**Project : Stock Market Prediction using LSTM Model**

Problem Statement : Predict the stock price of SBI for 30 days in future using LSTM and do comparison with actual prices.

Solution:

Import Libraries:



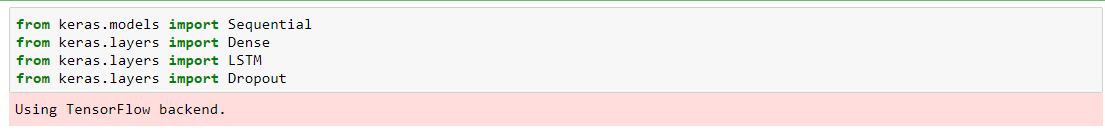
The data that we are going to use for this algorithm is downloaded by NSEpy API. For training our algorithm, we will be using the SBI stock prices from 1st January 2015 to 31st December 2018. For the sake of prediction, we will use the SBI stock prices for the month of January 2019



Make a dataframe from the above fetched data:



Import other required libraries:



Import training data set:



Now we are only interested in the closing price of the stock. Therefore, we will filter all the data from our training set and will retain only the values for the Close column:



We will use MinMaxScaler to scale our data between 0 and 1:



Now we are going to predict the closing stock price of the data based on the closing stock prices for the past 60 days:



There are 988 records in the training data. We execute a loop that starts from 61st record and stores all the previous 60 records to the feature\_set list. The 61st record is stored in the labels list.

We need to convert both the feature\_set and the labels list to the numpy array before we can use it for training:



In order to train LSTM on our data, we need to convert our data into the shape accepted by the LSTM. We need to convert our data into three-dimensional format. The first dimension is the number of records or rows in the dataset which is 988 in our case. The second dimension is the number of time steps which is 60 while the last dimension is the number of indicators. Since we are only using one feature, i.e Close, the number of indicators will be one.



We will add four LSTM layers to our model followed by a dense layer that predicts the future stock price

Instantiate the Sequential class:

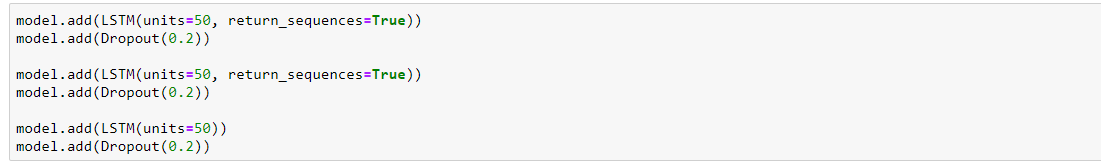


Creating LSTM and Dropout Layers:





Adding 3 more LSTM Layers:



To make our model more robust, we add a dense layer at the end of the model:



We use the mean squared error as loss function and to reduce the loss or to optimize the algorithm, we use the adam optimizer:



Algorithm Training:



Testing our LSTM

We have successfully trained our LSTM, now is the time to test the performance of our algorithm on the test set by predicting the closing stock prices for the month of January 2019

Again we will use NSEpy API to fetch data and then store it in as a dataframe:





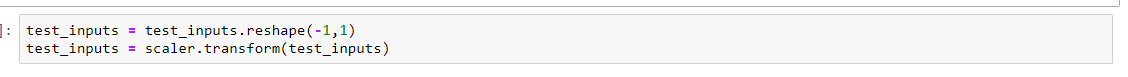
For each day of January 2019, we want our feature set to contain the opening stock prices for the previous 60 days. For the 1st of January, we need the stock prices for the previous 60 days. To do so, we need to concatenate our training data and test data before preprocessing



The input for each day should contain the opening stock prices for the previous 60 days. That means we need opening stock prices for the 20 test days for the month of January 2019 and the 60 stock prices from the last 60 days for the training set



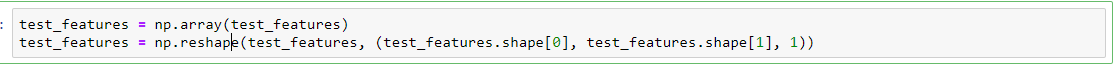
As we did for the training set, we need to scale our test data:



Let's prepare our final test input set that will contain previous 60 stock prices for the month of January:



Now we need to convert our data into the three-dimensional format which can be used as input to the LSTM:



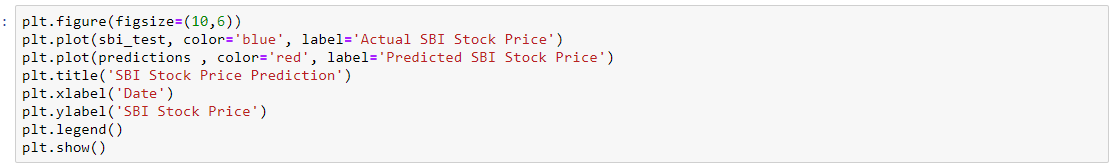
Making Predictions:

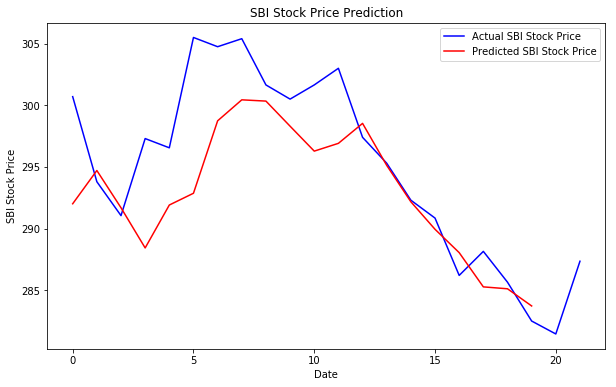


Since we scaled our data, the predictions made by the LSTM are also scaled. We need to reverse the scaled prediction back to their actual values



Plotting the predictions and Comparision:





Findings and Conclusion:

So here we can see that LSTM does a great job in stock predictions. A long short-term memory network (LSTM) is one of the most commonly used neural networks for time series analysis. The ability of LSTM to remember previous information makes it ideal for such tasks.