Stock Price Prediction using LSTM, Facebook Prophet and ARIMA

Gulab Patel[1] **and Ajay Kumar Patel**[1,*]

[1]Indian Institute of Technology (BHU), Mathematical Sciences, Varanasi, 221005, India
[+]gulabpatel923@gmail.com
[*]ajaykrpatel.rs.mat17@gmail.com

ABSTRACT

Now a days forecasting the stock prices has been a difficult task for many of the researchers and analysts. In fact, stock market being highly volatile, there is a huge amount of uncertainty and risk associated with them. For a good and successful investment, many investors are keen in knowing the future situation of the stock market. Good and effective prediction systems for stock market help traders, investors, and analyst by providing supportive information like the future direction of the stock market. This article presents three innovative method to predict the future closing prices of stocks using combination of deep learning approach using Long Short-Term Memory (LSTM), Facebook prophet and Auto Regressive Integrated Moving Average (ARIMA) time series model to predict the future closing prices of stocks. These three models are the most efficient model for time series prediction.

Keywords: Stock, LSTM, Facebook Prophet, ARIMA, Mean Square Error(MSE), Root Mean Square Error (RMSE).

1 Introduction

The Stock market forecast itself is a rough and high hazard, because of ambiguous and unforeseeable nature of it, as well as complicated financial problems. It is affected by noise and many hidden factors. Nobody knows the specific time of bearish or bullish the market. The Forecast situation becomes more difficult when the exchange market reflects an economic system without stability, regulation and money discipline. The basic reason for permanent fluctuation in economical data of any country are the relations between state and market, the gap between rich and poor, population growth, structural change, international debt and finance, and in some cases permanent sanctions. There are three main categories of stock market analysis and prediction such as (a) Fundamental analysis, (b) Technical analysis and (c) Time series analysis. Time series analysis is an approach to analyze time series data to extract meaningful characteristics of data and generate other useful insights applied in business situation. Generally, time-series data is a sequence of observations stored in time order.

Time series analysis helps to understand time based patterns of a set of metric data points which is critical for any business. Techniques of time series forecasting could answer business questions like how much inventory to maintain, how much website traffic do you expect in your e-store, how many product will be sold in the next month — all of these are important time series problems to solve. The basic objective of time series analysis usually is to determine a model that describes the pattern of the time series and could be used for forecasting. Thus; making a robust forecasting model for a stock price prediction for given unmodified data is a challenging issue.

Linear regression is one of the solutions for stock market prediction however, the idea works better for short terms[1]. Most of the stock forecasting techniques with time series data normally use either a linear such as AR, MA, ARIMA, ARMA, CARIMA, etc.[2–5] or non-linear models (ARCH, GARCH, ANN, RNN, LSTM, etc.)[6,7] and another technique which is very usefull nowadays name as FBProphet[8–10]. It uses time as a regressor and tries to fit several linear and nonlinear function of time as components. By default, FBProphet will fit the data using a linear model but it can be changed to the nonlinear model (logistics growth) from its arguments. We have tried using the historical stock prices as the basis for time series analysis to forecast future stock prices. Many different statistical models we were applied since long like LSTM, ARIMA and FBProphet.

This paper presents a methodology of using Long Short-Term Memory (LSTM) cells, a type of RNN, in combination with a time series model, called as Auto Regressive Integrated Moving Average (ARIMA), and Facebook Prophet. The output of these three models are combined in a Feed forward Neural Network for predicting the final value of future price. The next section of the paper will be methodology where we will explain about each process in detail. After that, we will have pictorial representations of the analysis that we have used and we will also reason about the results achieved.

2 Methodology

Various types of libraries and tools can be developed by the combination of different factors like network topology, training method etc. For this experiment, we have considered Long Short-Term Memory(LSTM), Auto Regressive Integrated Moving Average (ARIMA) and FBProphet networrks.
In this section we discuss the methodologies for our considered model. Our model consists of several stages which are given below as follows:-

- **Stage 1: Raw Data:**
  In this stage, the historical "APPLE" stock data is collected from the link https://finance.yahoo.com/quote/AAPL/history/ and this historical data is used for the prediction of future stock prices.

- **Stage 2: Data Preprocessing:**
  The pre-processing stage involves

  (a) Data discretization: Part of data reduction but with particular importance, especially for numerical data.

  (b) Data transformation: Normalization.

  (c) Data cleaning: Fill in missing values.

- **Stage 3: Visualization:** Here, we plot the graph and Visualize the behaviour of closing prices of APPLE dataset,

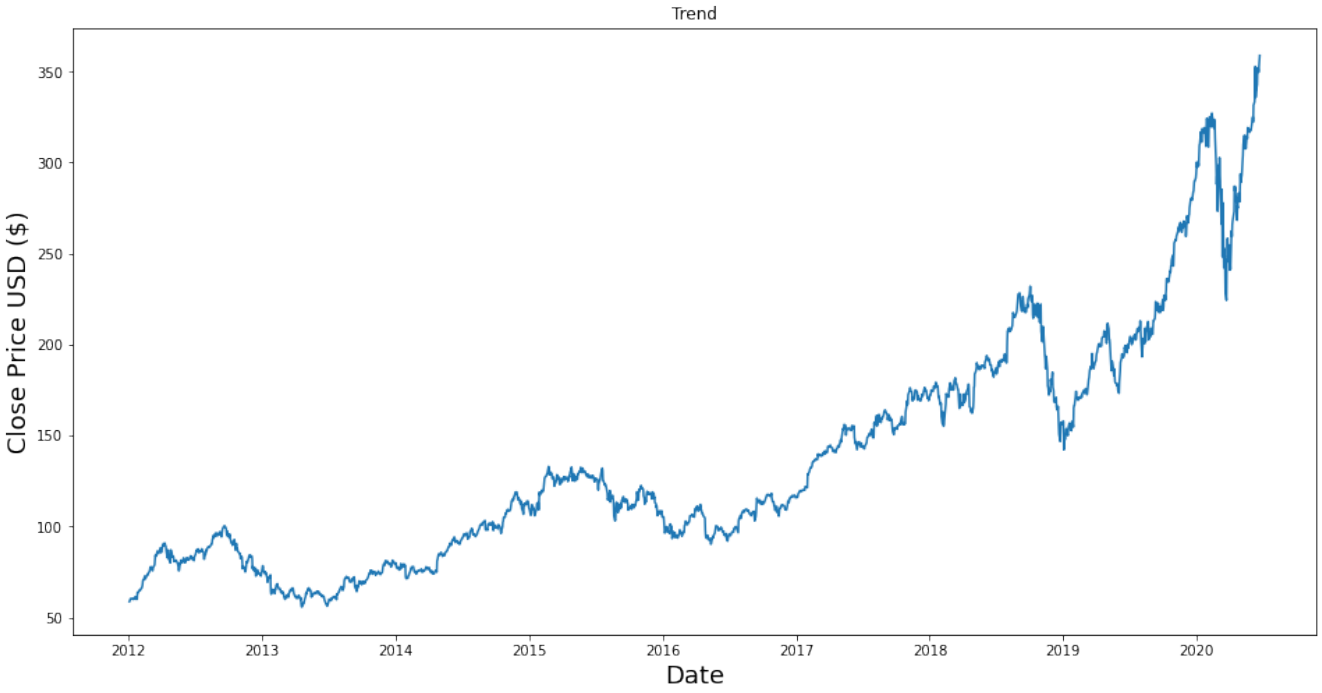

Figure 1. APPLE dataset

After the dataset is transformed into a clean dataset, the dataset is divided into training and testing sets so as to evaluate. Here, we take 80% for training set and 20% for testing set of the total dataset.

- **Stage 4: Feature Extraction:**
  In this layer, only the features which are to be fed to the models are chosen. We choose the feature from Date, Open, High, Low, Close, Volume and Adj Close. For our models we do time series forecasting and we select price and date as considered features.

- **Stage 5: Training of Models:**

- **1: Training LSTM:** The data is fed to the LSTM network and trained for prediction assigning random biases and weights. Our LSTM model is composed of a sequential input layer followed by 2 LSTM layers, one dense layer and then finally a dense output layer.
- **2: Training FBProphet:** FBProphet follows the similar API as sklearn model. We create an instance of the Prophet class and then call its fit and predict methods.

  The input to Prophet is always a dataframe with two columns: ds and y. The ds (datestamp) column should be of a format expected by Pandas, ideally YYYY-MM-DD for a date or YYYY-MM-DD HH:MM:SS for a timestamp. The y column must be numeric, and represents the measurement we wish to forecast.
- **3: Training ARIMA:**In ARIMA model, there are three parameters lag order(p), degree of difference(d) and order of moving average(q). The values of d is calculated from making the data stationary and p and q is calculated from the partial auto-correlation and auto-correlation plot. Then we fed the training set to the ARIMA model.

- **Stage 6: Output Generation:**
  For LSTM, the output value generated by the output layer of the LSTM is compared with the labeled value. The error or the difference between the target and the obtained output value is minimized by using back propagation algorithm which adjusts the weights and the biases of the network. For FBProphet and ARIMA, we feed the test set to the model and it uses the learned parameters during the training phase and predicts the value for test set.

2.1 Long Short-Term Memory (LSTM) Networks

LSTM uses one of the most common forms of RNN[11]. This time recurrent neural network is meant to avoid long-term dependence problems and is suitable for processing and predicting time series. Proposed by Sepp Hochreiter and Jurgen Schmidhuber in 1997,[12] the LSTM model consists of a unique set of memory cells that replace the hidden layer neurons of the RNN, and its key is the state of the memory cells. The LSTM model filters information through the gate structure to maintain and update the state of memory cells. Its door structure includes input, forgotten, and output gates. Each memory cell has three sigmoid layers and one tanh layer. Figure 2 displays the structure of LSTM memory cells. The forgotten gate in the LSTM unit determines which cell state information is discarded from the model. As shown in Figure 2, the memory cell accepts the output h_{t-1} of the previous moment and the external information x_t of the current moment as inputs and combines them in a long vector $[h_{t-1}, x_t]$ through σ transformation to become

$$f_t = \sigma(W_f.[h_{t-1}, x_t] + b_f) \tag{1}$$

where W_f and b_f are, respectively, the weight matrix and bias of the forgotten gate, and σ is the sigmoid function. The forgotten gate's main function is to record how much the cell state C_{t-1} of the previous time is reserved to the cell state C_t of the current time. The gate will output a value between 0 and 1 based on $h_t - 1$ and x_t, where 1 indicates complete reservation and 0 indicates complete discardment.

The input gate determines how much of the current time network input x_t is reserved into the cell state C_t, which prevents insignificant content from entering the memory cells. It has two functions. One is to find the state of the cell that must be updated; the value to be updated is selected by the sigmoid layer, as in Eq (2). The other is to update the information to be updated to the cell state. A new candidate vector $\tilde{C}_t$ is created through the tanh layer to control how much new information is added, as in Eq (3). Finally, Eq (4) is used to update the cell state of the memory cells:

$$i_t = \sigma(W_i.[h_{t-1}, x_t] + b_i) \tag{2}$$

$$\tilde{C}_t = tanh(W_c.[h_{t-1}, x_t] + b_c) \tag{3}$$

$$C_t = f_t * C_{t-1} + i_t * \tilde{C}_t \tag{4}$$

The output gate controls how much of the current cell state is discarded. The output information is first determined by a sigmoid layer, and then the cell state is processed by tanh and multiplied by the output of the sigmoid layer to obtain the final output portion:

$$O_t = \sigma(W_o.[h_{t-1}, x_t] + b_o) \tag{5}$$

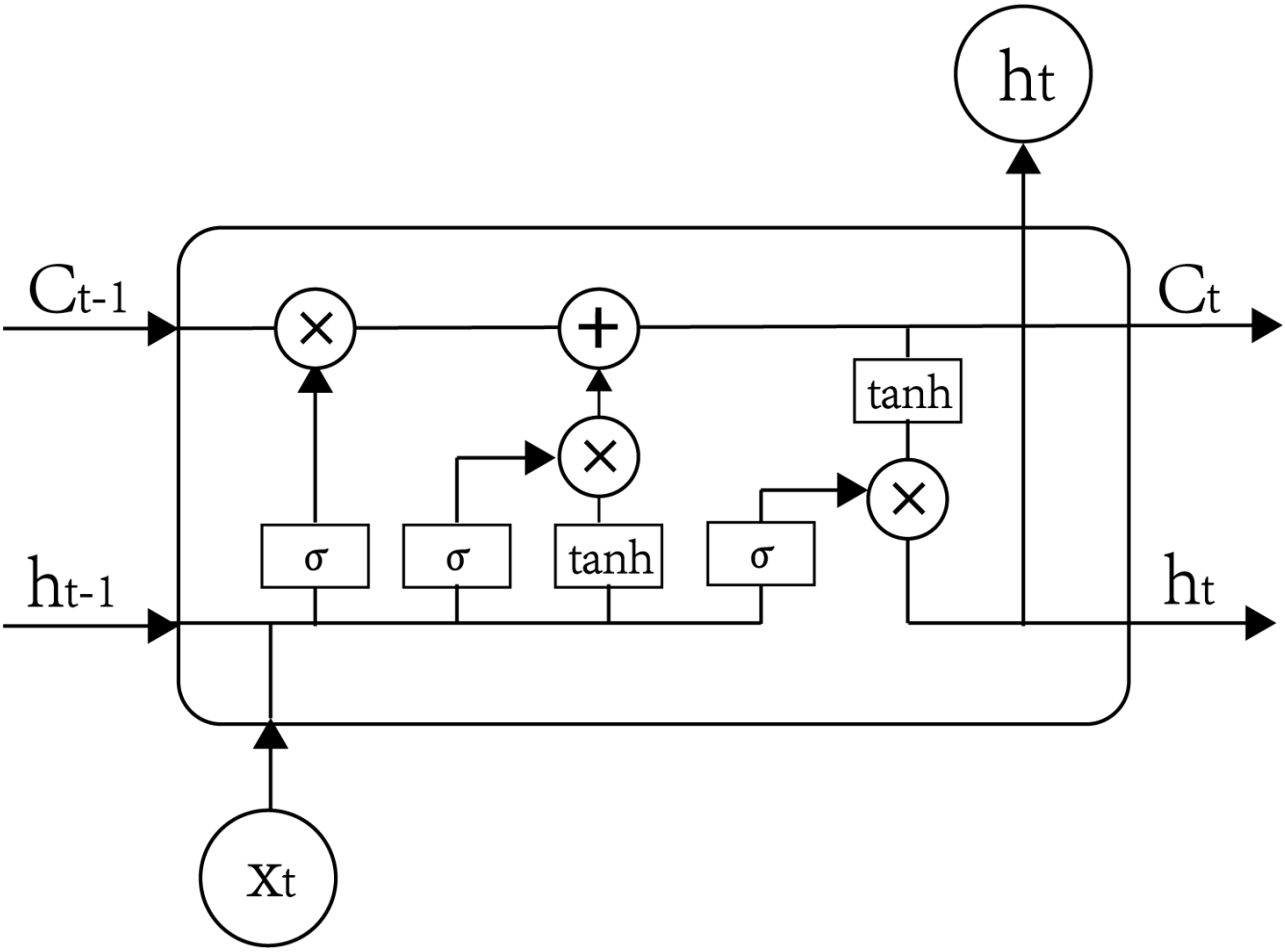

Figure 2. Structure of long short term memory(LSTM).

The final output value of the cell is defined as:

$$h_t = O_t * tanh(C_t) \tag{6}$$

Prediction: LSTM is a great neural network model and it is able to learn the pattern of our data well. Now, from the following figure [3] we see how good LSTM is for time series prediction

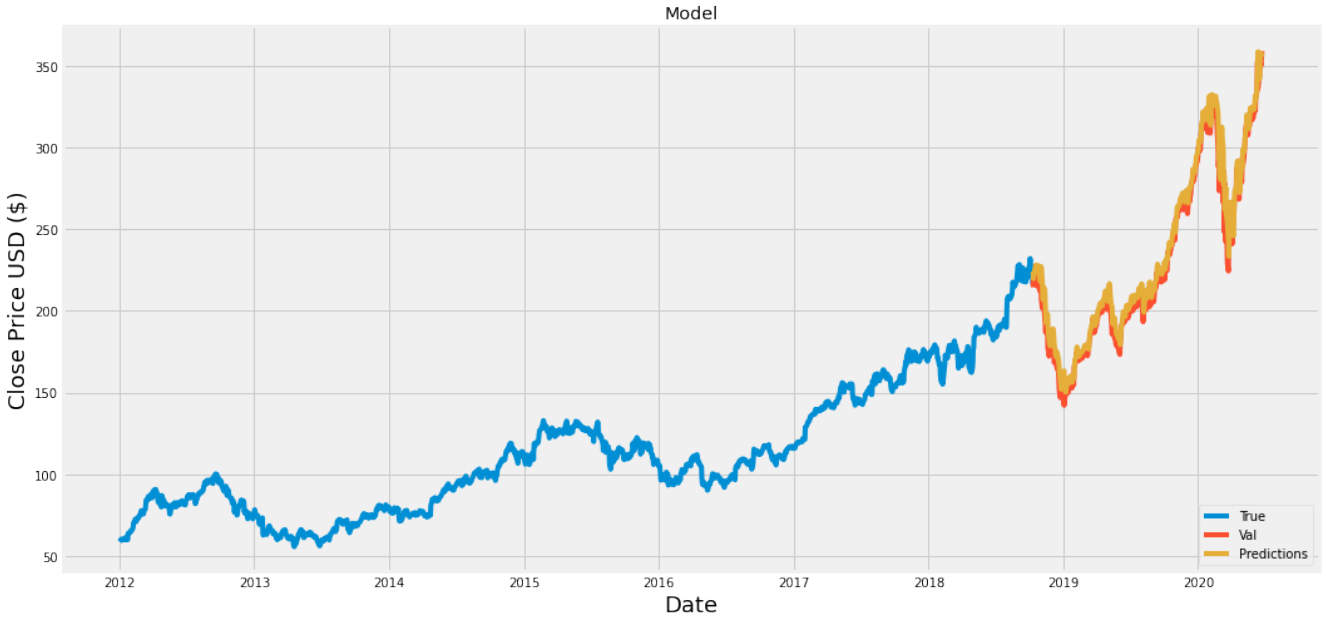

Figure 3. Time series prediction using LSTM

2.2 Facebook Prophet(FBProphet)

Facebook made its Prophet forecasting tool available to the public in early 2017. In the article announcing Prophet citeprophet, the authors justify the decision after observing that "completely automatic forecasting techniques can be brittle, and they are often too inflexible to incorporate useful assumptions or heuristics" and that "analysts who can produce high quality forecasts are quite rare because forecasting is a specialized data science skill requiring substantial experience." In addition to the ease of use, Prophet promises to handle trend, seasonality, holidays and to deal better with outliers, missing observations and trend changes.

In this article, we shall cover some background on how Prophet fills the existing gaps in generating fast reliable forecasts followed by a demonstration using Python.

What's new in Prophet: It provides intuitive parameters which are easy to tune. Even someone who lacks deep expertise in time-series forecasting models can use this to generate meaningful predictions for a variety of problems in business scenarios.

2.2.1 Highlights of Facebook Prophet

These are some good features of FBProphet:

- Very fast, since it's built in Stan, a programming language for statistical inference written in C++.
- An additive regression model where non-linear trends are fit with yearly, weekly, and daily seasonality, plus holiday effects:
  - A piecewise linear or logistic growth curve trend. Prophet automatically detects changes in trends by selecting changepoints from the data
  - A yearly seasonal component modeled using Fourier series
  - A weekly seasonal component using dummy variables
  - A user-provided list of important holidays.
- Robust to missing data and shifts in the trend, and typically handles outliers .
- Easy procedure to tweak and adjust forecast while adding domain knowledge or business insights.

The Prophet Forecasting Model

We use a decomposable time series model with three main model components: trend, seasonality, and holidays. They are combined in the following equation:

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

- **g(t):** piece-wise 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

Prophet in action (using Python)

Currently implementations of Prophet are available in both Python and R. They have exactly the same features. Prophet() function is used do define a Prophet forecasting model in Python. For the detail description about most important parameters go through the link https://facebook.github.io/prophet/docs/trend_changepoints.html and https://www.analyticsvidhya.com/blog/2018/05/generate-accurate-forecasts-facebook-prophet-python-r/

There are different terminology that are used in FBProphet time series forecasting models. A brief introduction about those terms is given below as:

For the time series prediction, the first step is to check whether the data is stationary. As we see in the figure [4] that our data is not stationary, we convert our data into stationary form. In the following figure our data is now stationary:

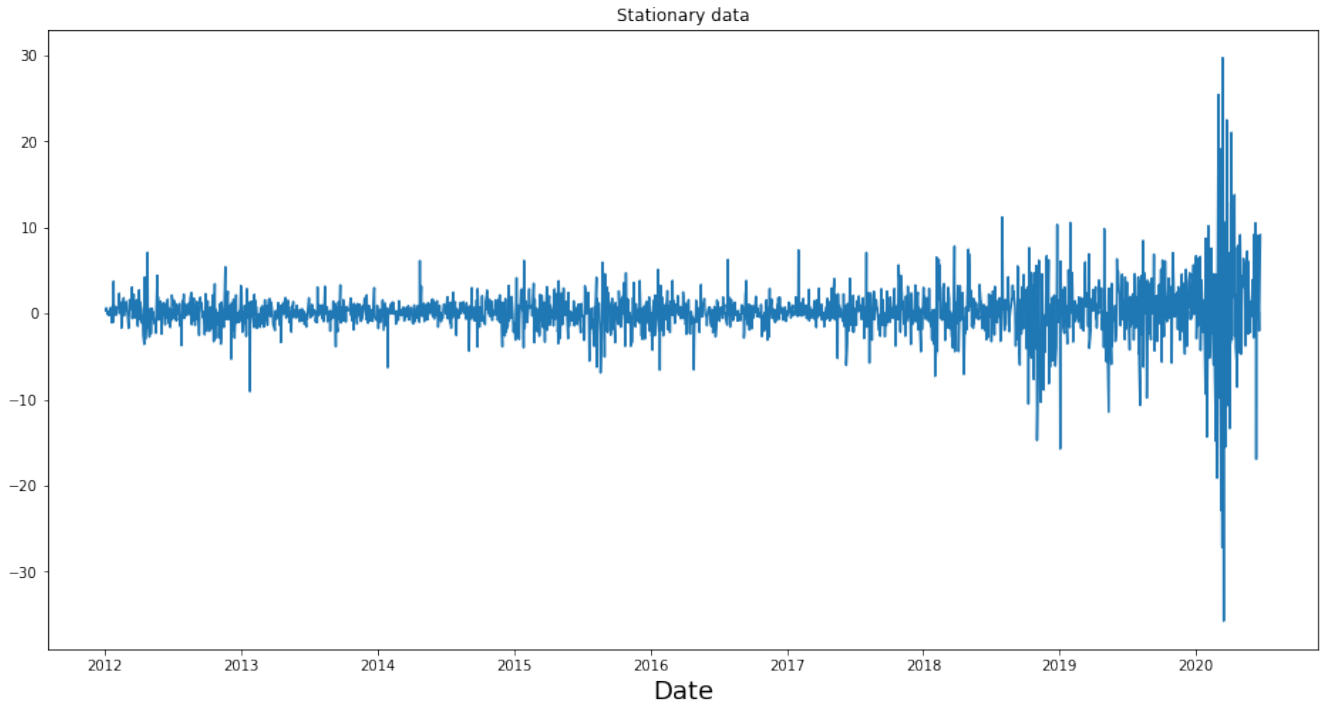

Figure 4. Stationary data

Trend: Trend is modelled by fitting a piece wise linear curve over the trend or the non-periodic part of the time series. The linear fitting exercise ensures that it is least affected by spikes/missing data.

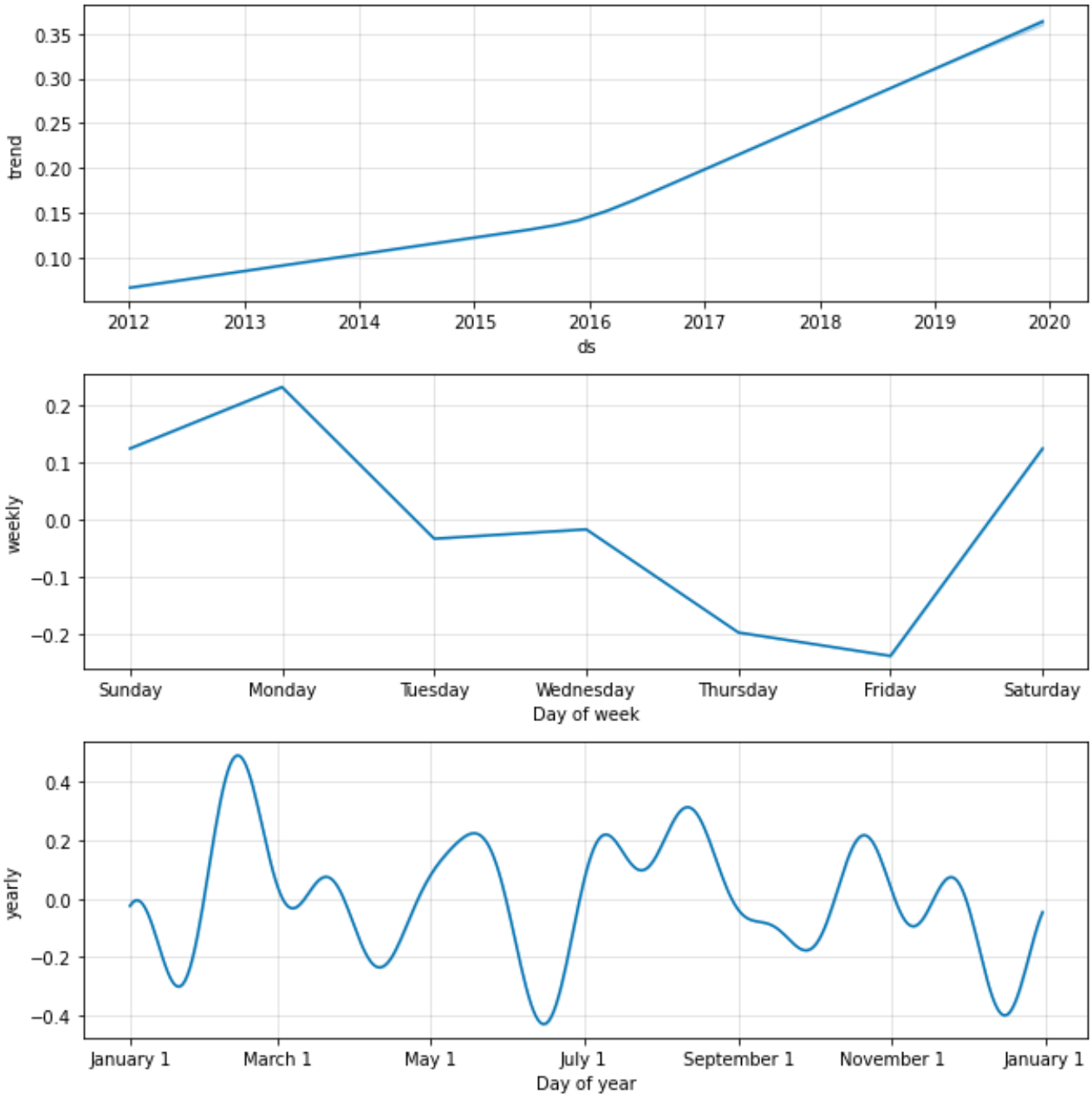

Figure 5. Trend of the prediction

Saturating growth : An important question to ask here is – Do we expect the target to keep growing/falling for the entire forecast interval? More often than not, there are cases with non-linear growth with a running maximum capacity.

Let's say we are trying to forecast number of downloads of an app in a region for the next 12 months. The maximum downloads is always capped by the total number of smartphone users in the region. The number of smartphone users will also, however, increase with time.

Changepoints : Another question to answer is whether my time series encounters any underlying changes in the phenomena e.g. a new product launch, unforeseen calamity etc. At such points, the growth rate is allowed to change. These changepoints are automatically selected. However, a user can also feed the changepoints manually if it is required. In the following graph. we do not see any significant change point because out data is very stationary.

Prediction: FBProphet follows the similar API as sklearn model. We create an instance of theProphet class and then call its fit and predict methods.The input to Prophet is always a dataframe with two columns: ds and y. In the following figure, we see FBProphet is a great tool for time series prediction:

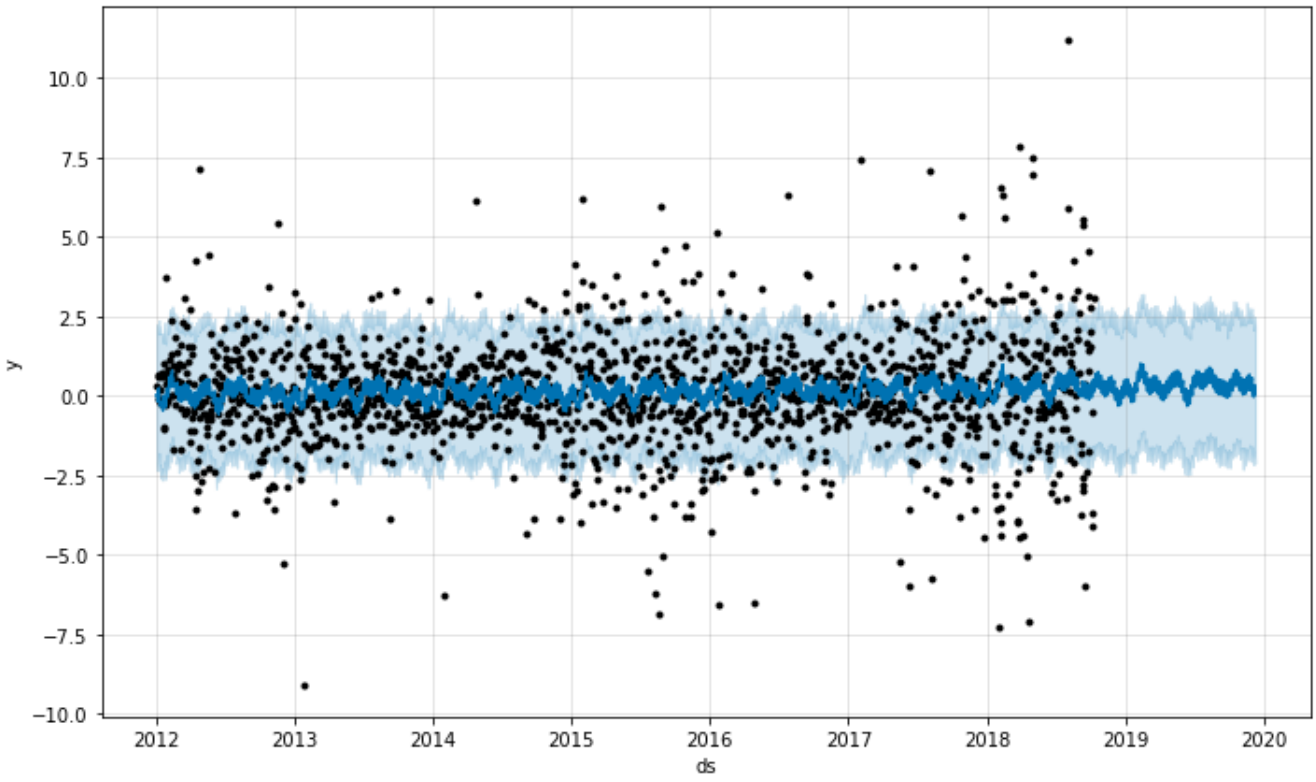

Figure 6. Prediction of APPLE data

2.3 Auto Regressive Integrated Moving Average (ARIMA)

The Auto-Regressive Integrated Moving Average (ARIMA)[13,14] method uses a higher level of univariate models of the type $\tilde{y}_t = f(\mathbf{y}_{t-1})$. In contrast to the Weighted Moving Average method, the general forecasting equations are no longer linear, and iterative techniques are to be used to find solutions. Upon closer inspection, the basic version of ARIMA equals the Weighted Moving Average method. The data of event which occurs after a fixed time interval is termed as a time series. Time series are an important class of data, which are regularly dealt in real time applications. Time series analysis deals with the fetching inferences and conclusions by analyzing a time series with help of statistical and mathematical methods. This paper uses a time series model called as Auto Regressive Integrated Moving Average (ARIMA), which is mainly used in non-stationary time series analysis.

ARIMA is an acronym that stands for AutoRegressive Integrated Moving Average. It is a generalization of the simpler AutoRegressive Moving Average and adds the notion of integration.

This acronym is descriptive, capturing the key aspects of the model itself. Briefly, they are

- **AR:** *Autoregression.* A model that uses the dependent relationship between an observation and some number of lagged observations.
- **I:** *Integrated.* The use of differencing of raw observations (e.g. subtracting an observation from an observation at the previous time step) in order to make the time series stationary.
- **MA:** *Moving Average.* A model that uses the dependency between an observation and a residual error from a moving average model applied to lagged observations.

The parameters of the ARIMA model are defined as follows:

- **p:** The number of lag observations included in the model, also called the lag order.

- **d:** The number of times that the raw observations are differenced, also called the degree of differencing.
- **q:** The size of the moving average window, also called the order of moving average.

A linear regression model is constructed including the specified number and type of terms, and the data is prepared by a degree of differencing in order to make it stationary, i.e. to remove trend and seasonal structures that negatively affect the regression model.

Prediction: ARIMA is a great tool for time series prediction and it is being used most and gives very accurate result. In the Figure 7 we see that ARIMA performs well for time series prediction:

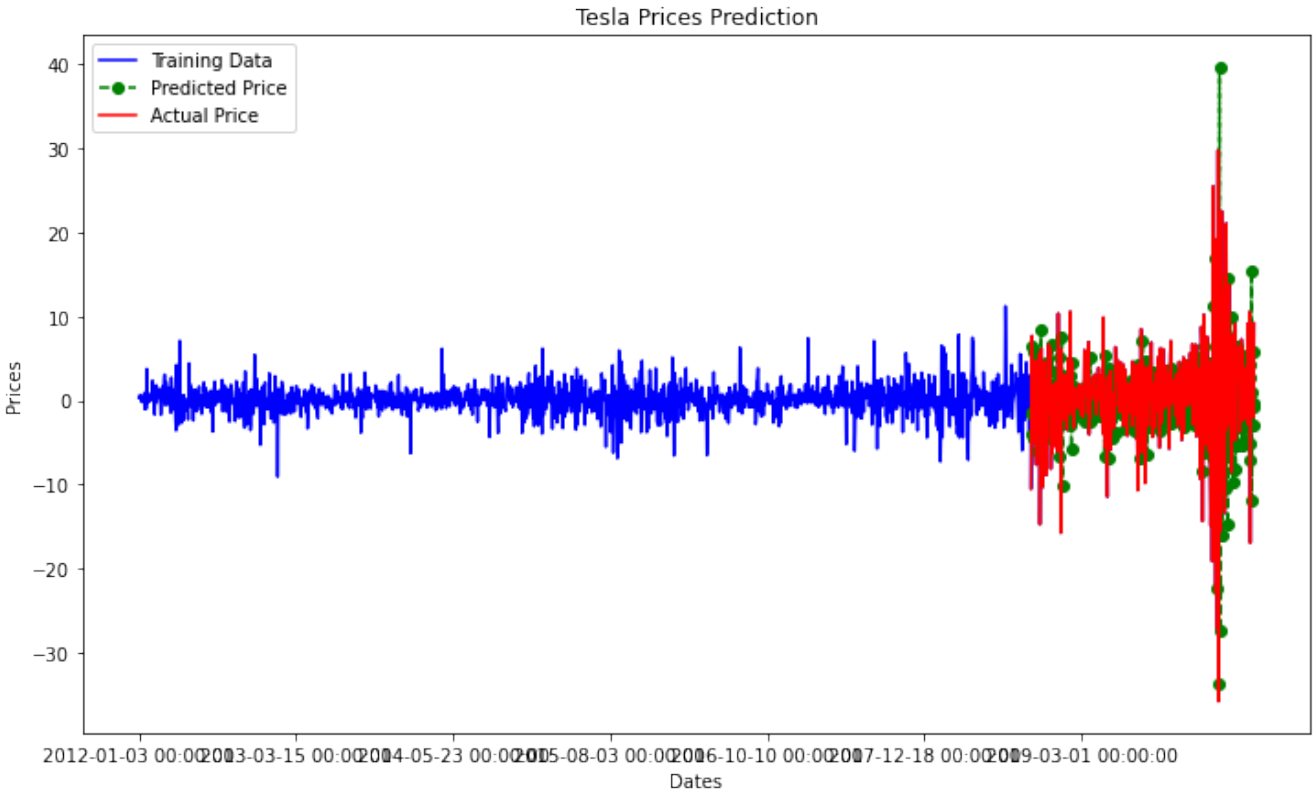

Figure 7. Time series prediction using ARIMA

3 Analysis and Experimental Results

We evaluated the prediction results and the established prediction model by the mean square error (MSE), root mean square error (RMSE). The error or the difference between the target and the obtained output value is minimized by using MSE and RMSE values. RMSE is the square root of the mean/average of the square of all of the error. The use of RMSE is highly common and it makes an excellent general purpose error metric for numerical predictions.

$$MSE = \frac{1}{N}\sum_{i=1}^{N}(\hat{y}_i - y_i)^2 \tag{7}$$

$$RMSE = \sqrt{\frac{1}{N}\sum_{i=1}^{N}(\hat{y}_i - y_i)^2} \tag{8}$$

where N denotes the number of samples, is the model prediction value, y_i is the real value, and is the mean value of y_i.

In the following table we compare the RMSE for the three above discussed approaches and we see all the three models are great for time series predictions, the table is given as follows:

Models Error	**MSE**	**RMSE**
LSTM	4.8788	2.2088
FBProphet	4.0716	1.9911
ARIMA	61.651	7.852

Table 1. LSTM, ARIMA and FBProphet testing results.

4 Conclusion:

In this article, we have presented a state-of-the-art of the following popular time series forecasting models with their salient features:

- LSTM
- FBProphet
- ARIMA

It has been seen that, the proper selection number of layes in LSTM network and hyper-parameters p,d and q in ARIMA model is very crucial for successful forecasting. In case of LSTM, we tune it for different LSTM and Dense layers and we find that two LSTM and two Dense layer is performs best. In case of ARIMA model we plot auto-correlation and partial auto-correlation and from the graph we choose the value of p and q, but still one need to tune the p and q value. From the plot of stationary data we choose the value of I, the most common value of I is one or two. As FBProphet is an advance tool it does most of the calculation internally so very easy to use. We have considered a few important performance measures for comparing the considered three forecasting models. As we see all the models performs well but we see FBProphet wins the battle here. However in some cases, significant deviation can be seen among the original observations and forecasted values. In such cases, we suggest that sufficient data, suitable data preprocessing and proper hyper-parameters tuning may improve the forecast performances. Time series forecasting is a fast growing area of research and as such provides many scope for future works.

References

1. Ariyo, A. A., Adewumi, A. O. & Ayo, C. K. Stock price prediction using the arima model. *2014 UKSim-AMSS 16th International Conference on Computer Modelling Simulation* 106–112, DOI: https://doi.org/10.1109/UKSim.2014.67 (2014).
2. Fama, E. F. Efficient capital markets: a review of theory and empirical work. *The Journal Finance* **25**, 383–417, DOI: https://doi.org/10.2307/2325486 (1970).
3. Farhath, Z. A., Arputhamary, B. & Arockiam, L. A survey on arima forecasting using time series model. *International Journal Computer Science Mobile Computing* **5**, 104–109, DOI: https://www.ijcsmc.com/docs/papers/August2016/V5I8201626.pdf (2016).
4. Ghosh, A., Bose, S., Maji, G., Debnath, N. & Sen, S. Stock price prediction using lstm on indian share market. *Proceedings 32nd International Conference on Computer Applications Ind. Engineering* **63**, 101–110, DOI: https://doi.org/10.29007/qgcz (2016).
5. Wichaidit, S. & Kittitornkun, S. Predicting set50 stock prices using carima (cross correlation arima). *2015 International Computer Science Engineering Conference (ICSEC), IEEE* 1–4, DOI: https://doi.org/10.1109/ICSEC.2015.7401453 (2015).
6. Roondiwala, M., Patel, H. & Varma, S. Predicting stock prices using lstm. *International Journal Science Research (IJSR)* **6**, 1754–1756, DOI: https://www.ijsr.net/archive/v6i4/ART20172755.pdf (2017).
7. Kim, T. & Kim, H. Y. Forecasting stock prices with a feature fusion lstm-cnn model using different representations of the same data. *PloS one* **14**, e0212320, DOI: https://doi.org/10.1371/journal.pone.0212320 (2019).
8. Fang, W.-X. *et al.* Combine facebook prophet and lstm with bpnn forecasting financial markets : the morgan taiwan index. *2019 International Symp. on Intell. Signal Processing Communication Systems (ISPACS)* 1–2, DOI: https://doi.org/10.1109/ISPACS48206.2019.8986377 (2019).
9. Weytjens, H., Lohmann, E. & Kleinsteuber, M. Cash flow prediction: Mlp and lstm compared to arima and prophet. *Electronic Commerce Research,* 1–21, DOI: https://doi.org/10.1007/s10660-019-09362-7 (2019).
10. Taylor, S. J. & Letham, B. Prophet: Forecasting at scale. DOI: https://research.fb.com/prophet-forecasting-at-scale/ (2017).
11. LeCun, Y., Bengio, Y. & Hinton, G. Deep learning. *Nature* **521**, 436–444, DOI: https://doi.org/10.1038/nature14539 (2015).

12. Abedinia, O., Amjady, N. & Zareipour, H. A new feature selection technique for load and price forecast of electrical power systems. *IEEE Transactions on Power Systems* **32**, 62–74, DOI: https://doi.org/10.1109/TPWRS.2016.2556620 (2017).

13. Reddy, V. Data analysis course, time series analysis and forecasting. DOI: https://http://www.trendwiseanalytics.com/training/Timeseries_Forecasting.pdf (2018).

14. Nau, R. Introduction to arima: Nonseasonal models. DOI: https://people.duke.edu/~rnau/411arim.htm (2018).