Revitalizing Stock Predictions with Machine Learning Algorithms – An Empirical Study

Vaibhav Gaur
Dept. Of Computer Science & Technology
Manav Rachna University
Haryana, India
vaibhav_gaur1819@manavrachna.net

Shubham Sood
Dept. Of Computer Science & Technology
Manav Rachna University
Haryana, India
shubham_sood1819@manavrachna.net

Lisha Uppal
Dept. Of Computer Science & Technology
Manav Rachna University
Haryana, India
lisha_uppal1819@manavrachna.net

Manpreet Kaur
Dept. Of Computer Science & Technology
Manav Rachna University
Haryana, India
manpreet.kathuria@gmail.com

***Abstract*— The accurate prediction of stock prices has a significant role to play in the present economic world. However, stocks have an ever-changing nature coupled with a risk factor that always seems much of a gamble, making it worthwhile for researchers to analyze them with machine learning algorithms. We thus seek to develop a machine-based model replacing the traditional yet unreliable time-series forecasting to predict the stocks from the past historical data of a firm and to uncover the underlying patterns to improve the accuracy of the prediction of the model. As a beginner, we have decided to use a couple of conventional machine learning algorithms to study the behavior of learning techniques for stock prediction. This paper presents an empirical study to study and analyze the behavior of decision Tree, Linear Regression, K-Nearest Neighbours, and LSTM learning algorithms to bet on the algorithm that best predicts the stock prices.**

***Keywords*— Stock, Stock Prediction, Machine Learning Algorithms, Decision Tree, Linear Regression, K-Nearest Neighbors, LSTM**

I. Introduction

In recent years Machine Learning has transformed the world in many ways from its applications in computer vision, medical science and many other domains. Humans have become dependent on data and information in society and due to this, technology is evolving. We could not help but wonder, can the same technique be used to predict something non-linear and non-stationary like the Stock Market? To simplify, stocks are known as a share that gives ownership in a company. If you own a given number of stocks in a company, it means you own a part of the company equal to the shares held as a proportion of the company's total shares. We can buy stocks from a company and sell them after a while and maybe make some profit, but since their factors are unpredictable, that's why we will take the help of specific algorithms to predict stock prices.

The stock market is one of the many exciting topics to be researched by students and scholars alike. Stock market prediction is a vast topic with a lot of dependencies like the company's recent product releases and its history etc. This all goes into the study of stock prices at the time of investment. There are ample amounts of data about stocks, but the most intriguing and challenging thing is predicting the price of these stocks based on old data, which is what we have done in our project. We have used the different Supervised Learning Algorithms and LSTM Architecture and tried to evaluate them to check which one of them best suits to predict the stock prices.

In the current research, we have used 2004 to May 2020 data of the stock prices of GOOG(Google). We have implemented various machine learning algorithms to predict future stock prices (30days) of Google.

The rest of the paper is organized as follows: - In Section II, we have discussed the dataset features and applied exploratory data analysis to get a better insight into the data. We have mentioned the related work done around a prediction system based on the Machine Learning algorithm in Section III. In Section IV, we have explained the various Machine Learning algorithms that are employed in the current research. In Section V, the experiments are conducted and the results of the study are analyzed. Finally, in Section VI, we have concluded and discussed the future scope of the research.

II. Data Analysis and visualization

A. About Dataset

Data collection is the most vital part of building machine learning models, as data plays a crucial role in the training and testing of the model. As we want our model to predict the accurate result, we should filter out the dataset on various aspects. In this proposed model, we have used primary data consisting of stock prices for the well-known company Google from Yahoo! Finance from the year 2004 to May 2020. The dataset includes several attributes, details of which are given below.

Various features of the gathered dataset which is used for Stock Price Prediction are as follows–

1) Date: This gives the date corresponding to which the stock prices are stored.

2) Open: The open is the opening price at which the market opens, i.e., the price for which the first trading executes.

3) High: It represents the highest price at which the stock has reached on the given day.

4) Low: It represents the most economical value at which the stock has concluded on the given day.

5) Close: The Close is the closing price, and it is the most crucial price as it is the closing price for the given period. Close works as an indicator for the strength- if Close is higher than Open, it is considered a fruitful day for the stock.

6) Adj Close: The adjusted closing price is the price that is quoted by the data provider after reviewing any corporate actions like dividends or stock splits.

978-1-7281-6916-3/20/$31.00 ©2020 IEEE

7) Volume: It refers to the number of shares sold or traded on a presented day.

We distribute the dataset in the training phase and in the testing phase, where 80% of the data goes to the training phase and 20% of the data goes to the testing phase. In the training phase, the model gets coached on the data and then the testing data gets tested on the trained model so that it can predict the stock price more accurately.

B. Visualizing the Data

The box plot, as represented in *Fig. 1,* shows the distribution of the data that are minimum and maximum value, median, first quartile and third quartile of each attribute. To summarize, we applied the exploratory analysis on the dataset using a boxplot.

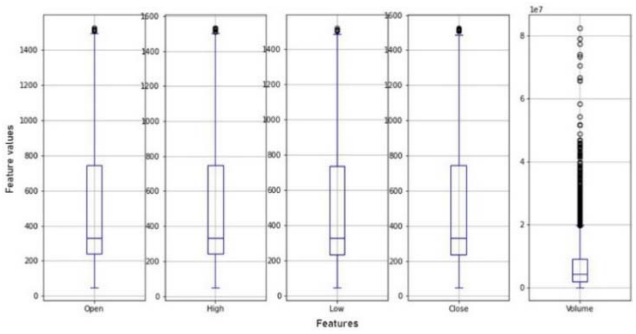

Fig. 1. Exploratory Analysis of the dataset.

The key observations from the above boxplot can be described as listed below:

1) Comparing the medians: As evident from Fig. 1, since the median line, roughly represented by the line dividing the box or promptly speaking the interquartile range into two equal parts is nearly the same for all the features except for the Volume of the shares, we can deduce that almost all of the elements present in the dataset exclusive of the Volume of the shares are spread around roughly equivalent medians. Thus, we can also conclude that there are only some small changes in the opening prices, closing prices, and the adjusted closing prices of the stocks.

2) Comparing the interquartile ranges and the whiskers: Interquartile ranges, stretch between the whiskers as represented by the box plots are a measure of scattering of the data, range of the score (another measure of dispersion) respectively. Thus, a plot with a broader area of the interquartile ranges and longer stretch for the whisker is prone to be more scattered than the one with a tight area for interquartile ranges and a shorter stretch for whiskers. Therefore, as apparent in the plot, all the features excluding the shares' Volume are widely scattered.

3) Potential Outliers: While examining a box plot, outliers are generally defined as the points that lie outside the whiskers. Thus, as perceptible from Fig. 1, the feature pertaining to the share's Volume has some values overshooting the maximum range of the whisker that is usually pre-set to 1.5 times of the Q3, i.e., the upper quartile range. We thus deftly drop these columns to render our dataset freer from the outliers.

III. Related work

In this section, we've laid a comprehensive yet concise description of some of the well-renowned work done around the stock prediction in recent years.

K. Hiba Sadia et al. in [1] proposed a prediction system based on Machine Learning algorithm Support Vector Machine (SVM) and Random Forest Classifier. We observed from this paper that the dataset they used was downloaded from Kaggle, which is more kind of raw and has to be pre-processed before making any prediction. We noticed that Kaggle does not provide the refined dataset for the particular company's Stock Prices; therefore, we have scratched other websites for the availability of the dataset.

A Logistic Regression based model using feature index values and significant time-effectiveness has been proposed by J. Gong et al. in [2]. They have mentioned that their daily trading prediction method is better in complexity and accuracy than any other models like the RBF-ANN model for prediction.

A prediction model where they measure the best algorithm before and after applying the Principal Component Analysis (PCA) has been presented by W. Khan et al. in [3]. They have concluded that the KNN algorithm gives better performance for stock prediction than SVM and Naive Bayes in terms of both accuracy and MAE (Mean Absolute Error).

Artificial Neural Network (ANN) has been used by Deepak et al. in [4] for predicting a stock market because ANN can be implemented on a broader range of parameters. In their model, they customize the features in ANN and then it is tested on the Support Vector Machine (SVM), which is a binary classifier. This paper's major take is that we can obtain live data from any company from the Yahoo Finance website.

A model that combined the historical data with the sentiments like news/tweets of a company has been proposed by Nayak et al. in [5]. They then used the model for daily prediction and observed the company's stock's monthly trend. They mentioned that on this type of dataset, Decision Tree performs better than Logistic Regression and SVM.

A model based on LSTM, CNN and RNN has been proposed by Selvin et al. in [6]. They evaluate the model performance based on percentage error. They train their model on Infosys' stock prices and can predict the stock prices of TCS and Cipla. From their experiment result, we observe that CNN, LSTM and RNN can spot some connection in the data and detect variation in the data.

LSTM has been applied by Nelson et al. in [7], predicting the price using the future movement of data and other technical measures. They can get a dividend gain in predicting the price but can increment it if variance could be less and make the LSTM model perform well.

We observed that recently LSTM has been giving excellent results than any other Neural Network Architecture. Thus, it was chosen for predicting the Stock Prices.

IV. EMPLOYING MACHINE LEARNING ALGORITHMS

Predicting the stock market change is very difficult because many factors can't be overlooked, for example– economic outlook, etc. When all these factors are combined, it is challenging to predict share prices with a high degree of accuracy. We have taken the help of Machine learning because it can identify specific patterns in the data and make precise predictions.

In our research, we used 2004 to May 2020 data of the stock prices of GOOG(Google). We have implemented various machine learning algorithms to predict future stock prices (30days) of Google.

A. Linear Regression

Variation in the stock market can be predicted using different regression models that are given below [8], linear regression and Support Vector Machine based regression model. Among the discussed models, linear regression is used in this scenario due to its simplicity [9]. Linear Regression is a standard statistical technique approach. Linear Regression generally tells us about the extent to which data linearly depends on a scalar variable (or dependent variable) and one independent variable. Regression performs operations on the dataset.

B. Decision Tree

Decision Tree is a graphical visualization that uses branching methodology to give us all the possible outcomes based on a specific condition. A decision tree is a predicting model that indicates the mapping from the observations about a thing to decisions about its target value. A tree has three essential elements that are nodes, branches, and leaves. A decision tree starts with a root and then branches off to several solutions. Branches keep on growing with an increasing number of decisions and the conditions. A leaf node is when we reach the end of the tree, i.e., we cannot further segregate it down to any other level. Each case that enters the tree is sorted based on its feature value and classified, starting from its root node to the leaf [10]. Its visualization is a flowchart diagram that effortlessly imitates human-level reasoning.

C. K-Nearest Neighbour

K-nearest neighbor algorithm can be implemented for Regression as well as for Classification problems. K-NN is a simple algorithm that stores all the available cases and arranges the new information or evidence based on a similarity measure [11]. KNN is also known to be the laziest learning that can make predictions using the training dataset directly and gives the k nearest instances and applying majority voting rules to determine the prediction outcome [12]. That implies when new information shows up, and then it is very well effortlessly grouped into a decent suite class by utilizing K-NN calculation.

D. LSTM

LSTM stands for Long Short Term Memory is a deep neural network. It is proposed by [13] to overcome the vanishing gradient problem that the Recurrent Neural Network (RNN) faced when implemented on the extended data sequence. In LSTM, the hidden layers that generally all neural network uses are replaced with memory cells, a series of gates. These gates *(Fig. 2)* decide whether to pass the detailed information to the next cell or ignore it. There are three gates in LSTM:-

a) Input Gate: Information goes in from this gate.

b) Forget Gate: This gate decides what information should get passed on to the next layer.

c) Output Gate: LSTM generated output goes out from this gate.

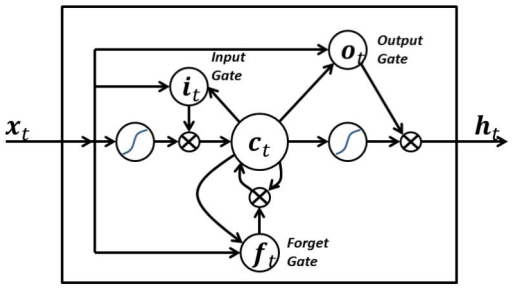

Fig. 2. Input, Forget and Output Gate in LSTM [14]

We have employed Keras library to implement the LSTM model, which works sequentially, which means layers are sequentially added one after the other. Consequently, we have used four layers of the LSTM model in which the first two-layers take the input of 50 features. The third layer, usually known as the "Dense" layer, serves as the connection between the second layer and the next Dense layer, which is the output layer. Ultimately, for compiling the model, we consider the two parameters – *Adam* optimizer and loss function as *mean square error*, which is further explained in Section V.C.4

The objective of applying LSTM is to evaluate the presumption of how LSTM will perform on time series data, i.e., stock price data in our case.

V. EXPERIMENTS AND RESULTS

This section includes the setup on which the experiments were operated along with the performance measure and the results and analysis obtained to understand the behavior of different algorithms in the stock prediction.

A. Experimental Setup

For our experiments, we used a laptop with the following hardware specification: -

- Nvidia GeForce GTX 1050 Ti GPU
- Intel® Core™ i5-8300H Processor @ 2.30 GHz
- 8GB DDR4 RAM Memory

Following libraries have been used for conducting the experiments: -

- NumPy – v1.19.1
- TensorFlow – v2.3.0
- Matplotlib – v3.3.0
- Keras – v2.4.3
- Pandas – v1.1.0

B. Evaluation

For evaluating the efficiency of the proposed models, we applied RMSE (Root Mean Square Error) to measure every proposed algorithm's accuracy [15]. The RMSE shows the average squared deviation of the actual value from the predicted value. And it gives information about how well the data is concentrated around the line of best fit. Hence, a good model is one whose RMSE value is lower. It is the most significant rule for fit if the fundamental motivation behind the model is forecast. RMSE value is calculated using (1)

$$RMSE = \sqrt{\frac{\sum_{i=1}^{n}(P_i - O_i)^2}{n}} \quad (1)$$

where P_i is the predicted value that we got from the model and O_i is the actual value from the data and n is the total number of observations.

C. Model Summary

In any Machine Learning algorithm, hyperparameters are the parameters that must be initialized before preparing the model. Hyperparameters can also be said as the data properties of a model training that an ML model will learn to enhance its functioning. There are various hyperparameters in ML and each algorithm is reliant on its parameters to yield the best result after the training. Hyperparameters that are used in the applied algorithm are discussed below.

1) Linear Regression

For optimizing the Linear Regression model, we have used the Learning Rate or abbreviated as α ('alpha'). Learning Rate for a linear regression can be defined as the constant where the regression intercept line meets the y-axis. α should be minimum for a regressor so that algorithm will slowly move towards the best intercept line.

2) Decision Tree

In Decision Tree, the parameter that we optimized is max_depth. The parameter determines the depth of the tree and how deep the decision node will go. When there are many features in the model, there will be more splitting in the decision tree and the tree will become complicated, which can lead to overfitting. We don't have a massive number of features in our case, so we kept the max_depth less.

3) K-NN

For the K-NN algorithm, we optimized the parameter K itself. K in K-NN regressor means the predicted output data-point will be the K nearest neighbours' average value. Keeping a higher value for K helps us to avoid the overfitting of the model.

4) LSTM

The LSTM model's architecture with the layers that we applied and the parameters that the model gets at the training phase can be seen in Table I.

TABLE I. LSTM ARCHITECTURE

Layer(type)	Output Shape	Parameters
lstm (LSTM)	(None, 60, 50)	10400
lstm_1(LSTM)	(None, 50)	20200
dense (Dense)	(None, 1)	1275
dense_1(Dense)	(None, 1)	26
Total parameters: 31,901 Trainable parameters: 31,901 Non-trainable parameters: 0		

Moreover, it might be also be taken into consideration that optimization is one of the core aspects of training a machine learning model, thus on an intuitive level, the essence of most of the machine learning models is to define an optimization algorithm that can reduce the cost function used to quantify the error between the predicted value and the expected value, more so we can also point that depending on the context of the problem, a cost function can either converge at its local minimum or a local maximum. And since the situation at our hand can be considered as a regression problem, it suits us quite well to use MSE (Mean Square Error) as defined in (2) as our loss function because, as evident from before, we have transformed our positively skewed data into a normal distribution. As MSE is more flexible in penalizing the outlier than the absolute mean error, it often ensures our dataset is not robust in considering outliers while making predictions.

$$MSE = \frac{\sum_{i=1}^{n}(P_i - O_i)^2}{n} \quad (2)$$

As for choosing our optimization algorithm, we decide to subside with Adam as the optimizer as unlike most of the stochastic optimization methods that maintain a single learning rate throughout the training, the optimization algorithms involving Adam as an optimizer calculates adaptive learning rates while updating the parameters as estimated from the first and second moment of gradients. Thus, we can say that an optimization method like Adam maintains the adaptive learning rate based on both the first and second moments of the gradients and keeps track of the exponentially decaying average of past gradients that are much more reliable, especially when dealing with sparse gradients.

D. Results and Analysis

After performing a series of experiments, we arrived at the following results.

The RMSE values of the proposed algorithms with the best-chosen hyperparameters are computed and are shown in Table II. We have observed that LSTM is giving the least RMSE as compared to other algorithms.

TABLE II. COMPARATIVE RMSE VALUE OF DIFFERENT ALGORITHMS

Algorithm	*Parameters*	*RMSE*
LSTM	**Optimizer = Adam, Loss Function = Mean Squared Error**	**21.83**
Linear Regression	$\alpha = 0.1$	54
Decision Tree	Max Depth = 5	59
KNN	K=13	149

After performing the series of observations, we have concluded that the LSTM model has been able to predict prices with the lowest RMSE of **21.83** at **200 epochs** (iterations) compared to different epochs variations, as shown in Table III.

TABLE III. LSTM MODEL'S RMSE VALUE WITH A DIFFERENT VARIATION OF EPOCHS

No. of Epochs	*RMSE*
50	26.94
100	23.86
200	**21.83**
300	31.37
400	49.21
500	25.21

As evident from Fig. 3, the LSTM model is the best-proposed model as it is depicting the near representation of predicted values from the actual stock price value and has the least RMSE value. Thus, for all intent and purposes, we can rely on the prices that LSTM has predicted. Whereas Linear Regression, Decision Tree and KNN cannot predict the prices as accurately as LSTM.

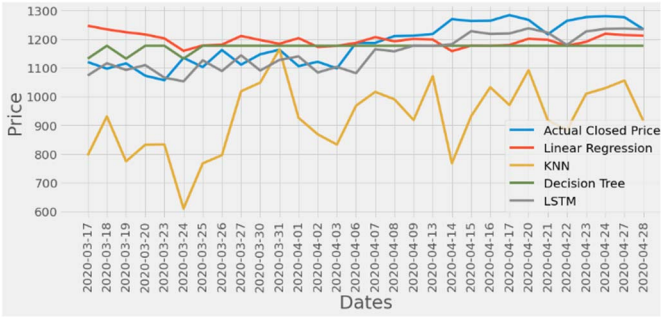

Fig. 3. Graphical Comparison of Actual Close Price and Predicted Closed Price of proposed algorithms.

The LSTM model that we worked on can thus have predicted the stock prices most accurately. A detailed comparison between the price predicted by all the models with the actual price can be seen in Fig. 4.

Date	Actual Price	LSTM	Linear Reg.	Decision Tree	K-NN
2020-04-15	1262.469971	1249.831055	1178.453431	1144.758281	889.541354
2020-04-16	1263.469971	1234.953613	1178.701617	1198.114363	999.325331
2020-04-17	1283.250000	1235.863892	1180.725238	1198.114363	993.635690
2020-04-20	1266.609985	1256.876099	1202.507832	1198.114363	1081.082257
2020-04-21	1216.339966	1236.862427	1199.284338	1198.114363	906.231156
2020-04-22	1263.209961	1186.866699	1178.426143	1144.758281	841.254103
2020-04-23	1276.310059	1244.269531	1191.902488	1198.114363	1002.173795
2020-04-24	1279.310059	1252.695312	1219.459388	1144.758281	971.730004
2020-04-27	1275.880005	1252.447876	1216.085958	1198.114363	1063.193852
2020-04-28	1233.670044	1248.345337	1213.198386	1144.758281	893.822223

Fig. 4. Actual Price of stocks and its comparison with the predicted values.

VI. CONCLUSION AND FUTURE WORK

Our research work has implemented several algorithms to test their accuracy over the stock prices dataset. That leads us to conclude that it's possible to identify the various patterns in the stock market, just like the trader does in real life.

For future work, we propose developing an API to check the Nifty50 P: E ratio, which can help us know whether it's the time to buy or sell the shares. That can be further upgraded by implementing Twitter Sentiment Analysis, which will help us understand people's opinions about a company and whether it would be profitable to invest.

We also propose to experiment with it further using the forex market as well. Still, since the market is very unpredictable and a slight change in a country can make a significant change in its currency, that's why a better-trained model would be required for it to predict forex prices and to help intraday traders.

REFERENCES

[1] Sadia, K. Hiba, Aditya Sharma, Adarrsh Paul, Sarmistha Padhi, and Saurav Sanyal. "Stock Market Prediction Using Machine Learning Algorithms." *IJEAT* (2019).

[2] Gong, Jibing, and Shengtao Sun. "A new approach of stock price prediction based on logistic regression model." *2009 International Conference on New Trends in Information and Service Science*. IEEE, 2009.

[3] Khan, W., M. A. Ghazanfar, M. Asam, A. Iqbal, S. Ahmad, and Javed Ali Khan. "PREDICTING TREND IN STOCK MARKET EXCHANGE USING MACHINE LEARNING CLASSIFIERS." *Science International* 28, no. 2 (2016).

[4] Deepak, Raut Sushrut, Shinde Isha Uday, and D. Malathi. "Machine learning approach in stock market prediction." *International Journal of Pure and Applied Mathematics* 115.8 (2017): 71-77.

[5] Nayak, Aparna, MM Manohara Pai, and Radhika M. Pai. "Prediction models for Indian stock market." *Procedia Computer Science* 89 (2016): 441-449.

[6] Selvin, Sreelekshmy, R. Vinayakumar, E. A. Gopalakrishnan, Vijay Krishna Menon, and K. P. Soman. "Stock price prediction using LSTM, RNN and CNN-sliding window model." In *2017 international conference on advances in computing, communications and informatics (icacci)*, pp. 1643-1647. IEEE, 2017.

[7] Nelson, David MQ, Adriano CM Pereira, and Renato A. de Oliveira. "Stock market's price movement prediction with LSTM neural networks." *2017 International joint conference on neural networks (IJCNN)*. IEEE, 2017.

[8] Nunno, Lucas. "Stock market price prediction using linear and polynomial regression models." *Computer Science Department, University of New Mexico: Albuquerque, NM, USA* (2014).

[9] Khan, Umair, Farhan Aadil, Mustansar Ali Ghazanfar, Salabat Khan, Noura Metawa, Khan Muhammad, Irfan Mehmood, and Yunyoung Nam. "A Robust Regression-Based Stock Exchange Forecasting and Determination of Correlation between Stock Markets." *Sustainability* 10, no. 10 (2018): 3702.

[10] Kotsiantis, Sotiris B., I. Zaharakis, and P. Pintelas. "Supervised machine learning: A review of classification techniques." *Emerging artificial intelligence applications in computer engineering* 160 (2007): 3-24.

[11] Aha, David W., Dennis Kibler, and Marc K. Albert. "Instance-based learning algorithms." *Machine learning* 6.1 (1991): 37-66.

[12] Gérard Biau and Luc Devroye, Lectures on the Nearest Neighbor Method, Springer, 2015.

[13] Hochreiter, Sepp, and Jürgen Schmidhuber. "Long short-term memory." *Neural computation* 9.8 (1997): 1735-1780.

[14] Greff, Klaus, Rupesh K. Srivastava, Jan Koutník, Bas R. Steunebrink, and Jürgen Schmidhuber. "LSTM: A search space odyssey." *IEEE transactions on neural networks and learning systems* 28, no. 10 (2016): 2222-2232.

[15] Siami-Namini, Sima, and Akbar Siami Namin. "Forecasting economics and financial time series: ARIMA vs. LSTM." *arXiv preprint arXiv:1803.06386* (2018).