Stock Market Prediction Using LSTMs

**Abstract:**

The trend of investing in stocks of different companies is increasing day by day. But it is very difficult to predict the stock price of any asset as a lot of factors affect it and some of the factors are natural and unknown. With every passing day interest in Machine Learning and Artificial Intelligence is increasing and they have also been useful for human beings as Artificial Intelligence has taken over a lot of work. This Paper tends to use LSTM stacked Recurrent Neural Networks to create a model for predicting stock market prices and analyzing the model and its accuracy.

**Keywords:** Stock market Prediction; RNN; LSTM; Forecasting; multivariate;

**Introduction:**

**Related Work:**

**Theoretical Overview:**

RNNs have been used for sequence prediction problems for a long time. RNN works on the principle of saving the output of a particular layer and feeding this back to the input so as to predict the output of the layer. They solved a lot of issues with feed-forward neural networks like:

1. RNN can handle sequential data while feed-forward neural networks can’t.

2. RNN can memorize previous inputs while feed-forward neural networks can’t.

But sometimes RNN also runs into the problem of Vanishing Gradient. Vanishing Gradient refers to the issue when the gradient becomes very small and the updates in parameters become insignificantly small. Due to this, learning long sequence data becomes very hard.

Wnew = Wold – learning\_rate \* gradient

Where: Wnew : new weight after updation

Wold : old weight

Also, RNN cannot store long-term memory. It has a very short-term memory due to which it cannot carry forward previous data to further layers.

To solve these problems Long-Short-Term-Memory (LSTM) was introduced. It is a special kind of RNN. LSTM contains both cell state(C) and hidden state (h). A Cell State carries the previous long-term information while Hidden State carries information from immediately previous events. Output at any level depends on the current cell state, hidden state, and the input data at the current level. It consists of three gates namely: Forget Gate, Input Gate, and Output Gate.

1. Forget Gate: This is the first step in a cell in LSTM. It decides which information of Long term memory should have less weight (can be forgotten) given the previous hidden state and new input data. It outputs a number between 0 and 1(sigmoid) where 0 shows completely forget this information and 1 shows keep this information. If we want to keep any information then it is multiplied by a number closer to 1(more weight) otherwise, it is multiplied by a number closer to 0(lesser weight).

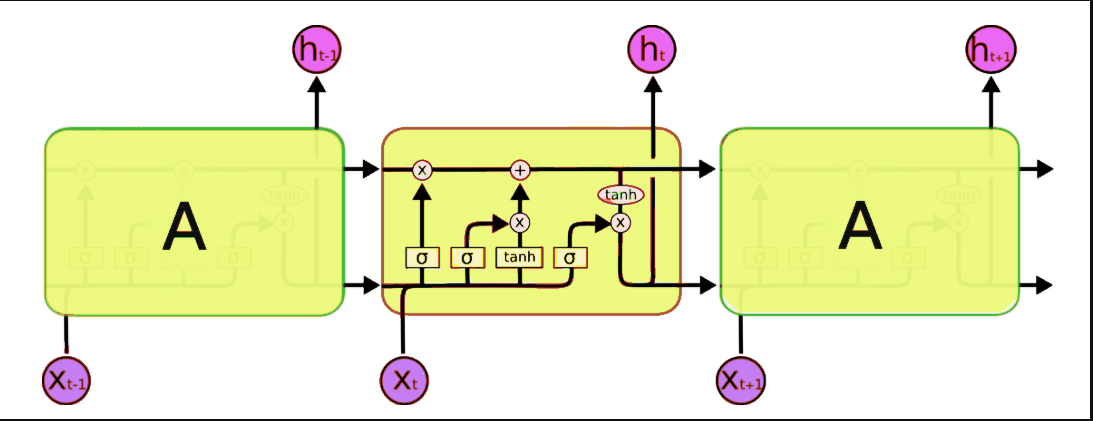
2. Input Gate: It is used to find the importance of the new information carried by the input. Firstly information is passed through the sigmoid activated layer (output between 0 and 1) and then passed through the tanh activated layer (output between -1 and 1). Negative values mean that this information is not going to be used in the cell state at the current timestamp and positive values will be used.

3. Output Gate: It decides the output and the new hidden state. It passes the current cell state through tanh activation and then passes the previously hidden layer and current input through a sigmoid activated layer. Then output the new hidden state.

The following image shows an LSTM network.

ht: Hidden state at any timestamp t.

Xt: Cell state at any timestamp t.



**Data and Work Done:**

The data has been taken from yahoo finance for Wipro Limited (WIPRO.NS). The work has been done for the time period 1/1/2005-17/4/2022. The data from 1/1/2005-31/12/2017 has been taken for training and testing has been done from 1/1/2018-17/4/2022.

To build this model, the following features have been used:

1. Open: Opening price on any particular day.

2. High: Highest price on that particular day.

3. Low: Lowest price on that particular day.

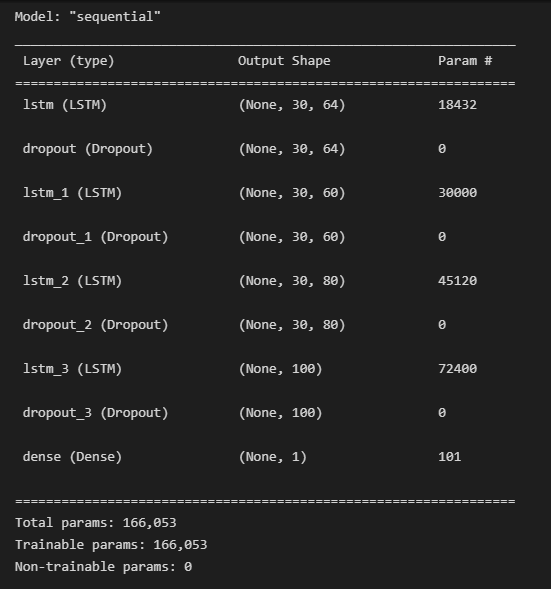
4. Close: Closing price on that particular day.

5. Adjusted Close: Adjusted closing price on that particular day.

6. Volume: Number of trades performed on that particular day.

7. Dollar: Price of US Dollar in INR.

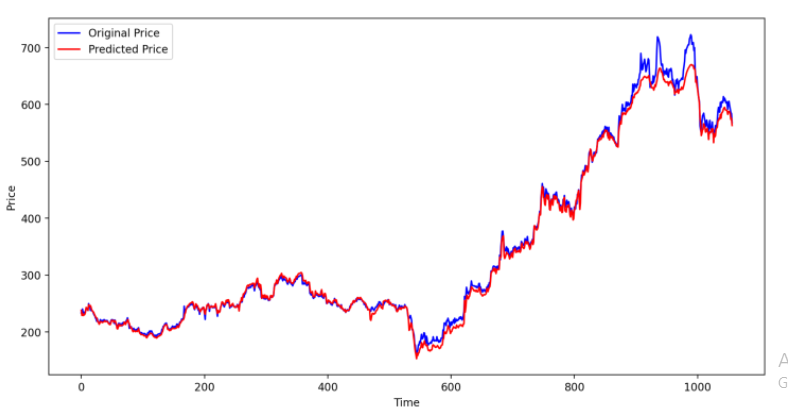
100 epochs have been used to train the model. The model has been trained on 3207 data points (days) and Tested on 1057 data points (days).The following image shows the model summary.



**Result and Analysis:**

After testing, this model showed the following results:

The below image shows a plot between the original and the predicted price.



In this figure, the blue line shows the real market price while the red line shows the predicted market price. Our Model was performing nicely continuously but around the 900th day of testing the stock price changed drastically due to which there was a big deviation between the predicted and original price.

The below table shows the different errors that are calculated to analyze the model.

|  |  |
| --- | --- |
| **Error** | **Value** |
| Mean Absolute Error | 6.833 |
| Root Mean Square Error | 10.790 |
| Normalized Root Mean Square Error | 0.019 |

Since it is difficult to define the accuracy in a regression problem, the accuracy in this case has been calculated by using the following method:

Let us say y\_pred is the predicted value and y\_test is the actual ground truth value on a particular day. Then Absolute error is calculated by using the following formula:

abs\_error = |y\_test – y\_pred|

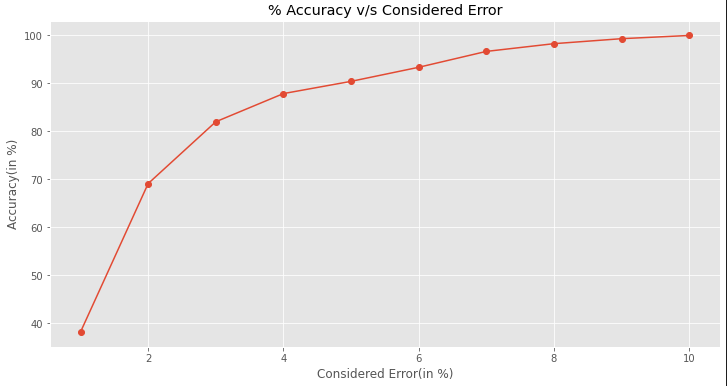
% abs\_error = (abs\_error/y\_test)\*100

Absolute % error has been calculated for each testing data. Then, we define a considered % error value (threshold). If that absolute % error is less than that considered % error then we consider that prediction as correct and if that %absolute error is more than %considered error then we classify that prediction as wrong. Now according to the number of correct predictions the accuracy of our model has been found.

The Results obtained for different considered error and corresponding accuracy has been tabulated below:

|  |  |
| --- | --- |
| Considered Error (in %) | Accuracy (in %) |
| 1 | 38.22 |
| 2 | 69.14 |
| 3 | 82.02 |
| 4 | 87.89 |
| 5 | 90.44 |
| 6 | 93.37 |
| 7 | 96.68 |
| 8 | 98.29 |
| 9 | 99.33 |
| 10 | 100 |

The following image shows a plot between Accuracy and Considered Error:



It can be clearly seen that accuracy is increasing continuously if we increase the considered error.

**Discussion and Conclusion:**

This paper shows the use of LSTM stacked RNN for predicting stock price for WIPRO limited. Different errors and accuracy has been clearly shown for this model. It is giving us good results and it is able to see the small variations in the stock prices.