Stock Price Prediction

ML Project

By: Lilit Hovsepyan ASDS YSU 2020 Machine Learning Lecturer: Tigran Sukiasyan

Table of Contents

1. Introduction

Predicting how the stock market will perform is one of the most difficult things to do. There are so many factors involved in the prediction – physical factors vs. phycological, rational and irrational behavior, etc. The successful prediction of a stock's future price will maximize investor's gains. And predicting stock prices has always been an attractive topic to both investors and researchers.

Can we use machine learning as a game changer in this domain? In recent years, the increasing prominence of machine learning in various industries have enlightened many traders to apply machine learning techniques to the field, and some of them have produced quite promising results.

We'll be using a dataset from Quandl (or download the stock.csv file attached) and for this project, I have used the 'Tata Global Beverages' dataset. Tata Consumer Products Limited, formerly Tata Global Beverages Limited, is an Indian multinational non-alcoholic beverages company headquartered in Mumbai, Maharashtra, India and a subsidiary of the Tata Group.

In this paper, we will work with historical data about the TATA stock prices of a publicly listed company. We will implement the Machine Learning algorithms to predict the future stock price, starting with simple algorithms, and then move on to advanced techniques, starting from the Linear Regression where we are going to see whether the first and the last working days of week have an impact on the close price of the stock or not. And we will end up with Facebook Prophet and LSTM methods which are widely used in the time series predictions sphere.

2. Related Work

In many other related works, there were lots of approaches to predict stock price with Machine Learning. But there were some main differences between that projects. In each of them the time period of the prediction was different, i.e. targeting price change can be less than a minute or a few days later or months later. The other difference was that the predictors used can range from global news and economy trend or the company's specific characteristics, and in some of the works this kind of information was added to the stock price prediction.

Some researches showed that by using short change time will lead to the higher accuracy and by adding global news to the predictions will increase the accuracy of the model, as well. But these well working models often rely on the company's specifics or on some particular information of the company. Sometimes for the long-term goals of the company it's required and highly preferred to have the prediction almost on a monthly basis. And in most of the cases it's quite difficult to add global news to the model, since for the long-run cases it's almost impossible to foresee what is going to be change in the politics and how it will be going to affect on the stock price in the future. And most of the cases in many of the projects you can see the classification problem solved.

Here we are going to use Linear regression, k-NN algorithm, ARIMA and afterwards shift into the widely used algorithms such as Facebook Prophet (open source library) and LSTM by showing that these fit the model very well compared with the above-mentioned algorithms.

3. Dataset and Features: Understanding the Problem

It’s important to establish what we’re aiming to solve.

Stock market analysis is divided into two parts: 1. Fundamental Analysis and 2. Technical Analysis. Both methods are used for researching and forecasting future trends in stock prices, and, like any investment strategy or philosophy, both have their advocates and adversaries.

- Fundamental analysis study everything from the overall economy and industry conditions to the financial condition and management of companies. Earnings, expenses, assets, and liabilities are all important characteristics to fundamental analysts. So, Fundamental analysis involves analyzing the company’s future profitability on the basis of its current business environment and financial performance.
- Technical analysis differs from fundamental analysis in that the stock's price and volume are the only inputs. The core assumption is that all known fundamentals are factored into the price, thus there is no need to pay close attention to them. Technical analysts do not attempt to measure a security's intrinsic value, but, instead, use stock charts to identify patterns and trends that suggest what a stock will do in the future. So, Technical Analysis, on the other hand, includes reading the charts and using statistical figures to identify the trends in the stock market.

Our focus will be on the technical analysis part. We’ll be using a dataset for ‘Tata Global Beverages’.

Table 1. TATA data first 5 rows

	Date	Open	High	Low	Last	Close	Total Trade Quantity	Turnover (Lacs)
0	2018-09-28	234.05	235.95	230.20	233.50	233.75	3069914	7162.35
1	2018-09-27	234.55	236.80	231.10	233.80	233.25	5082859	11859.95
2	2018-09-26	240.00	240.00	232.50	235.00	234.25	2240909	5248.60
3	2018-09-25	233.30	236.75	232.00	236.25	236.10	2349368	5503.90
4	2018-09-24	233.55	239.20	230.75	234.00	233.30	3423509	7999.55

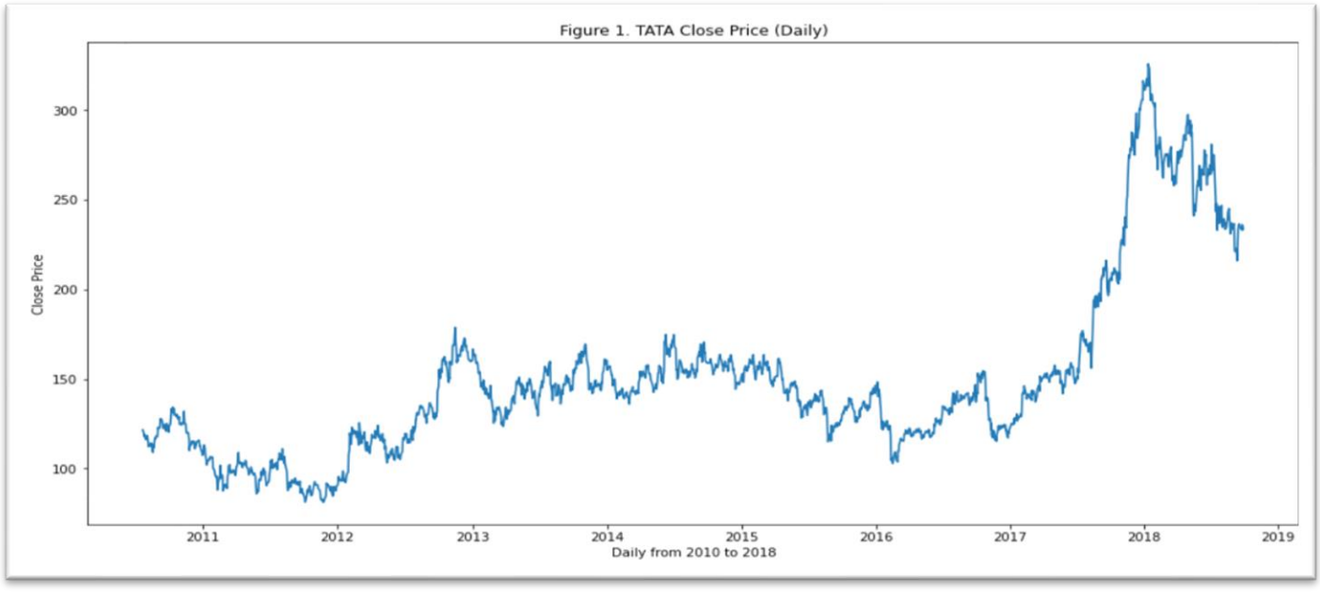
As we can see from the Table 1. we have a time series data which shape is (2035, 8). Since the domain of our problem is prediction of a TATA price, we decided to proceed with daily time series data. As we can see clearly there are multiple variables in the dataset – date, open, high, low, last, close, total trade quantity and turnover.

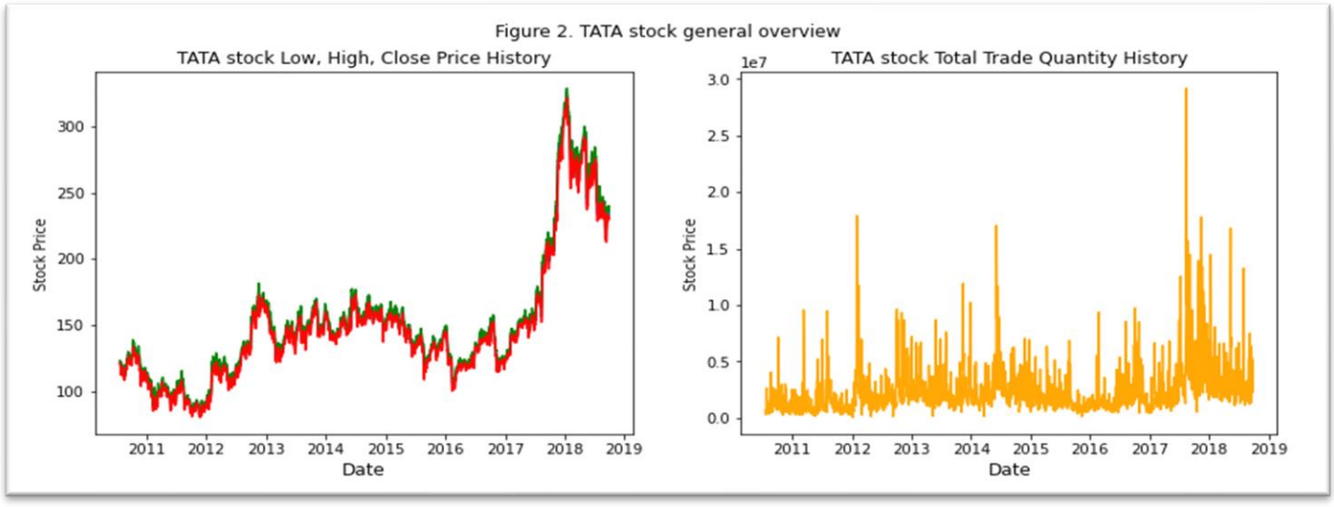
- The columns Open and Close represent the starting and final price at which the stock is traded on a particular day.
- High, Low and Last represent the maximum, minimum, and last price of the share for the day.
- Total Trade Quantity is the number of shares bought or sold in the day and Turnover (Lacs) is the turnover of the particular company on a given date.

While splitting the data into train and test set, we cannot use random splitting since that will destroy the time component. So here we have set the last year’s data into test and the 4 years’ data before that into train set, which is almost the first 1000 rows for the train set and the rest 1035 row for test set. So, the shape of the training set is (1000,8) and of the test set is (1035,8).

In the data there were no NAs. Date feature was needed to convert into datetime format since it came out that it had an object type. For normalizing the dataset, we used MinMaxScaler. As mentioned above you can easily download the data from Quandl or from the attached stock.csv file.

There is another important thing to note concerning with the dataset is market is closed on weekends and public holidays. The profit or loss calculation is usually determined by the closing price of a stock for the day, hence we will consider the closing price as the target variable. Let's plot the target variable to understand how it's shaping up in our data:





4. Methods

In this paper we are going to use Linear regression, k-NN algorithm, ARIMA and afterwards shift into the widely used algorithms such as Facebook Prophet (open source library) and LSTM by showing that these fit the model very well compared with the above-mentioned algorithms. Let's get familiar with the main ideas of these models.

4.1 Linear Regression

Linear Regression is a machine learning algorithm based on supervised learning. It performs a regression task. Regression models a target prediction value based on independent variables. It is mostly used for finding out the relationship between variables and forecasting. The linear regression model returns an equation that determines the relationship between the independent variables and the dependent variable. The equation for linear regression can be written as:

$$y = \theta_0 + \theta_1 x_1 + \cdots + \theta_n x_n$$

While training the model we are given:

x: input training data

y: labels to data (supervised learning)

When training the model, it fits the best line to predict the value of y for a given value of x. The model gets the best regression fit line by finding the best $\hat{\theta}_0$ ($\hat{\theta}_0$: intercept) and $\hat{\theta}_i$ ($\hat{\theta}_i$: coefficient of x_i) values.

In our case the y variable is the close price of the TATA stock, but initially we don't have a set of independent variables. We have only the dates instead. That's why we will use the date column to extract features like day, month, year, Monday / Friday etc. and then fit a linear regression model. Apart from this, we will add our own set of features that we believe would be relevant for the predictions. For instance, my hypothesis is that the first and last days of the week could potentially affect the closing price of the stock far more than the other days. So, I have created a feature that identifies whether a given day is Monday/Friday or Tuesday/Wednesday/Thursday. Afterwards we will calculate the cost function by using the RMSE. The cost function helps us to figure out the best possible values for θ_0 and θ_i which would provide the best fit line for the data points.

4.2 k-NN Algorithm

K-nearest neighbor (KNN) algorithm is a nonparametric classification algorithm that assigns query data to be classified to the category to which most of its neighbors belong. We use the Euclidean distance metric to find K-nearest neighbors from a sample set of known classifications. Suppose that the known data set has four feature variables $\{f_1, f_2, f_3, f_4\}$ and four categories $\{y_1, y_2, y_3, y_4\}$. The steps to search the category of the new data *i* through the KNN algorithm are as follows. Firstly, the Euclidean distance of the feature variables of the data *i* and the other data *j* (*j* = 1, 2, …, n) in the training data set is calculated:

$$D_j = \sqrt{\sum_{g=1}^{4} \left(f_g(i) - f_g(j)\right)^2}, j = 1, 2, \ldots, n$$

Secondly, all the data in the training set are sorted in ascending order according to the distance from data *i*. Thirdly, K data points with the smallest distance from data *i* are selected. Finally, the category with the largest proportion of these K data points will be considered as the category of data *i*. An important parameter to be determined in the KNN algorithm is K, which represents the number of the nearest neighbors to be considered when classifying unknown samples. A machine learning model has two types of parameters. The first type of parameters are the parameters that are learned through a machine learning model while the second type of parameters are the hyper parameter that we pass

to the machine learning model. So, in this KNN algorithm we have to specify the value of K. Normally we randomly set the value for these hyper parameters and see what parameters result in best performance. However randomly selecting the parameters for the algorithm can be exhaustive. instead of randomly selecting the values of the parameters, a better approach would be to develop an algorithm which automatically finds the best parameters for a particular model. Grid Search is one such algorithm. For this case I created the list of values of K [3,5,7,9] (only odd values since in case of even numbers there is a risk of a tie in the decision of which class you should set a new instance). The Grid Search algorithm basically tries all possible combinations of parameter values and returns the combination with the highest accuracy. But here let's not forget about one important thing as well which is cross validation. Normally the dataset is splatted into train and test sets one for training and the other one for testing the model. But there might be the case when the accuracy obtained on one test is very different to accuracy obtained on another test set using the same algorithm. The solution to this problem is to use k-Fold Cross-Validation for performance evaluation where k is any number. You divide the data into k folds. Out of the k folds, k-1 sets are used for training while the remaining set is used for testing. The algorithm is trained and tested k times, each time a new set is used as testing set while remaining sets are used for training. Finally, the result of the K-Fold Cross-Validation is the average of the results obtained on each set. In our case I took k=5 folds.

4.3 ARIMA

In statistics and econometrics, and in particular in time series analysis, an autoregressive integrated moving average (ARIMA) model is a generalization of an autoregressive moving average (ARMA) model. Both of these models are fitted to time series data either to better understand the data or to predict future points in the series. There are three important parameters in ARIMA:

- p (past values used for forecasting the next value)
- q (past forecast errors used to predict the future values)
- d (order of differencing)

Differencing is a method of transforming a non-stationary time series into a stationary one. The first differencing value is the difference between the current time period and the previous time period. If these values fail to revolve around a constant mean and variance then we find the second differencing using the values of the first differencing. We repeat this until we get a stationary series.

Parameter tuning for ARIMA consumes a lot of time. So, we will use auto ARIMA which automatically selects the best combination of (p, q, d) that provides the least error. In our case we took the p and q to change between 1 and 3, and d to be as 1.

4.4 Facebook Prophet

Prophet is an open source software released by Facebook's Core Data Science team. It's a procedure for forecasting time series data based on an additive model where non-linear trends are fit with yearly, weekly, and daily seasonality, plus holiday effects. It works best with time series that have strong seasonal effects and several seasons of historical data. Prophet is robust to missing data and shifts in the trend, and typically handles outliers well. The Prophet package provides intuitive parameters which are easy to tune. Even someone who lacks expertise in forecasting models can use this to make meaningful predictions.

It's based on decomposable (trend + seasonality + holidays) models. They are combined in the following equation:

$$y(t) = g(t) + s(t) + h(t) + \varepsilon_t$$

- g(t): piecewise linear or logistic growth curve for modelling non-periodic changes in time series
- s(t): periodic changes (e.g. weekly/yearly seasonality)
- h(t): effects of holidays (user provided) with irregular schedules
- ε_t: error term accounts for any unusual changes not accommodated by the model

Using time as a regressor, Prophet is trying to fit several linear and non-linear functions of time as components.

4.5 LSTM

Humans don't start their thinking from scratch every second. As you read this paper, you understand each word based on your understanding of previous words. You don't throw everything away and start thinking from scratch again. Your thoughts have persistence.

Traditional neural networks can't do this. One of the appeals of RNNs is the idea that they might be able to connect previous information to the present task, such as using previous video frames might inform the understanding of the present frame. If RNNs could do this, they'd be extremely useful. Sometimes, we only need to look at recent information to perform the present task. For example, consider a language model trying to predict the next word based on the previous ones. If we are trying to predict the last word in "the clouds are in the *sky*," we don't need any further context – it's pretty obvious the next word is going to be sky. In such cases, where the gap between the relevant information and the place that it's needed is small, RNNs can learn to use the past information. But there are also cases where we need more context. Consider trying to predict the last word in the text "I grew up in France… I speak fluent French." Recent information suggests that the next word is probably the name of a language, but if we want to narrow down which language, we need the context of France, from further back. It's entirely possible for the gap between the relevant information and the point where it is needed to become very large. Unfortunately, as that gap grows, RNNs become unable to learn to connect the information.

Long Short-Term Memory networks – usually just called "LSTMs" – are a special kind of RNN, capable of learning long-term dependencies.

So, LSTMs are widely used for sequence prediction problems and have proven to be extremely effective. The reason they work so well is because LSTM is able to store past information that is important, and forget the information that is not. LSTM has three gates:

- *The input gate:* The input gate adds information to the cell state
- *The forget gate:* It removes the information that is no longer required by the model
- *The output gate:* Output Gate at LSTM selects the information to be shown as output

5. Results

Table 2.	LSTM	Prophet	Linear Regression	k-NN	ARIMA
RMSE	9.7	50.0	50.8	63.5	95.9

Let's look at the results of each model implemented, starting from the Linear Regression. As mentioned above our y variable (target variable) is the close price of the stock. What concerns with x variables (features) we didn't have them initially. For this reason, we've extracted that features from the date variable, i.e. day, month, year, Monday / Friday etc. Apart from this we've created another feature which was the "mon_fri" (Monday, Friday) which we thought that would be relevant for the predictions. That feature identifies whether a given day is Monday/Friday or any other working day (Tuesday/Wednesday/Thursday). By implementing the linear regression we've got the following results: RMSE = 50.8 and the graph in Figure 3.

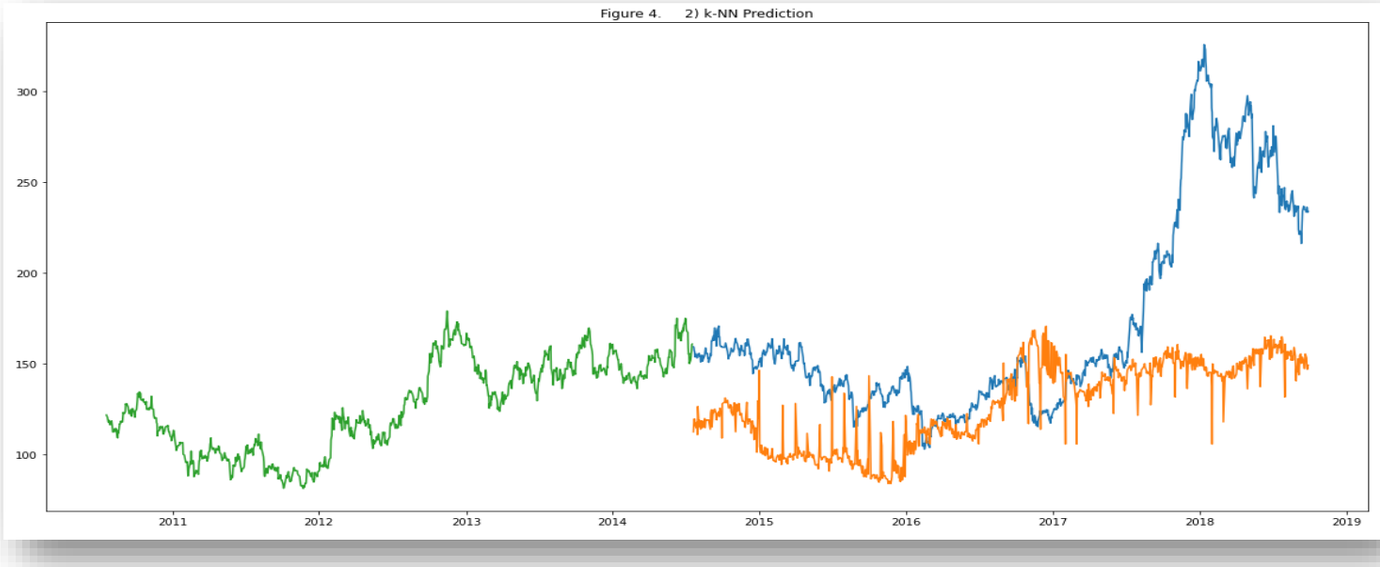
RMSE is the Root Mean Square Error which is the following:

$$\text{RMSE} = \sqrt{\frac{\sum_{i=1}^{n}(P_i - O_i)^2}{n}}$$

Root mean square error takes the difference for each observed (O_i) and predicted (P_i) value, then take the square and sum all of them. Afterwards divide the result by the number of observations and take the square root.

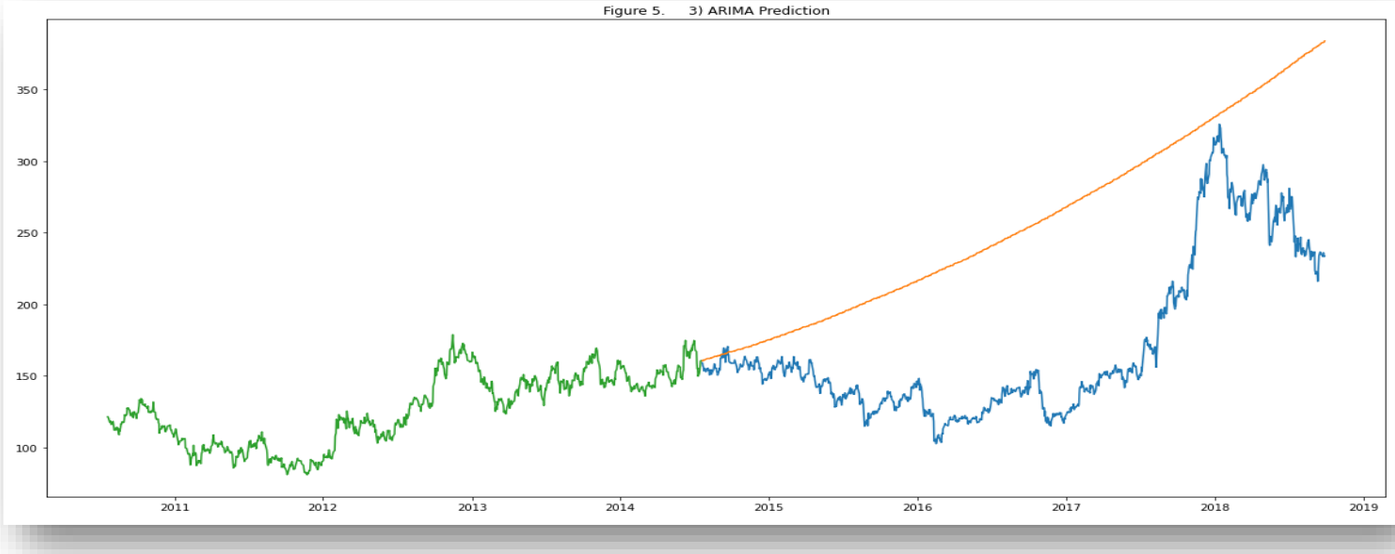
In Figure 3. The green line represents the prediction, blue line train set and the orange line is standing for the test set. Here we can clearly see that by this model the rapid ups and downs aren't taken into account although we have the RMSE quite close to the Prophet model (Table 2.). We know that the linear regression is a simple technique and quite easy to interpret, but there are a few obvious disadvantages. One problem in using regression algorithms is that the model overfits to the date and month column. Instead of taking into account the previous values from the point of prediction, the model will consider the value from the same date a month ago, or the same date/month a year ago.

Now let's look at the results of k-NN model. As mentioned above we took K list of odd numbers [3,5,7,9], and by using grid search and by taking the k = 5 folds in cross validation, this approach calculated the model with RMSE = 63.5 and the graph of prediction you can see below:



Here the green line represents train set, the blue line test set and the orange line is our prediction. From the graph we can see that starting from the middle of 2014 and ending with the middle of 2017 we have almost closer results to the test set. But after 2017 we see that the high jump in the stock prices aren't predicted and here is almost constant trend in the prediction line. While comparing with the linear regression results, we can say that in LR case there were an increasing trend in the prediction. That's much closer to the real price data.

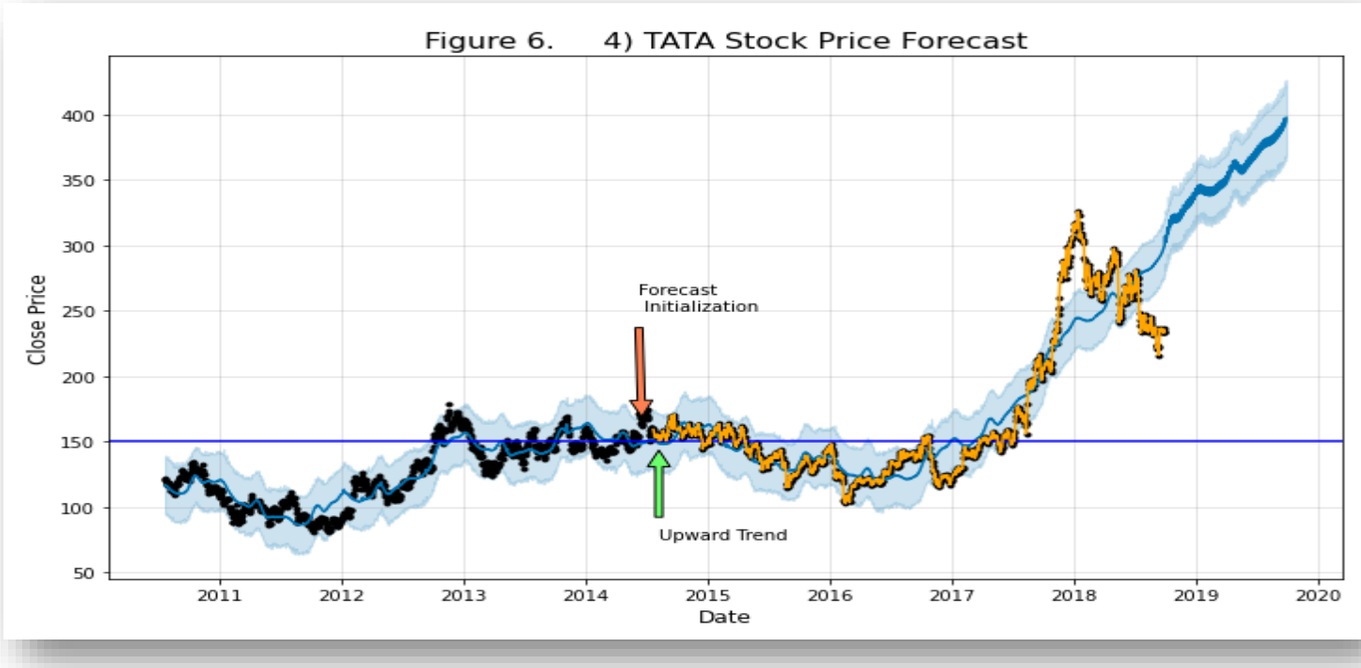
Here we can jump to the next model which is ARIMA. As we know, an auto ARIMA model uses past data to understand the pattern in the time series. Using these values, the model captured an increasing trend in the series.



Of course, there is an increasing trend in the observed data, but as you can see from the blue line there is a pattern of decrease which is not predicted by the way, according to the prediction there is no decrease at all, only increase. The RMSE = 95.9 which is quite high compared with other models.

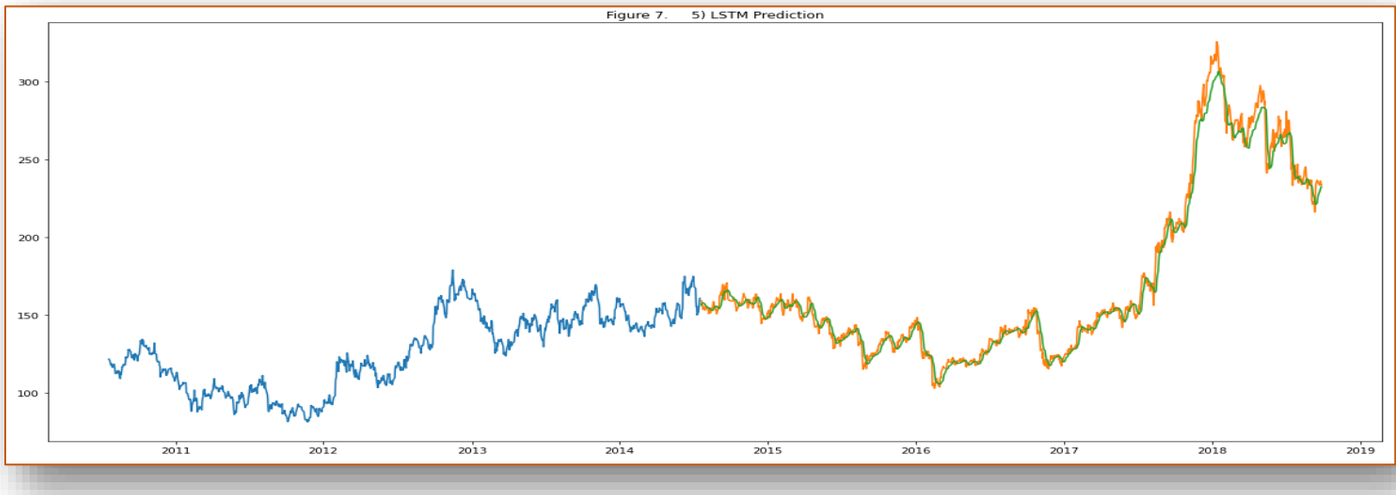
Here we are going to talk about the most popular models in times series predictions, which are Facebook Prophet and LSTM.

Let's first introduce the Prophet and then will go through the LSTM result



Prophet (like most time series forecasting techniques) tries to capture the trend and seasonality from past data. This model usually performs well on time series datasets, but fails to live up to its reputation in this case. As it turns out, stock prices do not have a particular trend or seasonality. It highly depends on what is currently going on in the market and thus the prices rise and fall. Hence forecasting techniques like ARIMA, SARIMA and Prophet would not show good results for this particular problem. Although, compared with previous models this performs very well, RMSE = 50 which is almost closer to the LR result. From the graph we can see that model fitted very well till 2018 but afterwards it couldn't detect the decrease in the prices. And this rapid decrease may be connected with some political decision which isn't taken into consideration.

Now let's see the LSTM results.



Already from the graph (Figure 7.) it's clearly visible that LSTM predicted very well as the test set (orange line) and the prediction (green line) are almost the same. This model could detect all the ups and downs fitted in a way that looking at the lines it's very difficult to separate which one is the line in observed dataset and which one is in predicted dataset. The RMSE = 9.7 also proves the above-mentioned opinion.

6. Conclusion

Concluding from our results, we see that using only past price data we could predict the future prices for a long time period. Lots of companies for their strategic goals (for 3-5 years) need long-term predictions. Here we've got the RMSE = 9.7 result for the long time period using LSTM method. And it's quite obvious that this model performs better than others. But in the future if I had more time and recourses, maybe I would rather go dipper into LSTM model, would try to add other features besides past price data, i.e. political decisions, S&P 500 changes over time etc.

7. Bibliography

ARAT, M. M. (2019, 07 12). A Complete Guide to K-Nearest-Neighbors with Applications in Python. *KNN* , p. 10. Retrieved from https://mmuratarat.github.io/2019-07-12/k-nn-from-scratch

Gandhi, R. (2018, 05 27). Introduction to Machine Learning Algorithms: Linear Regression. *Linear Regression*, p. 5. Retrieved from https://towardsdatascience.com/introduction-to-machine-learning-algorithms-linear-regression-14c4e325882a

Glen, J. (2020, 03 01). Fundamental Analysis vs. Technical Analysis. *Fundamental Analysis vs. Technical Analysis*. Retrieved from http://www.businessdictionary.com/article/1104/fundamental-analysis-vs-technical-analysis-d1412/

Harrison, O. (2018, 09 11). Machine Learning Basics with the K-Nearest Neighbors Algorithm. *towardsdatascience*. Retrieved from https://towardsdatascience.com/machine-learning-basics-with-the-k-nearest-neighbors-algorithm-6a6e71d01761

Limited, T. C. (2020, 05 22). *tataconsumer*. Retrieved from TATA Global Beverages Limited: https://www.tataconsumer.com/?reload

Sangarshanan. (2018, 10 03). Time series Forecasting — ARIMA models. *towardsdatascience*. Retrieved from https://towardsdatascience.com/time-series-forecasting-arima-models-7f221e9eee06

Sean J. Taylor, B. L. (2017, 02 23). Prophet: forecasting at scale. *research.fb*. Retrieved from https://research.fb.com/prophet-forecasting-at-scale/

SRIVASTAVA, P. (2017, 12 10). Essentials of Deep Learning : Introduction to Long Short Term Memory. *analyticsvidhya*. Retrieved from https://www.analyticsvidhya.com/blog/2017/12/fundamentals-of-deep-learning-introduction-to-lstm/