STOCK PRICE PREDICTION USING LSTM

Abstract:

Aremarkable amount of profit can be attained through an effective stock prediction paradigm. This is certainly not an easy task because of the inconsistent marketing factors acting as a blockade. At present, numerous organizations rely on efficient and accurate prediction techniques for achieving high profits. A definitive amount of profit can be earned with the usage of various machine learning patterns such as the Long Short-Term Memory (LSTM)rather than standard protracted methods.

Keywords: Stock Market, Deep Learning, Stock Price Prediction, Long Short-Term Memory

Introduction:

1. Fundamentals:

The buying and selling of shares transpires every day at the stock market for companies which are registered publicly. Stock exchange is the mediator that allows buying and selling of shares. Stock Market Prediction plays an important role in the arbitration of the worth of company stock in the forthcoming and the trading of financial instruments on an exchange.

The absolute idea of predicting stock prices is to gain significant profits but it is a burdensome task. There are a lot of other elements involved in the prediction such as physical factors, rational and irrational behavior, etc. All these elements combine to make share prices dynamic and volatile making it difficult to predict prices with high accuracy. Therefore researchers and numerous organizations have been relying on machine learning patterns method which are used to provide higher accuracy. One of these methods, i.e. **Long Short Term Memory (LSTM)** has been playing a significant role in the prediction domain.

2. Long Short Term Memory (LSTM):

LSTMis a conventional form of Neural Networks which can proficiently make use of long-term dependencies, i.e., it can have memory of previous inputs for a very long time. Traditional Recurrent Neural Network(RNN) are not good at capturing long-term dependencies. When we tend to work with a massive data set and multiple RNN layer, we are at the risk of vanishing gradient problem. To overcome this problem, LSTM was introduced. LSTM is capable of remembering RNN’s weight and their inputs over a very long

period of time. In addition to the hidden state, cell state is passed down to the next block. LSTM uses three main gates. First one is the **Forget Gate** which removes the information that is no longer useful in the cell state. Then the **Input Gate** adds the additional information to the cell state. Finally, the additional useful information to the cell state is also added by an **Output Gate**. This mechanism has allowed network to learn the conditions for when to forget, ignore or keep information in the memory cell.

Data Set:

Thisdataset of NSE-TATA-GLOBAL-BEVERAGES(from year 2013 to 2018) (**Fig.1**) stock priceswith attributes: date, opening price, high price, low price, closing price, last price, Total Trade Quantity and Turnover.

	A	B	C	D	E	F	G	H
1	Date	Open	High	Low	Last	Close	Total Trade Quantity	Turnover (Lacs)
2	08-10-2018	208	222.25	206.85	216	215.15	4642146	10062.83
3	05-10-2018	217	218.6	205.9	210.25	209.2	3519515	7407.06
4	04-10-2018	223.5	227.8	216.15	217.25	218.2	1728786	3815.79
5	03-10-2018	230	237.5	225.75	226.45	227.6	1708590	3960.27
6	01-10-2018	234.55	234.6	221.05	230.3	230.9	1534749	3486.05
7	28-09-2018	234.05	235.95	230.2	233.5	233.75	3069914	7162.35
8	27-09-2018	234.55	236.8	231.1	233.8	233.25	5082859	11859.95
9	26-09-2018	240	240	232.5	235	234.25	2240909	5248.6
10	25-09-2018	233.3	236.75	232	236.25	236.1	2349368	5503.9
11	24-09-2018	233.55	239.2	230.75	234	233.3	3423509	7999.55
12	21-09-2018	235	237	227.95	233.75	234.6	5395319	12589.59
13	19-09-2018	235.95	237.2	233.45	234.6	234.9	1362058	3202.78
14	18-09-2018	237.9	239.25	233.5	235.5	235.05	2614794	6163.7
15	17-09-2018	233.15	238	230.25	236.4	236.6	3170894	7445.41
16	14-09-2018	223.45	236.7	223.3	234	233.95	6377909	14784.5
17	12-09-2018	216.35	223.7	212.65	221.65	222.65	4570939	10002.01
18	11-09-2018	222.5	225.4	214.85	216.35	216	3508990	7735.81
19	10-09-2018	222.5	235.15	220.65	221.05	222	7514106	17130.29
20	07-09-2018	221	224.5	219.1	223.15	222.95	1232507	2742.84
21	06-09-2018	224	225	218.2	220.95	221.05	1738824	3856.72
22	05-09-2018	222	224.6	215.2	222.1	222.4	3023097	6674.93
23	04-09-2018	238.2	238.2	222.6	223.45	223.7	3554859	8163.82
24	03-09-2018	236	243.55	235.05	236.85	236.7	5242852	12538.39
25	31-08-2018	237	239.75	232.95	234.65	234.3	3353833	7913.21
26	30-08-2018	235.35	237.3	232.1	237.3	236	1921327	4516.57
27	29-08-2018	233.85	237.7	232.7	234.2	234.55	1394661	3280.33
28	28-08-2018	237	239.3	231.3	232.9	233.35	2374782	5571.77
29	27-08-2018	231.8	239.35	231.05	236.8	237.05	1990020	4689.94
30	24-08-2018	234.5	237.2	230.2	231.5	231	1838417	4289.35

Fig.1NSE-TATA-GLOBAL-BEVERAGES Dataset

Implementation:

Firstly, we will import the required libraries and modules like pandas, numpy, matplotlib, keras.models, keras.layers and sklearn.preprocessing.

```
import pandas as pd
import numpy as np

import matplotlib.pyplot as plt
%matplotlib inline

from matplotlib.pyplot import rcParams
rcParams['figure.figsize'] = (20,10)

from keras.models import Sequential #layer by layer model
from keras.layers import LSTM, Dense
# Dense is used to feed i/o to layers

from sklearn.preprocessing import MinMaxScaler
# Scaling features in a given range
```

Next, we will read data from the 'NSE-Tata-Global-Beverages' dataset as shown in **Fig.2**. For this purpose pd.read_csv() is used and to display the read data, head() is used.

```
#Reading Dataset
df = pd.read_csv('NSE-Tata-Global-Beverages.csv')
df.head()
```

	Date	Open	High	Low	Last	Close	Total Trade Quantity	Turnover (Lacs)
0	2018-10-08	208.00	222.25	206.85	216.00	215.15	4642146.0	10062.83
1	2018-10-05	217.00	218.60	205.90	210.25	209.20	3519515.0	7407.06
2	2018-10-04	223.50	227.80	216.15	217.25	218.20	1728786.0	3815.79
3	2018-10-03	230.00	237.50	225.75	226.45	227.60	1708590.0	3960.27
4	2018-10-01	234.55	234.60	221.05	230.30	230.90	1534749.0	3486.05

Fig.2 Reading data

Next, we will assign 'Date' attribute as index and visualize the closing price with respect to Year as shown in **Fig.3**.

```
# Analysing the closing prices from dataframe
df['Date'] = pd.to_datetime(df.Date, format = "%Y-%m-%d")
df.index = df['Date']

plt.figure(figsize=(16,8))
plt.plot(df['Close'], label="Closing Price History")
```

Fig.3 Analysing Closing Price with respect to Year

Now, we will sort data according to the 'Date' attribute and assign it to new variable along with closing price (as this is the dependent variable in our dataset) for scaling purpose (**Fig.4**).

```
# Sorting dataset on datetime and filtering
data = df.sort_index(ascending=True, axis=0)
new_df = pd.DataFrame(index=range(len(df)), columns=['Date','Close'])

for i in range(len(data)):
    new_df['Date'][i] = data['Date'][i]
    new_df['Close'][i] = data['Close'][i]


new_df.head()
```

	Date	Close
0	2013-10-08 00:00:00	155.8
1	2013-10-09 00:00:00	155.55
2	2013-10-10 00:00:00	160.15
3	2013-10-11 00:00:00	160.05
4	2013-10-14 00:00:00	159.45

Fig.4 New data with Date attribute and Close Price attribute

Now, we will scale the new variable but first we will split the data into training and testing variable. This is done for predicting data on small scale first.

```
# Scaling the filtered dataset
new_df.index = new_df.Date
new_df.drop('Date', axis=1, inplace=True)

final_df = new_df.values

train_data = final_df[0:987,:]
test_data = final_df[987:,:]

sc = MinMaxScaler(feature_range=(0,1)) #Here given Range of scaling is 0 to 1.
sc_data = sc.fit_transform(final_df) #Scaled Data

x_train_data, y_train_data = [], []

for i in range(60, len(train_data)):
    x_train_data.append(sc_data[i-60:i,0])
    y_train_data.append(sc_data[i,0])

x_train_data, y_train_data = np.array(x_train_data), np.array(y_train_data)

x_train_data = np.reshape(x_train_data,(x_train_data.shape[0],x_train_data.shape[1],1))
```

Next, we will build and train the model on training data.

```
# Building and training LSTM model

lstm_model = Sequential()
lstm_model.add(LSTM(units=50, return_sequences=True, input_shape=(x_train_data.shape[1],
 1)))
lstm_model.add(LSTM(units=50))
lstm_model.add(Dense(1))

lstm_model.compile(loss='mean_squared_error', optimizer='adam')
lstm_model.fit(x_train_data,y_train_data,epochs=1,batch_size=1,verbose=2)
# epochs is the no of times the model will walk thru the entire training data.
# batch_size is the no of samples to work thru before updating the model parameters.

inputs_data = new_df[len(new_df)-len(test_data)-60:].values
inputs_data = inputs_data.reshape(-1,1)
inputs_data = sc.transform(inputs_data)


927/927 - 44s - loss: 0.0011
```

Next, we'll perform prediction on test data with our model and store the prediction values alongside the actual values in the variable as shown in **Fig.5**.

```
# Taking sample of df to predict stock price using LSTM model
x_test = []
for i in range(60,inputs_data.shape[0]):
    x_test.append(inputs_data[i-60:i,0])
```

```
x_test = np.array(x_test)

x_test = np.reshape(x_test,(x_test.shape[0],x_test.shape[1],1))
pred_close_price = lstm_model.predict(x_test)
pred_close_price = sc.inverse_transform(pred_close_price)


# Saving the lstm model in the system
lstm_model.save('saved_lstm_model.h5')


train_data = new_df[:987]
test_data = new_df[987:]
test_data['Predictions'] = pred_close_price


test_data
```

Date	Close	Predictions
2017-10-09	208.3	204.030746
2017-10-10	208.45	205.057953
2017-10-11	209.4	205.764893
2017-10-12	212	206.360245
2017-10-13	210.25	207.233780
...	...	...
2018-10-01	230.9	233.424728
2018-10-03	227.6	232.532333
2018-10-04	218.2	230.973389
2018-10-05	209.2	227.713303
2018-10-08	215.15	222.715164

248 rows × 2 columns

Fig5. Actual Closing Price and Predicted Closing Price

Now, we will visualize the predicted sample data on the whole data as shown in **Fig.6** and visualize predicted sample data on the actual sample data as shown in **Fig.7**.

```
# Visualizing the predicted and actual closing price
plt.plot(train_data['Close'])
plt.plot(test_data[['Close','Predictions']])
plt.title('Stock Price Prediction', fontsize = 30)
plt.xlabel('Date', fontsize=25)
plt.ylabel('Closing Price', fontsize=25)
plt.legend(['Train','Test','Predicted'], loc='upper left', prop={'size': 20})
```

```
<matplotlib.legend.Legend at 0x21d58d9cf70>
```

Fig.6 Predicted test data on whole data

```
plt.plot(test_data['Close'])
plt.plot(test_data['Predictions'])
plt.title('Stock Price Prediction', fontsize = 30)
plt.xlabel('Date', fontsize=25)
plt.ylabel('Closing Price', fontsize=25)
plt.legend(['Test','Predicted'], loc='upper right', prop={'size': 20})
```

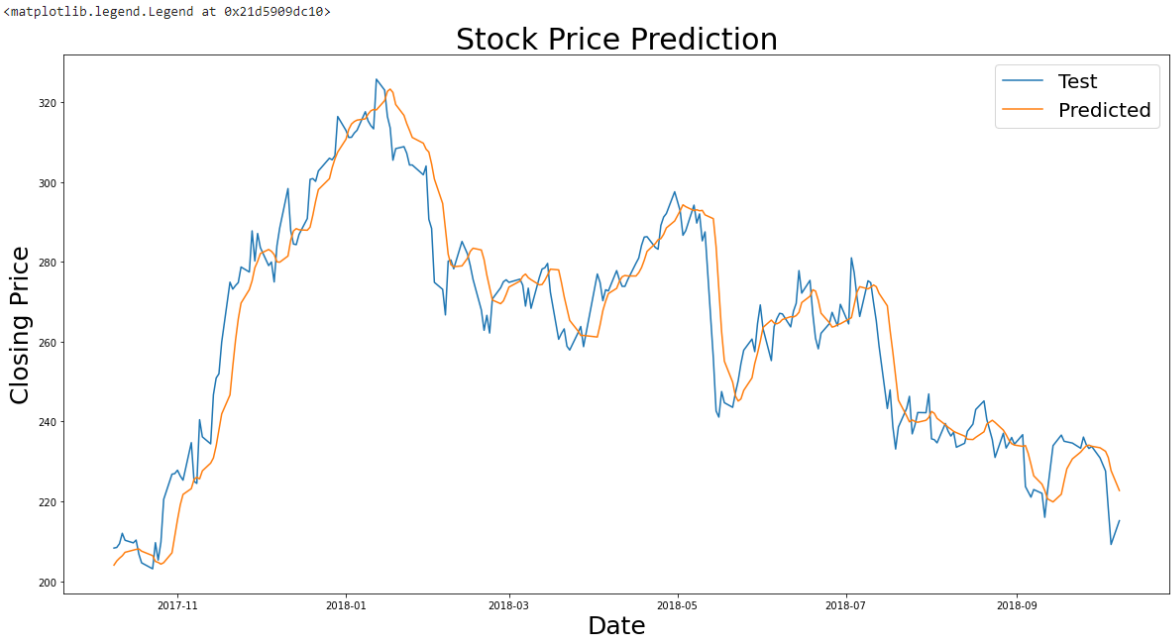

Fig.7 Predicted test data on Actual test data

Result

The Stock price prediction method in this paper is highly accurate. The loss of model is **0.0011** (lower the loss higher the accuracy). Here we used LSTM model to implement stock price prediction as it is a best model for time series data.

Finally, we will visualize plots of whole actual data (**Fig.8**) and predicted data (**Fig.9**).

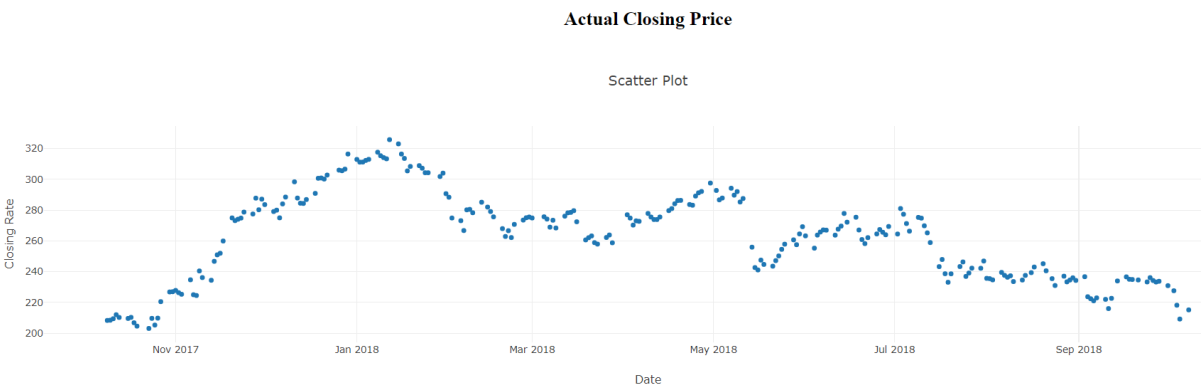

Fig.8 Actual Closing Price of whole data

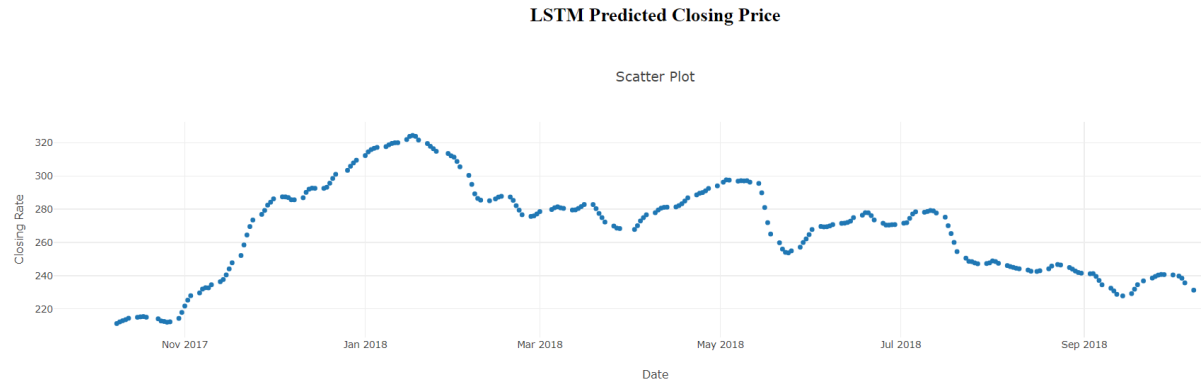

Fig.9 Predicted Closing Price of whole data

Conclusion

In order to fulfill the need of industry to get maximum returns on investment with minimum risk, we have implemented a model known as stock price prediction model which helps you to predict the ups and downs of prices. We have a historical dataset to train and test the model. We have used the LSTM model to implement predictions. Our mean squared error value is 0.0011, it shows that our model's prediction is almost similar to close price.

References

Nandakumar, R., Uttamraj, K. R., Vishal, R., &Lokeswari, Y. V. (2018). Stock price prediction using long short term memory. *International Research Journal of Engineering and Technology*,*5*(03).

Ghosh, Achyut, Soumik Bose, GiridharMaji, Narayan Debnath, and Soumya Sen. “Stock price prediction using LSTM on Indian share market.” In Proceedings of 32nd international conference on, vol. 63, pp. 101-110. 2019.

Adil MOGHAR,Mhamed HAMICHE. “Stock Market Prediction Using LSTM Recurrent Neural Network.” International Workshop on Statistical Methods and Artificial Intelligence (IWSMAI 2020), April 6-9, 2020, Warsaw, Poland.

Jia H. (2016). ”Investigation into the effectiveness of long short term memory networks for stock price prediction.”arXiv preprint arXiv:1603.07893.

https://www.sciencedirect.com/science/article/pii/S1877050918307828

https://ezproxy.svkm.ac.in:2142/document/8920761

Machine Learning to Predict Stock Prices | by RoshanAdusumilli | Towards Data Science