Abstract—Designing robust systems for precise prediction of future prices of stocks has always been considered a very challenging research problem. Even more challenging is to build a system for constructing an optimum portfolio of stocks based on the forecasted future stock prices. We present a deep learning-based regression model built on a long-and-short-term memory network (LSTM) network that automatically scrapes the web and extracts historical stock prices based on a stock's ticker name for a specified pair of start and end dates, and forecasts the future stock prices. We deploy the model on 75 significant stocks chosen from 15 critical sectors of the Indian stock market. For each of the stocks, the model is evaluated for its forecast accuracy. Moreover, the predicted values of the stock prices are used as the basis for investment decisions, and the returns on the investments are computed. Extensive results are presented on the performance of the model. The analysis of the results demonstrates the efficacy and effectiveness of the system and enables us to compare the profitability of the sectors from the point of view of the investors in the stock market.

Keywords—Stock Price Prediction, Regression, Long and Short-Term Memory Network, Multivariate Time Series, Portfolio Management, Huber Loss, Accuracy Score, MAE.

I. INTRODUCTION

Designing robust frameworks for precise prediction of future prices of stocks has always been considered a very challenging research problem. The advocates of the efficient market hypothesis affirm the impossibility of an accurate forecasting of future stock prices. However, propositions demonstrate how complex algorithms and sophisticated and optimally designed predictive models enable one to forecast future stock prices with a high degree of precision. One classical method of predicting stock prices is the decomposition of the stock price time series [1]. Most of the recent works in the literature for forecasting stock prices use approaches based on machine learning and deep learning [2-4]. Bollen et al. contend that emotions profoundly affect an individual's buy or sell decisions [3]. The authors propose a mechanism that computes the public's collective mood from the Twitter feeds and investigate whether the collective moods have any impact on the Dow Jones Industrial Average. The application of convolutional neural networks (CNN) in designing predictive systems for forecasting future stock prices is proposed in some works [4].

Researchers have proposed different approaches to the technical analysis of stocks. Numerous methods have been suggested for dealing with the technical analysis of stock prices. Most of these approaches are based on searching and detecting well-known patterns in the sequence of stock price movement so that the investors may devise profitable strategies upon identifying appropriate patterns in the stock price data. For this purpose, a set of indicators has been recognized for characterizing the stock price patterns.

This paper presents a deep learning-based regression model built on an LSTM network. The system automatically scrapes historical stock prices using its ticker in the NSE and makes a robust prediction of the future values on a daily interval. The model is deployed on 75 stocks chosen from 15 sectors of the Indian stock market, and the return on investment as per the prediction of the model is computed for each stock. The results are analyzed to evaluate the accuracy of forecasting and also for comparing the return on investments for the sectors on which we carry out our study.

The three major contributions of the current work are as follows. First, the work presents a deep learning-based regression model built on an LSTM network that is capable of forecasting future stock prices with a significantly high level of precision. Second, the forecasted output of the LSTM model is used to make investment decisions on 75 different stocks from 15 critical sectors of the stock market of India. The results show the efficacy and effectiveness of the predictive models. Third, the study also enables one to understand the relative profitability of the sectors from the point of view of the investors in the stock market.

We organize the paper as follows. Section II provides a brief discussion on some of the related works. Section III provides a detailed discussion on the data used and the methodology used. Section IV provides the design details of the deep learning regression model built on LSTM. Section V presents extensive results of the performance of the model. Finally, Section VI concludes the paper.

II. RELATED WORK

The literature on the construction of robust portfolio systems of stocks using sophisticated predictive models is quite rich. Researchers have proposed numerous techniques and approaches for precise forecasting of future movements of stock prices. Broadly, these methods are classified into four types. The works of the first category use different types of regression methods, including the ordinary least square (OLS) regression and its other variants like penalty-based regression, polynomial regression etc. [5-7]. The second category of propositions is based on various econometric methods like autoregressive integrated moving average (ARIMA), cointegration, quartile regression etc. [8-10]. The approaches of the third category use machine learning, deep learning, and reinforcement learning algorithms [11-13]. These approaches are based on predictive models built on
complex algorithms and architectures. The works of the fourth category are based on hybrid models built on machine learning and deep learning with inputs of historical stock prices and the sentiments in the news articles on the social web [14-15]. The forecasting performances of these models are found to be the most robust and accurate.

The most difficult challenge in designing a robust and accurate system framework for predicting future stock prices is handling the randomness and the volatility exhibited by the time series. Moreover, the predicted values of stock should be finally used to guide an investor in making a wise investment in the stock market for maximizing the profit out of the investment. Most of the currently existing works in the literature either have not considered this objective or have done it in a somewhat ad-hoc manner. In the current work, we utilize the power of a deep learning model in learning the features of a stock price time series and then making a prediction of its future values with a high level of precision. The predicted stock price is used as a guide for investment in the top five stocks in the fifteen important sectors of the Indian economy. The expected buy and sell profit for each of the stocks is computed, and an overall profit in investment for all the sectors is computed. The work therefore shows the relative profitability of the sectors in addition to demonstrating the effectiveness and efficacy of the predictive model.

III. DATA AND METHODOLOGY

As we have mentioned earlier, our goal is to design a predictive model based on LSTM networks and use the predicted values as a guide to investments in various stocks of different sectors of the Indian economy. The profit earned from the investment will provide us an idea about the robustness and accuracy of the models and will also give us an insight into the comparative idea of the profitability of those sectors. We use the Python programming language for developing our proposed system. The proposed system works on the interaction among five functions. The functions are as follows: (1) load_data, (2) create_model, (3) comb_df (4) plot_graph, and (5) predict. In the following, we describe these in detail.

(1) load_data: This function scrapes the data from the web and carries out all preprocessing before the data are used for building the model. The function uses the following parameters: (i) ticker, (ii) no_steps, (iii) scale, (iv) shuffle, (v) forward_step, (vi) split_by_date, (vii) test_size, and (viii) variables_col. We discuss the role of each parameter in the following.

The parameter ticker refers to the ticker of the stock that we want to load. The yahoo_fin API in Python is used for extracting the historical records of the stock price data from during a start and an end date mentioned in the program. Our proposed model is constructed using the historical stock price records of the National Stock Exchange (NSE) of India. For example, for the Bajaj Auto stock, we use the ticker string, "BAJAJ-AUTO.NS". Every stock listed in the NSE has a unique ticker string, which is used for extracting the data from the web automatically using the yahoo_fin API.

The parameter, no_steps, refers to the number of records used in one round of prediction. The default value of this parameter is set to 50, so that we need to feed in our model a sequence of past 50 records to predict the next value in the series. In other words, 50 past records are used to predict the next day’s stock price. These values are tunable, however, and can be changed based on the requirement.

The parameter, scale, is a Boolean variable that indicates whether to scale the values of the stock prices in the range 0 to 1. By default, this parameter is set to ‘True’ so that the input values are scaled in the interval [0, 1]. The MinMaxScalar function of the sklearn module of Python is used for scaling the columns (i.e., the variables). The scaling is needed since the neural network converges faster when the variables are scaled, and the variables with higher values do not get any opportunity to undesirably dominate the model.

The parameter shuffle is a Boolean variable, which, if set to True, will shuffle the records so that the model does not learn from the sequence of the records in the dataset. The default value of the parameter is True.

The forward_step parameter refers to the future lookup step to predict. The default value is 1, which implies that the stock price for the next day is predicted.

The parameter split by date is a Boolean variable whose value determines whether the records are split into the training and test sets by the date attribute of the stock price records. When set to True, the training and the test datasets will be created based on the date attribute of the records. On the other hand, if it is set to False, the training and the test sets will be created by randomly choosing the records. The default value of the parameter is set to True.

The parameter test_size is a float variable that indicates the size of the test dataset as a fraction of the size of the total dataset size. Hence, if test_size has a value of 0.2, it indicates that 20% of the total number of records are allocated to the test dataset, and accordingly, the training dataset consists of 80% of the total number of records.

The variables_col parameter refers to the list of features of the stock price records used in the model. By default, all the features captured by the yahoo_fin API are used in this list. The features are: adjusted close (adj_close), open, high, low, close, and volume.

In summary, the load_data function extracts the stock records using the stock_info.get_data function in the yahoo_fin API. It then adds the date column from the index if it does not exist in the original data. If the scale argument is set as True, the function scales all the attributes of the stock price in the interval [0, 1] using the MinMaxScalar class of the sklearn module in Python. Further, it adds a new column, future, that indicates the target values to predict by shifting the adj_close column (the target variable) by the value of the parameter, forward_lookup. Finally, the function shuffles and divides the data into the training and the test dataset using the date column and returns the training and the test datasets.

(2) create_model: This function constructs the deep learning model. We will describe the architecture of the model in the next section. The function create_model uses the following arguments: (i) sequence_length, (ii) no_features, (iii) no_units, (iv) cell_type, (v) no_layers, (vi) dropout_rate, (vii) loss, (viii) optimizer, (ix) batch_size, and (x) epochs.

The parameter sequence_length refers to the number of past records used by the model for predicting. As mentioned
earlier, the default length of the input sequence is 50, implying 50 consecutive daily stock price multivariate data are used as the input. In other words, while the value of the parameter sequence_length can be changed based on our requirement, the default value of the parameter is set to 50.

The parameter no_features parameter refers to the number of features present in the input stock price data. We use five features: open, high, low, volume, and adjclose. Among these, the feature adjclose is used as the target variable, while the remaining variables are used as the predictors. Hence, the default value of the argument no_features is set to 5.

The argument, no_units, refers to the number of nodes in each LSTM layer of the model. We will discuss more about this when we will describe the architecture of the model. While the number of LSTM units can be changed based on the complexity of the problem, we use a default value of 256 for this parameter.

The parameter cell_type refers to the type of cells used in the recurrent neural network model. As we create an LSTM model, the cell_type is set to LSTM.

The parameter, no_layers, refers to the number of LSTM layers used in the model. The default value of the parameter is set to 2, implying two LSTM layers are used. However, based on the complexity of the problem, no_layers can also be changed to any desirable value.

The drop_out parameter is used to control the training of the model and regularize it. The use of dropout prevents overfitting of the model. We will discuss this point further in the architecture of the model in Section IV. In simple terms, dropout refers to the fraction of nodes in the LSTM layer that is put off randomly during the training so that the model does not get an opportunity to learn to minutely from the training data. The higher the value used for dropout more likely the model will not be overfitted. However, if the dropout rate is too high, the model may enter into an undesirable state of underfitting. We use a default value of 0.3 for the drop_out parameter, which is considered a standard value of this purpose.

The parameter loss refers to the function used for evaluating the loss exhibited by the model during the training and the validation phase. The loss function can be of multiple types such as Huber loss (HL), mean absolute error (MAE), mean square error (MSE). We use Huber loss as the default value of the loss parameter.

The optimizer parameter is set to Adam. The other possible optimizers are: stochastic gradient descent (SGD), RMSProp, Adadelta, Adagrad, Adamax, Nadam, and Ftrl.

The parameter batchsize refers to the number of records (i.e., the data points) used in one iteration of the training. An optimum value of batchsize minimizes the execution time of an iteration while keep the batchsize as large as possible. We find the optimum value of batchsize as 64 using a gridsearch method. The optimum value of the batchsize is set to 64.

Finally, the parameter, epochs, indicates the number of times the learning algorithm of the model executes over the entire set of records in the training data. Again, using a gridsearch method, we find the optimum value of epochs as 100. Hence, the default value of the parameter is set to 100.

In summary, the function create_model constructs an LSTM model with an input of sequential data of 50 records and predicts the next value in the sequence using a solitary node in its output layer. The learns from the five features in the input data using 256 nodes in the two LSTM layers. It is trained over 100 epochs with a batch size of 64 using the Adam as the optimizer. The Huber loss function is used for evaluating the training and validation process of the model.

(3) comb_df: This function takes two parameters, model and data. The first parameter is the model returned by the create_model function, while the second parameter is the data returned by the load_data function. The function comb_df constructs a Pandas DataFrame object in which it packs the predicted values along with the actual values of stock price. Furthermore, the function perform some important computations using the determination of buy and sell profits from the transactions of a stock. We explain these in the following. The function receives the predicted values of the future stock prices from the return of the function predict. We describe the predict function later in this section.

If the predicted future price (i.e., the predicted adj_close for the next day) exceeds the current price, then the function computes the difference between the actual future price and the current price to compute the buy profit. In other words, if the model predicts a rise in the price the next day, the investors are advised to buy the stock on the current day. The buy profit is, however, computed based on the difference between the actual price of the stock on the next day and the price of the current day. The stock price here refers to the adj_close value, which is the target variable in our study. After computing the buy profits on all possible days in the test cases, the function computes the total buy profit by summing them up.

On the other hand, if the predicted future price is lower than the current price, then to compute the sell profit, the function calculates the difference between the current price and the actual price on the next day. The function computes the total sell profit by summing up all sell profits on the test data points.

The function computes the total profit by adding up the total buy profit with the total sell profit.

Since the number of data samples in the test dataset may not be equal for all stock, we also introduce into this function another task – computation of the values of profit per trade. This value is computed as the ratio of the total profit to the total number of data points in the test dataset.

Additionally, the function also computes a mean absolute score (MAS) and an accuracy score (AS). MAS depicts an average value of error in prediction. A value of 15 indicates that, on average, the predicted values deviate from the actual values by a unit of 15. AS, on the other hand, expresses the fraction of test cases in which the buy or sell profit is positive. If AS for a stock is 0.98, it implies that 98% of the trades on the test data points yielded profit from either sell or buy transactions.

It is important to note that in computing the buy and sell profits, it is assumed that the investor will trade on all the days in the test dataset. If the forecasted price of the following day exceeds the price of the current day, then the investor will buy the stock; else, there will be a sell. The computations are shown on the basis of buy/sell of one stock.
only. However, if $X$ number of shares of a stock are transacted, the buy and sell profits will just get multiplied by the number of shares transacted. In summary, the $\text{comb\_df}$ function uses two arguments- one is the model created by the $\text{create\_model}$ function, and the other is the data dictionary created by the $\text{load\_data}$ function. It returns a DataFrame containing the features of the input data samples together with the actual and the forecasted prices of the stock of the test dataset points.

(4) plot_graph: This function takes the DataFrame returned by the function $\text{comb\_df}$, and plots the actual and the predicted prices (i.e., the $\text{adj\_close}$ values) on the same graph using the $\text{pyplot}$ function defined in the $\text{matplotlib}$ module of Python.

(5) predict: This function receives two parameters – model and data. The former parameter is the return of the create_model function, while the latter is the return of the load_data function. The function computes the predicted values for all the data points in the test dataset and returns a DataFrame containing the predicted values of the stock prices.

IV. THE MODEL ARCHITECTURE

In the previous section, we describe in detail the five functions constituting our predictive framework, including the function responsible for data extraction and preprocessing. In this section, we will dissect the architecture and further details of the model design. Before discussing the details of the model architecture, we discuss the working principles of LSTM networks and their suitability in handling sequential data such as time series of historical stock price data very briefly.

LSTM is an adaptation of a recurrent neural network (RNN) that is capable of interpreting and then forecasting sequential data like time series and text [12]. The networks are capable of maintaining their state information in their designated memory cells which are called gates. The state information stored in the memory cells is aggregated with the past information available at the forget gates. The input gates receive the currently available information, and using the information at the forget gates and the input gates, the network computes the predicted value for the next time slot and makes it available at the output gates [12].

Fig. 1 shows the design of the proposed LSTM model. The model uses daily stock records of the past 50 days as its input, and each record has five features. This is indicated by the input data shape of (50, 5). The input data is passed on the first LSTM layer consisting of 256 nodes. Hence, the output of the LSTM layer has a shape of (50, 256). This indicates that each of the 256 LSTM nodes processes the 50 records from the input and extracts a total of 256 features from each record. The first LSTM layer is followed by a dropout layer that randomly puts off 30% of the nodes at the first LSTM layer in order to prevent any overfitting of the model. A second LSTM layer follows that receives the output from the first LSTM layer and further extracts 256 features from the data. The second LSTM layer is also controlled by a dropout layer with a 30% dropout rate. The output of the second LSTM layer is passed on finally to a dense layer having 256 nodes at its input and one node at the output. The single node at the output of the dense layer produces the final output of the model as the forecasted value of adjusted close for the next day. The training and validation of the model is carried out over 100 epochs using a batch-size of 64. The activation function used at all layers in the model is the rectified linear unit (ReLU) except for the final output layer, in which the sigmoid activation function is used. The Huber loss function is used to compute the loss, and the mean absolute error (MAE) is used for computing the error. The performance results of the model presented in Section IV are achieved using these parameters. However, alternative choices can be made, and the performance of the model can be studied further. The values of the parameters, epoch and batchsize, are, however, determined using a gridsearch method and should not be changed.

We use the Huber loss function as it effectively acts as a combination of the mean squared error (MSE) and the mean absolute error (MAE) [16]. The Huber loss function is quadratic when the error is smaller than a threshold value, while it behaves in a linear way when the error is larger than the threshold. When the function is linear, it is less sensitive to the outliers than the MSE. On the other hand, the quadratic part allows it to converge faster and yield more accurate results than the MAE.

V. EXPERIMENTAL RESULTS

This section presents extensive experimental results on the performance of the predictive model. We apply the model to forecast the $\text{adj\_close}$ values of stock price for the next day. Based on the forecasted price and the current price of the stock, the predictive model first determines whether a buy or a sell strategy will be recommended to the investor. While the buy profit is computed as the difference between the current price and the actual price of the stock on the next day, the sell profit is determined by calculating the difference between the actual stock price on the next day and the current price of the stock. It is assumed that trading (either buy or sell) is carried out on all the days in the test data. Total profit is computed as the sum of the total buy profit and the total sell profit. Since stock prices widely vary for different companies, we compute the ratio of the total profit to the mean stock price for the data points in the test case. The value of the ratio gives a robust measure of the profitability measure of the investment in the stock. The predictive system also computes the total number of sample points in the test set for each stock and computes the ratio of the total profit to the number of sample points in the test.
dataset. The value of this ratio is a measure of the profit earned per trade (i.e., buy or sell transaction) for the stock.

We choose fifteen sectors of the Indian economy. These fifteen sectors are: (i) auto, (ii) banking, (iii) consumer durables, (iv) capital goods, (v) fast-moving consumer goods (FMCG), (vi) healthcare, (vii) information technology (IT), (viii) large-cap, (ix) metal, (x) mid-cap, (xi) oil and gas, (xii) power, (xiii) realty, (xiv) small-cap, and (xv) telecom. For each of these sectors, we identify the top five stocks which have the most significant impact on the index of the corresponding sector. The predictive model is deployed to compute the profit in investment in the stocks. Finally, the average of the ratio of the total profit to the mean value of the stock over the test period is computed. This metric is used as a measure of the profit (i.e., the return) of investment for a sector. The models are implemented using Python 3.7.4 on TensorFlow 2.3.0 and Keras 2.4.5 frameworks. The epochs are run on the Google Colab GPU runtime environment. Each epoch approximately took 1 second to execute. In the following, we present the results of all the sectors. It may be noted that the experiments were carried out from March 3 to March 5, 2021. Hence, the stock price on the next day refers to prices from March 4 to March 6.

Auto sector: The five significant auto sector stocks listed in NSE are: (i) Maruti Suzuki India (MSU), (ii) Mahindra and Mahindra (MMH), (iii) Tata Motors (TMO), (iv) Bajaj Auto (BAJ), and (v) Hero MotoCorp (HMC). The weights in percent values for these stocks for computing the overall index of the auto sector index in NSE are as follows: MSU-18.88, MMH-15.97, TMO-11.97, BAJ-10.23, HMC-8.66 [17]. For all these stocks, we deploy our proposed predictive framework to compute the following: predicted price on the next day, Huber loss, mean absolute error, accuracy score, total buy profit, total sell profit, total profit, mean stock price over the test period, the ratio of the total profit to the mean stock price, the number of test point samples, the profit per trade computed as the total profit per test sample points. Finally, the average of the ratio of the total profit to the mean stock price is computed for the five stocks to arrive at a measure of the profitability of investment for the overall sector. Table I presents the results for the auto sector stocks. As an example, Fig. 2 depicts the convergence of the Huber loss in training and validation for the Maruti Suzuki stock.

![Fig. 2. The Huber loss convergence for the stock data of Maruti Suzuki. The x-axis plots the number of epochs, and the y-axis, the Huber loss](image)

**Banking sector:** The five critical stocks in this sector as per the listing in the NSE and their corresponding weights in percent are: HDFC Bank (HDB)-26.06, ICICI Bank (ICB)-19.56, Axis Bank (AXB)-15.93, State Bank of India (SBI)-13.27, and Kotak Mahindra Bank (KTB)-12.37. Table II presents the results for the banking sector stocks. As an example, the plot of the predicted and the actual price for the State Bank of India stock is presented in Fig. 3.

![Fig. 3. The actual and the predicted price of the SBI stock over the entire training and test periods – the output of the plot_graph function](image)

**Capital Goods sector:** The top five stocks in this sector as per the listing in the NSE are: ABB, Bharat Electronics (BEL), Havells India (HVL), Larsen and Toubro (LNT), and Siemens India (SIM). Table III presents the results for the capital goods sector stocks. As an example, in Fig. 4, we show the convergence of the Huber loss in the training and validation for the ABB stock.

**TABLE I. THE RESULTS OF THE AUTO SECTOR STOCKS**

| Metric       | BAJ   | HMC   | MMH   | MSU   | TMO   |
|--------------|-------|-------|-------|-------|-------|
| Next day’s pred. price | 4149  | 3362  | 843   | 7446  | 333   |
| Huber loss   | 0.00042 | 0.00028 | 0.00020 | 0.00020 | 0.00020 |
| Mean absolute error | 1.0974 | 1.5590 | 1.6435 | 225.90 | 17.27 |
| Accuracy score | 0.9584 | 0.9770 | 0.9792 | 0.9857 | 0.9816 |
| Total buy profit | 103415 | 912200 | 181900 | 1164205 | 102315 |
| Total sell profit | 298601 | 509230 | 181077 | 1178980 | 101984 |
| Total profit | 1062016 | 1012458 | 363067 | 2343165 | 204256 |
| Mean stock price | 1507 | 1484 | 1344 | 877 | 1260 |
| Profit per trade | 767 | 688 | 1344 | 877 | 1260 |
| Number of test cases | 911 | 912 | 1282 | 858 | 1253 |
| Avg. profit/mean price | 0.987 | 0.948 | 0.987 | 0.977 | 0.980 |

**TABLE II. THE RESULTS OF THE BANKING SECTOR STOCKS**

| Metric       | AXB   | HDB   | ICB   | KTB   | SBI   |
|--------------|-------|-------|-------|-------|-------|
| Next day’s pred. price | 1481  | 135  | 1178  | 1554  | 1844  |
| Huber loss   | 0.00089 | 0.00024 | 0.00012 | 0.00059 | 0.00026 |
| Mean absolute error | 1.0794 | 1.9872 | 1.9792 | 1.9872 | 1.9872 |
| Accuracy score | 0.9671 | 0.9822 | 0.9748 | 0.9757 | 0.9817 |
| Total buy profit | 259098 | 20596 | 115837 | 260986 | 284522 |
| Total sell profit | 259098 | 20596 | 115837 | 260986 | 284522 |
| Total profit | 518197 | 411320 | 231670 | 527970 | 570507 |
| Mean stock price | 729  | 56  | 202 | 614 | 368 |
| Profit per trade | 1099 | 941 | 1089 | 1099 | 1099 |
| Number of test cases | 911 | 912 | 913 | 911 | 1095 |
| Profit per trade | 551.74 | 45.42 | 248.09 | 440.84 | 321.01 |
| Avg. profit/mean price | 0.942 | 0.9 | 0.942 | 0.942 | 0.942 |

**TABLE III. THE RESULTS OF THE CAPITAL GOODS SECTOR STOCKS**

| Metric       | ABB   | BEL   | HVL   | LNT   | SIM   |
|--------------|-------|-------|-------|-------|-------|
| Next day’s pred. price | 0.00015 | 0.00012 | 0.00018 | 0.00013 | 0.00022 |
| Huber loss   | 0.2790 | 1.101 | 13.11 | 10.48 | 41.29 |
| Mean absolute error | 0.9844 | 0.9797 | 0.9861 | 0.9861 | 0.9861 |
| Accuracy score | 0.9999 | 0.9999 | 0.9999 | 0.9999 | 0.9999 |
| Total buy profit | 201196 | 113857 | 212626 | 286242 | 295898 |
| Total sell profit | 109312 | 112663 | 210254 | 308077 | 297410 |
| Total profit | 310508 | 226501 | 422876 | 594319 | 593388 |
| Mean stock price | 398 | 202 | 236 | 207 | 502 |
| Total profit/ Mean price | 1089 | 1121 | 1792 | 1011 | 1182 |
| Number of test cases | 1092 | 913 | 1253 | 914 | 907 |
| Profit per trade | 440.16 | 248.09 | 337.49 | 228.94 | 654.14 |
| Avg. profit/mean price | 0.912 | 0.9 | 0.912 | 0.9 | 0.912 |
**Consumer Durable sector:** The top five stocks in this sector as per the listing in the NSE are: Titan Company (TIT), Havells India (HVL), Voltas (VOL), Whirlpool India (WHR), and Bata India (BAT). Table IV presents the results for the consumer durable goods sector stocks. As an example, in Fig. 5, we show the plot of the predicted and the actual values of the stock price of Havells India.

![Fig. 5. The actual and the predicted price of the Havells India stock over the entire training and test periods – the output of the plot_graph function](image)

**FMCG sector:** The critically important stocks of this sector and their corresponding percentage weights in NSE are: Hindustan Unilever (HUL)-26.80, ITC (ITC)-25.08, Nestle India (NST)-8.09, Britannia Industries (BRT)-6.22, and Dabur India (DBR)-4.46. Table V presents the results for the FMCG sector stocks. The plot of the convergence of the Huber loss in the training and validation for the HUL stock is depicted in Fig. 6.

![Fig. 6. The training and the validation for the stock data of HUL. The x-axis plots the number of epochs, and the y-axis, the Huber loss](image)

**TABLE V. THE RESULTS OF THE FMCG SECTOR STOCKS**

| Metric          | BRT | DBR | HUL | ITC | NST |
|-----------------|-----|-----|-----|-----|-----|
| Next day’s pred. price | 5846 | 519 | 2233 | 225 | 16929 |
| Huber loss      | 0.00011 | 0.0015 | 0.0069 | 0.00018 | 0.00012 |
| Mean absolute error | 3.292 | 10.06 | 55.77 | 6.27 | 566.77 |
| Accuracy score  | 0.9702 | 0.902 | 0.9365 | 0.9808 | 0.8330 |
| Total buy profit | 524253 | 91694 | 124461 | 6250 | 1368113 |
| Total sell profit | 374469 | 70839 | 336792 | 62254 | 1787462 |
| Total profit    | 1148769 | 147326 | 617053 | 125834 | 373537 |
| Mean stock price | 969 | 819 | 4199 | 96 | 357 |
| Total profit Mean price | 1776 | 932 | 1422 | 1311 | 973 |
| Number of test cases | 1249 | 910 | 1250 | 1249 | 905 |
| Profit per trade | 919.93 | 155.74 | 336.84 | 10035 | 4109.70 |
| Avg. profit/mean price | 1283 |

**Healthcare sector:** The five most critical stocks of listed in the NSE and their corresponding percentage weights are: Sun Pharmaceutical Industries (SNP)-15.99, Dr. Reddy's Laboratories (DRL)-13.38, Divi's Laboratories (DVL)-10.67, Cipla (CPL)-9.96, and Aurobindo Pharma (ARP)-5.99. Table VI presents the results of this sector. Fig. 7 depicts the predicted and the actual price of the stock of Dr. Reddy's Lab.

![Fig. 7. The actual and the predicted price of the stock of Dr. Reddy's Lab over the entire training and test periods – the output of plot_graph function](image)

**TABLE VI. THE RESULTS OF THE HEALTHCARE SECTOR STOCKS**

| Metric          | ARP | CIP | DVL | DRL | SNP |
|-----------------|-----|-----|-----|-----|-----|
| Next day’s pred. price | 5892 | 781 | 1458 | 4422 | 601 |
| Huber loss      | 0.00223 | 0.00184 | 0.00082 | 0.00117 | 0.00122 |
| Mean absolute error | 15.30 | 14.18 | 16.82 | 18.26 | 16.70 |
| Accuracy score  | 0.9975 | 0.9975 | 0.9967 | 0.9938 | 0.9905 |
| Total buy profit | 170308 | 165322 | 2852660 | 899727 | 176762 |
| Total sell profit | 146041 | 124156 | 325494 | 803884 | 130574 |
| Total profit    | 316249 | 310764 | 535774 | 1631610 | 349338 |
| Mean stock price | 214 | 268 | 635 | 1282 | 235 |
| Total profit Mean price | 1581 | 1159 | 883 | 1257 | 1489 |
| Number of test cases | 1246 | 1248 | 1577 | 1251 | 1255 |
| Profit per trade | 271.47 | 248.96 | 137.12 | 1288 | 279.20 |
| Avg. profit/mean price | 1274 |

**IT sector:** The most significant stocks of this sector and with their corresponding percentage weights are: Infosys (IFY)-26.42, Tata Consultancy Services (TCS)-25.92, Wipro (WIP)-9.84, Info Edge India (INE)-9.29, and Tech Mahindra (TEM)-8.89. Table VII presents the results for the IT sector stocks. The plot of the convergence of the Huber loss in the training and validation for the TCS stock is shown in Fig. 8.

![Fig. 8. The training and the validation for the stock data of TCS. The x-axis plots the number of epochs, and the y-axis, the Huber loss](image)

**TABLE VII. THE RESULTS OF THE IT SECTOR STOCKS**

| Metric          | INE | IFY | TCS | TEM | WIP |
|-----------------|-----|-----|-----|-----|-----|
| Next day’s pred. price | 5909 | 1113 | 3244 | 941 | 460 |
| Huber loss      | 0.00011 | 0.00007 | 0.00010 | 0.00021 | 0.00016 |
| Mean absolute error | 111.84 | 29.70 | 111.79 | 97.24 | 27.91 |
| Accuracy score  | 0.9811 | 0.9800 | 0.9774 | 0.9743 | 0.9816 |
| Total buy profit | 281802 | 160038 | 287602 | 83785 | 58701 |
| Total sell profit | 299230 | 156409 | 287399 | 810150 | 58560 |
| Total profit    | 561032 | 116547 | 576991 | 164797 | 117224 |
| Mean stock price | 1202 | 1451 | 791 | 645 | 115 |
| Total profit Mean price | 843 | 1311 | 720 | 425 | 1029 |
| Number of test cases | 609 | 1250 | 799 | 700 | 1259 |
| Profit per trade | 815.29 | 254.83 | 314.04 | 355.42 | 938.21 |
| Avg. profit/mean price | 837 |

**Large-Cap sector:** The most significant stocks of this sector in the NSE are: Reliance Industries (RIL), Tata Consultancy Services (TCS), HDFC Bank (HDB), ICICI Bank (ICB), and the Kotak Mahindra Bank (KTB). Table VIII presents the results for the large-cap sector stocks. The plot of the predicted and the actual price for the stock of Reliance Industries is presented in Fig. 9.

![Fig. 9. The actual and the predicted adjusted close price of the stock of Reliance Industries over the entire training and test periods](image)
### TABLE VIII. THE RESULTS OF THE LARGE CAP SECTOR STOCKS

| Metric | HDI | ICB | KTB | RIL | TCS |
|--------|-----|-----|-----|-----|-----|
| Next day’s pred. price | 1.579 | 0.820 | 1.174 | 0.055 | 2.264 |
| Huber loss | 1.4580 | 0.9205 | 0.0012 | 0.0089 | 0.0010 |
| Mean absolute error | 10.78 | 23.14 | 10.71 | 25.79 | 117.87 |
| Accuracy score | 0.9808 | 0.9700 | 0.9792 | 0.9720 | 0.8754 |
| Total buy profit | 2264464 | 57076 | 241635 | 261257 | 287062 |
| Total sell profit | 285276 | 57430 | 238310 | 261771 | 282759 |
| Total profit | 45052 | 114320 | 480163 | 510108 | 569901 |
| Mean stock price | 386 | 178 | 441 | 392 | 991 |
| Total profit/ Mean price | 1.391 | 1.041 | 1.095 | 1.15 | 2.21 |
| Number of test cases | 1250 | 910 | 964 | 1251 | 798 |
| Profit per trade | 364.02 | 125.85 | 49.09 | 424.38 | 714.04 |
| Avg. profit/mean price | 10.8 |

**Metal sector:** The five most significant stocks of this sector in the NSE and their corresponding weights are: Tata Steel (TSL)-22.47, Hindalco Industries (HIND)-20.68, JSW Steel (JSW)-15.92, Coal India (CIL)-13.28, and Jindal Steel and Power (JIN)-5.71. Table IX presents the results for the metal sector stocks. Fig. 10 shows the plot of the Huber loss in the training and validation for the Tata Steel stock. 

### TABLE IX. THE RESULTS OF THE METAL SECTOR STOCKS

| Metric | CIL | HIND | JIN | JSW | TSL |
|--------|-----|------|-----|-----|-----|
| Next day’s pred. price | 140 | 300 | 70 | 200 | 700 |
| Huber loss | 0.0010 | 0.0001 | 0.0010 | 0.0001 | 0.0010 |
| Mean absolute error | 11.98 | 32.09 | 14.41 | 13.71 | 51.18 |
| Accuracy score | 0.9689 | 0.9512 | 0.9511 | 0.9604 | 0.9679 |
| Total buy profit | 3909 | 40389 | 11245 | 39809 | 12985 |
| Total sell profit | 9074 | 39947 | 110635 | 40126 | 132074 |
| Total profit | 17155 | 80336 | 223108 | 79935 | 258259 |
| Mean stock price | 198 | 109 | 186 | 107 | 256 |
| Total profit/ Mean price | 92 | 73 | 119 | 74 | 103 |
| Number of test cases | 402 | 125 | 1058 | 609 | 1248 |
| Profit per trade | 36.82 | 64.17 | 210.88 | 91.98 | 298.85 |
| Avg. profit/mean price | 760 |

**Mid-Cap sector:** The critically important stocks in this sector listed in the NSE are: Adani Enterprise (ADE), Berger Paints (BRP), Biocon (BIO), Info Edge India (INE), and Torrent Pharmaceuticals (TRP). Table X presents the results for the mid-cap sector stocks. The plot of the predicted and the actual price for the stock of Adani Enterprises is presented in Fig. 11.

### TABLE X. THE RESULTS OF THE MID-CAP SECTOR STOCKS

| Metric | ADE | BRP | BIO | INE | TRP |
|--------|-----|-----|-----|-----|-----|
| Next day’s pred. price | 292 | 906 | 195 | 5000 | 2438 |
| Huber loss | 0.0007 | 0.0009 | 0.0009 | 0.0068 | 0.0001 |
| Mean absolute error | 5.47 | 5.48 | 17.94 | 133.84 | 41.11 |
| Accuracy score | 0.9485 | 0.9702 | 0.9781 | 0.9811 | 0.9906 |
| Total buy profit | 21173 | 70551 | 42780 | 279500 | 340578 |
| Total sell profit | 28117 | 70551 | 42780 | 279500 | 340578 |
| Total profit | 59807 | 140257 | 85742 | 361732 | 679136 |
| Mean stock price | 573 | 1199 | 105 | 332 | 6246 |
| Total profit/ Mean price | 1014 | 1199 | 817 | 683 | 1085 |
| Number of test cases | 931 | 931 | 691 | 740 | 142 |
| Profit per trade | 65.5 | 153.02 | 104.44 | 815.29 | 344 |
| Avg. profit/mean price | 900 |

**Oil and Gas Sector:** The stock of significance in this sector and their corresponding percentage weights are: Reliance Industries (RIL)-30.92, Oil and Natural Gas Corporation (ONG)-11.98, Bharat Petroleum Corporation (BPL)-10.68, GAIL India (GAL)-7.76, and Indian Oil Corporation (IOC)-7.38. Table XI presents the results for the metal sector stocks. Fig. 12 shows the plot of the Huber loss in the training and validation for the Oil and Natural Gas Corporation stock.

### TABLE XI. THE RESULTS OF THE OIL AND GAS SECTOR STOCKS

| Metric | BPL | GAL | IOC | ONG | RIL |
|--------|-----|-----|-----|-----|-----|
| Next day’s pred. price | 0.0021 | 0.0023 | 0.0021 | 0.0024 | 0.0008 |
| Huber loss | 0.977 | 0.979 | 0.975 | 0.9720 |
| Mean absolute error | 0.977 | 0.973 | 0.975 | 0.9720 |
| Accuracy score | 0.977 | 0.973 | 0.975 | 0.9720 |
| Total buy profit | 7877 | 31147 | 27885 | 41974 | 269127 |
| Total sell profit | 7910 | 30949 | 27885 | 41974 | 269127 |
| Total profit | 15787 | 62127 | 55154 | 83579 | 530898 |
| Mean stock price | 109 | 61 | 48 | 83 | 392 |
| Total profit/ Mean price | 1448 | 1019 | 1149 | 1005 | 1354 |
| Number of test cases | 1251 | 1186 | 1256 | 1247 | 1251 |
| Profit per trade | 126.20 | 52.39 | 88.82 | 66.78 | 424.38 |
| Avg. profit/mean price | 1199 |

### TABLE XII. THE RESULTS OF THE POWER SECTOR STOCKS

| Metric | JSE | NTP | PWG | SJP | TPW |
|--------|-----|-----|-----|-----|-----|
| Next day’s pred. price | 23 | 163 | 218 | 25 | 369 |
| Huber loss | 0.0017 | 0.0007 | 0.0023 | 0.0074 | 0.0048 |
| Mean absolute error | 0.34 | 0.40 | 0.65 | 17.37 | 13.13 |
| Accuracy score | 0.9405 | 0.9993 | 0.9628 | 0.9538 | 0.9626 |
| Total buy profit | 405 | 405 | 405 | 405 | 270 |
| Total sell profit | 405 | 405 | 405 | 405 | 270 |
| Total profit | 810 | 810 | 810 | 810 | 540 |
| Mean stock price | 316 | 316 | 316 | 316 | 316 |
| Total profit/ Mean price | 156 | 166 | 259 | 215 | 314 |
| Number of test cases | 328 | 791 | 645 | 320 | 688 |
| Profit per trade | 18.56 | 20.41 | 40.50 | 5.66 | 80.31 |
| Avg. profit/mean price | 214 |

**Power sector:** The stocks listed in the NSE which are of critical importance in this sector are: National Thermal Power Corporation (NTP), JSW Energy (JSE), Power Grid Corporation of India (PWG), Torrent Power (TPW), and...
SJVN (SJN). Table XII presents the results for the power sector stocks. Fig. 13 shows the plot of the Huber loss in the training and validation for the stock of Power Grid Corporation of India stock.

**Small-Cap sector:** Some of the critical small-cap stocks listed in the NSE are: Ipec Lab (IPC), Mindtree (MIN), SRF, Tata Communications (TCM), and Trent (TRN). Table XIV presents the results for the small-cap sector.

**Telecommunication sector:** Some important telecom sector stocks listed in the NSE are: Bharti Airtel (BHA), Tata Communications (TCM), Finolex Cables FICL), Honeywell Automation (HNA), and Vodafone Idea (VDI). Table XV presents the results for the stocks of the telecom sector.

**Summary:** Considering the metric average of the ratio of total profit to the mean value of the stock for the five stocks as the metric for overall profitability of a sector, we observe that the FMCG sector turns out to be the most profitable sector, while the power sector is the least profitable one.

**VI. CONCLUSION**

In this paper, we have proposed a deep learning regression model built of an LSTM architecture. The model is regularized using two dropout layers and has the ability to automatically learn the composition of the data for extracting historical stock price data using the yahoo_fin API. We deployed the model for analyzing 75 critical stocks from 15 sectors of the Indian stock market. An extensive set of results is analyzed on the performance of the model that demonstrates the efficacy and effectiveness of the model. The results also enable us to compare the profitability of different sectors.

**REFERENCES**

[1] J. Sen, "A study of the Indian metal sector using time series decomposition-based approach", Int. J. Basari et al. (eds), Selected Studies of Economics and Finance, pp. 105-152, Cambridge Scholars Publishing, UK, 2018.

[2] J. Sen, "Stock price prediction using machine learning and deep learning frameworks", Proc. of the 6th International Conference on Business Analytics and Intelligence, Dec, 2018, Bangalore, India.

[3] J. Bollen, H. Mao, and X. Zeng, "Twitter mood predicts the stock market", Journal of Comp. Science, vol. 2, pp. 7046-7056, 2011.

[4] S. Mehtab and J. Sen, "Stock price prediction using convolutional neural network on a multivariate time series", Proc. of the 3rd National Conference on Machine Learning and Artificial Intelligence (NCMLAF'20), Feb, 2020, New Delhi, India.

[5] 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 sectors”, Jour. of Eco. Lib., vol. 3, no. 2, pp. 303-326, 2016.

[6] X. Zhong and D. Enke, "Forecasting daily stock market return using dimensionality reduction", Expert System with Application, vol. 97, pp. 60-69, 2017.

[7] S. S. Roy, D. Mittal, A. Basu, and A. Abraham, “Stock market forecasting using LASO linear regression model”, Proc. of Afro-European Conf. of Int. Applications, pp. 771-181, 2015.

[8] Y. Ning, L. C. Wah, and L. Erda, “Stock price prediction based on error correction model and Granger causality test”, Cluster Computing, vol. 22, pp. 4849-4858, 2019.

[9] L. Wang, F. Ma, J. Liu, and L. Yang, “Forecasting stock price volatility: New evidence from the GARCH-MIDAS model”, Int. Journal of Forecasting, vol. 36, no. 2, pp. 684-694, 2020.

[10] Y. Du, “Application and analysis of forecasting stock price index based on combination of ARIMA model and BP neural network”, Proc. of CCDC, pp. 2854-2857, Jun 9-10, Shenyang, China, 2018.

[11] S. Mehtab and J. Sen, “Stock price prediction using CNN and LSTM-based deep learning models”, Proc. of Int. Conf. on Decision Aid Sc. and Appl. (DASA), pp. 447-453, Nov 8-9, 2020, Bahrain.

[12] S. Mehtab, J. Sen, and D. Dutta, “Stock price prediction using machine learning and LSTM-based deep learning models”, In: Thampi, S. M. et al. (eds.) Machine Learning and Metaheuristics Algorithms and Applications (SOMMA’20), pp. 88-106, vol 1386, Springer, Singapore.

[13] B. Yang, Z-J. Gong, and W. Yang, “Stock market index prediction using deep neural network ensemble”, Proc. of IEEE CCC, pp. 3382-3887, Jul 26-28, Dalian, China, 2017.

[14] S. Mehtab and J. Sen, “A robust predictive model for stock price prediction using deep learning and natural language processing”, Proc. of ICBAI, Dec 5-7, Bangalore, India, 2019.

[15] K. Nam and N. Seong, "Financial news-based stock movement prediction using causality analysis of influence in the Korean stock market", Decision Support Systems, vol. 117, pp. 100-112, 2019.

[16] A. Geron. Hands-On Machine Learning with Scikit-Learn, Keras, and TensorFlow, 2nd Edition, OReilly Media Inc, USA, 2019.

[17] NSE Website: http://www1.nseindia.com