SAPIENZA
UNIVERSITÀ DI ROMA

Final project: Stock market data analysis using Twitter data

Name: Debodeep Banerjee
Matricola: 1901253

Contents

1 Introduction

Money market has always been great matter of anticipation and prediction and so as stock market. In this case. The accurate prediction of prediction of stock market data is one of the most intriguing task of the investors. While inaccurate prediction may lead to bankruptcy, accurate prediction will come as a boon. In this report, I encounter a stock market prediction problem. Data of 2015 to 2021 of ONGC (Oil and Natural Gas Corporation). Over the years, several techniques for modeling time series data have been developed. However, in this project we have implemented LSTM model in order to predict the future data. Further, specifically in the stock data, the volatility of stock data highly depends on news and/ or social media. To address this idea, I have mined twitter with a programming module twint and ran a sentiment analysis algorithm which provides scores to those twits. Combining the scores and the stock data, another LSTM model has been developed. Eventually we compare both the models to draw suitable conclusions. The entire analysis has been done with python programming language.

2 Data Collection

For this task, two separate sets of data have been collected. First, historical stock price of 2015 to 2021 of ONGC has been collected with a module named *investpy*. Further, in order to combine the data with the twitter data, all the twits of ONGC have been collected using a module *twint*. Further the twits were pre-processed and only those twits written in English were taken into account.

3 Methodology

As mentioned earlier, two models were used in this specific task. Though both the models are essentially LSTM models, the very difference between these two models are while the first models involves only the closing stock prices of ONGC, the second model considers the twits also. In order to analyse the twits, help of natural language processing techniques were necessary. After collecting all the tweets, pre prcessing has been done and necessary stopwords were removed. India, being a multilingual country, there were several twits which were in languages other than English. All such cases has been ignored and only those twits with English language have been taken into account. Once the cleaned twits are obtained, the next task is to judge the sentiment of the sentences. For this purpose, one brilliant library called *vader* was used. Vader is one of the very famous libraries used in financial economics. The brilliance of vader lies in the fact that it analyses the words and predicts the sentiments of the sentences, that is, whether a sentence has a positive sentiment or a negative sentiment. In order to do this, vader provides few scores to the sentences. For this project, the compound scores have been considered. Finally, the stock data and the compound scores were combined before sending it to an LSTM model. Last but not the least, hyper parameter tuning strategy was applied to both the models for the better ment of performance.

4 Exploratory Data Analysis

4.1 area plot

in this case, we plot the entire graph of the closing price of ONGC stocks under a graphical area. In this case we see that the graph dips significantly in the span 2020-2021. This is natural as the entire country was under lock down and the companies also suffered.

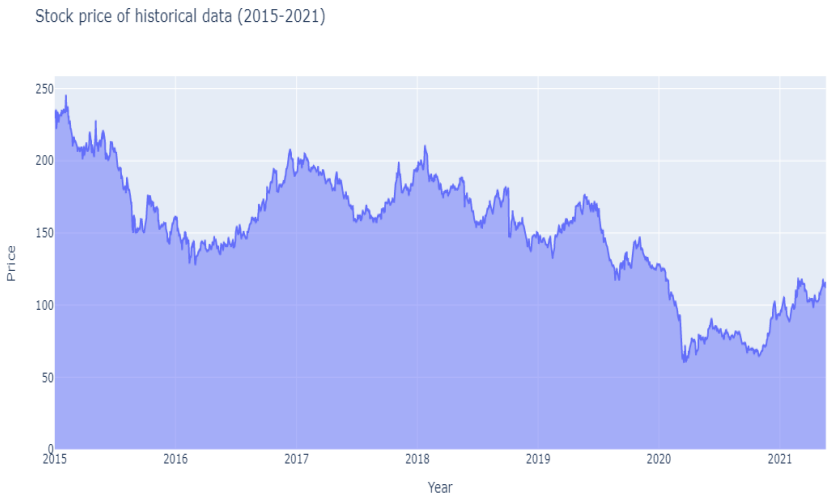

Figure 1: Stock Price of ONGC (2015-2021)

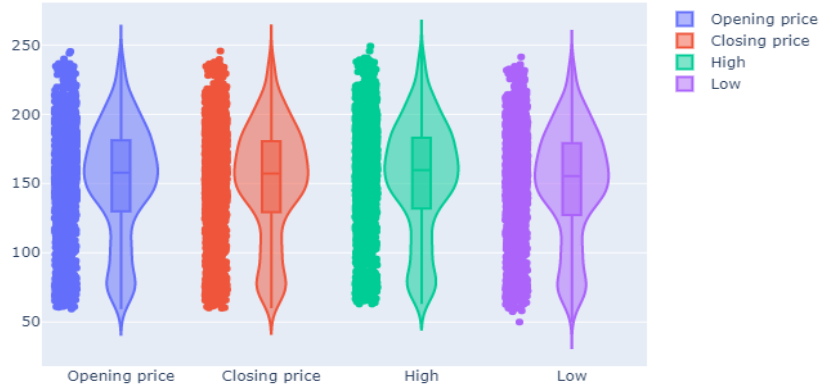

Figure 2: Stock Price of ONGC (2015-2021)

4.2 Moving average and exponential average

In order to understand the data more critically, we consider the plots of moving averages. It is a widely used technique used by traders and investors. It can be calculated for different prices, such as the open, close, high, and low. It is a backward-looking indicator and relies on past price data for 1a certain period The moving

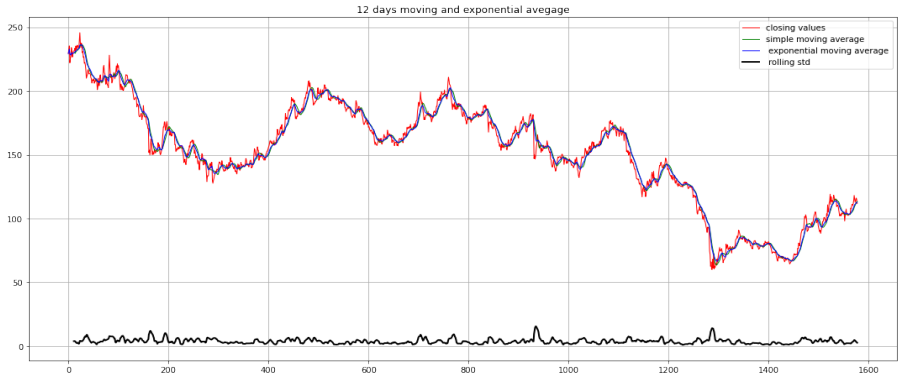

Figure 3: 12 days moving and exponential average

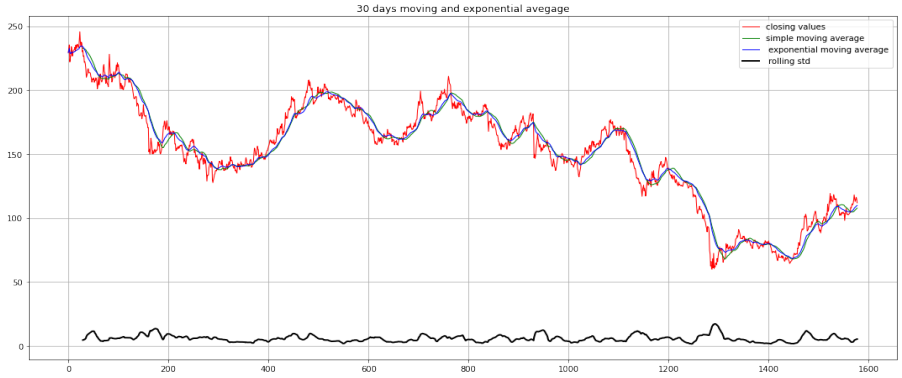

Figure 4: 30 days moving and exponential average

average is a great technique to check minute fluctuation(s) in the data. A moving average of lower period is more sensitive to minor fluctuation of the prices. Exponential moving averages increases the sensitivity by applying more weights to recent prices. The application of EMA essentially depends on the number of previous days the weights are being applied on the most recent value. The fundamental difference between EMA and simple moving average is that in order to calculate a EMA, one needs to obtain the previous days' EMA as well. The forluma for simple moving average is,
SMA $= \frac{A_1+A_2+.....+A_n}{n}$ where, $A_i = textpriceoftheasset$
$n = textnumberofperiods$
In 2, 3 and 4 we see that the 12 days moving average curve is following the price trend more accurately than the 30 days moving average (3) and the 60 days moving average (4) fails to capture all the minute ups and downs of the daily closing and opening values. But if we consider the exponential moving averages, we can notice that the in all the cases the exponential moving averages performs pretty well to reduce the lag even for a longer period of time.

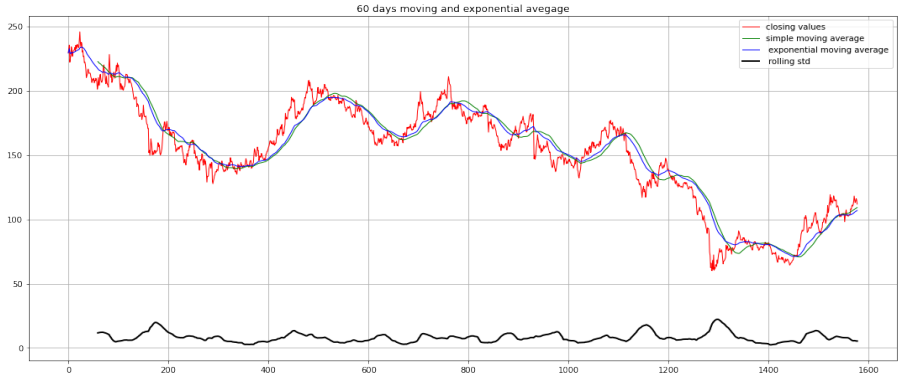

Figure 5: 30 days moving and exponential average

4.3 Daily return

Daily return is a very useful metric to check the volatility of an asset. It checks the ups and downs on a daily basis. The formula for daily return is as follows:
daily return $= \frac{P_1}{P_0} - 1$
where, $P_0 =$ initial value, $P_1 =$ next day value
In 5, we can see massive volatility of closing price in 2020 where we observe the daily return dipped to -15%

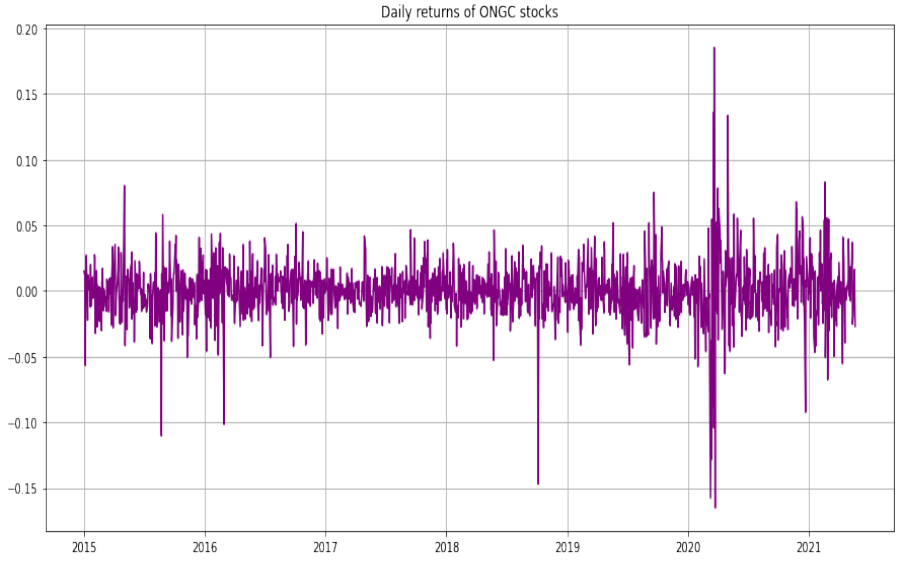

Figure 6: Daily Return of ONGC stock data

and we see a rise of more than 15%. We may intuitively say that the pandemic might have had an effect on the daily returns though finding the reasons of price volatility is beyond the scope of this report.

5 Forcasting

5.1 Data pre processing

In order make the data prepared for the analysis, it needed to be processed to some extent. As the main aim has been to conduct the analysis with the closing values only, the 'Close' column is extracted from the parent data frame. Next, the original values has been normalized using the MinMaxScaler function of Scikit-learn library for better fitting. Next, the data has been splitted into train set, validation set and test set respectively. Here 70% of the data has been considered as training data and test and validity is 15% each. Here it is important to mention that time series data, being dependent on each previous day, a sixty day window has been constructed based on which the train and validation data were fitted into the LSTM model.

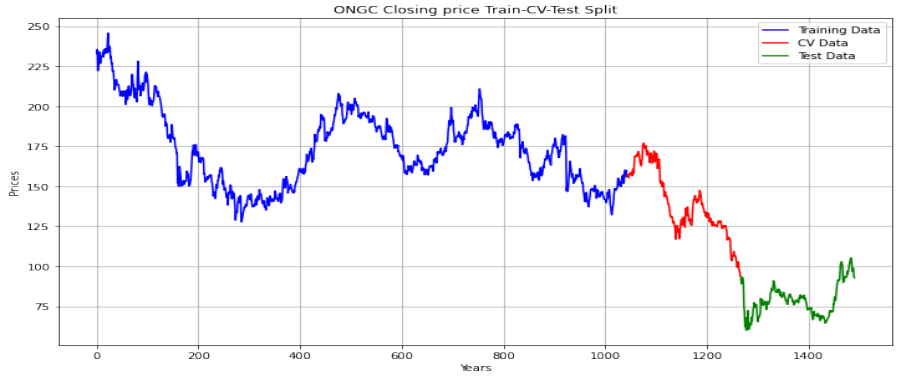

Figure 7: Train test split

5.2 Simple LSTM

In this case, the LSTM model has been used. The reason behind choosing an LSTM framework is that LSTMs are very powerful in sequence prediction problems because they're able to store past information. This is important in this case because the previous price of a stock is crucial in predicting its future price. In the model, the initial number of units are chosen as 256 and after implementing a 20% drop outs, we introduce another layer wit 64 neurons.For this model, we have used the tanh activation function. Under these condition and 100 epochs with batch size being 70, the final validation loss has been 0.0018. Finally we fit the model with the test data and end up with an RMSE of 19.178.

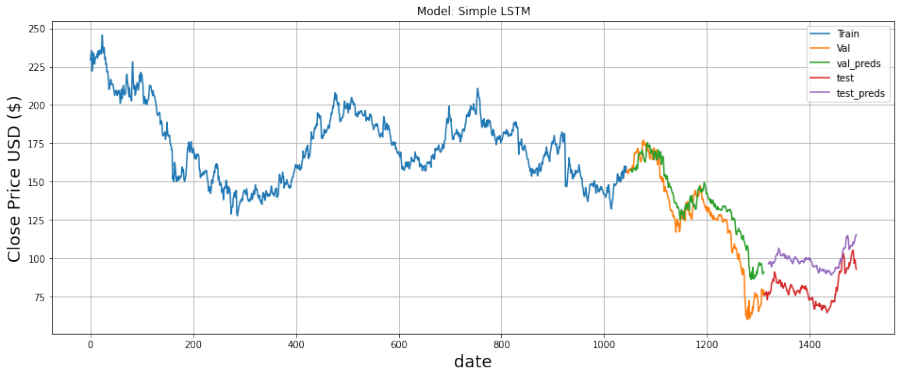

Figure 8: Simple LSTM model

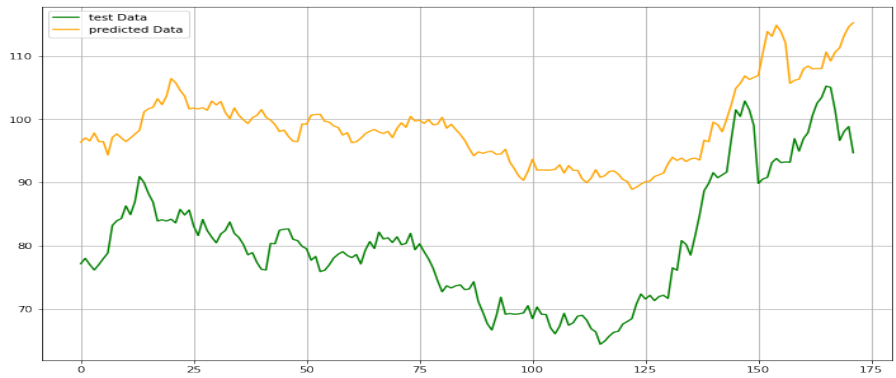

Figure 9: Test: True vs Prediction- zoomed in

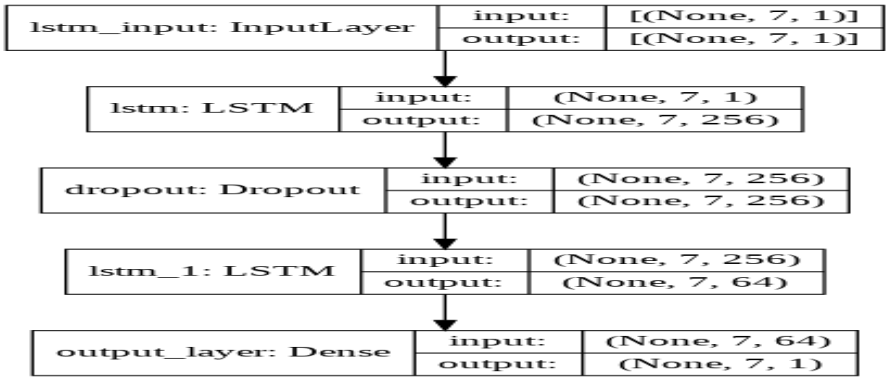

Figure 10: Model architecture

5.3 LSTM with news data

In this case we combine the compound scores along with the stocks data and conduct the same LSTM model with three LSTM layers of units 512, 128, 256 respectively. We see that the model gives us pretty good result with a train and validation RMSE 3.012.

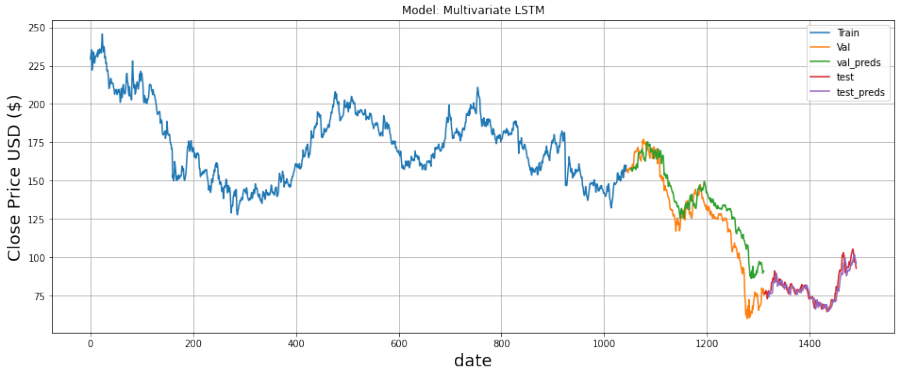

Figure 11: LSTM news model with news data prediction

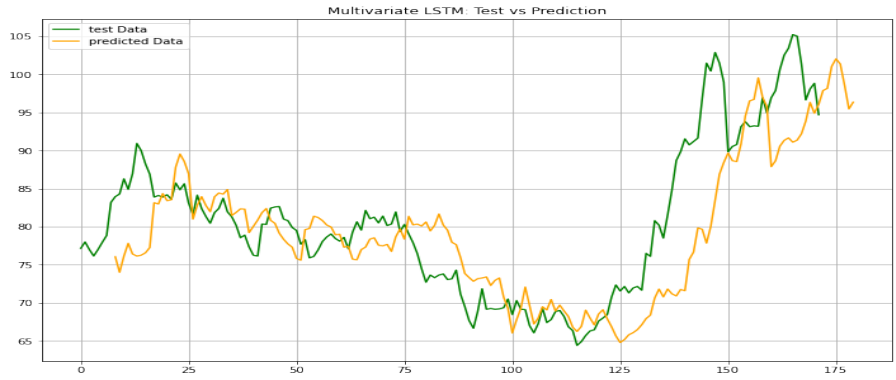

Figure 12: Test: True vs Prediction- zoomed in

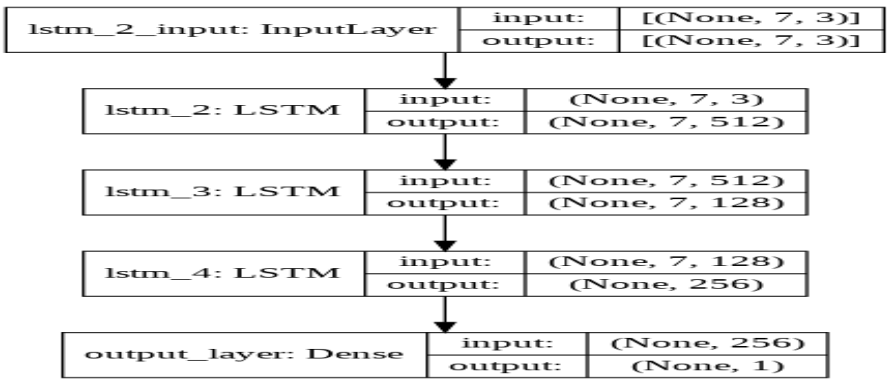

Figure 13: Mpdel architecture

5.4 LSTM Autoencoder

In this set up, we implement an LSTM autoencoder. In this structure, the LSTM mechanism reads the data and encodes it. This part is called the encoder part. Then in order to decode the input, we first use a *RepeatVector* layer and then use the same encoder structure but in a reverse manner. The LSTM AE gave us a pretty satisfactory result with an RMSE of 3.39.

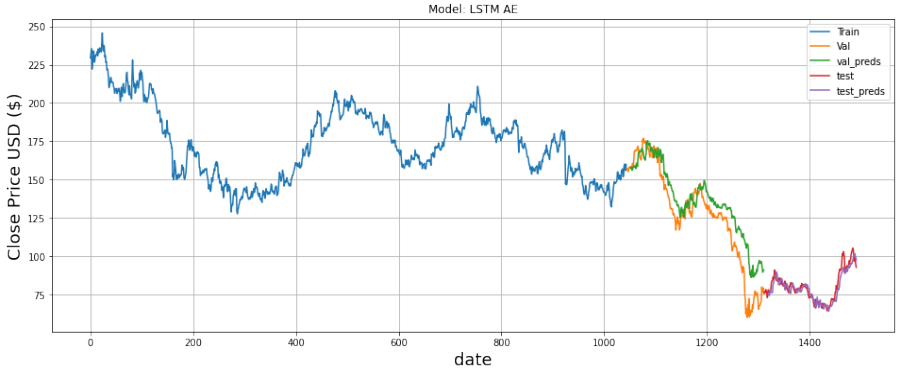

Figure 14: LSTM AE model with news data prediction

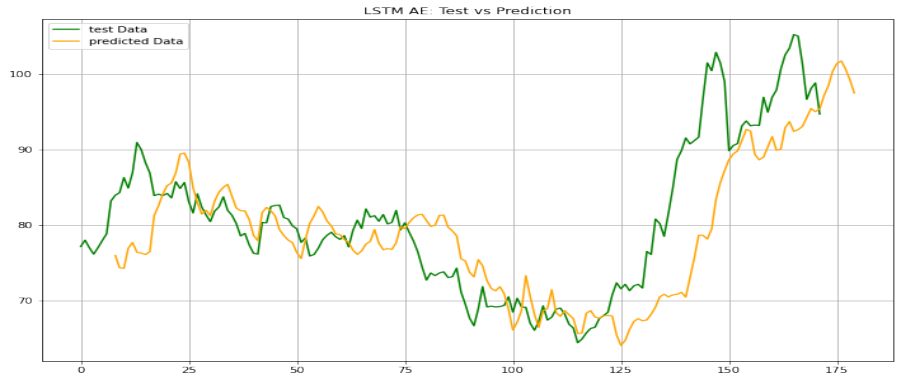

Figure 15: Test: True vs Prediction- zoomed in

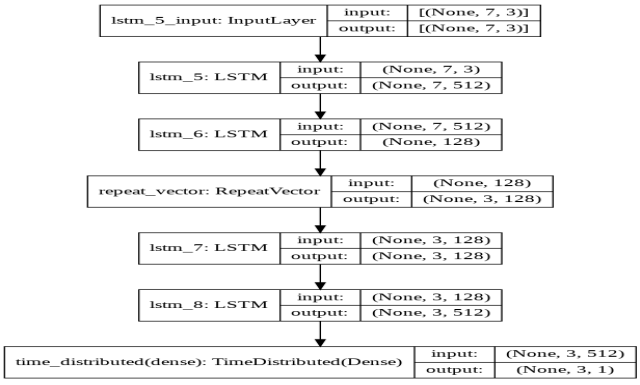

Figure 16: Model architecture

5.5 Comparison table

An over all picture of all the comparisons are given below.

Model	RMSE
Simple LSTM	19..178
Multivariate LSTM	3.012
LSTM AE	3.39

6 Conclusion and future work

From the entire task, it has been observed that the LSTM model with news polarity has out performed the standard LSTM model. However, it must be admitted that all the model almost worked seimilarly and hence there mught be further chance of improvement.Additional feature engineering to the twitter data could be very helpful. But amidst all these things, we can understand that news and twitter have become intertwined with stock market data and our model has been able to address the issue successfully.