International Journal of Latest Engineering and Management Research (IJLEMR)
ISSN: 2455-4847
www.ijlemr.com || Volume 07 – Issue 11 || November 2022 || PP. 54-67

Stock Price Prediction Using Machine Learning and Deep Learning Frameworks

Moses Praveen, MCA
Research Scholar,
Department of Computer Science and Information
Himalayan University, India

Dr. Balaji Koturu, M.Sc, Bed, P.hD
Guide,
Department of Computer Science and Information
Himalayan University, India

Abstract: Prediction of Stock price is always been a difficult job for the researchers in financial realm. While the *Efficient Market Hypothesis* claims that it is unfeasible to predict stock prices precisely, there are work in the literature that have demonstrated that stock price movements can be forecasted with a reasonable degree of accuracy, if appropriate variables are chosen and suitable predictive models are built using those variables. In this work, we present a robust and accurate framework of stock price prediction using statistical, machine learning and deep learning methods. We use daily data on stock prices at five minutes interval of time from the *National Stock Exchange* (NSE) of India and aggregate these granular data suitably to build the forecasting framework for stock prices. We contend that this framework, by combining several machine learning and deep learning methods, can accurately model the volatility of the stock price movement, and hence it can be utilized for short-term forecasting of the stock price. Eight classification and eight regression models, including one on deep learning-based approach, have been built using data of two stocks listed in the **NSE - Tata Steel and Hero Moto. Extensive results have been presented on the performance of these models.**

Keywords: Stock Price Prediction, Multivariate Regression, Logistic Regression, Decision Tree, K-Nearest Neighbor, Artificial Neural Networks, Random Forest, Bagging, Boosting, Support Vector Machines, LSTM.

I. Introduction

Prediction of upcoming movement of stock prices has been the subject matter of many research works. On one hand, we have proponents of the *Efficient Market Hypothesis* who claim that stock prices cannot be predicted. On the other hand, there are work that have shown that, if correctly modeled, stock prices can be predicted with a fairly reasonable degree of accuracy. The latter have focused on choice of variables, appropriate functional forms and techniques of forecasting. In this regard, Sen and Datta Chaudhari propose a novel approach of stock price forecasting based on a time series decomposition approach of the stock prices time series [1-8].

There is also an extent of literature on technical analysis of stock prices where the objective is to identify patterns in stock movements and profit from it. The literature is geared towards making money from stock price movements, and various indicators like Bollinger Band, *Moving Average Convergence Divergence* (MACD), *Relative Strength Index* (RSI), *Moving Average*, *Momentum Stochastics*, *Meta Sine Wave* etc., have been devised towards this end. There are also patterns like *Head and Shoulders*, *Triangle*, *Flag*, *Fibonacci Fan*, *Andrew's Pitchfork* etc., which are extensively used by traders for gain. These approaches provide the user with visual manifestations of the indicators which helps the ordinary investors to understand which way stock prices may move.

In this paper, we propose a granular approach to stock price prediction by combining statistical and machine learning methods of prediction on technical analysis of stock prices. We present numerous approaches for short-term stock price movement forecasting using various classification and regression techniques and compare their performance in prediction of stock price movement. We believe, this approach will provide several useful information to the investors in stock market who are particularly interested in short-term investments for profit. This work is an extended version of our previous work [9]. In the present work, we have extended our predictive framework by including five more classification and five more regression models including an advanced deep learning model.

The rest of the paper is organized as follows. In Section II, we present a clear statement of our problem at hand. Section III provides a brief review of the literature on stock price movement modeling and prediction. .In Section IV, we present a detailed discussion on the methodology that we have followed in this work. Section-V describes the details of all the predictive models built in this work and the results they have produced. A comparative analysis has been presented on the performance of the models in Section VI. Finally Section VII concludes the paper.

II. Problem Statement

The object of our work is to take stock price data at five minutes interval from the *National Stock Exchange* (NSE) of India and develop a robust forecasting framework for the stock price movement. Our contention is that such a granular approach can model the inherent dynamics and can be fine-tuned for immediate forecasting of stock price or stock price movement. Here, we are not addressing the problem of forecasting of long-term movement of stock price. Rather, our framework will be more relevant to a trade-oriented framework.

At any point of time in the Indian economy, given the appetite of financial market players including individuals, domestic institutions and foreign financial institutions, there is a finite amount of fund that are deployed in the stock market. This amount discounts the entire macro economics of that time. This fund would be distributed among various stocks. Thus, in the very short run, if some stock prices are rising, some other stock prices should be falling. Using our proposed framework, it will be possible for an investor to predict the price of a stock in the next slot of time, given the historical movement pattern of the stock. The approach builds in indicators like momentum, pivot points and range, all based on daily data on stock prices at five minutes interval of time.

III. Related Work

The literature attempting to prove or disprove the efficient market hypothesis can be classified in three strands, according to choice of variables and techniques of estimation and forecasting. The first strand consists of studies using simple regression techniques on cross sectional data [10-14]. The second strand of the literature has used time series models and techniques to forecast stock returns following economic tools like *Autoregressive Integrated Moving Average* (ARIMA), Granger Causality Test, *Autoregressive Distributed Lag* (ARDL) and *Quantile Regression* to forecast stock prices [15-18]. The third strand includes work using machine learning tools for prediction of stock returns [19- 23].

The major downside of the existing propositions in literature for stock price prediction is their in ability to predict stock price movement in a short-term interval. The current work attempts to address this shortcoming by exploiting the power of deep neural networks in stock price movement modeling and prediction.

IV. Methodology

In Section II, we mentioned that the goal of this work is to develop a robust forecasting framework for short-term price movement of stocks. We use the Meta stock tool [24] for collecting data on short-tem price movement of stocks. Particularly, we collected the stock data for two companies - Tata Steel and Hero Moto Corp. The data is collected at every 5 minutes interval in a day, for all the days in which the *National Stock Exchange* (NSE) was operating during the years 2019 and 2020. The raw data for each stock consisted of the following variables: (i) Date, (ii) Time, (iii) Open value of the stock, (iv) High value of the stock, (v) Low value of the stock, (vi) Close value of the stock, and (viii) the Volume of the stock traded in a given interval. The variable *Tim*e refers to the time instance at which the stock values are noted as each record is collected at 5 minutes interval of time. Hence, the time interval between two successive records in the raw data was 5 minutes. The raw data in this form is collected for two stocks-Tata Steel and Hero Moto Corp- for two years. In addition to the six variables in the raw data that we have mentioned, we collected also the NIFTY index at 5 minutes interval for the same period of two years, in order to capture the overall market sentiment that each time instant, so that more accurate and robust forecasting can be made using the combined information of historical stock prices and the market sentiment index. Therefore, the raw data for both the stocks now consists of seven variables. Since 5minutes interval is too granular, we make some aggregation of the raw data. We break the total time interval in a day into threes lots as follows: (1) morning slot that covers the time interval 9:00 AM till 11:30 AM, (2) afternoon slot that covers the time interval 11:35 AM till 1:30 PM and (3) evening slot that covers the time interval 1:35 PM till the time of closure of NSE in a given day. Hence, the daily stock information now consists of three records, each record containing stock price information for a timeslot.

Using the eight variables in the raw data and incorporating the aggregation of data using time slots, we derive the following variables that we use later on in our forecasting models. We used two approaches in forecasting- regression and classification. These approaches involved little differences in some of the variables, which we will describe and explain later in this Section.

The following eleven variables are derived and used in our forecasting models:

a) **Month:** it refers to the month to this a given record belongs. This variable is coded into numeric data, with "1" referring to the month of January and "12" referring to the month of December. The value of the variable *Month* lies in the range [1, 12].

b) Day_Month: this variable refers to the day of the month to which a given record belongs. It is a numeric variable lying within the range [1,31].For example, the date 14th February2019 will have a value14 against the variable *Day_Month*.

c) Day_Week: it is a numeric variable that refers to the day of the week corresponding to a given stock record. This variables lies in the range [1,5].Mondayiscodedas1,while Friday is coded as 5.

d) Time: it is a numeric variable that refers to the time slot to which a given record belongs. The morning, afternoon and evening slots are coded as 1, 2 and 3 respectively. Thus if a stock record is noted at time instance 2:45PM, the value of the *Time* variable corresponding to that record would be 3.

e) Open_Perc: it is a numeric variable that is computed as a percentage change in the value of the *Open* price of the stock over two successive times lots. The computation of the variable is done as follows. Suppose, we have two successive slots: $S_1$and S_2. Both of them consist of several records at five minutes interval of time. Let the *Open* price of the stock for the first record of $S_1$is $X_1$and that for $S_2$is X_2. The *Open_Perc* for the slot $S_2$is computed as $(X_2 - X_1)/X_1$in terms of percentage.

f) Sensex_Perc: it is a numeric variable that is computed as a percentage change in the NIFTY index over two successive time slots. The computation of the variable is done as follows. We compute the means of the NIFTY index values for two successive time slots S_1andS_2. Let us assume the means are M_1 and M_2 respectively. Then the *Sensex_Perc* for the slot S_2 is computed as $(M_2 - M_1)/M_1$ in terms of percentage.

g) Low_Diff: it is a numeric value that is computed as the difference between the *Low* values of two successive slots. For two successive slots S_1 and S_2, first we compute the mean of all *Low* values of the records in both the slots. If $L_1$and L_2 refers to the mean of the *Low* values for S_1 and S_2 respectively, then *Low_Diff*f or S_2 is computed as $(L_2 - L_1)$.

h) High_Diff: it is a numeric value that is computed as the difference between the *High* values of two successive slots. The computation is identical to that of *Low_Diff* except for the fact that *High* values are used in this case.

i) Close_Diff: it is a numeric value that is computed as the difference between the *Close* values of two successive slots. It computation is similar to the *Open_Perc* variable, except for the fact that we use the *Close* values in the slots and we don't compute any percentage. Hence, if two Successive slots S_1 and S_2 have close values C_1 and C_2respectively, then *Close Diff* for S_2 is computed as (C_2-C_1).

j) Vol_Diff: It is a numeric value that is computed as the difference between the *Volume* values of two successive slots. For two successive slots $S_1$and S_2, we compute the mean values of *Volume* for both the slots, say $V_1$and V_2 respectively. Now, the *Vol Diff* for S_2 is computed as $(V_2 - V_1)$.

k) Range_Diff: it is a numeric value that is computed as the difference between the *Range* values of two successive slots. For two successive slots $S_1$and S_2, suppose the *High* and *Low* values are H_1, H_2, L_1 and L_2 respectively. Hence, the *Range* value for S_1 is $R_1 = (H_1 - L_1)$ and for S_2 is $R_2 = (H_2 - L_2)$. The *Range_Diff* for the slot S_2 is computed as $(R_2 - R_1)$.

After, we compute the values of the above eleven variables for each slots for both the stocks for the time frame of two years (i.e., 2019 and 2020), we develop the forecasting framework. As mentioned earlier, we followed two broad approach in forecasting of the stock movements - regression and classification.

In regression approach, based on the historical movement of the stock prices we predict the stock price in the next slot. We use Open_Perc as the response variable, which is a continuous numeric variable. The objective of the regression technique is to predict the Open_Perc value of the next slot given the stock movement pattern and the values of the predictor still the previous slot. In other words, if the current time slot is S 1 , the regression techniques will attempt to predict Open_Percfor the next slot S 2 . If the predicted Open_Perc is positive, then it will indicate that there is an expected rise the stock price in S 2 , while a negative Open_Perc will indicate a fall in the stock price in the next slot. Based on the predicted values, an potential investor can make his/her investment strategy in stocks.

In the classification approach, the response variable Open_Perc is a discrete variable belonging to one of two classes. For developing the classification-based forecasting approaches, we converted Open_Perc into a categorical variable that takes up one of the two values 0 and 1. The value 0 indicating negative Open_Perc values and 1 indicating positive Open_Perc values. Hence, if the current slot is S 1 and if the forecast model expects a rise in the Open_Perc value in the next slot S 2 , then the Open_Perc value for S 2 will be 1. An expected negative value of the Open_Perc in the next slot will be indicated by a 0 value for the response variable.

For both classification and regression approaches, we experimented with three cases which are described below.
Case I: We used the data for the year 2019 which consisted of 19, 385 records at five minutes interval. These records were aggregated into 745 time slot records for building thepredictive model. We used the same dataset for testing the forecast accuracy of the models for both the stocks and made a comparative analysis of all the models.

Case II: We used the data for the year 2020 which consisted of 18, 972 records at five minutes interval. These granular data were aggregated into 745 time slot record for building the predictive model. We used the same dataset for testing the forecast accuracy of the model for both the stocks and carried out an analysis of the performance of the predictive models.

Case III: We used that data for 2019 as the training dataset for building the models and test the models using the data for the year 2020 as the test dataset. We, again, carried out an analysis on the performance of different models in this approach.

We use eight approaches to classification and eight approaches to regression for building our forecasting framework. The classification techniques used are: (i) Logistic Regression, (ii) K-Nearest Neighbor (iii) Decision Tree, (iv) Bagging, (v) Boosting, (vi) Random Forest, (vii) Artificial Neural Network, and (viii) Support Vector Machines. For measuring accuracy and effectiveness in these approaches, we use several metrics such as, sensitivity, specificity, positive predictive value, negative predictive value, and classification accuracy.

The eight regression methods that we built are:(i) Multivariate Regression,(ii)Decision Tree,(iii)Bagging, (iv) Boosting, (v) Random Forest, (vi) Artificial Neural Network, (vii) Support Vector Machine, and (viii) Long and Short Term Memory Network. While all the classification techniques are machine learning-based approaches, one regression technique - Long and Short Term Memory Network -is a deep learning method. For comparing the performance, of the regression methods, we use several metrics such as Root Mean Square Error (RMSE), and correlation coefficient between the actual and predicted values of the response variable, e.g., Open_Perc. We will observe in Section V that the deep learning regression method far outperforms the machine learning-based approaches.

V. Performance Results

In this Section, we offer a detailed discussion on the forecasting techniques that we have used and the results obtained using those techniques. We first discuss the classification techniques and then the regression techniques.

For both the stocks and for all the three cases, we computed the prediction accuracy of the classification models using several metrics. We define the metrics below.

Sensitivity: It is the ratio of the true positives to the total number of positives in the test dataset expressed as a percentage. Here, positive refers to the records belonging to the class 1. The term true positives refers to the number of positive cases that the model correctly identified.
Specificity: It is the ratio of the true negatives to the total number of negatives in the test data set expressed as a percentage. Here negative refers to the records belonging to class0. The term true negatives refers to the number of negative cases that the model correctly identified.
Positive Predictive Value (PPV): It is the ratio of the number of true positives to the sum of the true positive cases and false positive cases expressed as a percentage.
Negative Predictive Value (NPV): It is the ratio of the number of true negative cases to the sum of the true negative cases and false negative cases expressed as a percentage.
Classification Accuracy (CA): It is the ratio of the number of cases which are correctly classified to the total number of cases expressed as a percentage.

The eight classification models that we built are now discussed in detail.

A. Logistic Regression Classification

This being a classification technique, we transformed the response variable Open_Percto a discrete domain from continuous domain. In other words, we transformed the response variable into a categorical variable that can assume values 0 or 1. We converted all negative or zero values of Open_Percto the class 0 and all non-zero positive values to class 1. We used the function glm in R for building the logistic regression model with three parameters being passed in the function:(i)the first parameter is the formula which is Open_Perc~. To include Open_Perc as the response variable and all the remaining variables as the predictors, (ii) the second parameter is &quot; family = binomial & quot; indicating that model is a binary logistic regression that involves two classes, and (iii) the third parameter is the R data object containing the training dataset. We used the predict function in R to compute the probability of the test records to belong to the two classes. We assumed a threshold value of 0.5 as the probability. In other words, when the probability of a record belonging to a class exceeds 0.5 we assume that record belongs to that class. Table I presents the results.

Table I Logistic Regression Classification Results

Stock	**Case I Trg data 2019 Test data 2019**		**Case II Trg data 2020 Test data 2020**		**Case III Trg data 2019 Test data 2020**	
Tata Steel	Sensitivity	93.00	Sensitivity	94.20	Sensitivity	94.20
	Specificity	97.25	Specificity	98.65	Specificity	93.63
	PPV	92.54	PPV	96.53	PPV	85.52
	NPV	97.43	NPV	97.71	NPV	97.59
	CA	96.11	CA	97.38	CA	93.79
Hero Moto	Sensitivity	42.86	Sensitivity	55.56	Sensitivity	72.46
	Specificity	94.32	Specificity	91.70	Specificity	81.08
	PPV	70.91	PPVNP	72.78	PPVNPV	60.48
	NPV	83.62	V	83.77		88.05
	CA	81.74	CA	81.38	CA	78.62

Table II KNN Classification Results

Stock	**Case I Trg data2019 Test data2019**		**Case II Trg data 2020 Test data 2020**		**Case III Trg data2019 Test data 2020**	
Tata Steel	Sensitivity	81.50	Sensitivity	72.46	Sensitivity	95.17
	Specificity	97.80	Specificity	97.49	Specificity	57.72
	PPV	93.14	PPV	92.02	PPV	46.45
	NPV	93.51	NPV	89.86	NPV	94.62
	CA	93.42	CA	90.34	CA	67.44
Hero Moto	Sensitivity	64.29	Sensitivity	60.87	Sensitivity	10.63
	Specificity	94.67	Specificity	93.05	Specificity	94.98
	PPV	79.59	PPVN	77.78	PPVNPV	45.83
	NPV	89.13	PV	85.61		72.67
	CA	87.25	CA	83.86	CA	70.90

B. K-Nearest Neighbor Classification

The *k*- nearest neighbor is an example of instance-based learning used for classification in which the training data is stored, so that a classification for a new unclassified record may be found simply by comparing it to the most similar records in the training set. The value of *k* determine show and the value of *k*=3 was finally chosen, This value of *k* was found to produce the best performance of the models with the minimum probability of model over fitting. Table II provides the performance of this approach on the two stocks.

Table III Decision Tree Classification Results

Stock	**Case I Trg data 2019 Test data 2019**		**Case II Trg data 2020 Test data 2020**		**Case III Trg data 2019 Test data 2020**	
Tata Steel	Sensitivity	96.00	Sensitivity	93.72	Sensitivity	98.07
	Specificity	97.43	Specificity	99.42	Specificity	92.86
	PPV	93.30	PPV	98.48	PPV	84.58
	NPV	98.52	NPV	97.54	NPV	99.18
	CA	97.05	CA	97.79	CA	94.34
Hero Moto	Sensitivity	40.66	Sensitivity	53.62	Sensitivity	36.23
	Specificity	92.18	Specificity	92.47	Specificity	89.77
	PPV	62.71	PPVN	74.00	PPVNPV	58.59
	NPV	82.78	PV	83.30		77.89
	CA	79.60	CA	81.38	CA	74.48

C. Decision Tree Classification

The *Classification and Regression Tree* (CART) algorithm produces decision trees that are strictly binary, containing exactly two branches for each decision node. CART recursively partitions the records in the training data set into subsets of records with similar values for the target attributes. The CART algorithm constructs the trees by carrying out an exhaustive hunt on each node for all available variables and all possible splitting values and selects the optimal split based on some "goodness of split" criteria.

We used the tree function defined in the tree library of R for classification of the stock records. Table III presents the results produced by the classification approach. While for Model I and Model III, the classification trees consisted of 9 nodes with *Close_Diff*, *Low_Diff*, *High_Diff* and *Month* as the predictors, the tree for Model II having the same number of nodes found *Close_Diff*, *Month* and *Low_Diff*as discriminating predictors.

D. Bagging Classification

Bootstrap Aggregation (Bagging) is an ensemble technique. It works as follows: Given a set *D*, of *d* tuples, for iteration *i*, a training set, D_i of *d* tuples is sampled with substitution form the original set of tuples *D*. Each training set is a bootstrap sample. Since the sampling is done with replacement, some tuples in *D* may not be included in D_i, while some may be included more than once. A classifier model M_i, is learned for each training set, D_i. To classify an unknown tuple *X*, each classifier, M_i returns its class predictions, which is considered as one vote. The bagged classifier counts the votes and assigns the class with maximum number of votes to *X*.

Table IV Bagging Classification Results

Stock	Case I Trg data 2019 Test data 2019		Case II Trg data 2020 Test data 2020		Case III Trg data 2019 Test data 2020	
Tata Steel	Sensitivity	96.00	Sensitivity	95.17	Sensitivity	98.07
	Specificity	97.43	Specificity	99.23	Specificity	92.86
	PPV	93.20	PPV	98.01	PPV	84.58
	NPV	98.52	NPV	98.09	NPV	99.18
	CA	97.05	CA	98.07	CA	94.34
Hero Moto	Sensitivity	68.13	Sensitivity	61.35	Sensitivity	41.06
	Specificity	96.80	Specificity PPV	95.95	Specificity	85.33
	PPV	87.32	NPV	85.81	PPV	52.80
	NPV	90.38		86.14	NPV	78.37
	CA	89.80	CA	86.07	CA	72.69

For carrying out classification on stock price data, we used *bagging* function defined in the *I pred* library of R. The value of the parameter *nbag-* that specifies the number of samples - was taken as 25. The results of classification are presented in Table IV.

Table V Boosting Classification Results

Stock	Case I Trg data 2019 Test data 2019		Case II Trg data2020 Test data2020		Case III Trg data2019 Test data2020	
Tata Steel	Sensitivity	100.00	Sensitivity	100.00	Sensitivity	84.62
	Specificity	100.00	Specificity	100.00	Specificity	98.17
	PPV	100.00	PPV	100.00	PPV	95.65
	NPV	100.00	NPV	100.00	NPV	93.05
	CA	100.00	CA	100.00	CA	93.79
Hero Moto	Sensitivity	100.00	Sensitivity	100.00	Sensitivity	54.60
	Specificity	100.00	Specificity	100.00	Specificity	79.67
	PPV	100.00	PPV	100.00	PPV	45.89
	NPV	100.00	NPV	100.00	NPV	84.75
	CA	100.00	CA	100.00	CA	73.66

E. Boosting Classification

Unlike Bagging, Boosting, assigns weights to each training tuple. A series of *k* classifiers is iteratively learned. After a classifier $M_i$is learned, the weights are updated to allow building of subsequent classifier M_{i+1}, to focus more closely on the training tuples that were misclassified by M_i. The final boosted classifier combines the votes of each individual classifier, where the weight of each classifier's vote is a function of its accuracy.

F. Adaptive Boosting (Ada Boost) is a very popular variant of Boosting.

We used *boosting* function of the *ada bag* library in R for classification of stock price data. Table V presents the results of Boosting classification.

G. Artificial Neural Network Classification

An *Artificial Neural Network* (ANN) consists of an input layer, one or more hidden layers, and an output layer. Each layer is made up of units. The inputs to the network corresponds to the attributes measured for each training tuple. The inputs are fed simultaneously into the units making up the input layer. These inputs pass through the input layers and are then weighted and fed simultaneously to a second layer of *neuron* units, known as the hidden layer. The output of the hidden layer can be input to another hidden layer and so on. The weighted outputs of the last hidden layer are input to units making up the output layer, which produces the network's prediction for given tuples.

Table VII ANN Classification Results

Stock	**Case I** **Trg data 2019** **Test data 2019**		**Case II** **Trg data 2020** **Test data 2020**		**Case III** **Trg data 2019** **Test data2020**	
Tata Steel	Sensitivity	95.00	Sensitivity	94.69	Sensitivity	100.00
	Specificity	96.70	SpecificityPPV	98.84	Specificity	32.82
	PPV	91.35		97.03	PPV	37.30
	NPV	98.14	NPV	97.90	NPV	100.00
	CA	96.25	CA	97.66	CA	52.00
	Sensitivity	63.19	Sensitivity	68.12	Sensitivity	7.73
Hero Moto	Specificity	93.61	SpecificityPPV	91.31	Specificity	100.00
	PPV	76.16	NPV	75.81	PPV	100.00
	NPV	88.72		87.76	NPV	73.06
	CA	86.17	CA	83.17	CA	73.66

We used the *neuralnet* function defined in the *neuralnet* library in Rfor classifying the stock price data. The raw data is normalized using the *min-max normalization* approach. Only the predictors are normalized, the response variable: *Open_Perc* is kept unchanged. The parameter *hidden* of the function *neuralnet* is changed to realize different number of hidden layers in the network. The parameter*s tepmax* is set to

Table VI Random Forest Classification Results

Stock	**Case I** **Trg data 2019** **Test data 2019**		**Case II** **Trg data 2020** **Test data 2020**		**Case III** **Trg data 2019** **Test data 2020**	
Tata Steel	Sensitivity	95.00	Sensitivity	92.75	Sensitivity	96.14
	Specificity	97.06	Specificity	98.46	Specificity	92.86
	PPV	92.23	PPV	96.00	PPV	84.32
	NPV	98.14	NPV	97.14	NPV	98.36
	CA	96.51	CA	96.83	CA	93.79
	Sensitivity	37.36	Sensitivity	46.93	Sensitivity	72.95
Hero Moto	Specificity	88.45	Specificity	88.03	Specificity	63.51
	PPV	51.13	PPV	60.00	PPV	44.41
	NPV	81.37	NPV	80.00	NPV	85.45
	CA	75.97	CA	75.72	CA	66.21

H. Random Forest Classification

Random forest is an ensemble technique. Each of the classifiers in the ensemble is a decision tree classifier so that the collection of classifiers is a forest. The individual decision trees are generated using a random selection of attribute sat each node to determine the split. More formally, each tree depends on the values of a random vector sampled independently and with the same distribution for all trees in the forest. During classification, each tree votes and the most popular class is returned.

We have used the random Forest function defined in the random Forest library in R for classification purposes of the stock price data. Table VI presents the results.

The maximum value of 10^6 so that maximum number of iteration capability of the neural net function can be utilized. In order to carry out classification exercise, the parameter *linear. Output* if set to FALSE in the *neural net* function. For the Tata Steel stock data, for all three models, we needed one node in the hidden layer to obtain the results presented in Table VII. However, for the Hero Motocorp data, we needed three nodes in the hidden layer for all the three models. Fig.1depicts the ANN model build for classification the Hero Motocorp stock data for Case III.

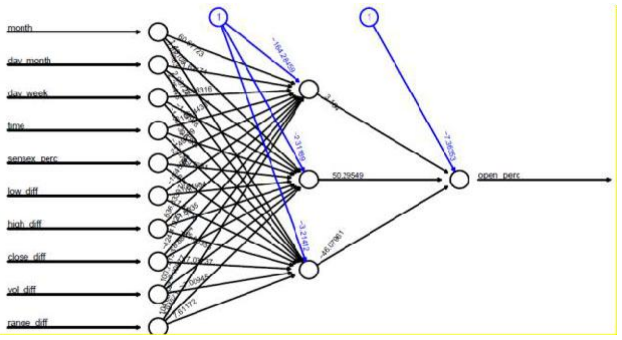

Fig.1.ANN model with three hidden nodes for Hero Motocorp Case III

H. SVM Classification

Support Vector Machines (SVMs) is a method of classification of both linear and nonlinear data. It uses a non linear mapping to transform the original training data into a higher dimension. Within this new higher dimension, it searches for the linear optimal *hyperplane* that separates the two classes. SVM finds this hyperplane using *support vectors* which are the *essential* and the *discriminating* training tuples to separate the two classes.

We have used the *ksvm* function defined in the *kernlab* library in R for carrying out classification of the stock price data. The function *ksvm* has an optional parameter called *kernel* which is set to *vanilladot* in our implementation. Table VIII presents the results of SVM classification.

Table VIII SVM Classification Results

Stock	Case I Trg Data 2019 Test Data 2019		Case II Trg Data 2020 Test Data 2020		Case III Trg Data 2019 Test Data 2020	
Tata Steel	Sensitivity	91.26	Sensitivity	96.50	Sensitivity	85.22
	Specificity	97.77	Specificity	97.33	Specificity	97.78
	PPV	94.00	PPV	93.24	PPV	94.69
	NPV	96.70	NPV	98.65	NPV	93.44
	CA	95.97	CA	97.10	CA	93.79
Hero Moto	Sensitivity	64.71	Sensitivity	72.00	Sensitivity	71.67
	Specificity	78.53	Specificity	78.40	Specificity	75.34
	PPVNP	18.13	PPVNP	34.78	PPVNP	20.77
	V	96.80	V	94.59	V	96.72
	CA	77.58	CA	78.12	CA	75.03

I. Multivariate Regression

In this regression approach, we used *Open_Perc* as the response variable and the remaining ten variables as the predictors to build predictive models for three cases mentioned earlier in Section IV. In all these cases, we use the programming language R for data management, model construction, testing of models and visualization of results.

Case I: We use 2019 data as the training data set for building the model and test the model using the same data set. For both the stocks, we used two approaches of multivariate regression - (i) *Backward Deletion* and (ii) *Forward Addition* of variables. Both the approaches yielded the same results for both the stocks.

For Tata Steel data set for the year 2019, we applied the vif function in the faraway library to detect the collinear variable in order to get rid of the multi collinearity problem. It was observed that the variables *Low_Diff*, *High_Diff* and *Close_Diff* exhibited multi collinearity. We retained the variable *Close_Diff* and left out the other two variables. Using the *drop1* function in case of the backward deletion technique, and *add1* in case of the forward addition technique, we identified the variables that were not significant in the model and did not contribute to the information content of the model. For identifying the variables that contributed least to the information content in the model at each iteration, we used the *Akaike Information Criteria* (AIC) - the variable that had least AIC value and non-significant *p*-value at each iteration, was removed from the model, in case of backward deletion process. On the other hand, the variable that had the lowest AIC and a significant p-value was added to the model at each iteration for the forward addition technique. It was found that *Close_Diff* and *Sensex_Perc* are the two predictors which were finally retained in the model.

For Hero Moto for the year 2019, the variables *Sensex_Perc*, *Close_Diff* and *Vol_Diff* were found to be significant for the model.

Table IX Multi Variate Regression Results

Stock	**Case I Trg Data 2019 Test Data 2019**		**Case II Trg Data 2020 Test Data 2020**		**Case III Trg Data 2019 Test Data 2020**	
Tata Steel	Correlation RMSE/Mean	0.97 32.40	Correlation RMSE/Mean	0.98 4.01	Correlation RMSE/Mean	0.98 24.08
Hero Moto	Correlation RMSE/Mean	0.56 116.23	Correlation RMSE/Mean	0.84 116.03	Correlation RMSE/Mean	0.53 116.03

Case II: For Tata Steel, in this model, the predictors *Close_Diff*, *Sensex_Perc* and *Day_Week* were found to be significant using both the backward deletion and forward addition technique. However, for Hero Moto, the significant variables were *Sensex_Perc*, *Low_Diff* and *Vol_Diff.*

Case III: In this case the model is identical to that in Case I. However, since the test data is difference there, we find different values for the correlation coefficient and RMSE.

For testing the prediction accuracy of the models built for both the stocks, we used the *predict* function for forecasting the values of the response variables. We computed the correlation coefficient between the actual *Open_Perc* values and the predicted *Open_Perc* values. We also computed the RMSE among the actual and predicted values, and hence derived the percentage of the RMSE with respect to the mean of the actual *Open_Perc* values. Table IX presents the results of the performance of the Multivariate Regression method.

TABLE X Decision Tree Regression Results

Stock	**Case I Trg Data 2019 Test Data 2019**		**Case II Trg Data 2020 Test Data 2020**		**Case III Trg Data 2019 Test Data 2020**	
Tata Steel	Correlation RMSE/Mean	0.96 40.13	Correlation RMSE/Mean	0.97 33.65	Correlation RMSE/Mean	0.04 198.04
Hero Moto	Correlation RMSE/Mean	0.98 30.92	Correlation RMSE/Mean	0.97 34.88	Correlation RMSE/Mean	0.02 195.17

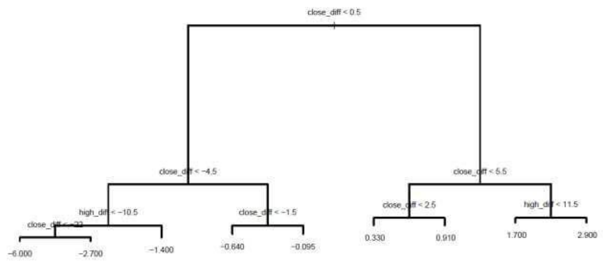

Fig 2. Decision Tree regression model for Tata Steel stock for Case II

J. Decision Tree Regression

We have used the same *tree* function in the *tree* library in R to carry out regression as we did in classification. However, in this case the response variable was kept as numeric and not converted to a factor variable unlike in the classification techniques. The *predict* function is used to predict the values of the response variable. For both the stocks, *Close_Diff*, *High_Diff* and *Low_Diff* are the predictors which are found to be important for all the three cases. The functions *cor* and *rmse* defined in the library *Metrics* are used to compute the correlation coefficient and the RMSE value for determining the prediction accuracy of the models. Table X presents the results of the performance of this regression method.

K. Bagging Regression.

For carrying out regression on stock price data, we use *bagging* function defined in the *ipred* library of R. The value of the parameter *n bag*-that specifies the number of samples-is taken as 100. The results of classification are presented in Table XI. We use the predict function in *ipred* library to predict the response variable values and *rmse* function in the *Metric* library to compute the RMSE values of the predicted values. The *cor* function in the base Ris used to compute the correlation between the original and the predicted values of the response variable. Table XI presents the performance results of the Bagging regression.

Table XI Bagging Regression Results

Stock	Case I Trg Data 2019 Test Data 2019		Case II Trg Data 2020 Test Data 2020		Case III Trg Data 2019 Test Data 2020	
Tata Steel	Correlation RMSE/Mean	0.97 34.22	Correlation RMSE/Mean	0.98 28.54	Correlation RMSE/Mean	0.96 65.24
Hero Moto	Correlation RMSE/Mean	0.70 101.03	Correlation RMSE/Mean	0.69 100.85	Correlation RMSE/Mean	0.51 119.76

L. Boosting Regression

We use the *black boost* function defined in the *m boost* library in R for building regression models on the stock price data. *boosting* function of the *adabag* library in R for classification of stock price data. As in other cases of regression, the *predict* and *rmse* functions are used to compute the predicted values and the RMSE values in the regression model. Table XII presents the results of Boosting regression.

Table XII Boosting Regression Results

Stock	Case I Trg Data 2019 Test Data 2019		Case II Trg Data 2020 Test Data 2020		Case III Trg Data 2019 Test Data 2020	
Tata Steel	Correlation RMSE/Mean	0.98 340.65	Correlation RMSE/Mean	0.99 22.39	Correlation RMSE/Mean	0.77 106.23
Hero Moto	Correlation RMSE/Mean	0.69 102.12	Correlation RMSE/Mean	0.69 99.67	Correlation RMSE/Mean	0.51 119.88

Table XIII Random Forest Regression Results

Stock	Case I Trg Data 2019 Test Data 2019		Case II Trg Data 2020 Test Data 2020		Case III Trg Data 2019 Test Data 2020	
Tata Steel	Correlation RMSE/Mean	0.99 15.23	Correlation RMSE/Mean	0.99 12.88	Correlation RMSE/Mean	0.97 59.77
Hero Moto	Correlation RMSE/Mean	0.95 53.09	Correlation RMSE/Mean	0.95 53.67	Correlation RMSE/Mean	0.51 119.60

M. Random Forest Regression

We use the *random Forest* function defined in the *random Forest* library in R for regression purposes. The response variable *Open_Perc* is kept as a numeric variable and not converted to a factor variable as it was done in case of Random Forest classification. The same *predict* and *rmse* functions are used as in other regression methods. Table X VIII presents the results of the performance of the Random Forest regression.

Table XIV ANN Regression Results

Stock	Case I Trg Data 2019 Test Data 2019		Case II Trg Data 2020 Test Data 2020		Case III Trg Data 2019 Test Data 2020	
Tata Steel	Correlation RMSE/Mean	0.97 31.70	Correlation RMSE/Mean	0.98 24.23	Correlation RMSE/Mean	0.98 72.91
Hero Moto	Correlation RMSE/Mean	0.73 95.38	Correlation RMSE/Mean	0.73 93.01	Correlation RMSE/Mean	0.42 359.91

N. Artificial Neural Network Regression

As in case of classification, we use the *neural net* function defined in the *neural net* library in R for regression on the stock price data. The predictors are normalized using *min- max normalization* before building the model. The *compute* function defined in the *neural net* library is used for computing the predicted values, while the parameter *hidden* is used to change the number of nodes in the hidden layer. The value of the parameter *stepmax* is set to 10^6 so as to exploit the maximum number of iterations executed by the neural net function. The parameter *linear. Output* is by default set to TRUE, and hence it is not altered. For the Tata Steel stock data, for all three cases, we needed one node in the hidden layer. However, for the Hero Motocorp data, we needed two nodes in the hidden layer for Case I and III and three nodes for Case II. Table XV presents the results of ANN regression.

Table XV SVM Regression Results

Stock	**Case I Trg Data 2019 Test Data 2019**		**Case II Trg Data 2020 Test Data 2020**		**Case III Trg Data 2019 Test Data 2020**	
Tata Steel	Correlation	0.96	Correlation	0.97	Correlation	0.88
	RMSE/Mean	39.61	RMSE/Mean	31.51	RMSE/Mean	71.07
Hero Moto	Correlation	0.71	Correlation	0.70	Correlation	0.52
	RMSE/Mean	101.75	RMSE/Mean	98.95	RMSE/Mean	117.14

O. Support Vector Machine Regression

For building regression model using SVM, we use the *svm* function defined in the *e1071* library in R. The *predict* function is used for predicting the response variable values using the regression model, and the *rmse* function is used to compute the RMSE values for the predicted quantities. Table XV presents the results of the SVM regression model.

P. Long and Short Term Memory Networks Regression

Finally, we apply a deep learning technique of regression using *Long and Short Term Memory* (LSTM) networks. First, we provide a brief discussion on the working principles of such networks. LSTM is a variant of *Recurrent Neural Networks* (RNNs) - neural networks with feedback loops [25]. In such networks output at the current times lot depends on the current inputs as well as previous state of the network. However, RNNs suffer from a problem that these networks cannot capture long-term dependencies due to vanishing or exploding gradient during back propagation in learning the weights of the links [25]. LSTM networks overcome such problems and hence such networks are quite effective in forecasting in multivariate time series. LSTM networks consist of memory cells that can maintain their states over time using memory and gating units that regulate the information flow into and out of the memory. There are different variants of gates used. The *forget gates* control what information to throw away from memory. The *input gates* are meant for controlling the new information that is added to cell state from the current input. The *cell state vector* aggregates the two components - the old memory from the *forget gate* and the new memory from the *input gate*. At the end, the *output gates* conditionally decide what to output from the memory cells. The architecture of an LSTM network along with the *Back propagation Through Time (BPTT)* algorithm for learning provides with such networks a very powerful ability to learn and forecast in a multivariate time series framework.

We use Python programming language and the Tensor flow deep learning framework for implementing LSTM networks and utilize those networks to predict the prices of Tata Steel and Hero Motocorp in a multivariate time series framework. For this purpose, we use *Close* price of the stocks as the response variable and the predictors chosen are - *Open*, *High*, *Low*, *Volume* and NIFTY index values. However, unlike for the machine learning techniques, we don't compute the differences between successive slots. Rather, we forecast the *Close* value of the nexts lot based on the predictor values in the previous lots. For both the stocks, and for all three cases, we use *Mean Absolute Error* (MAE) as the loss function and *ADAM* as the optimizer. However, we train the network with different *epoch* values and *batch sizes* of the data under different cases and for different stocks in order to obtain the optimum performance of the network. The *Sequential* constructor in the Tensor flow frame work has been used to build the LSTM model. Fig 3 shows the way loss converged for both the training and test data sets with epoch values in Case III for the Tata Steel stock.

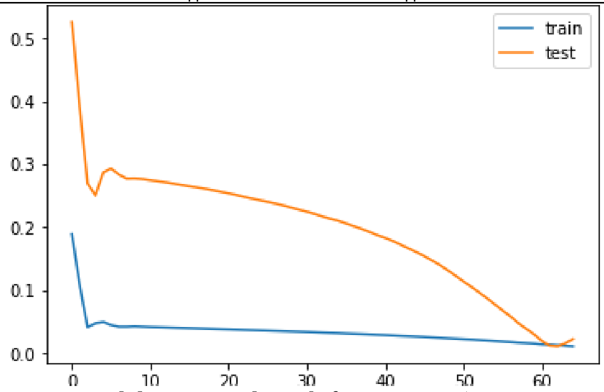

Fig.3.Loss convergence in LSTM model Tata Steel stock for Case III (x- axis: No of epoch, y-axis: Percentage of loss)

Table XVI LSTM Regression Results

Stock	**Case 1 Trg Data 2019 Test Data 2019**		**Case II Trg Data 2020 Test Data 2020**		**Case III Trg Data 2019 Test Data 2020**	
Tata Steel	Correlation RMSE/Mean	1.00 7.94	Correlation RMSE/Mean	1.00 4.04	Correlation RMSE/Mean	1.00 2.36
Hero Moto	Correlation RMSE/Mean	1.00 0.70	Correlation RMSE/Mean	1.00 0.20	Correlation RMSE/Mean	1.00 0.28

For the Tata Steel stock, we use 70 epochs with a batch size of 72 for Case I and Case II. For Case I, at the completion of the epoch 63, the loss was found to have converged to as low as 0.0106, which then slowly increased to a value of 0.0553, at the end of the epoch 70. For Case II, the loss converged to 0.0114 at the completion of the epoch 65, and then slowly increased to 0.0388 at the end of epoch 70. For Case III, 65 epochs with batch size 72 were used. The loss converged to 0.0114 at the end of epoch 63, and slowly increased to 0.0228 at the completion of the epoch70.

In case of the Hero Motocorp stock data, we employ 65 epochs with batch size of 72 for Case I and II. For Case I, the loss converged to 0.0286 on completion of the epoch 65, while for Case II, the loss was as low as 0.0097 at the end. For Case III, 55 epochs with batch size 72 were used. The loss consistently decreased over the successive epochs finally converging to a value0.0135atthe end. Table XVI presents the results of LSTM regression on the two stock prices data.

Table XVII Best Performing Classification Techniques

Stock	**Case I Trg Data 2019 Test Data 2019**		**Case II Trg Data 2020 Test Data 2020**		**Case III Trg Data 2019 Test Data 2020**	
Tata Steel	Sensitivity Specificity PPV NPV CA	BOOST BOOST BOOST BOOST BOOST	Sensitivity Specificity PPV NPV CA	BOOST BOOST BOOST BOOST BOOST	Sensitivity Specificity PPV NPV CA	ANN BOOSTB OOST ANN BAG
Hero Moto	Sensitivity Specificity PPV NPVCA	BOOST BOOST BOOST BOOST BOOST	Sensitivity Specificity PPV NPVCA	BOOST BOOST BOOST BOOST BOOST	Sensitivity Specificity PPV NPVCA	RFANN ANNSV M LR

VI. Brief Analysis of Results

In this Section, we provide a very brief analysis of the results and compare the performance of various models.

Among the classification techniques, Boosting quite expectedly performed the best for Case I and Case II, on all metrics. These two cases actually measured the performance of the model on the training data, and Boosting surely performs best when tested on the training data. However, for the Case III, when the test data and training data are different sets-the former being the stock data for the year 2020, while the latter is for the year 2019 - Bagging and Logistic Regression are found to have performed best on classification accuracy for Tata Steel and Hero Motocorp stock data respectively. ANN is found to have performed best on two of the metrics for both the stocks - sensitivity and NPV for Tata Steel stock data, and specificity and PPV for the Hero Motocorp stock data. Since the Hero Motocorp stock data was very unbalanced for both the years 2019 and 2020, the technique that has yielded the highest value of sensitivity for this data - Random Forest - can also be assumed to have performed very well.

For the regression techniques, we find that the deep learning technique - LSTM - outperforms all machine learning techniques on all metrics for both the stocks by large margins. In Table XVIII, we have also listed the best performing machine learning technique for both the stocks corresponding to each metric. It is interesting to note that for Case I and II and for Tata Steel stock, Random Forest performed best among the machine learning techniques, while for the Hero Motocorp stock, Decision Tree produced the best results under those two cases. Multivariate Regression produced the lowest RMSE/Mean value for both the stocks under Case III, while ANN and SVM produced the best correlation values for Tata Steel stock and the Hero Motocorp stock respectively.

Table XVIII Best Performing Regression Techniques (DL & ML)

Stock	**Case I Trg Data 2019 Test data 2019**		**Case II Trg Data 2020 Test Data 2020**		**Case III Trg Data 2019 Test Data 2020**	
Tata Steel	Correlation	LSTM RF	Correlation	LSTM RF	Correlation	LSTM ANN
	RMSE/Mean	LSTM RF	RMSE/Mean	LSTM RF	RMSE/Mean	LSTM MV
Hero Moto	Correlation	LSTM DT	Correlation	LSTM DT	Correlation	LSTM SVM
	RMSE/Mean	LSTM DT	RMSE/Mean	LSTM DT	RMSE/Mean	LSTM MV

VII. Conclusion

In this paper, we propose a framework of stock price movement prediction in the short-term time period using eight classification and eight regression models. These models are based on machine learning and deep learning approaches. We built, fine-tuned and tested these models using the data of two stocks listed in the *National Stock Exchange* (NSE)- Tata Steel and Hero Moto. The stock price data were collected at five minutes interval of time using the Meta stock tool over a period of two years-January 2019 to December 2020. The raw data was suitably cleaned, appropriate variable transformation done, and the predictors and the response variable were identified for building the classification and regression-based predictive models. In addition to building eight classification and seven regression models using machine learning approach, we also created one deep learning-based regression model using *Long and Short Term* (LSTM) memory network. Extensive results have been presented on the performance of these eight classification and eight regression models. It has been observed that while among the classification techniques, *Artificial Neural Networks* (ANNs) has, on the average, produced the highest level of accuracy, LSTM out performed, by a large margin, all other regression models. The results have given us a significant insight into stock price forecasting. Since regression using LSTM provided us with a very accurate prediction on stock price movement in a short interval of time, this approach can be used as a single solution to this problem. The use of classification-based approaches and other regression techniques for predicting price movement lose their significance and applications in this field.

References

[1] J. Sen and T. Datta Chaudhuri, "An alternative framework for time series decomposition and forecasting and its relevance for portfolio choice - a Comparative study of the Indian consumer durable and small cap sector," Journal Economics Library, vol 3, no. 2, pp. 303 -326,2016.

[2] J. Sen and T. Datta Chaudhuri, "Decomposition of time series data of stock markets and its implications for prediction - an application for the Indian auto sector," In Proceedings of the 2nd National Conference on Advances in Business Research and Management Practices(ABRMP'2016), Kolkata, India, pp 15-28.

[3] J. Sen and T. Datta Chaudhuri, "An investigation of the structural characteristics of the Indian IT sector and the capital goods sector-an application of the R programming language in time series decomposition and forecasting," Journal of Insurance and Financial Management, vol 1, no 4, pp. 68 - 132, 2016.

[4] J. Sen and T. Datta Chaudhuri, "A time series analysis-based forecasting framework for the Indian healthcare sector," Journal of Insurance and Financial Management, vol 3, no. 1, pp. 66 -94, 2017.

[5] J. Sen and T. Datta Chaudhuri, "A predictive analysis of the Indian FMCG sector using time series decomposition-based approach, "Journal of Economics Library, vol 4, no 2, pp. 206 - 226, 2017.

[6] J. Sen, "A time series analysis-based forecasting approach for the Indian realty sector," International Journal of Applied Economic Studies, vol 5, n0 4, pp 8 - 27, 2017.

[7] J. Sen, "A robust analysis and forecasting framework for the Indian mid cap sector using time series decomposition,"Journal of Insurance and Financial Management, vol 3, no 4, pp 1- 32, 2017.

[8] J. Sen and T. Datta Chaudhuri, "Understanding the sectors of Indian economy for portfolio choice," International Jouranl of BusinessForecastingandMarketingIntelligence,vol4,n02,pp178-222, 2018.

[9] J. Sen and T. Datta Chaudhuri, "A robust predictive model for stockprice forecasting," In Proceedings of the 5thInternational Conference on Business Analytics amd Intelligence, Bangalore, India, Dec11-13,2017.

[10] S. Basu, "The relationship between earnings yield, market value and return for NYSE common stocks: further evidence," Journal of Economics, vol 12, no 1, pp. 129 - 156, 1983.

[11] J. Jaffe, D. B. Keim, and R. Wester field, "Earnings yields, market values and stock returns," Journal of Finance, vol 44, pp. 135 - 148, 1989.

[12] B. Rosenberg, K. Reid, and R. Lanstein, "Persuasive evidence of market in efficiency,"Journal of Portfolio Management, vol11, pp.9 -17, 1985.

[13] E. F. Fama and K. R. French, "Size and book-to-market factors in earnings and returns,"Journal of Finance, vol 50, no 1, pp.131 - 155, 1995.

[14] A. Chui and K. Wei, "Book-to-market, firm size, and the turn of the year effect: evidence from Pacific basin emerging markets," Pacific Basin Finance Journal, vol 6, no 3- 4, pp. 275 - 293, 1998.

[15] J. E. Jarrett and E. Kyper, "ARIMA modeling with intervenation to forecast and analyze Chinese stock prices," International Journal of Engineering Business Management, vol 3, no 3, pp. 53 - 58, 2011.

[16] A. Adebiyi, O. Adewumi, and C. K. Ayo, "Stock price prediction using the ARIMA model, "In Proc. Of the International Conference on Computer Modelling and Simulation, Cambridge, UK, pp. 105 - 111, 2020.

[17] P Mondal, L. Shit, and S. Goswami, "Study of effectiveness of time series modeling (ARMA) in forecasting stock prices," International Journal of Computer Science, Engineering and Applications, vol 4, pp.13-29, 2020.

[18] S. Mishra, "The quantile regression approach to analysis of dynamic interaction between exchange rate and stock returns in emerging markets: case of BRIC nations," IUP Journal of Financial Risk Management, vol 13, no 1, pp 7 -27, 2016.

[19] M. Mostafa, "Forecasting stock exchange movements using neural networks: empirical evidence from Kuwait," Expert Systems with Applications, vol 37, pp 6302 - 6309, 2010.

[20] G. Dutta, P. Jha, A. Laha, and N. Mohan, "Artificial neural network models for forecasting stock price index in the Bombay Stock Exchange," Journal of Emerging Market Finance, vol 5, no 3, pp 283-295, 2006.

[21] Q. Wu, Y. Chen, and Z. Liu, "Ensemble model of intelligent paradigms for stock market forecasting," In Proceedings of the IEEE1st International Workshop on Knowledge Discovery and Data Mining, Washington DC, USA, pp 205 - 208.

[22] T. A. Siddiqui and Y. Abdullah, "Developing a nonlinear model to predict stock prices in India: an artificial neural networks approach, "IUP Journal of Applied Finance, vol 21, no 3, pp. 36 - 39, 2015.

[23] M. Jaruszewicz and J. Mandziuk, "One day prediction of Nikkei index considering information from other stock markets," In Proceedings of the International Conference on Artificial Intelligence and Soft Computing, Japan, pp. 1130 - 1135.

[24] Meta stock Website: http://www.metastock.com

[25] V. Zoca, G. Spacagna, D. Slater, and P. Roelants, Python Deep Learning, Packt Publishing Ltd., UK, 2017.