) models. The results of the same are then compared and the most accurate and efficient model is found. Another motivation for research in this field is that it possesses many theoretical and experimental challenges.

In rest of the paper Section II illustrates proposed system which tells briefly about what we want to do and then Section III has implementation of proposed system which gives idea about everything that is implemented in this project and then in Section IV we have attached the results that we got and then we can concluded according to our results in Section V.

II. PROPOSED SYSTEM

The whole purpose of this project is to assist the stock market traders in making profit. This project helps traders in "Buy low, Sell high" [1] strategy. The practice of buying a security when its price is (or is perceived to be) low and selling it when its price is high. The ability to buy low and sell high requires one to be able to determine roughly when the low and high prices for a security occur. There are a number of technical indicators analysts use to find these, but critics of the practice contend it is impossible or at least excessively risky [2]. This project helps avoiding that risky part by forecasting the future prices.

Architecture of artificial neural networks: We used Multilayer perceptron and Feed forward network. Both are having sequential model(Linear stack of layers). Multi-layer perceptron was implemented using Keras library in python [3]. Numbers of Hidden layers we have used is 8. The activation function we used is reLU (rectified linear units) [4]. One of the advantages of reLU activation function is its ability to increase the training speed of neural network. "mean_squared_error" is used as a Model loss function. The loss function, also called the objective function is the evaluation of the model used by the optimizer to navigate the weight space. Adam (Adaptive Moment Estimation) optimizer [5] is used. Adam's biggest advantage is it uses adaptive learning rates and is well suited for large amount of data. Number of epochs chosen after taking execution time and

978-1-5386-3852-1/17/$31.00 ©2017 IEEE

forecasting accuracy in consideration is 200. Feed-forward neural network used in R uses a single hidden layer and lagged inputs.

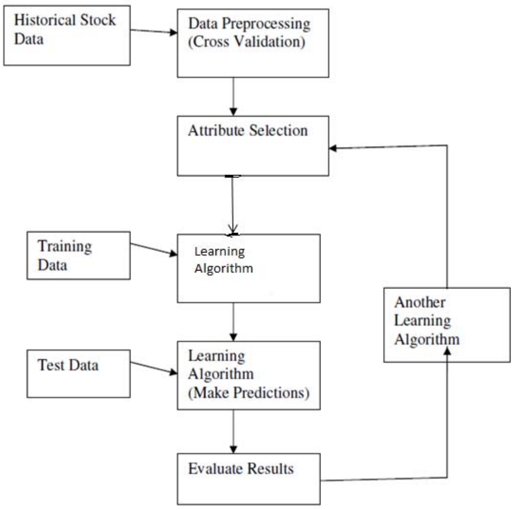

Historical Stock Data
Data Preprocessing (Cross Validation)
Attribute Selection
Training Data
Learning Algorithm
Another Learning Algorithm
Test Data
Learning Algorithm (Make Predictions)
Evaluate Results

Fig. 1 Basic Flowchart for Stock Prediction

III. Implementation of Proposed Methodology

The data used for forecasting can be downloaded from internet [6]. Web scraping [7] is implemented using python and R both. Data that was extracted ranged from date 1-Jan-2007 to 17-Sep-2016 (9 years). Before data was used in the various methods, pre-processing of data was considered. The days which had no trading or no data available, they needed to be processed. There are various options available to process this type of condition like:

1. Ignore the days with no trading and choose the days with trading.
2. Assign 0 to the days with zero trading.
3. Build a model that can approximate the value for no trading.

A model was not created to determine the value for the days with no trading because it was feared that the values calculated may contribute significantly to the final error of the networks. Initial experiments were conducted, utilizing techniques 1 and 2 above, and it was found that, technique 1 resulted in lower error values. As a result, technique 1, from the above list, has been employed.

The data which we have consist of OHLC (open, high, low and close) and volume. Out of this, we used open price to forecast open price or close price while forecasting close price. The data is divided into two parts train set and test set. Dividing data also helped in avoiding over and under fitting. Data is divided into time order. The data of the earlier period is used as train set and later period data is used as train set. Training set is used to train the model and then test set is used to calculate the accuracy of the model.

The various models which we have implemented are

1. Holt-winters filtering [8]: Computes Holt-Winters Filtering of a given time series. Unknown parameters are determined by minimizing the squared prediction error. Here we used two variants of Holt-winters, one with seasonal data and other without seasonal data [9] (*implemented in R).

2. Neural Networks: Here a single hidden layer neural network with type feed-forward is used to predict the future prices[10] (*implemented in R).

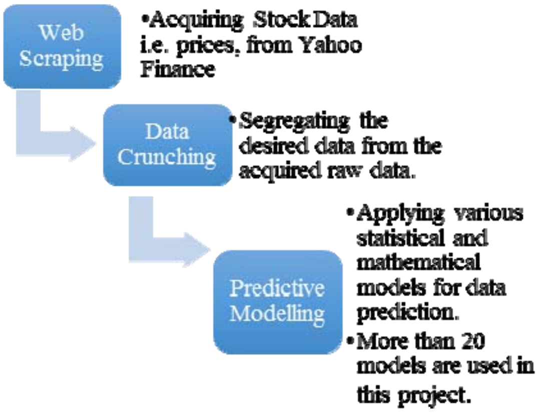

Web Scraping
•Acquiring Stock Data i.e. prices, from Yahoo Finance
Data Crunching
•Segregating the desired data from the acquired raw data.
Predictive Modelling
•Applying various statistical and mathematical models for data prediction.
•More than 20 models are used in this project.

Fig. 2 General flow in prediction analytics

3. ARIMA model: This approach uses an Autoregressive Integrated Moving Average model as a generalization of the ARMA model which are used mainly for non-stationarity function predictions. Two variants are used, one determining the parameter automatically and the other with a fixed parameter approach [11] (*implemented in R)

The mathematical formula for ARIMA can be given as,

$$\hat{Y}t = \mu + \phi 1\ yt\text{-}1 + \ldots + \phi p\ yt\text{-}p - \theta 1et\text{-}1 - \ldots - \theta qet\text{-}q \qquad (1)$$

4. Linear Model: This TSLM (Time Series Modelling) uses a linear model as a base and adds trend and season components to the computation [12] (*implemented in R).

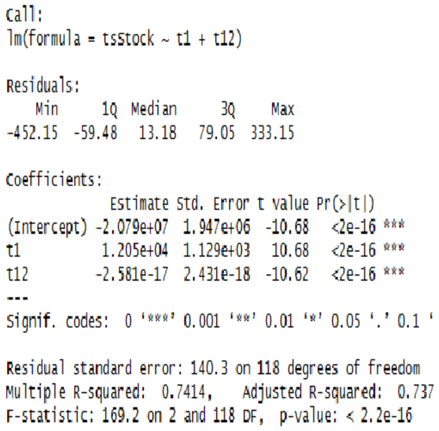
5. STL Model: The Seasonal Trend Loess model [13] (*implemented in R).

6. Support Vector Machines: In Support Vector Machines (SVMs) we used Radial Basis Function (Gaussian) kernel, a.k.a. RBF kernel, as a learning algorithm for data classification. For efficiency purposes, 33% of the data is used for testing the network and rest of the data is used for training the network [14] (*implemented in Python).

7. Multi-Layer Perceptron Model (MLPs):Multi-Layer Perceptron are one of the simplest and most efficient neural networks that can be used for prediction purposes. It is a feed-forward network that learns through back-propagation technique, minimizing errors in every iteration until the desired value is achieved [15] (*implemented in Python).

8. Twitter sentiment analysis: Here we downloaded the recent 100 tweets related to NIFTY and processed there sentiment. Sentiment gives a very good indication about how people are thinking about a certain stock whether they are positive or negative about the stock.

We begin by creating a polynomial trend [16] of the Stock's price. The summary of our trend is shown in fig 3, and the polynomial trend is shown in the fig 4.



```
Call:
lm(formula = tsStock ~ t1 + t12)

Residuals:
    Min      1Q  Median      3Q     Max
-452.15  -59.48   13.18   79.05  333.15

Coefficients:
              Estimate Std. Error t value Pr(>|t|)
(Intercept) -2.079e+07  1.947e+06  -10.68   <2e-16 ***
t1           1.205e+04  1.129e+03   10.68   <2e-16 ***
t12         -2.581e-17  2.431e-18  -10.62   <2e-16 ***
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 '

Residual standard error: 140.3 on 118 degrees of freedom
Multiple R-squared:  0.7414,    Adjusted R-squared:  0.737
F-statistic: 169.2 on 2 and 118 DF,  p-value: < 2.2e-16
```

Fig 3: Summary of polynomial trend

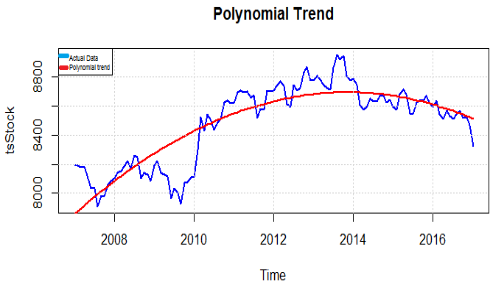


Fig 4: Polynomial Trend

Now that we have one generalized Trend, we create another function based on the loss window for seasonal extraction called STL (Seasonal, Trend and Loss) as shown in fig 5. The overview of both functions is shown in Fig 6.

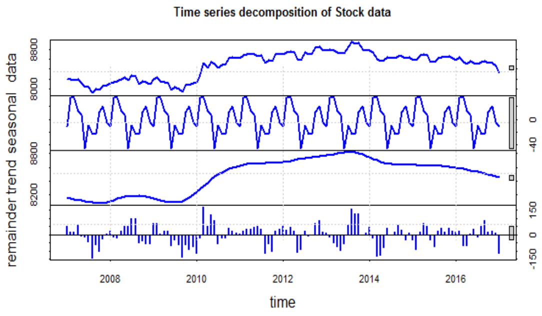


Fig 5: Time series decomposition of data

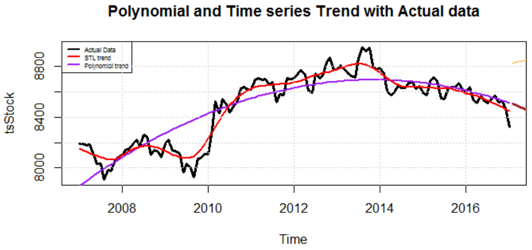


Fig 6: Polynomial and Time series trend with Actual data.

IV. Results

For accuracy we took data till 2014 and performed forecasting for the rest of the time for which we had data. After that we compared the result of each method (we got average value for each month as a result for forecasting) with the actual value of the stock and result was obtained. We predicted the open price of the stock (Nifty 50) and best method to forecast the opening price of stock is feed forward neural network. We also observed that for different stock different methods can provide better result and this is true for different types of prices (Open, low, high and low) too. All 21 methods are implemented on R platform and the results are given below in table 1, 2 and 3.

Table 1: Error Calculation of Results Obtained by Every Method Implemented in R Using polynomial trend (ME: Mean Error, RMSE: root-mean-square error, MAE: Mean absolute error, MPE: Mean Percent Error, MAPE: Mean Absolute Percent Error, ACF1: First order Auto-correlation coefficient)

Methods	**ME**	**RMSE**	**MAE**	**MPE**	**MAPE**	**ACF1**
Holt winters with seasonal data	-817.4501976	840.0280676	817.4501976	-9.472914127	9.472914127	0.827043581
Holt winters without seasonal data	624.8432068	659.1552717	624.8432068	7.227932052	7.227932052	0.697360254
Feed forward neural network	-925.2259633	957.1125759	925.2259633	-10.72245388	10.72245388	0.847569072
Auto Arima	-934.4091159	967.8160735	934.4091159	-10.82903774	10.82903774	0.849558232
Arima	-539.6683745	557.8539684	539.6683745	-6.25521347	6.25521347	0.794805538
Linear model	-977.592138	1007.563108	977.592138	-11.32847594	11.32847594	0.849544443

STL	-504.2921685	520.5117625	504.2921685	-5.845237243	5.845237243	0.768575376

Table 2: Error calculation of Results obtained by every method implemented in R using Time series trend

Methods	**ME**	**RMSE**	**MAE**	**MPE**	**MAPE**	**ACF1**
Holt winters with seasonal data	-171.2930454	180.6783156	171.2930454	-1.986892974	1.986892974	0.41398822
Holt winters without seasonal data	-351.5597615	356.5589133	351.5597615	-4.074267608	4.074267608	0.425959981
Feed forward neural network	-540.0584884	556.6065307	540.0584884	-6.259556605	6.259556605	0.774273406
Auto Arima	-497.7757022	511.5899434	497.7757022	-5.769440698	5.769440698	0.750163001
Arima	-546.2326753	564.4232407	546.2326753	-6.331376678	6.331376678	0.784807012
Linear model	-977.592138	1007.563108	977.592138	-11.32847594	11.32847594	0.849544443
STL	-977.592138	1007.563108	977.592138	-11.32847594	11.32847594	0.849544443

Table 3: Error calculation of Results obtained by every method implemented in R using Time series trend using actual data

Methods	**ME**	**RMSE**	**MAE**	**MPE**	**MAPE**	**ACF1**
Holt winters with seasonal data	-430.3869843	444.6796069	430.3869843	-4.989021661	4.989021661	0.736470136
Holt winters without seasonal data	-484.9258825	503.7336645	484.9258825	-5.621639239	5.621639239	0.703620828
Feed forward neural network	-156.525133	165.4955047	156.525133	-1.815983423	1.815983423	0.378886413
Auto Arima	-178.8611917	186.7089664	178.8611917	-2.074573194	2.074573194	0.369718377
Arima	-189.6564637	208.5686193	189.6564637	-2.200198333	2.200198333	0.296523295
Linear model	-559.4130349	581.4327851	559.4130349	-6.484772593	6.484772593	0.765990827
STL	-157.6955043	175.8374393	157.6955043	-1.83014208	1.83014208	0.42818805

In python, we predicted the Open price of the stock from 1/12/16 to 31/1/17 using Support vector machine, and from 05/12/2016 to 08/02/2017 using multi-layer perceptron, and the accuracy was calculated after comparing the predicted value with the actual value. The results are given in Table 4.

Table 4: Error calculation of Results obtained by methods implemented in Python

Method	**ME**	**RMSE**	**MAE**	**MPE**	**MAPE**
RBF	51.62	163.05	126.29	0.58767	1.518612
MLP	10.02	61.33	49.96	0.11516	0.6056538

V. Conclusion

This research entailed the development of various methods to find the most accurate model for prediction of prices of the stock. We increased the nodes of neural network as suggested by author of Stock Price Prediction and Trend Prediction Using Neural Networks and we got better results. From the above listed methods we have found that Feed Forward Neural network provides the highest accuracy for the opening price of stock. We also observed that different method can be efficient for different type of stocks and prices. The least amount of mean absolute percentage error that we got is1.81598342% for feed forward neural network using actual raw data as it is and the maximum error is 11.32847594% which is obtained using linear model with polynomial trend. The result obtained was the opening price of the stock and that too was average for a full month. So an improvement in this system can be achieved by forecasting the opening price of each day.

Acknowledgment

We would primarily like to thank our mentor Dr. Sudha Gupta, for guiding us throughout the entire project work, motivating us to face the encountered challenges and helping us at all stages to take the project to higher standards of refinement. Without her immense support and guidance, this project would not have been possible.We would like to thank "Mr. Rob Hyndman", as his work in statistics really inspired us to work with the techniques we used in our project.

References

[1] Investopedia. A Look At the Buy Low, Sell High StrategyAvailable from: http://www.investopedia.com/articles/investing/081415/look-buy-low-sell-high-strategy.asp. [cited 27th july 2017]

[2] Definition of "buy low, sell high". Available from: http://www.thefreedictionary.com. [cited 27th july 2017]

[3] Keras. Available: https://keras.io. [cited 27th july 2017]

[4] reLU.Available from: https://github.com/Kulbear/deep-learning-nano-foundation/wiki/ReLU-and-Softmax-Activation-Functions. [cited 27th july 2017]

[5] Adam optimizer. Available from: https://arxiv.org/abs/1412.6980. [cited 27th july 2017]

[6] Yahoo Finance. Available from: http://www.finance.yahoo.com. [cited 27th july 2017]

Google Finance. Available from:https://www.google.com/finance. [cited 27th july 2017]

[7] Web scraping. Available from: https://www.webharvy.com. [cited 27th july 2017]

[8] Holt winters filtering. Available from: https://www.otexts.org/fpp/7/5. [cited 27th july 2017]

[9] Holt winters filtering. Available from: http://stat.ethz.ch/R-manual/R-patched/library/stats/html/HoltWinters.html. [cited 27th july 2017]

[10] Neural network. Available from: http://www.inside-r.org/packages/cran/forecast/docs/nnetar. [cited 27th july 2017]

[11] Arima Model. Available from: http://stat.ethz.ch/R-manual/R-patched/library/stats/html/arima.html. [cited 27th july 2017]

[12] Tslm. Available from: http://www.inside-r.org/packages/cran/forecast/docs/tslm. [cited 27th july 2017]

[13] STL Model. Available from:http://stat.ethz.ch/R-manual/R-devel/library/stats/html/stl.html. [cited 27th july 2017]

[14] Support Vector Machine. Available from: http://scikit-learn.org/stable/modules/svm.html. [cited 27th july 2017]

[15] MLP. Available from: https://keras.io/models/sequential/. [cited 27th july 2017]

[16] Polynomial Trend. Available from: https://en.wikipedia.org/wiki/Polynomial_regression. [cited 27th july 2017]

[17] Pritam Radheshyam Charkha. "Stock Price Prediction and Trend Prediction Using Neural Networks", 2008 First International Conference on Emerging Trends in Engineering and Technology , 29/7/2008, Page no: 592-594.

[18] A. Adebiyi., Aderemi O. Adewumi and Charles K. Ayo. "Stock Price Prediction Using the ARIMA Model", 2014 UKSim-AMSS 16th International Conference on Computer Modelling and Simulation, 23/2/2015, Page no: 106-12G. Eason, B. Noble, and I.N. Sneddon, "On certain integrals of Lipschitz-Hankel type involving products of Bessel functions," Phil. Trans. Roy. Soc. London, vol. A247, pp. 529-551, April 1955.

[19] S. R. Gupta, F. S. Kazi, S. R. Wagh and N. M. Singh , "Probabilistic framework for evaluation of smart grid resilience of cascade failure", IEEE Innovative Smart Grid Technologies Conference(ISGT) Asia, Kuala Lumpur, Malaysia, 20-23 May 2014, pp 255-260

[20] Sudha Gupta, Ruta Kambli, Sushma Wagh, and Faruk Kazi, "Support Vector Machine based Proactive Cascade Prediction in Smart Grid using Probabilistic Framework" IEEE Transactions on Industrial Elctronics, vol. 62, issue-4, pp. 2478-2486, April 2015.