Solve the problem of forecasting the change in stock prices using Machine Learning methods.

by
Hena Agnon

Abstract

The research topic "Solve the problem of forecasting the change in stock prices with using machine learning methods" mainly aims to evaluate several models of machine learning to predict the stock price of three companies namely Amazon, Apple and Google. Stock exchange prices being time series, many techniques in machine learning have been implemented to predict the time series. Time series prediction can be applied in various fields, especially in the financial field. This thesis examined a study on KERAS time series prediction with LSTM (Long Short-Term Memory), ARIMA and PROPHET models, these models were developed to predict daily adjusted close price of selected stocks with the last 10 years of data collected from Yahoo Finance (from April 1, 2011 to April 1,2021). The results of the prediction show that all three-time series models have a high potential in predicting time series. Keras with LSTM gives better results compared to the other two time series prediction models.

Chapter 1. Introduction

"The Stock Market is a device for transferring money from the impatient to the patient."-Warren Buffet (American Investor & CEO of Berkshire Hathaway)

"The four most dangerous words in investing are: This time it's different."- Sir JohnTempleton (British Investor)

1.1. Terms and Definition

What is Stock and Stock Market?

The Stock market is a prediction of buying and selling by a company on a given business day. The stock exchange is like an auction where the investors can sell and buy the shares of stocks and the investors who hold the share of these stocks in a company or organization are called as shareholders or stockholders. A stock exchange is an exchange where traders and stockbrokers buy and sell shares. The stock market mainly deals with the securities and these securities are involved in such a way that it is a fixed interest security and the trade in stock exchange means transfer of a security or stock from buyer to seller [1].

The financial activities of the stock market are carried through institutionalized formal exchanges or Over the Counter (OTC) marketplace which will be mainly operated under a regular set of regulations. The stock market and the stock exchange, both the term is used interchangeably in business but the latter is subset of the former (example: If one trades in share market that means that they can buy and sell shares on one of the stock exchange that is part of entire stock market.

The stock market which is primarily used for trading stocks can also be helpful to other financial securities like corporate bonds, exchange traded funds and some other basic derivatives such as commodities, currencies and bonds that is traded in the stock market. In today's world most of the people can do the purchase in online but some stuff can only be bought outside the online for which the stock market plays a crucial role in which it is designated in a similar fashion of market for various trading securities in a controlled, secure and managing environment. The main factor of the stock market is even though it brings hundreds of thousands of participants together who wish to buy or sell shares making them to have a fair pricing and transparency in transactions [2].

The working of the stock market is quite simple as it allows general public to take a choice whether or not to invest their shares in a particular organization, by allowing them to buy or sell an organization shares to the public through the process of initial public offerings. This activity will help an organization to deal with its investors. In order to initiate these processes an organization needs a marketplace where it can sell its shares and the place of this required marketplace is called as stock market.

What is Stock Market Prediction?

The future value of an individual stock, a market or the whole market can be aimed to be predicted, this is known as stock market prediction. It generally uses technical analysis of charts or the fundamental analysis of a company or combination of two. Since all investors wants to predict the value of stocks there will be more stock market prediction by self-styled experts in media and published by investment advisors and brokers [3].

In short, the stock market prediction is all about determining the future value of a company with which the company's financial instrument traded on an exchange. The success of the stock market price of the future will give a significant profit in the field of stock market. There are various aspects of the stock market prediction which are as follows: Fundamental analysis, Technical analysis, and the Machine learning.

Fundamental Analysis is mainly based on the past performance of the company which gives the credibility of its accounts and it is mainly done with the use of performance ratio which is mainly helpful for doing it.

Technical Analysis is not considered with the company's fundamental, here the techniques are used to predict the future price of the stocks and these techniques includes of the Exponential Moving Average (EMA), oscillators and volume indicators etc. Candle stick patterns developed by Japanese rice merchants are most used technical analysis nowadays.

Machine Learning is another way in which stock market can be predicted and it uses many algorithms with which all these algorithms are based on the Artificial Neural Networks (ANN) and Genetic Algorithms. The main aspect of the artificial neural network is the use of time series forecasting with which using this forecasting the anticipating trend changes of the numerous stock markets and time series datasets.

The predictive analysis is being utilized by organizations or companies, individuals everywhere throughout the world by extracting the historical data. The different techniques in executing the forecast framework like Machine Learning, Technical Analysis, Fundamental Analysis, and Time series Forecasting. By advancing the innovation, the Artificial

Intelligence includes the man-made consciousness which enables the framework to prepare and improve as a matter of fact [4].

What is Time series forecasting?

The most important aspect of machine learning involves with the time series forecasting and it is important because most of the prediction problems including time component are dealt here. Before we see time, series forecasting let us see more about the time series. The time series is modelled through a stochastic process and mainly constituted with the four components which are: Level, Trend, Seasonality and Noise.

Level is a value for baseline in the series if it is straight line.

Trend is linear increasing or decreasing behavior of the series over time.

Seasonality is cycles of behavior or repeat patterns over time.

Noise is a change in observations which cannot be explained for the given model.

The time series forecasting is all about making predictions about the future and the process of predicting future is called as extrapolation in statistical handling of the time series. The modern field mainly focus on the topic and hence refer to it as a time series forecasting. Forecasting mainly involves with taking the historical data that fits the model and using them to predict future observations and important aspect which has to be considered is future cannot happen and it can be predicted only with estimation of already happened events.

The purpose of time series forecasting involves with mainly two folds which are: to understand or model the stochastic mechanisms which leads to an observed series and to predict or forecast the future values of the series based on the already known historical data of that series. The time

series forecasting is mainly used in R and Python and it has many kind of models which are as follows: Naïve, Exponential smoothing, ARIMA, Dynamic linear models, FBProphet, LSTM. Out of which ARIMA, FBProphet and LSTM are mainly used in this paper.

Figure 1: The Amazon company stock price representation in Yahoo Finance. [5]

The figure 1. shows an example of time series data representation of a Amazon company over a period of 10 years.

1.2. Overview of approach

This thesis primarily centered around the stock market price prediction of tree world-famous companies (Amazon, Apple and Google) by using time series models. Yahoo Finance is utilized for gathering 10 years of data. Close price value in the dataset is used for the future price prediction. For this KERAS with LSTM (Long Short Term-Memory), Prophet and ARIMA algorithms are used.

Python version 3 programming language is used for doing the time series analysis. Jupyter Notebook is used for the processing and

visualization of data in the primary stage import libraries and python packages for the functioning of the model. Data is split for training and testing purpose. Time series models are developed in the subsequent stage. Data is predicted and forecasted in the final stage. By calculating the RMSE value of each algorithm efficiency and accuracy of each algorithm is identified and Visualized with the help of a Bar diagram.

List of Technologies used for implementation

- Programming Language - Python Version 3.

- Libraries – Pandas, NumPy, Sklearn, TensorFlow, Keras.

- IDE – Jupyter Notebook

- Python Package – Anaconda 3

1.3. Document Structure

The document contains eight chapters. In chapter one it includes problem definition, overall view of the project. In chapter two it highlights about the project motivation and research problem. In chapter three it describes the detail study of time series models, analysis and forecasting in existing research papers. In chapter four it deals with the CRISP DM Approach and defines about algorithm used. In chapter five it defines about the Core Structure - Hardware and Software, Artefact Diagram. In chapter six it defines data Collection, Pre-Processing etc. In chapter seven express the deployment of forecasting models - testing and evaluation. In chapter eight describes the conclusion and future work of the project.
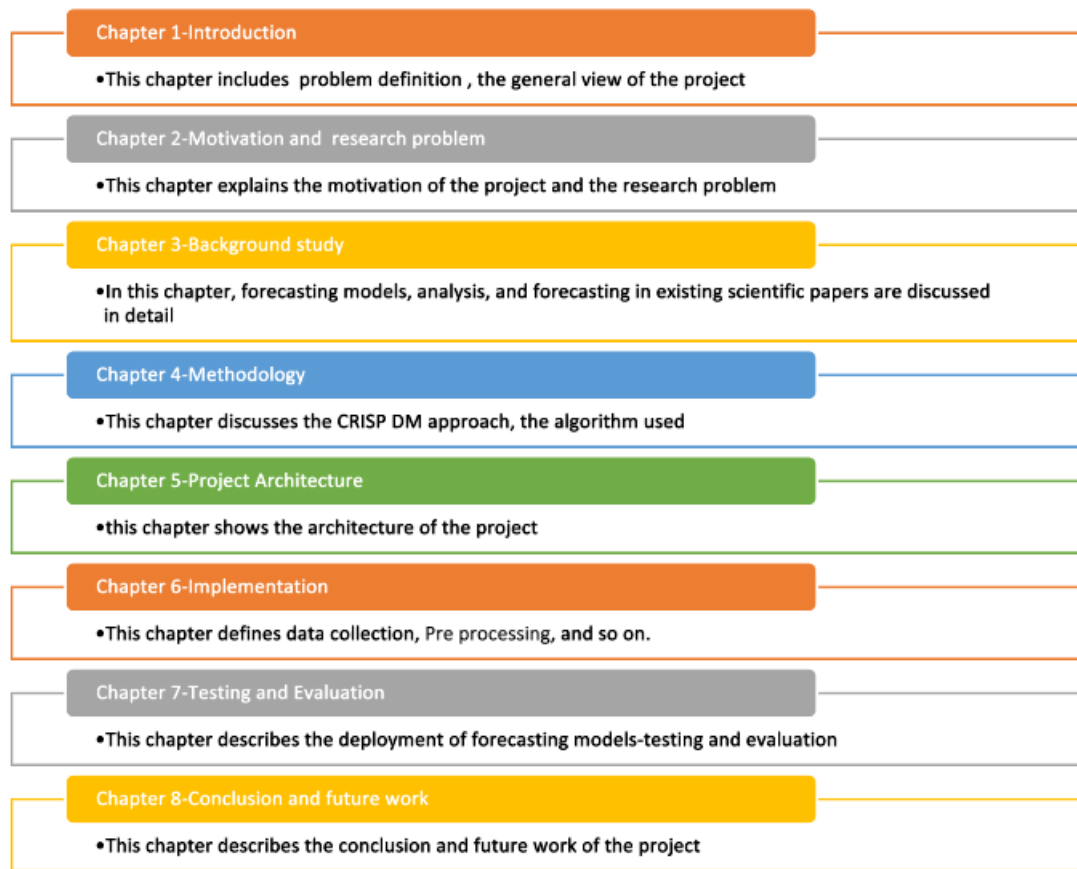
1.4. Dissertation Roadmap

Chapter 1-Introduction
• This chapter includes problem definition , the general view of the project

Chapter 2-Motivation and research problem
• This chapter explains the motivation of the project and the research problem

Chapter 3-Background study
• In this chapter, forecasting models, analysis, and forecasting in existing scientific papers are discussed in detail

Chapter 4-Methodology
• This chapter discusses the CRISP DM approach, the algorithm used

Chapter 5-Project Architecture
• this chapter shows the architecture of the project

Chapter 6-Implementation
• This chapter defines data collection, Pre processing, and so on.

Chapter 7-Testing and Evaluation
• This chapter describes the deployment of forecasting models-testing and evaluation

Chapter 8-Conclusion and future work
• This chapter describes the conclusion and future work of the project

Figure 2: Dissertation Roadmap

Chapter 2. Motivation and Research Problem

Stock market or equity market has an overpowering impact in today's economy. A rise or fall in the share price has an important role in determining the investor's gain. Stock market forecasting is always very tricky. No one can see the future– the world is inherently uncertain and surprising things will happen. Even if it is known what's going to happen, however, you might not know how markets will respond. Investment in the stock market is regarded as high risks and high gains and so attracts a large number of investors and economists. However, information regarding a stock is normally incomplete, complex, uncertain and vague, making it a challenge to predict the future economic performance. People invest in the stock market based on some analysis.

2.1. Research Question

- Identifying which is the most suitable time series model for Stock Market Price Prediction?

- How the prediction of Stock Market Price helps the development of Economy?

Predicting day by day stock market price helps the investors to identify the which company is more suitable for the investing their money. By predicting the Stock Market Price Prediction, Government can also identify what are the factors that influencing the rise and falls of Stock Price?

2.2. Forecasting Models

1. KERAS with LSTM (Long Short Term-Memory).

2. Prophet model.

3. ARIMA (Autoregressive integrated moving average) model.

Chapter 3. Background Study

This chapter discussed about the background study of the research papers that are related to the project.

3.1. Literature Review

3.1.1. Stock Exchange: The Birth

Stock is a word which is used for symbolizing an investor's ownership in a company and those who hold the stock are called as shareholders. The concept of modern stock world began in the late 16th century. When merchants want to start the huge business they don't have a single great investor so they decided to collect savings from all the investors who are interested to give their particular share and contribute to the large business and hence become business partners. This all started back in 1602, when Dutch East India Company issued the first paper of shares, which allowed the shareholders to conveniently buy, sell and trade their stock with other shareholders[8].

3.1.2. Stock Market Price Prediction system – Modular NeuralNetwork

It is based on the buying and selling time prediction system for the stocks on Tokyo stock exchange and it concentrates mainly on the modular neural networks. A number of learning algorithms and prediction methods were developed for the prediction system which is also called as TOPIX (Tokyo Stock Exchange Prices Indexes) and this prediction system gave an accurate prediction, simulation on stocks trading [9].

3.1.3. Stock Price Prediction using ARIMA(Auto-regressive Integrated Moving Average) Model

In terms of economics and finance the stock price prediction plays a crucial role and which gives more interests towards the better development of the predictive models. That is where the Auto-regressive Integrated Moving Average (ARIMA) models playing an important role in the time series prediction. This paper mainly concentrates on the stock price prediction using the ARIMA model and it mainly takes the data from the New York Stock Exchange (NYSE) and Nigeria Stock Exchange (NSE) used with stock predictive models and the result obtained

from this paper states that ARIMA model is strong for short term prediction and it also favors the existing techniques for the stock price prediction [10].

3.1.4. Stock price prediction using RNN, CNN, and LSTM

The rise and fall in share market price decide the fate of the investor's profit. The existing algorithms which are linear and non-linear focus on predicting the stock index movement for the single company but on the other hand the proposed method is a model independent approach and the data is not fitted to a specific model rather identifying the latent dynamics in the data using deep learning. This performs the price predictions for the NSE (Nigeria Stock Exchange) listed companies and compares their performance and applied a sliding window approach for predicting the future values on short term basis [11].

3.1.5. Stock price prediction using time series data

Researchers have taken some time to predict the data with algorithms fast and accurate enough to make stock prices prediction. Investor's mainly want their stock prices to forecast with much smarter techniques and this method worked well for them. In this paper mainly concentrates on the three main time series forecasting algorithms which are LSTM (Long Short-Term Memory), Auto-regressive Integrated Moving Average (ARIMA), Facebook prophet time series algorithms [12].

3.1.6. Stock Market Trend Prediction

A trend analysis of the stock market is useful for an appreciation of the changes in the stock market attributes over time. The predictions can be made with a certain level of accuracy, but the objective of the predictions is to reduce the risk. Linear regression, FBProphet, Bayesian Regression are used to predicted and compare the stock price. A new risk feature for long-term stock market prediction is developed using the forecasting models [13]

3.1.7. Neural Networks for Stock Price Prediction

This research focuses primarily on how previous information and neural networks can be used to enhance predictive ability. The daily stock pricing is a complex problem in the real world, considering political and world events, so we have to derive information and thus event awareness is derived mainly from the headlines. This paper also takes many economic measures and uses the neural networks to input them along with event information. This method also gives us a less predictive error than the multiple regression analysis [14].

- **Comparison of Neural Networks and ARIMA Models**

This calculates the forecasting performance of ARIMA, and artificial neural networks model published for the New York Stock Exchange. The prominent techniques used here are fall into two categories mainly which are statistical and computing techniques. In which the ARIMA falls into the statistical computing techniques and the artificial neural networks falls into the category of computing techniques and that too soft computing technique. The use of ARIMA as a time series forecasting is very essential but the use of artificial neural network is most accurate, and this paper mainly predicts the use of these and compares which is good for the stock market prediction [15].

3.1.8. Stock Price Prediction and Trend Prediction – NeuralNetworks

This mainly constitutes on the analyzing the feed forward network using back propagation learning algorithm with early stopping and the radial basis neural network is used to predict the trend of the stock price. In this research, Fundamental data or Technical indicators are not used as a basic objective of the research to predict the uses of artificial neural networks for future prices based on past prices [16].

3.1.9. Comparisons of Artificial Neural Networks Architecture inStock

The up's and downs of the stock market is predicted with the artificial neural networks in this paper and it provides a numerical analysis of concrete financial time series. This concept consists of algorithms such as Convolutional Neural Networks (CNN), Long Short-Term Memory (LSTM) and Multi-Layer Perceptron (MLP) recurrent neural techniques.

This paper mainly shows that neural networks are there to predict financial time series movements when trained on plain time series data [17].

3.1.10. Stock Price Prediction Using LSTM

The forecast of the stock market price prediction is complex task for many researchers but however the investors are highly interested in the research of the stock market price prediction and even many investors are keenly interested in the future situation of the stock market prices and considering all these in the mind, this paper mainly represent the Long Short Term Memory

(LSTM) and Recurrent Neural Network (RNN) approach to predict the stock market prices for the future [18].

Chapter 4. Methodology

This chapter discussed about three topics

1. CRISP-DM methodology.

2. Tools and technology used in this project.

3. Forecasting models - KERAS with LSTM , PROPHET, and ARIMA

4.1. CRISP DM Diagram

The CRISP DM represents cross industry process for data mining. It is a strong and very much demonstrated system. Its adaptability and handiness when examining to gruesome business issues. It is the brilliant line that almost every customer engagement goes through.

This model is a glorified sequence of times. Many individuals of the companies can act in an alternative way and it is important to track back past tasks and retrace certain activities on a regular basis. The model does not try to take every imaginable path through the information mining process. This is a 6-phase model. [19].
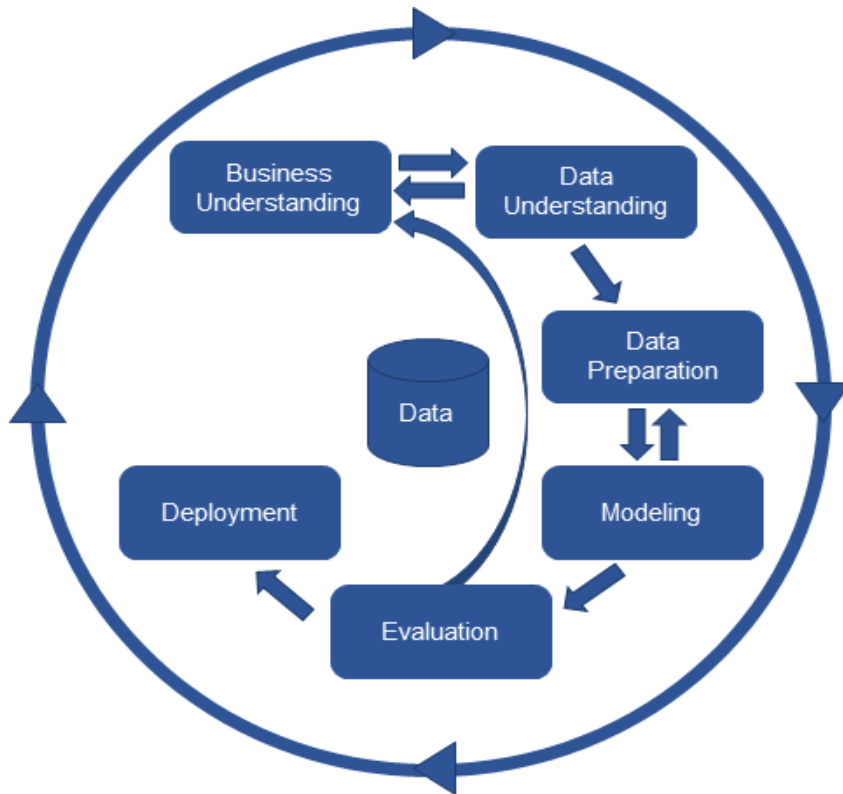
Figure 3 : CRISP – DM Diagram. [20].

- Business Understanding: To identify the business objectives and data mining goals.

- Data Understanding: Collecting of data from sources, exploring and verifying thequality of data.

- Data preparation: This phase is the time-consuming phase. Data selecting, cleaning,constructing, integrating are takes place in this phase.

- Modelling: Selecting the techniques for the modelling, creating the test design buildingand accessing of models.

- Evaluation: Evaluation of results.

- Deployment: Planning of deployment and monitoring.

William Vorhies, one of the makers of CRISP-DM, contends that since all information science ventures start with business understanding, have

information that must be accumulated and cleaned, and apply information science calculations, "Fresh DM gives solid direction to even the most exceptional of the present data science exercises" [19].

4.2. Tools and Technologies

4.2.1. Python Programming Language

Python is used in this project because Python is an interpreted, object oriented and high-level programming language. It was created by Guido Van Rossum in the year 1990. It is cross platform programming language. This programming language is freely usable for commercial purposes. Python is easy to use and very powerful. Also, this language is versatile in nature. The companies like Google, Facebook, Netflix etc. are using python.

Advantage of Python:

Python is a object oriented programming language the we can used for scientific and nonscientific programming. Simple high-level language with lots of libraries. Python is a platform independent programming language. We can use python in servers to create web applications. It is used in database systems because it can read and modify the data very easily. One of the major advantages of the python is we can used to handle big data for performing complex mathematics.

Python can work on different platforms. Python has a simple syntax that is very similar to English language. Prototyping of the python code is very fast. The recent version of the python is python version 3 [21].

4.2.2. Pandas

Pandas package is one of the most fast and powerful packages in python programming language. It is very flexible and easy to use in open source data analysis. It is also using for visualizing the data. That is why pandas are called the backbone of data science projects. By using pandas, we can clean, transform, and analyze the data. Pandas can extract data from csv file into data frames. We can calculate statistics such as average, mean, max and min etc. Visualizing the data with the aid of matplotlib is another feature of pandas. We can plot line graph, bar diagram, histogram etc. [22].

The main essential data structure pandas are Panel. Panel is data structure which helps to manipulate the data sets and time series.

We can install pandas using pip command and import the panda's library with the help of import function. Pandas have two core components.

- Series
- DataFrame

Series:

One dimensional array is called series in pandas. The function 'pd. Series' is used to create constructor which can include lot of arguments. The commonly used argument name is called data that specifies the all elements in the series. For the manual casting panda's series also used dtype keyword. The series of elements collectively called as Index of the series [23].

The parameters in panda series are data, index, dtype and copy.

DataFrame:

DataFrame is the two-dimensional array in pandas. The function 'pd. DataFrame' is used to create a DataFrame. The DataFrame is not scalar. The index is rows in DataFrame. Column is another label [23].

The parameters in panda DataFrame are data, index, columns, dtype and copy.

Advantage of Pandas

- It can present the data in Series or DataFrame data structures.

- Panadas package contain multiple methods for data filtering.

- It handles the data very efficiently.

- It can create the data flexible.

Disadvantages of Pandas

- Learning slope is very steeper in pandas.

- It has difficult syntax.

- It has bad documentation.

- Pandas cannot used for three-dimensional arrays [24].

4.2.3. NumPy

NumPy stands for numerical python. NumPy is a library used in python programming. It is fast and versatile. It is used to process powerful N dimensional arrays. NumPy consist of functions that which can work in the field of linear algebra, Fourier transform and metrices. NumPy data structures performed in

- Size – It needs only less space.

- Performance – It is faster than lists

- Functionality – SciPy and NumPy are the main functions for linear algebra [25].

4.2.4. Sklearn

Scikit Learn also known as Sklearn is a simple and efficient tool using for predictive analytics. It is free accessible and reusable. It is also an open source library package [26].The main features of Sklearn are Preprocessing, Model selection, Regression, Classification, Clustering.

4.2.5. TensorFlow

TensorFlow is a prominent framework used in machine learning and deep learning. The wordTensorFlow is combined from two words.

- Tensor – Multi dimensional array

- Flow – the flow of data

It is a free and open source library. It is released on 9 November 2015. It is developed by Google Brain Team. To increase the speed TensorFlow is totally developed on the python programming language. TensorFlow can train and run neural network programs. It supports platforms such as Windows, MacOS, Linux and Android [27].

Advantages of TensorFlow

- Graphs: When comparing to other libraries It has a better graph visualization.

- Library Management: Google developed TensorFlow. It has the advantages of quickupdates and frequent releases with new features and libraries.

- Debugging, Scalability and pipelining

- It has excellent community support [27].

Disadvantage of TensorFlow

- Computation speed is very high.

- It is very hard to find the errors because of its unique structure.

- It does not need any super lower matter [27].

4.2.6. Keras

Keras is open source neural network library in python. It is a high-level framework. It can easily build a neural network model in a limited time. The main advantage of Keras are user friendly, Fast deployment, multiple back ends , modularity [28].

It runs on both CPU and GPU. It supports convolutional and recurrent neural networks. It is installed simply by using pip install keras.

4.2.7. Jupyter Notebook

Jupyter notebook is a web based open source interactive application. It allows to develop codes and documents in a single page. We can use Jupyter Notebook for cleaning, modelling,

training, and evaluating of data. The prominent feature of Notebook is we can visualize the data [29].

The main features of Jupyter Notebook are we can execute the code within the browser. We can display the outputs in multiple formats such as HTML, pdf etc. Markdown markup language helps the formatting of texts in the console. Notebook documents are saved internally in ipynb format. We can install Jupyter Notebook by using pip command in the command prompt [30].

4.2.8. Anaconda3

Anaconda is open source distributor of R and Python programming. It provides tools for doing data analysis and machine learning techniques. It includes various python libraries and hundreds of scientific packages. Navigator is the GUI of Anaconda where the user can open

various tools and applications. Another important factor of anaconda is anaconda cloud where we can save our code [33].

4.3.   Forecasting models

The prediction of stock market price id done with following Algorithms. Time series is sequence of numbers measured over a regular period. Based on the frequency, times series can be classified in to yearly, monthly and daily.

The Univariate time series forecasting that use previous values in the time series to estimate the future values. Multivariate time series forecasting type of forecasting which we use different predictors.

4.3.1. ARIMA Model

ARIMA is a prediction algorithm short for Auto Regressive Integrated Moving Average, built on the principle that the past time series values can be used for the purposes of predicting future values only. ARIMA is a class of models that describe such time series on the basis of their own past value , i.e. their own limitations and delayed forecasting errors. ARIMA models can be based on any non-seasonal time series which exhibits trends and which is not a white natural variability.There are 3 parameters in ARIMA model p, d, q [34].

- p denotes Auto Regressive.

- q denotes Moving Average.

- d denotes the number of differences required to station the time series

When a time series is seasonal, the seasonal patterns must be introduced and SARIMA – Seasonal ARIMA. Auto Regressive means in ARIMA a linear regression model that uses its own lag predictors. Linear models of regression work well if the predictors are not interrelated or independent. Differences are the most popular approach and difference may be required depending on the nature of the sequence. Therefore, the value of d is the minimum number of differences required to stabilize the series. And if the time series is still, then d = 0. "p" is a "Regressive Automatic Term" (AR) order. The number of Y deficiencies to be used as predictors is mentioned. "q" is the order of the term "moving average." It refers to the amount of late prediction mistakes that the ARIMA model will produce [34].

Auto Regressive model (AR) is one in which Yt only relies on its own deficiencies. Yt is a 'ytlagging' element.

$$Y_t = \alpha + \beta_1 Y_{t-1} + \beta_2 Y_{t-2} + .. + \beta_p Y_{t-p} + \epsilon_1$$

Equation 1 – Equation of AR [34].

The moving average model (MA only) is one in which Yt only depends on lagging predictionerror.

$$Y_t = \alpha + \epsilon_t + \phi_1 \epsilon_{t-1} + \phi_2 \epsilon_{t-2} + .. + \phi_q \epsilon_{t-q}$$

Equation 2 – Equation of MA [34].

An ARIMA model is a model that differs the time series to make it stationary, at least once, and combines the terms AR and MA. So, the equation is

$$Y_t = \alpha + \beta_1 Y_{t-1} + \beta_2 Y_{t-2} + .. + \beta_p Y_{t-p} \, \epsilon_t + \phi_1 \epsilon_{t-1} + \phi_2 \epsilon_{t-2} + .. + \phi_q \epsilon_{t-q}$$

Equation 3 – Equation of ARIMA [34].

Now, Equation of ARIMA in words

Predicted Yt = constant + Linear Y-lag (up to p-lag) + Linear Lagged Forecast Combination (up to q lags) [34].

4.3.2. FB Prophet Model

Prophet is a Facebook open source library focused on decomposable models (trend + seasonality + holidays). It gives us the ability to predict time series with accuracy and supports custom seasonality and holidays. The Prophet package offers easy-to-tune, intuitive parameters. Those who have no experience in forecasting models can use this to make accurate predictions on a wide variety of business problems. The three main components of prophet model are trend, seasonality, and holidays [35].

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

Equation 4: components of prophet [35].

- g(t): linear curve in part for modeling non-periodic time series transition.

- s(t): daily changes (seasonality weekly / yearly).

25

- h(t): holidays effects of irregular timescales.

- εt: error term is responsible for any unexpected alteration that is not suited to thepattern.

Trend:

Trend is formed by a linear piece curve that matches the pattern or non-periodic sections. Thelinear fitting method ensures that the spikes / missing data.

Seasonality:

The Prophet adapts and predicts the effect of seasonality to a scalable model using foursequences. The following method approximates the seasonal effects of s(t):

$$s(t) = \sum_{n=1}^{N} \left(a_n \cos\left(\frac{2\pi nt}{P}\right) + b_n \sin\left(\frac{2\pi nt}{P}\right) \right)$$

Equation 5: Seasonality effect [35].

Holidays and Events:

Holidays and events contribute to predictable time series shocks. Prophet allows analysts toidentify past and future events in a personalized manner. There is a window during certain days and parameters are used to model the impact of holidays and events [35].

4.3.3. Keras with LSTM

LSTM is a sort of recurring neural network. A recurrent neural network is a neural networkwhich seeks to form time or sequence-based behavior for instance language, stock prices,

energy demand, etc. The output of the neural network is calculated by the input of same network layer at t+1 time [36].
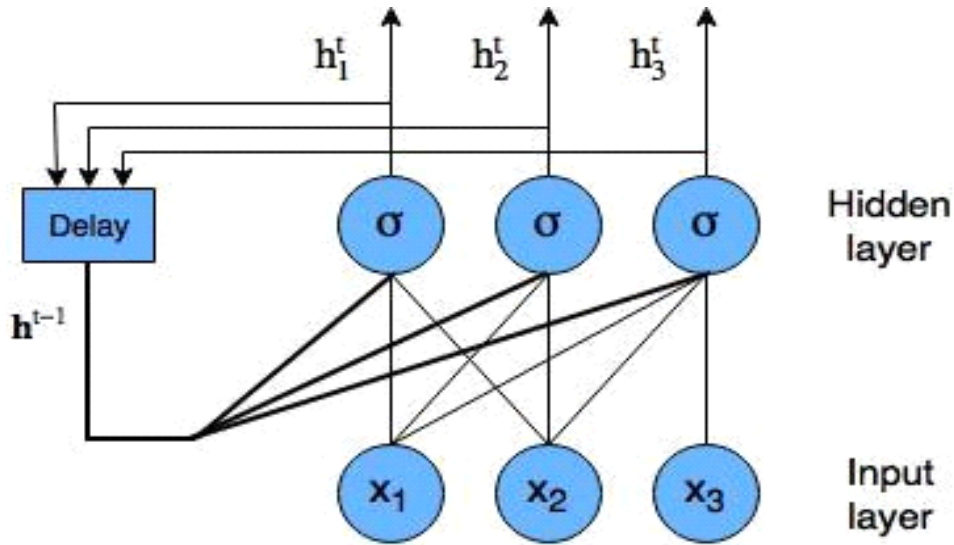
Figure 4: Recurrent Neural Network with nodes [36].

At the time of training of model and prediction, the recurrent neural network became unrolled.
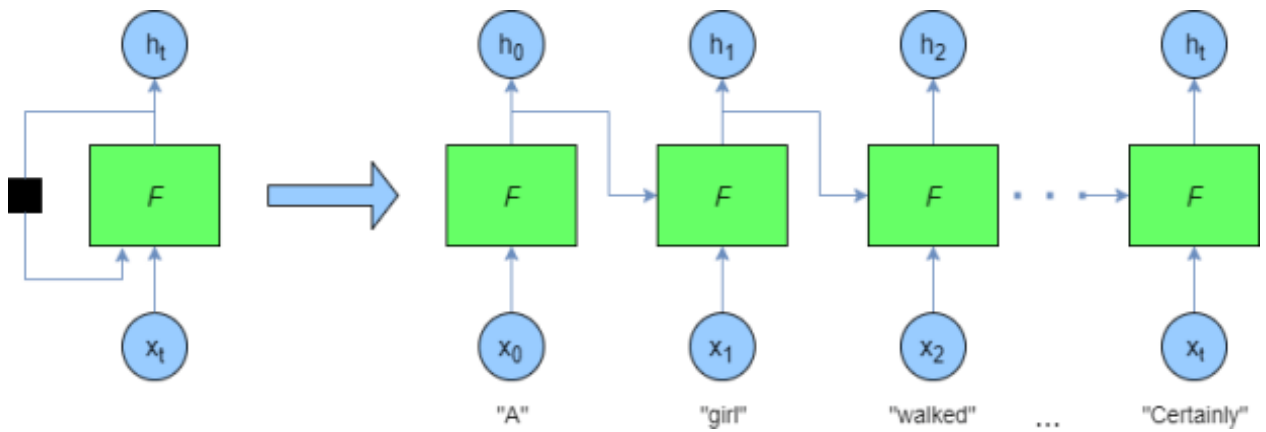
Figure 5: Unrolled recurrent neural network [36].

LSTM is a recurrent neural network with LSTM cellblocks replacing the neural network's regular layers. The different components of these cells are called the input gate, forget gate and the output gate [36].

- Input layer: The activity of the data contraptions refers to the non-cooked data, whichcan enter the system.

- Hidden layer: Determine every secret unit's endeavor. The exercises of the systems entered and the loads on the connections between the data and the components. At least one hidden layer may also be present.

- Output layer: The output direct depends on the actions of the hinged units and theload between the secret and output devices.
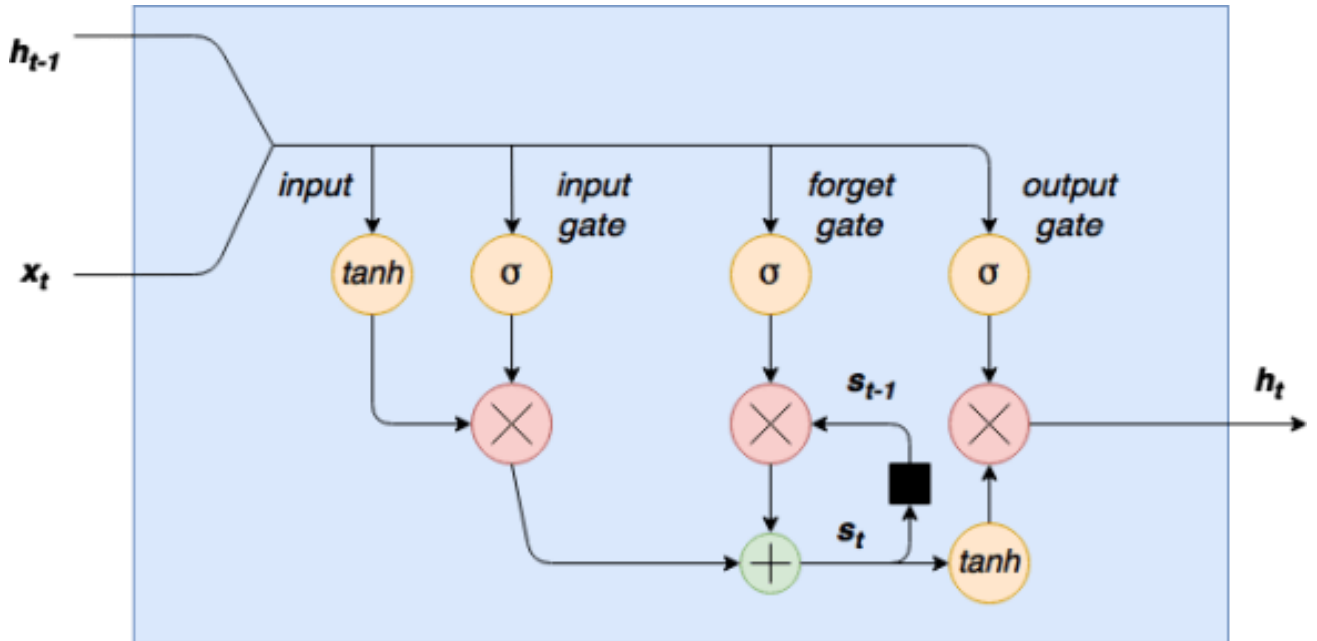
Figure 6: Cell diagram of LSTM [36].

LSTMs are designed explicitly to prevent the problem of long-term dependence. The long-term retention of knowledge is basically their automatic comportment, not something they try to understand. All repetitive neural networks are shaped like a chain of neural network repeating modules. This repeating module has a structure in standard RNNs, like a single layer of tanh [37].
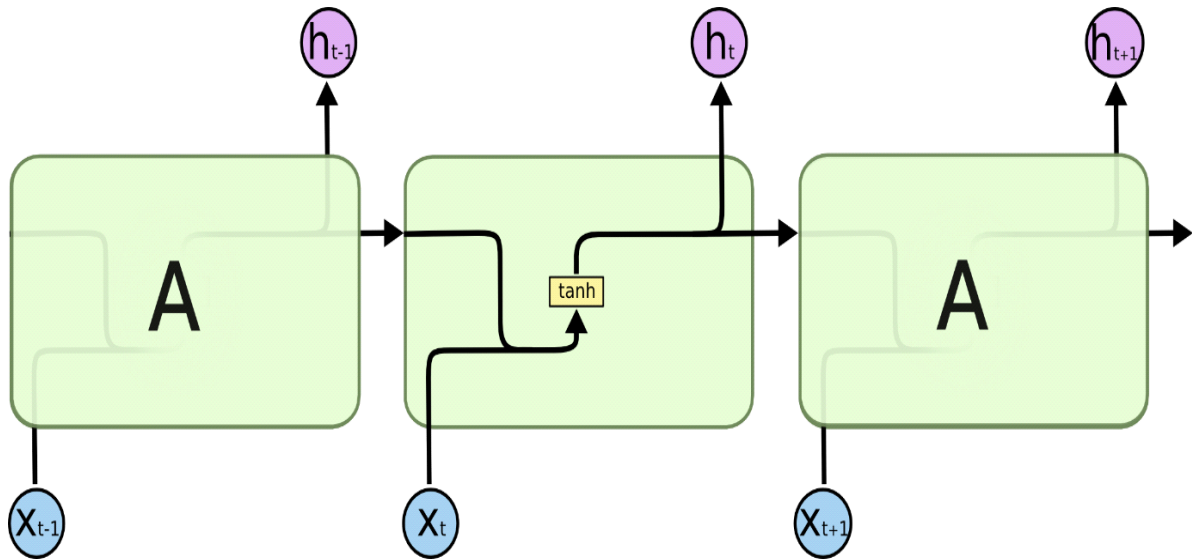
Figure 7: RNN contains Single layer [37].

LSTMs have a structure like this chain, but the repetitive module which have different structure. There are four, interacting very specifically, instead of making a single neural network layer.
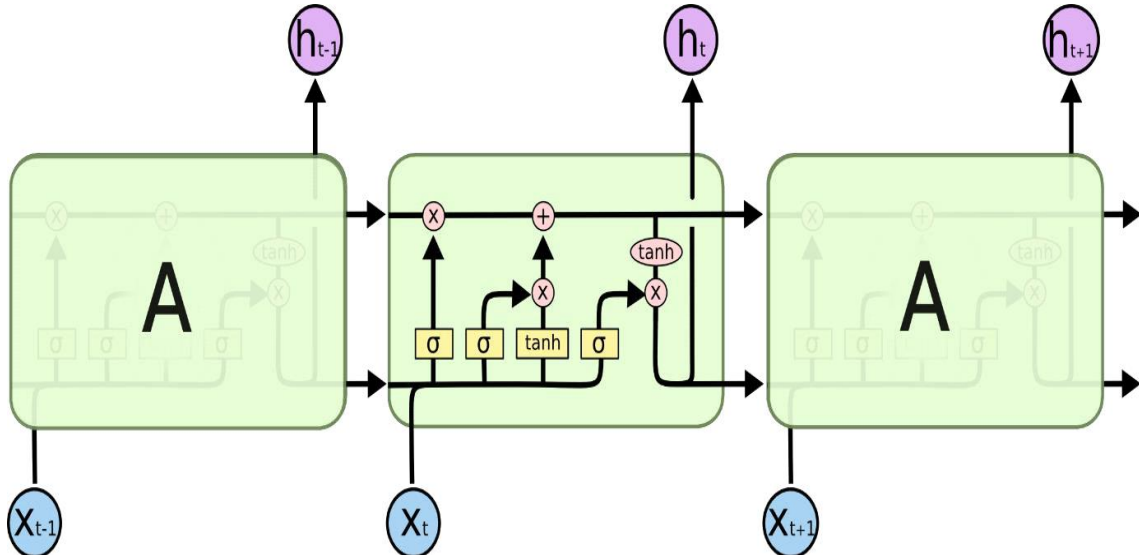
Figure 8: LSTM with interactive layers [37].

Due to its large-range and short-range data dependencies, LSTMs may help deal with disappearance and exploding gradients. LSTMs in many

cases work a little better than GRUs, but due to the larger latent state size, they are more costly to train and operate. LSTMs are the prototypical, non-trivial state controlled latent autoregressive variable model [38].

Chapter 5. PROJECT ARCHITECTURE

In this section, we discuss different modules used in the program. Firstly 10 years of historical data is collected from the yahoo finance. Yahoo finance is a financial website that provide stock market data's, news, live updates etc. These data are processed and normalized for doing analysis. Time series analysis models are used to identify the performance of the processed data with the aid of Jupyter Notebook. Jupyter Notebook is an IDE which allows majority of python library functions. Python 3.8.2 version is used not only for processing the data but also it is used for the predicting and forecasting of stock market data.

5.1.1. Analysis Phase

The below figure shows the architecture of the analysis phase. The analysis phase is classified into preparation phase and experimental phase.
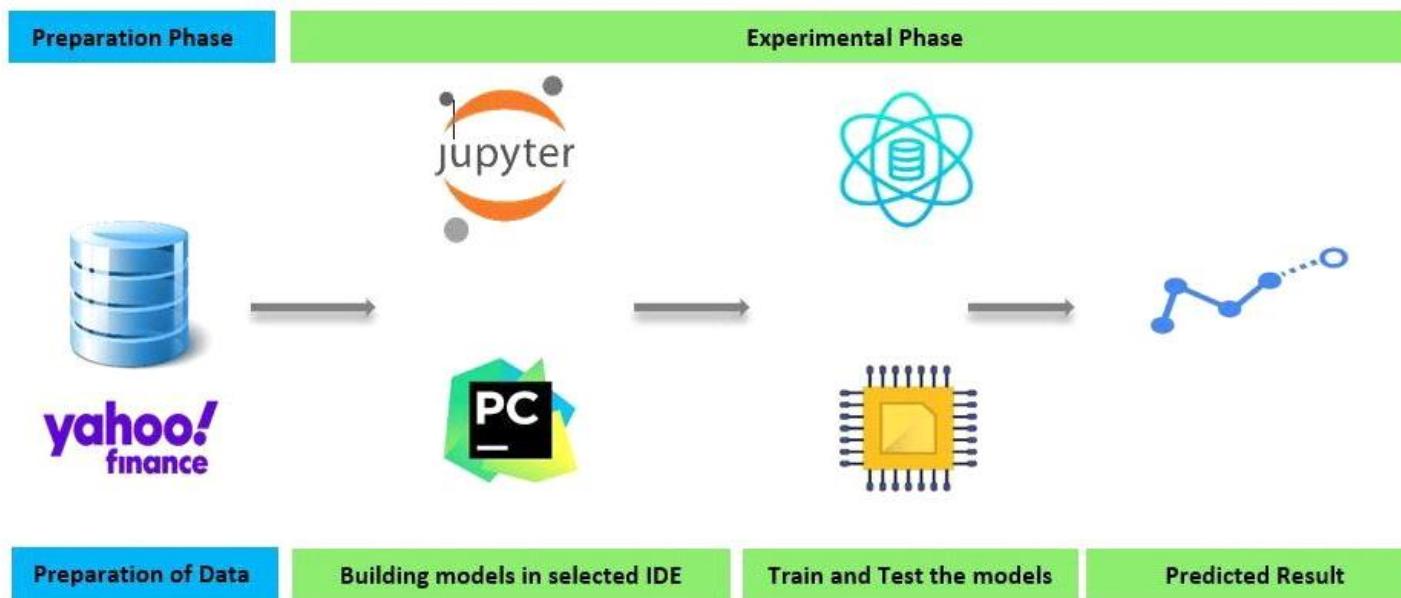
Figure 9: Architecture of Analysis Phase.

Preparation Phase:

Preparing of the data is the initial stage of the project. The raw data is collected from yahoo finance using python library fix yahoo finance. The collected data consist of seven attributes. Date attribute refers the date. Open attribute refers the opening value of the stock. Close attribute refers the closing value of the stock. High attribute refers the highest stock value in a

day. Low attribute refers the lowest stock value in a day. Adj Close attribute refers the closing price after adjustments. Volume is the amount of transactions made in a day. For pre-processing of data steps like cleaning, format and select the data which is suitable for the algorithms. In this project we collected the ten years of data from different banks.

Experiment Phase:

In this phase Jupyter Notebook is used to perform the training and testing of time series models. Different time series models are developed using ARIMA, PROPHET and KERAS with LSTM. For the predicting the stock price these models are used.
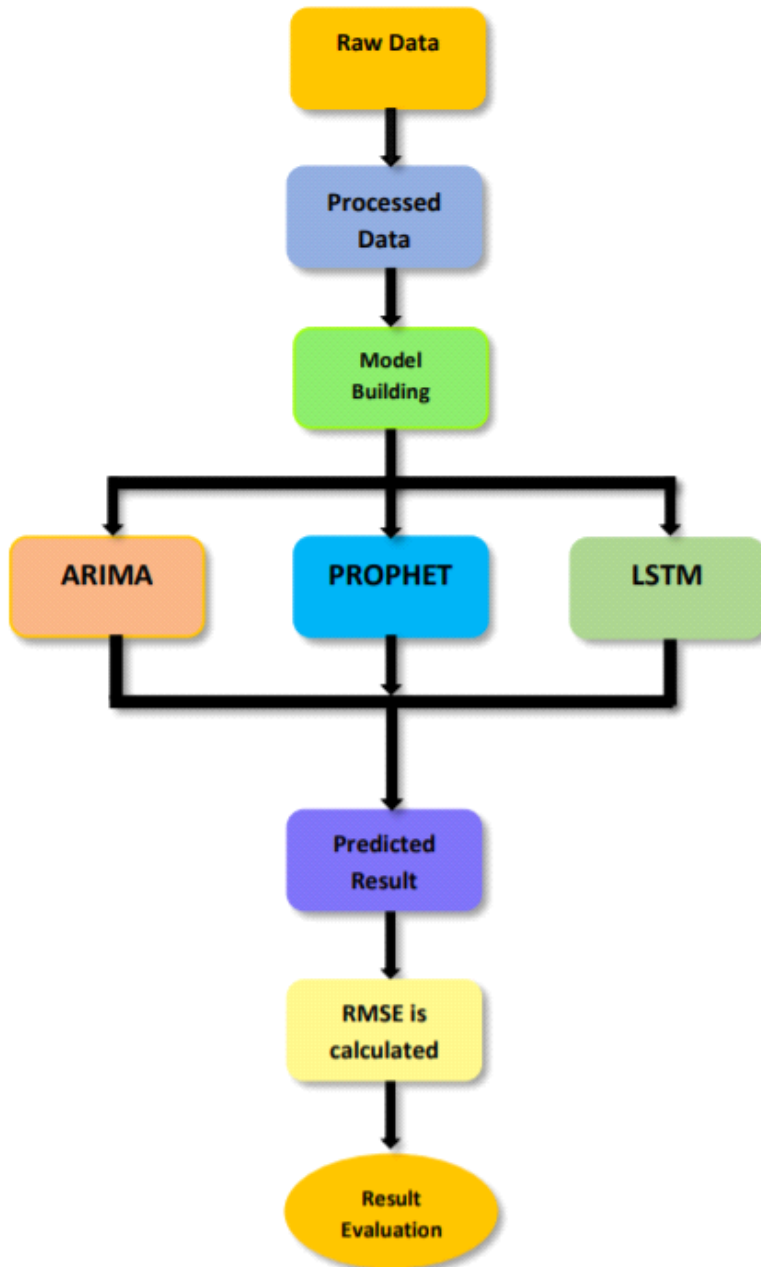
Figure 10: Step by Step procedure of Analysis Phase.

Chapter 6. Implementation

In this chapter we are going to discuss about the step by step procedure of Analysis phase.

6.1. Data Collection

The historical data of Banks are collected from yahoo finance using

Yfinance library and saved into the directory in csv format.

```
#importing of libraries
import pandas
from pandas_datareader import data as pdr
import yfinance as yf |

import datetime
yf.pdr_override()

import warnings
warnings.filterwarnings('ignore')

stocks = ["AMZN"]
start = datetime.datetime(2011,4,1)
end = datetime.datetime(2021,4,1)

data = pdr.get_data_yahoo(stocks, start=start, end=end)

#file saved in the directory
data.to_csv('C://Users//HP//Desktop//Stock market prediction//AMAZON.csv')

data.head(5)
```

Figure 12: Collect and save data from yahoo finance.

After running the above code, the output is shown in figure 13

```
[*********************100%***********************] 1 of 1 completed
```

Date	Open	High	Low	Close	Adj Close	Volume
2011-03-31	179.309998	181.570007	178.500000	180.130005	180.130005	4826500
2011-04-01	181.580002	183.250000	178.589996	180.130005	180.130005	5684100
2011-04-04	180.889999	183.610001	180.690002	182.940002	182.940002	4186400
2011-04-05	182.100006	186.360001	181.800003	185.289993	185.289993	5569200
2011-04-06	186.149994	188.270004	181.119995	182.759995	182.759995	5430700

Figure 13: Collected data.

6.2. Implementation of ARIMA model

For the implementation of ARIMA model, first we introduce the python
packages and libraries.

```python
import pandas as pd
import numpy as np
import itertools

import warnings
warnings.filterwarnings("ignore")

import statsmodels.api as sm
import matplotlib
import matplotlib.pyplot as plt
plt.style.use('fivethirtyeight')

matplotlib.rcParams['axes.labelsize'] = 14
matplotlib.rcParams['xtick.labelsize'] = 12
matplotlib.rcParams['ytick.labelsize'] = 12
matplotlib.rcParams['text.color'] = 'k'
```

Figure 14: Importing Packages and libraries.

The above figure 14 represents the initial stage in which packages are
imported. We import Pandas, NumPy and Itertools. Python Itertools is a
module which offers different functions to produce complex iterators.
Warnings are imported to avoid the future warnings at the time of
execution. Matplotlib library is used to plot variables in graphical
format. Stats models is a python module that contain classes and
functions. These classes and functions are used to estimate the wide
range of statistical model and testing.

```
AMAZON = pd.read_csv('C:/Users/HP/Desktop/Stock market prediction/AMAZON.csv')
AMAZON.head(5)
```

	Date	Open	High	Low	Close	Adj Close	Volume
0	2011-04-01	181.580002	183.250000	178.589996	180.130005	180.130005	5684100
1	2011-04-04	180.889999	183.610001	180.690002	182.940002	182.940002	4186400
2	2011-04-05	182.100006	186.360001	181.800003	185.289993	185.289993	5569200
3	2011-04-06	186.149994	188.270004	181.119995	182.759995	182.759995	5430700
4	2011-04-07	182.779999	185.169998	181.759995	184.910004	184.910004	4564000

Figure 15: Reading of the AMAZON stock prices data from the directory and its output.

The figure 15 shows the reading of dataset in csv format from the directory. This dataset is stored in a variable called AMAZON. After that we print the first 5 rows of the dataframe by using head().we can also use tail to print the last 5 rows of the dataframe.

```
AMAZON.Date = pd.to_datetime(AMAZON.Date, format='%Y%m%d', errors='ignore')
cols = ['High', 'Low', 'Open', 'Volume', 'Adj Close']
AMAZON.drop(cols, axis=1, inplace=True)
AMAZON = AMAZON.sort_values('Date')
AMAZON.isnull().sum()
```

```
Date     0
Close    0
dtype: int64
```

```
AMAZON['Date'].min()
```

```
'2011-04-01'
```

```
AMAZON['Date'].max()
```

```
'2021-03-31'
```

Figure 16: Data preprocessing Amazon data example

The figure 16 shows the checking of missing values, removing of columns and sort the data in date format. After that we are checking the minimum and maximum value of the date.

```
#creation data column as index
AMAZON = AMAZON.set_index('Date')
AMAZON.index
```

```
Index(['2011-04-01', '2011-04-04', '2011-04-05', '2011-04-06', '2011-04-07',
       '2011-04-08', '2011-04-11', '2011-04-12', '2011-04-13', '2011-04-14',
       ...
       '2021-03-18', '2021-03-19', '2021-03-22', '2021-03-23', '2021-03-24',
       '2021-03-25', '2021-03-26', '2021-03-29', '2021-03-30', '2021-03-31'],
      dtype='object', name='Date', length=2516)
```

Figure 17: Indexing with AMAZON time series data.

```
#creating monthly mean from 2011
monthly_mean['2011']
```

```
Date
2011-04-30     184.576500
2011-05-31     198.166667
2011-06-30     191.667272
2011-07-31     215.195001
2011-08-31     199.458694
2011-09-30     223.242857
2011-10-31     226.838095
2011-11-30     205.480951
2011-12-31     183.933811
Freq: M, Name: Close, dtype: float64
```

Figure 18: Creating AMAZON monthly mean.

The figures 17 and 18 show the indexing of date with the time series data. The current date time data is very difficult to work. So, we used the average close price value of the month and starting of every month as timestamp.

```
#Plotting the monthly mean
monthly_mean.plot(figsize=(15, 6), title = 'AMAZON STOCK PRICES')
plt.savefig('amznstock.pdf')
plt.show()
```


Figure 19: Visualizing Close time series data.

```
# Visualising the Distinct components: trend, seasonality, and noise.
from pylab import rcParams
rcParams['figure.figsize'] = 18, 8
decomposition = sm.tsa.seasonal_decompose(monthly_mean, model='additive')
fig = decomposition.plot()
plt.savefig('amznstock_compnent.pdf')
plt.show()
```

Figure 20: Code for implementing time series decomposition method.

The figure 20 describes the code for implementing the time series decomposition method which is used to decompose our data in to three components: trend, seasonal and noise. The below figure 21 shows the graph for our AMAZON stock prices. The graph clearly showing the unstable nature of closing price.
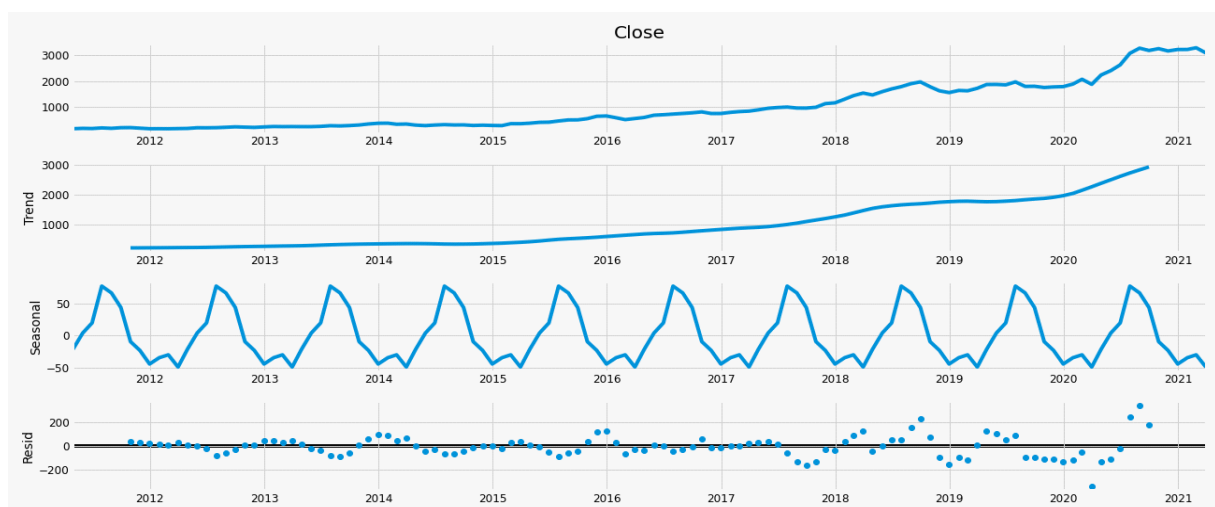
Figure 21: Time series decomposition graph.

Now we are going to predict the closing stock price by building the ARIMA model. For the initially we are selecting the p, d, q parameters. These three parameters reflect seasonality, trend, and noise.

```
# Define the p, d and q parameters to take any value between 0 and 2
p = d = q = range(0, 2)
# Generate all different combinations of p, q and q triplets
pdq = list(itertools.product(p, d, q))
# Generate all different combinations of seasonal p, q and q triplets
seasonal_pdq = [(x[0], x[1], x[2], 12) for x in list(itertools.product(p, d, q))]
```

Figure 22: Code for Generating all Seasonal
ARIMA Combinations.

Here p is the autoregressive part in the model. It enables us to include the effect of past values in our model. d is the integrated part of the model. It means the number of past time points deduct from the present value. q is the moving average part of the model which allows us to determine the error model. In seasonal ARIMA, defined as ARIMA(p, d, q)(P, D, Q)s, in relation to seasonal effects. The s denotes period of time series 4 in quarterly and 12 in annually. Now, we have to find the value of ARIMA(p, d, q)(P, D, Q)s. For that we used grid search to iteratively explore all the different combinations in parameters. For that SARIMAX() functionis used.

```python
import warnings
warnings.filterwarnings("ignore")

l_param = []
l_param_seasonal=[]
l_results_aic=[]
for param in pdq:
    for param_seasonal in seasonal_pdq:
        try:
            mod = sm.tsa.statespace.SARIMAX(monthly_mean,
                                            order=param,
                                            seasonal_order=param_seasonal,
                                            enforce_stationarity=False,
                                            enforce_invertibility=False)

            results = mod.fit()

            print('ARIMA{}x{}12 - AIC:{}'.format(param, param_seasonal, results.aic))

            l_param.append(param)
            l_param_seasonal.append(param_seasonal)
            l_results_aic.append(results.aic)
        except:
            continue

minimum=l_results_aic[0]
for i in l_results_aic[1:]:
    if i < minimum:
        minimum = i
i=l_results_aic.index(minimum)

mod = sm.tsa.statespace.SARIMAX(monthly_mean,
                                order=l_param[i],
                                seasonal_order=l_param_seasonal[i],
                                enforce_stationarity=False,
                                enforce_invertibility=False)

results = mod.fit()
print("\n\n")
print(results.summary().tables[0])

print(results.summary().tables[1])

print(results.summary().tables[2])
```

Figure 23: Seasonal ARIMA Model.

The figure 23 shows the implementation of SARIMAX function to fit the seasonal ARIMA model.

The SARIMAX results suggested SARIMAX(1,1,1)×(0,1,1,12) with lowest AIC value of
1250.205. The coef column describes the weight and effect on the time series in each function. The column P>|z| shows the value of each characteristic weight.

```
                              SARIMAX Results
==============================================================================
Dep. Variable:                         Close   No. Observations:              132
Model:             SARIMAX(1, 1, 1)x(0, 1, 1, 12)   Log Likelihood         -621.103
Date:                       Mon, 12 Apr 2021   AIC                        1250.205
Time:                              09:02:19   BIC                        1260.821
Sample:                          04-30-2010   HQIC                       1254.507
                               - 03-31-2021
Covariance Type:                        opg
==============================================================================
                 coef    std err          z      P>|z|      [0.025      0.975]
------------------------------------------------------------------------------
ar.L1          0.6428      0.213      3.012      0.003       0.224       1.061
ma.L1         -0.4625      0.247     -1.872      0.061      -0.947       0.022
ma.S.L12      -0.6977      0.103     -6.747      0.000      -0.900      -0.495
sigma2      7810.4782    676.913     11.538      0.000    6483.753    9137.203
==============================================================================
Ljung-Box (L1) (Q):               0.29   Jarque-Bera (JB):            65.64
Prob(Q):                          0.59   Prob(JB):                     0.00
Heteroskedasticity (H):          31.91   Skew:                         0.11
Prob(H) (two-sided):              0.00   Kurtosis:                     6.87
==============================================================================
```

Figure 24: SARIMAX Result for AMAZON stock
prices

```
#Plotting the diagnostics results
results.plot_diagnostics(figsize=(20, 10))
plt.savefig('applstock_diag.pdf')
plt.show()
```

Figure 25: Code for Plotting result diagnostics.

Now we are going to plot the diagnostics results with the help of plot
diagnostics. This figure shows the Standardized residual, Histogram
plus estimated density, Normal Q-Q and Correlogram. In figure 26,
Histogram plus estimated density shows the red line of KDE follows
closely with the N(0,1) where in stands for standard notation with
standard deviation 1 and mean 0. In the Q-Q Plot shows that blue dots

follow the trend of standard normal distribution. The standard residual does not show any obvious seasonality and it is confirmed by the correlogram plot. In the Correlogram plot time series residual have low correlation.
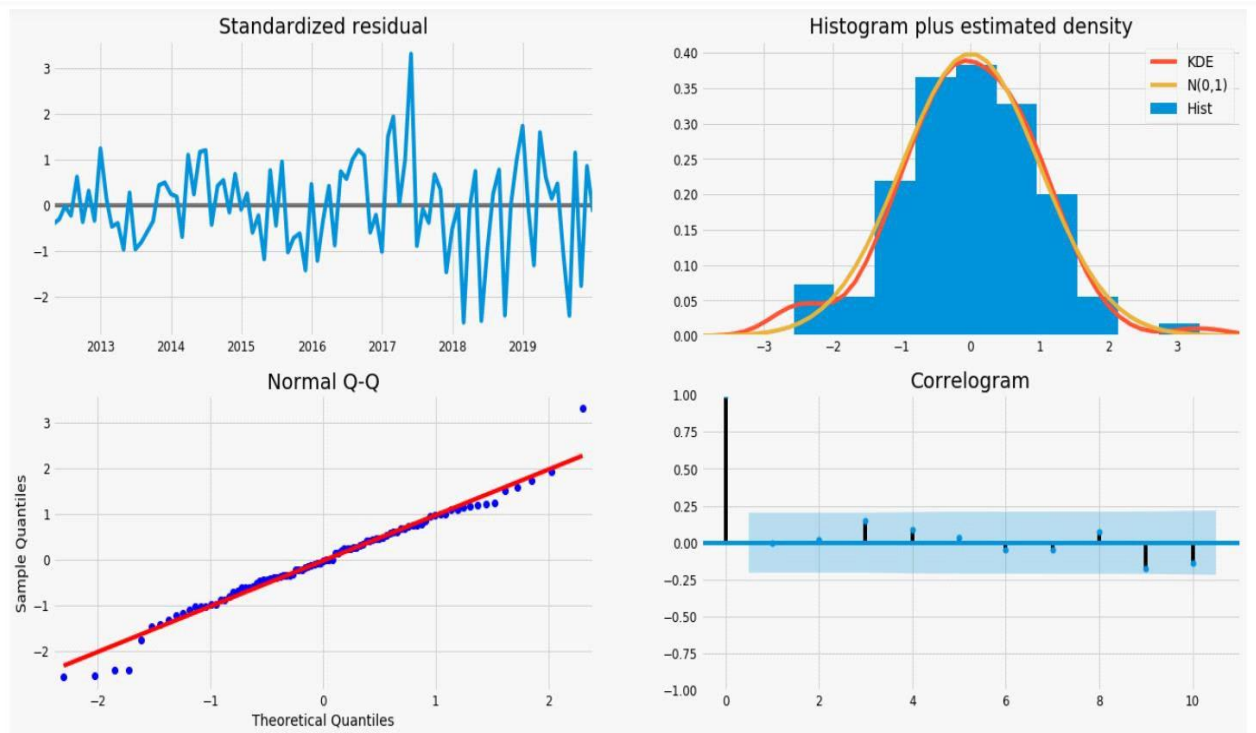
Figure 26: Result Diagnostics graph.

The above figure helps us to conclude that our model is good enough to forecast the future stock price values.

```
pred = results.get_prediction(start=pd.to_datetime('2011-12-31'), dynamic=False)
pred_uc = results.get_forecast(steps=24)
pred_ci = pred.conf_int()


ax = monthly_mean['2012':].plot(label='observed')
pred.predicted_mean.plot(ax=ax, label='Predicted', alpha=.7, figsize=(12, 7))
pred_uc.predicted_mean.plot(ax=ax, label='Forecast')

ax.fill_between(pred_ci.index,
                pred_ci.iloc[:, 0],
                pred_ci.iloc[:, 1], color='k', alpha=.2)

plt.title('AMAZON STOCK PRICES')
ax.set_xlabel('Date')
ax.set_ylabel('close price')
plt.legend()
plt.savefig('amznstock_pred.pdf')

plt.show()
```

Figure 27: Code for plotting the graph of observed, predicted, and forecasting results.

For validating the forecast, we begin by comparing forecast values to real time series value. It helps us to understand the prediction accuracy. The attribute get_prediction() and coef_int() which helps to get the values and confidence interval for time series prediction. The dynamic = FALSE argument which means the forecast are generated at each point using the entire history.

The figure 27 explains the code for predicting and forecasting the closing price in the graph. We took start date as 31-12-2011 for prediction and 24 months for forecasting. Now first we are plotting the observed value. Then we are going to plot the predicted value and forecasted value. After all these we defined the title, y-label, and x-label of the graph.
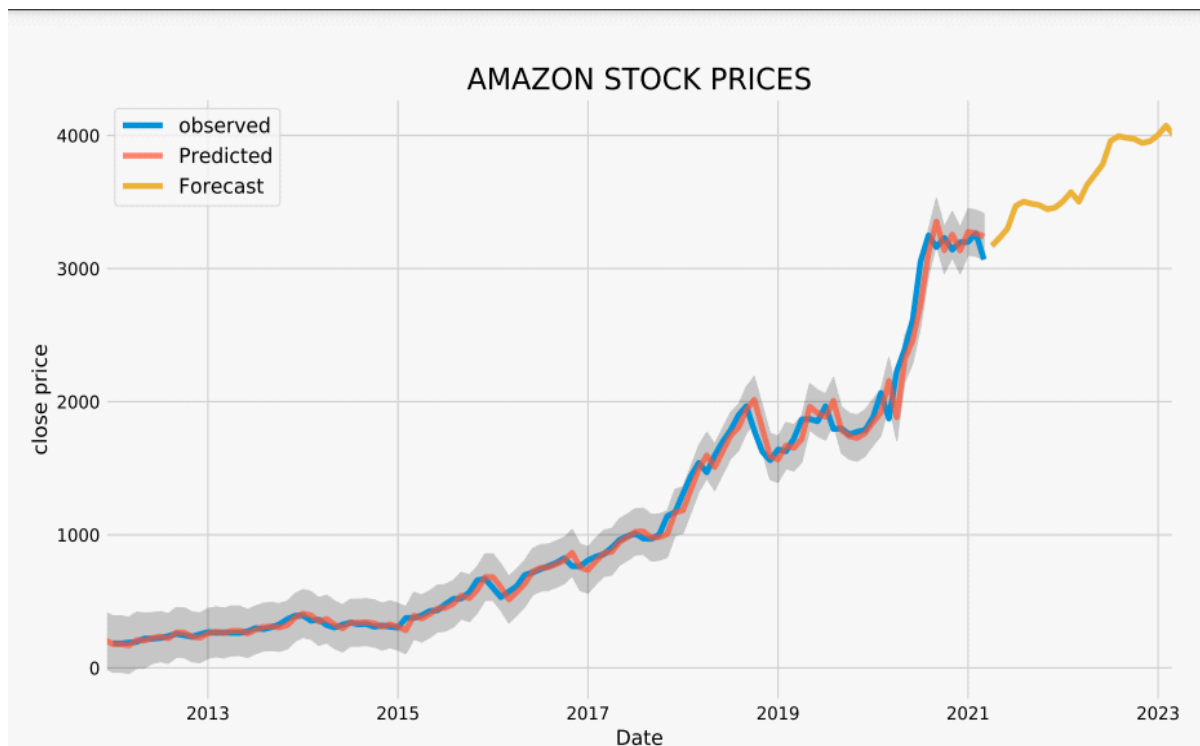
Figure 28: Visualizing of predicted and forecasting graph of Closing Price.

6.3. Implementation of Prophet model

For the implementation of Prophet model, first we introduce the python packages and libraries.

```python
#imports
import pandas as pd
import numpy as np
from fbprophet import Prophet
import matplotlib.pyplot as plt
from functools import reduce

%matplotlib inline
import warnings
warnings.filterwarnings('ignore')

import logging, sys
logging.disable(sys.maxsize)


pd.options.display.float_format = "{:,.2f}".format
```

Figure 29: Importing necessary libraries.

The figure 29 represents the importing of necessary libraries for building , predicting, and forecasting of Prophet model. We import prophet from the FBProphet. Warnings are imported to avoid the future warnings at the time of execution. Matplotlib library is used to plot variablesin graphical format.

```python
AMAZON = pd.read_csv('C:/Users/HP/Desktop/Stock market prediction/AMAZON.csv')
```

```python
AMAZON.describe()
```

	Open	High	Low	Close	Adj Close	Volume
count	2,516.00	2,516.00	2,516.00	2,516.00	2,516.00	2,516.00
mean	1,004.71	1,015.30	992.65	1,004.35	1,004.35	4,252,377.31
std	883.72	893.85	871.76	882.81	882.81	2,347,787.13
min	169.62	174.55	166.97	173.10	173.10	881,300.00
25%	299.96	303.59	296.27	299.70	299.70	2,752,825.00
50%	665.58	674.39	659.29	664.65	664.65	3,649,200.00
75%	1,680.00	1,699.93	1,661.03	1,675.07	1,675.07	4,962,775.00
max	3,547.00	3,552.25	3,486.69	3,531.45	3,531.45	24.134.200.00

Figure 30: Importing and describing of Data.

The figure 30 shows the reading of data from the directory and it stored in a variable. Describe function is used for describing the count, mean, std, min, max etc. of each attribute in the dataset. Parsing of date means creating the date column as index.

```
AMAZON = AMAZON[['Date','Close']]

AMAZON.columns = ['ds', 'y']
AMAZON.head(10)
```

	ds	y
0	2011-04-01	180.13
1	2011-04-04	182.94
2	2011-04-05	185.29
3	2011-04-06	182.76
4	2011-04-07	184.91
5	2011-04-08	184.71
6	2011-04-11	184.04
7	2011-04-12	180.48
8	2011-04-13	182.29
9	2011-04-14	181.82

Figure 31: Data Preparation.

The figure 31 shows the data preparation. For the working of prophet, we changed the names of Date and Close attributes into ds and y. y is the attribute that we are used to predict the future.

```
#prophet
AMAZON.set_index('ds').y.plot(figsize=(15,8), title = 'AMAZON STOCK PRICES')
```

Figure 32: Code for Visualizing the data.(amazon data example)

The figure 32 describes the code for plotting the graph. We set the ds attribute which contains date as index and define the size of the plot. Matplotlib library is used for the plotting of graph. The output of the code is visualized in figure 33.

Figure 33: Visualization of Data.

After visualizing the data, we called prophet() for running the prophet model and assigned in a variable called model. By using fit method, we fit our stock price.

```
model = Prophet()
model.fit(AMAZON)
```

```
<fbprophet.forecaster.Prophet at 0x25186f0c9c8>
```

Figure 34: Code for implementing the model.

```
future = model.make_future_dataframe(365, freq='d')

future_boolean = future['ds'].map(lambda x : True if x.weekday() in range(0, 5) else False)
future = future[future_boolean]

future.tail()
```

	ds
3127	2022-03-25
3130	2022-03-28
3131	2022-03-29
3132	2022-03-30
3133	2022-03-31

Figure 35: Predicting and forecasting the future stock price.

The figure 36 shows the prediction and forecasting of future variables. For the creation of future dates, we used a helper function in prophet called make_future_dataframe with the forecasting period of 365 days. Stocks are traded only in weekdays. So, we need to remove the weekends for that we used Boolean expression. The Boolean expression stating that Monday is 0 and Saturday is 5 and if a day is not equal to 0-4 then return as false.

```
In [10]: ▶ forecast = model.predict(future)

In [11]: ▶ forecast.tail()
```

Out[11]:

	ds	trend	yhat_lower	yhat_upper	trend_lower	trend_upper	additive_terms	additive_terms_lower	additive_terms_upper	weekly	weekly_lower
2474	2020-01-20	83.30	75.39	90.57	83.30	83.30	-0.41	-0.41	-0.41	0.41	0.41
2475	2020-01-21	83.28	74.86	90.07	83.28	83.28	-0.48	-0.48	-0.48	0.50	0.50
2476	2020-01-22	83.26	75.33	89.70	83.26	83.26	-0.79	-0.79	-0.79	0.37	0.37
2477	2020-01-23	83.24	75.17	90.16	83.24	83.24	-0.81	-0.81	-0.81	0.52	0.52
2478	2020-01-24	83.22	74.09	89.40	83.22	83.22	-1.10	-1.10	-1.10	0.41	0.41

Figure 36: Forecasting the stock price.

48

The figure 37 shows the forecasted value of stock price. For forecasting the price value, we created forecast and call the predict from the model and pass the predict in the future dataframe. After executing the program, we get the output with lot of attributes. Yhat is the attribute that denote the forecasted value.

```
In [13]: ▶ plt.figure(figsize=(16,6))
            model.plot(forecast);

            <Figure size 1152x432 with 0 Axes>
```
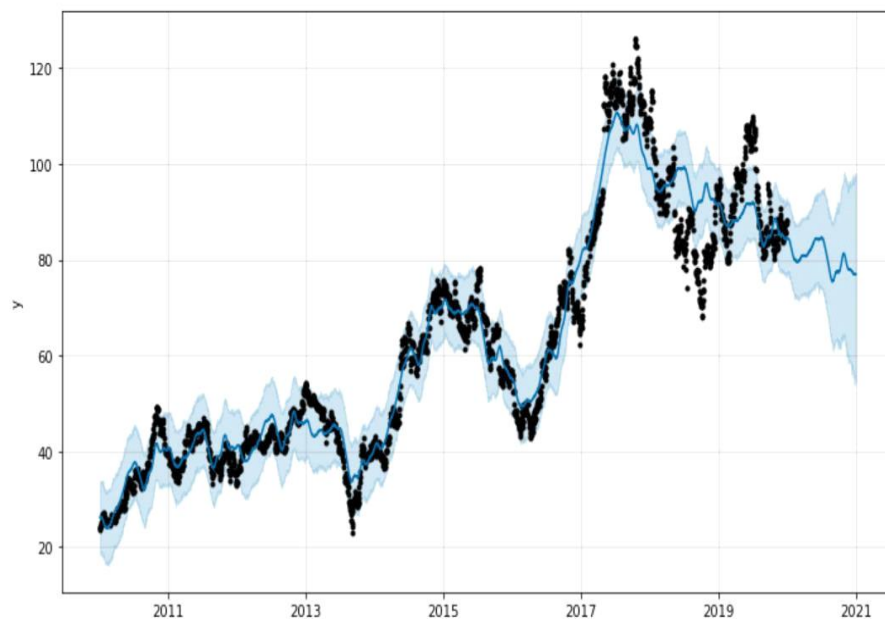

Figure 37: Visualization of forecasted stock price using plot function.

After visualizing the forecasted graph. We visualized the components of the graph by usingplot components.

```
model.plot_components(forecast);
```

Figure 38: Code for displaying components.

The above code is used to visualize the components of the forecasted graph.
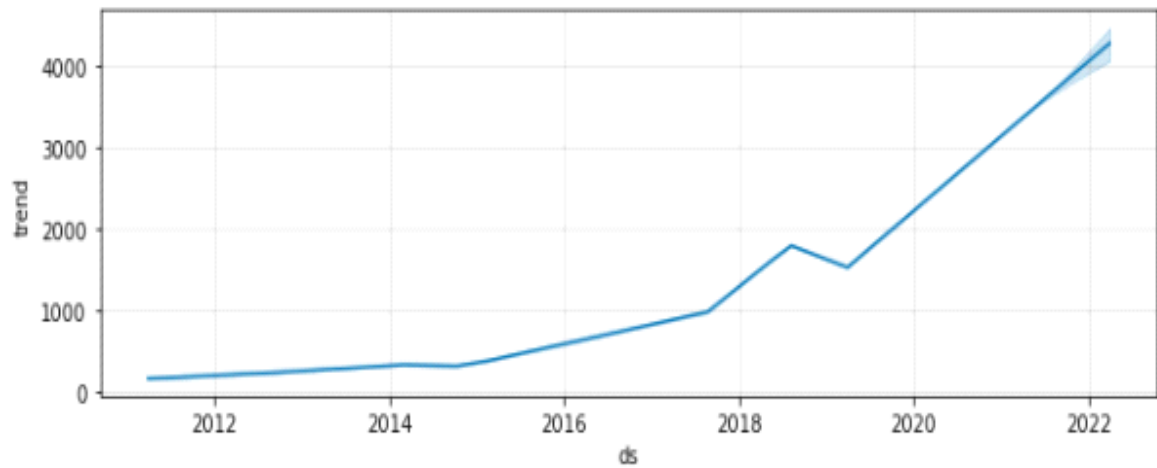
Figure 39: Trend component of the graph.
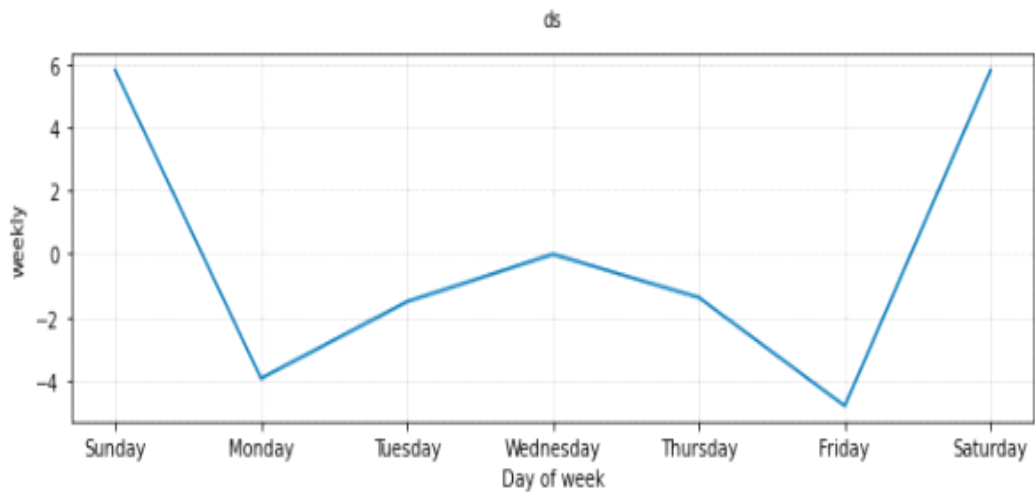
Figure 40: Weekly Component of the graph.
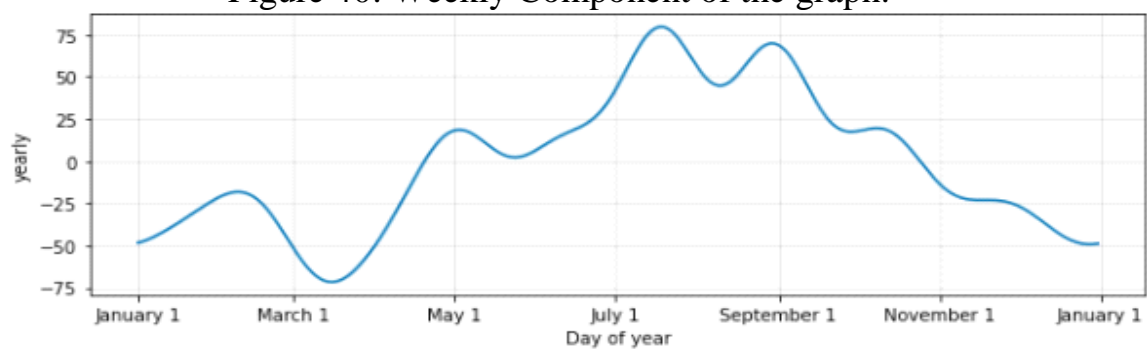
Figure 41: Yearly Component of the graph.

Now, we are going to visualize the forecasted graph. The figure 42 shows the code for visualizing the forecasted graph. In the graph we visualize ds, yhat, yhat_lowe and yhat_upper.

```
AMAZON_forecast = forecast[['ds','yhat', 'yhat_lower', 'yhat_upper']]
df = pd.merge(AMAZON, AMAZON_forecast, on='ds', how='right')
df.set_index('ds').plot(figsize=(16,8), title= 'AMAZON STOCK PRICES',color=['#F29F0E', "#000000", "#0EB6F2", "#0EB6F2"], gri
```

Figure 42: Code for visualizing the Forecasted graph.

Figure 43: Forecasted Graph.

Y – Actual Stoke Price.

Yhat – Forecasted Stock Price.

Yhat_lower and Yhat_upper – Uncertainty interval of stock price.

6.4. Implementing the KERAS – LSTM model

For the implementation of LSTM model, we need to import the libraries initially. The main packages and libraries such as Pandas, NumPy, Matplotlib, Sklearn and keras. Layers. Matplotlib library is used to plot variables in graphical format.

```
#importing Libraries

import pandas as pd
import numpy as np

import matplotlib
import matplotlib.pyplot as plt
import matplotlib.dates as mdates

%matplotlib inline

from sklearn import linear_model
from keras.layers import LSTM,Dense,Dropout
from sklearn.preprocessing import MinMaxScaler
from sklearn.metrics import mean_squared_error, r2_score
from sklearn.model_selection import TimeSeriesSplit
```

Figure 44: Importing of packages and libraries.

```
AMAZON = pd.read_csv('C:/Users/HP/Downloads/AMZN (1).csv', na_values=['null'],index_col='Date',parse_dates=True,infer_datetime_format=True)
AMAZON.head(5)
```

Date	Open	High	Low	Close	Adj Close	Volume
2010-04-01	135.80	136.51	131.18	131.81	131.81	8785800
2010-04-05	132.85	133.74	130.78	131.49	131.49	5816500
2010-04-06	131.23	136.00	131.18	135.56	135.56	7950300
2010-04-07	135.96	136.08	133.86	134.87	134.87	5945400
2010-04-08	134.71	141.25	134.71	140.96	140.96	12689100

Figure 45: Reading the data from directory.

The figure 45 shows the reading of dataset which is in csv format from the directory and assigned to a variable. Parse date is used to create the

date column as index. The output is printed using head(). In the figure we clearly see that date is transferred as index column.

The below figure 46 shows the Preprocessing steps. At first, we check the shape of the date that we import for the analysis. After that we describe the data using describe function. At last we check the missing values in the dataset.

```
AMAZON.shape
```

```
(2769, 6)
```

```
AMAZON.describe()
```

	Open	High	Low	Close	Adj Close	Volume
count	2,769.00	2,769.00	2,769.00	2,769.00	2,769.00	2,769.00
mean	926.74	936.55	915.60	926.44	926.44	4,429,499.89
std	877.56	887.52	865.80	876.68	876.68	2,530,122.21
min	105.93	111.29	105.80	108.61	108.61	881,300.00
25%	257.30	259.74	254.57	257.73	257.73	2,833,300.00
50%	533.74	539.20	526.57	533.16	533.16	3,801,900.00
75%	1,604.00	1,622.72	1,590.72	1,602.91	1,602.91	5,257,500.00
max	3,547.00	3,552.25	3,486.69	3,531.45	3,531.45	42,421,100.00

```
AMAZON.isnull().values.any()
```

Figure 46: Preprocessing of Dataset

Figure 47: Visualizing the Graph with Close Price.

The figure 47 shows the graph of Closing stock price. Matplotlib library is used to plot the graph. Date is act as the x values and Closing stock price act as the y value of the graph.

```
#Correlation Analysis
X=AMAZON.drop(['Close'],axis=1)
X=X.drop(['Adj Close'],axis=1)

X.corrwith(AMAZON['Close']).plot.bar(
        figsize = (16, 6), title = "Correlation with Close", fontsize = 20,
        rot = 90, grid = True)
```

Figure 48: Correlation Analysis.

The figure 48 shows the correlation analysis of the data. For doing correlation analysis first we drop the close from df_final. After that correlation graph is plotted with all other attributes. The title of the graph is 'Correlation with Close'. Having a font size of 20.
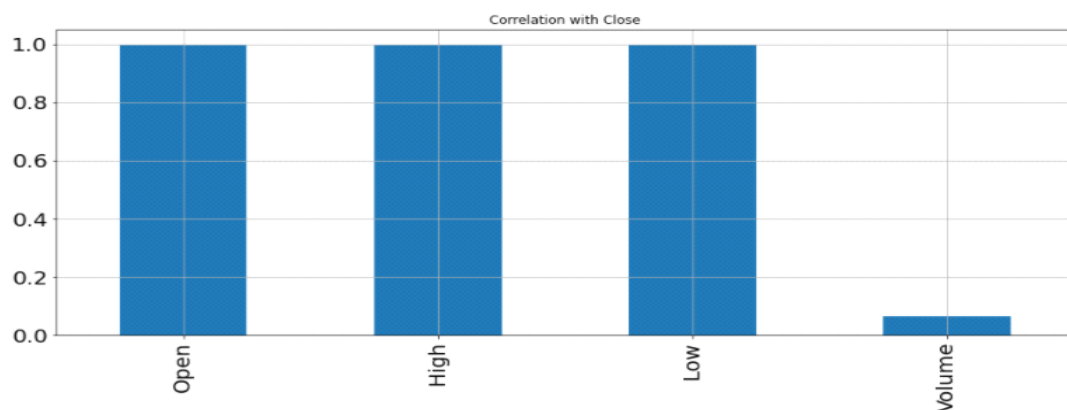
Figure 49: Correlation Graph.

```
from sklearn.preprocessing import MinMaxScaler
scaler = MinMaxScaler()
feature_minmax_transform_data = scaler.fit_transform(test[feature_columns])
feature_minmax_transform = pd.DataFrame(columns=feature_columns, data=feature_minmax_transform_data, index=test.index)
feature_minmax_transform.head()
```

Date	Open	High	Low	Volume
2010-04-01	0.01	0.01	0.01	0.19
2010-04-05	0.01	0.01	0.01	0.12
2010-04-06	0.01	0.01	0.01	0.17
2010-04-07	0.01	0.01	0.01	0.12
2010-04-08	0.01	0.01	0.01	0.28

Figure 50: Normalizing the data.

The figure 50 shows the process of normalizing the data. For normalizing the data first, we import MinMaxScaler from Sklearn. Preprocessing. After that define scalar is equal to MinMaxScaler.

The below figure 51 shows the code for displaying the shape of feature column and target column. After printing the shapes, we shift the target array because we want to predict n+1thday value. Then we took last 180 rows as validation set. Finally, we print the shape and the target close price.

```
display(feature_minmax_transform.head())
print('Shape of features : ', feature_minmax_transform.shape)
print('Shape of target : ', target_adj_close.shape)

# Shift target array because we want to predict the n + 1 day value


target_adj_close = target_adj_close.shift(-1)
validation_y = target_adj_close[-180:-1]
target_adj_close = target_adj_close[:-180]

# Taking Last 90 rows of data to be validation set
validation_X = feature_minmax_transform[-180:-1]
feature_minmax_transform = feature_minmax_transform[:-180]
display(validation_X.tail())
display(validation_y.tail())

print("\n -----After process------ \n")
print('Shape of features : ', feature_minmax_transform.shape)
print('Shape of target : ', target_adj_close.shape)
display(target_adj_close.tail())
```

```
-----After process------

Shape of features :  (2589, 4)
Shape of target :  (2589, 1)
```

	Close
Date	
2020-07-08	3,182.63
2020-07-09	3,200.00
2020-07-10	3,104.00
2020-07-13	3,084.00
2020-07-14	3,008.87

Figure 51: Displaying dataset after Normalization.

```
ts_split= TimeSeriesSplit(n_splits=10)
for train_index, test_index in ts_split.split(feature_minmax_transform):
        X_train, X_test = feature_minmax_transform[:len(train_index)], feature_minmax_transform[len(train_index): (len(train_index)+len(test_index))]
        y_train, y_test = target_adj_close[:len(train_index)].values.ravel(), target_adj_close[len(train_index): (len(train_index)+len(test_index))].values.ravel()
```

```
X_train.shape
```

(2124, 4)

```
y_train.shape
```

(2124,)

```
X_test.shape
```

(212, 4)

```
y_test.shape
```

(212,)

Figure 52: Train and Test Splitting of Data.

The figure 52 shows the splitting of training and testing data using
timeseries split. Then weprinted the shapes of xtrain, ytrain, xtest, ytest.
After that we started to process the data for building the LSTM model.
For creating the models, we import Sequential from keras.models,
Dense from keras.layers,

```
from keras.models import Sequential
from keras.layers import Dense
import keras.backend as K
from keras.callbacks import EarlyStopping
from keras.optimizers import Adam
from keras.models import load_model
from keras.layers import LSTM
K.clear_session()
model_lstm = Sequential()
model_lstm.add(LSTM(16, input_shape=(1, X_train.shape[1]), activation='relu', return_sequences=False))
model_lstm.add(Dense(1))
model_lstm.compile(loss='mean_squared_error', optimizer='adam')
early_stop = EarlyStopping(monitor='loss', patience=5, verbose=1)
history_model_lstm = model_lstm.fit(X_tr_t, y_train, epochs=200, batch_size=8, verbose=1, shuffle=False, callbacks=[early_stop])
```

Figure 53: LSTM Model Building.

After getting the benchmark model we started to process the data for
building the LSTM model.For creating the models, we import
Sequential from keras.models, Dense from keras.layers,

Keras.backend, Early stopping from keras.callbacks, Adam from keras.optimizer, Loadmodel from keras.models and LSTM from keras.layers.

```python
#Evaluation of model
y_pred_test_lstm = model_lstm.predict(X_tst_t)
y_train_pred_lstm = model_lstm.predict(X_tr_t)
print("The R2 score on the Train set is:\t{:0.3f}".format(r2_score(y_train, y_train_pred_lstm)))
r2_train = r2_score(y_train, y_train_pred_lstm)

print("The R2 score on the Test set is:\t{:0.3f}".format(r2_score(y_test, y_pred_test_lstm)))
r2_test = r2_score(y_test, y_pred_test_lstm)
```

```
The R2 score on the Train set is:      0.999
The R2 score on the Test set is:       0.956
```

Figure 54: Evaluation of model using R2 Score.

R2 score is regression score function. It is statistical measure that which is used to describe the proportion of variance of a dependent variable that is explained in regression model. It can be negative or positive. Here the R2 score for train set is 0.999 and R2 score for test set is 0.956.

```python
y_pred_test_LSTM = model_lstm.predict(X_tst_t)
```

```python
plt.figure(figsize=(12,6))
plt.plot(y_test, label='True')
plt.plot(y_pred_test_LSTM, label='LSTM')
plt.title("LSTM's_Prediction - AMAZON STOCK PRICES")
plt.xlabel('Observation')
plt.ylabel('INR_Scaled')
plt.legend()
plt.show()
```

Figure 55: Code for Visualizing of LSTM
Prediction
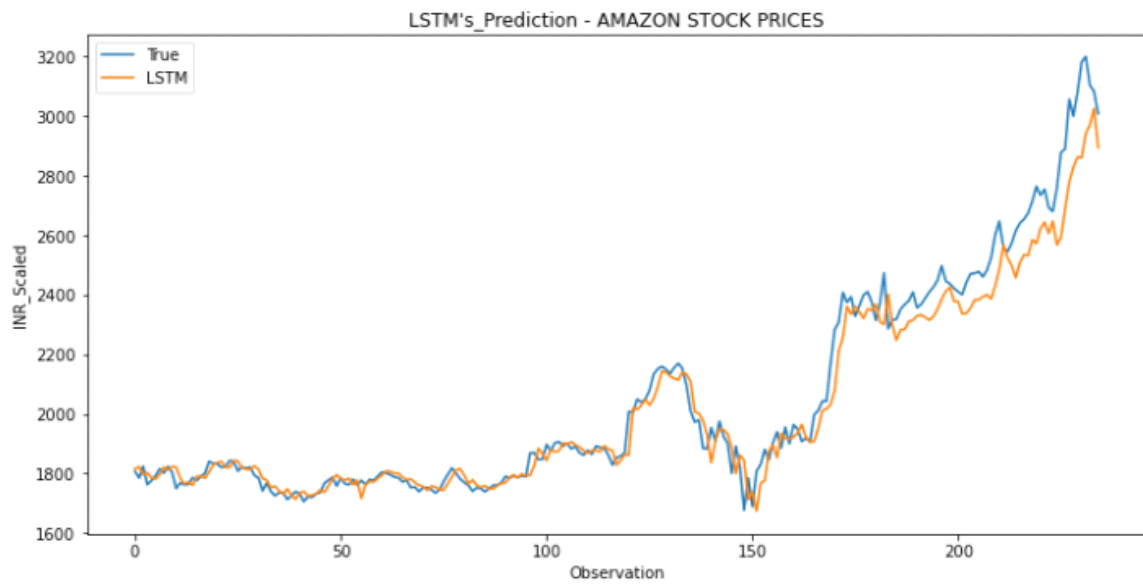
Figure 56: Predicted Graph.

6.5. Predicted Graphs of all compagnies – ARIMA model

Sl.No	Bank Name	Predicted Graphs
1	AMAZON	
2	APPLE	
3	**GOOGLE**	

Table 1: Predicted Graphs - ARIMA.

6.6. Predicted Graphs of compagnies – Prophet

Sl.No	Bank Name	Predicted Graphs
1	AMAZON	
2	APPLE	
3	GOOGLE	

Table 2: Predicted Graphs - Prophet.

• Predicted Graphs of all banks – Keras-LSTM model

Sl.No	Bank Name	Predicted Graphs
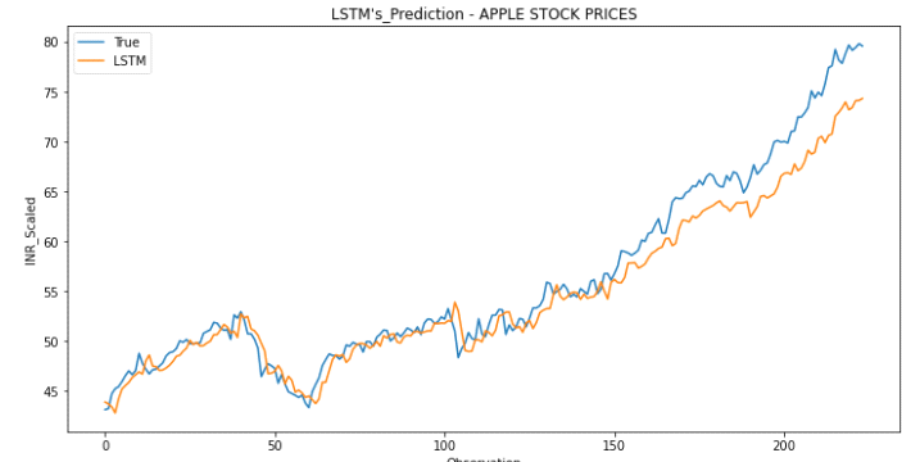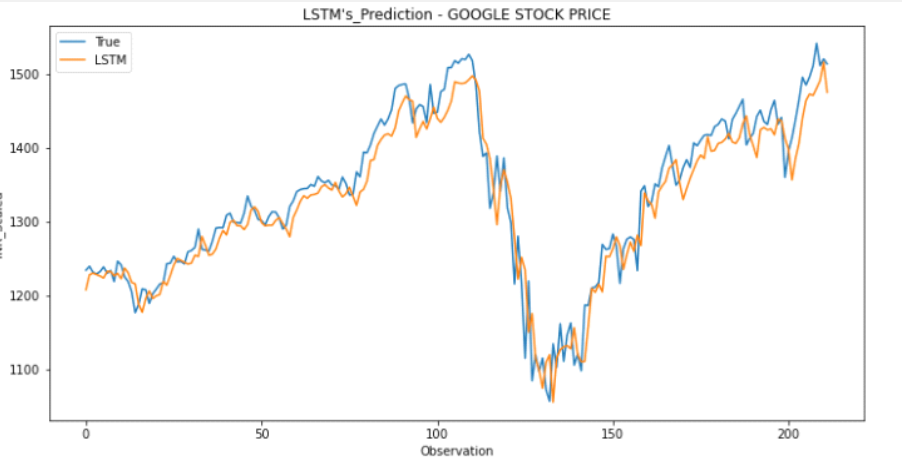
1	AMAZON	
2	APPLE	
3	GOOGLE	

Table 3: Predicted Graphs - Keras with LSTM.

Chapter 7. Testing and Evaluation

7.1. RMSE – Root Mean Square Error Value

The RMSE value is the square root of Mean Square Error. RMSE measures the difference between predicted value and the original value. We know that lower RMSE value gives the better performance in time series models. The RMSE value is calculated for three time series models and compared between each of them. The final phase of the project is the execution ofevaluated result.

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

Equation 6: Formulae for RMSE.

The Equation 6 is the formulae for calculating the RMSE value. Here n stands for positive integer, Pi for ideal distance and Oi for the performance.

7.2. RMSE value – ARIMA Model

7.2.1. AMAZON

```
y_forecasted = pred.predicted_mean
y_truth = monthly_mean['2011-12-31':]

from sklearn import metrics
print('Mean Absolute Error:', metrics.mean_absolute_error(y_truth, y_forecasted))
print('Mean Squared Error:', metrics.mean_squared_error(y_truth, y_forecasted))
print('Root Mean Squared Error:', np.sqrt(metrics.mean_squared_error(y_truth, y_forecasted)))

Mean Absolute Error: 54.62281905757571
Mean Squared Error: 7358.8388235904
Root Mean Squared Error: 85.78367457500524
```

Figure 57: Validation Report of AMAZON stock price.

The figure 57 shows the Validation report of AMAZON stock price. Here we import metrics from Sklearn for calculating the errors between the predicted model and actual model. We also calculated RMSE , MAE, and MSE.

RMSE of AMAZON =85.78

7.2.2. APPLE

```
y_forecasted = pred.predicted_mean
y_truth = monthly_mean['2011-12-31':]

from sklearn import metrics
print('Mean Absolute Error:', metrics.mean_absolute_error(y_truth, y_forecasted))
print('Mean Squared Error:', metrics.mean_squared_error(y_truth, y_forecasted))
print('Root Mean Squared Error:', np.sqrt(metrics.mean_squared_error(y_truth, y_forecasted)))

Mean Absolute Error: 2.4626403305743603
Mean Squared Error: 13.477614122293293
Root Mean Squared Error: 3.6711870181581996
```

Figure 58:Validation Report of APPLE stock price.

The figure 58 shows the Validation report of APPLE stock price. Here we import metrics from Sklearn for calculating the errors between the predicted model and actual model. We also calculated RMSE , MAE, and MSE.

RMSE of APPLE 3.67.

7.2.3. GOOGLE

```
y_forecasted = pred.predicted_mean
y_truth = monthly_mean['2011-12-31':]

from sklearn import metrics
print('Mean Absolute Error:', metrics.mean_absolute_error(y_truth, y_forecasted))
print('Mean Squared Error:', metrics.mean_squared_error(y_truth, y_forecasted))
print('Root Mean Squared Error:', np.sqrt(metrics.mean_squared_error(y_truth, y_forecasted)))

Mean Absolute Error: 35.488297471602756
Mean Squared Error: 2782.8208529234244
Root Mean Squared Error: 52.752448786036695
```

Figure 59: Validation Report of GOOGLE stock price.

The figure 59 shows the Validation report of GOOGLE stock price. Here we import metrics from Sklearn for calculating the errors between the predicted model and actual model. We also calculated RMSE , MAE, and MSE.

RMSE of GOOGLE = 52.75.

7.3. RMSE Value – PROPHET Model

7.3.1. AMAZON

```
from sklearn.metrics import mean_squared_error,mean_absolute_error
print('Mean Absolute Error:',mean_absolute_error(metric_AMAZON.y,metric_AMAZON.yhat))
print('Mean Squared Error:',mean_squared_error(metric_AMAZON.y,metric_AMAZON.yhat))
print('Root Mean Squared Error:',np.sqrt(mean_squared_error(metric_AMAZON.y,metric_AMAZON.yhat)))

Mean Absolute Error: 77.65379620247134
Mean Squared Error: 16629.598921057743
Root Mean Squared Error: 128.95580220004737
```

Figure 60: Validation Result of AMAZON stock price.

The figure 60 shows the Validation report of AMAZON stock price. Here we import mean squared error and mean absolute error from Sklearn for calculating the errors between the predicted model and actual model. We also calculated RMSE , MAE, and MSE.

RMSE of AMAZON = 128.95.

7.3.2. APPLE

```
from sklearn.metrics import mean_squared_error,mean_absolute_error
print('Mean Absolute Error:',mean_absolute_error(metric_APPLE.y,metric_APPLE.yhat))
print('Mean Squared Error:',mean_squared_error(metric_APPLE.y,metric_APPLE.yhat))
print('Root Mean Squared Error:',np.sqrt(mean_squared_error(metric_APPLE.y,metric_APPLE.yhat)))

Mean Absolute Error: 2.7754684944520216
Mean Squared Error: 19.73795036145253
Root Mean Squared Error: 4.4427413115612
```

Figure 61: Validation Report of APPLE stock price.

The figure 61 shows the Validation report of APPLE stock price. Here we import mean squared error and mean absolute error from Sklearn for calculating the errors between the predicted model and actual model We also calculated RMSE , MAE, and MSE.

RMSE of APPLE = 4.44.

7.3.3. GOOGLE

```
from sklearn.metrics import mean_squared_error,mean_absolute_error
print('Mean Absolute Error:',mean_absolute_error(metric_GOOGLE.y,metric_GOOGLE.yhat))
print('Mean Squared Error:',mean_squared_error(metric_GOOGLE.y,metric_GOOGLE.yhat))
print('Root Mean Squared Error:',np.sqrt(mean_squared_error(metric_GOOGLE.y,metric_GOOGLE.yhat)))

Mean Absolute Error: 36.74722554240493
Mean Squared Error: 3538.3798347479387
Root Mean Squared Error: 59.48428224958202
```

Figure 62: Validation Report of GOOGLE.

The figure 62 shows the Validation report of GOOGLE stock price. Here we import mean squared error and mean absolute error from Sklearn for calculating the errors between the predicted model and actual model. We also calculated RMSE , MAE, and MSE.

RMSE of GOOGLE = 59.48.

7.4. RMSE Value – KERAS with LSTM Model

7.4.1. AMAZON

```
from sklearn import metrics
print('Mean Absolute Error:', metrics.mean_absolute_error(y_test, y_pred_test_LSTM))
print('Mean Squared Error:', metrics.mean_squared_error(y_test,y_pred_test_LSTM ))
print('Root Mean Squared Error:', np.sqrt(metrics.mean_squared_error(y_test,y_pred_test_LSTM )))

Mean Absolute Error: 48.74962753706782
Mean Squared Error: 5610.731840834851
Root Mean Squared Error: 74.90481854216624
```

Figure 63: Validation Report of AMAZON sotock price.

The figure 63 shows the Validation report of AMAZON sotock price. Here we import metrics from Sklearn for calculating the errors between the predicted model and actual model. We also calculated RMSE , MAE, and MSE. After all we print all the values.

RMSE of AMAZON sotock price = 74.90.

7.4.2. APPLE

```
from sklearn import metrics
print('Mean Absolute Error:', metrics.mean_absolute_error(y_test, y_pred_test_LSTM))
print('Mean Squared Error:', metrics.mean_squared_error(y_test,y_pred_test_LSTM ))
print('Root Mean Squared Error:', np.sqrt(metrics.mean_squared_error(y_test,y_pred_test_LSTM )))

Mean Absolute Error: 1.7893405207563127
Mean Squared Error: 5.830101872065758
Root Mean Squared Error: 2.4145603889871463
```

Figure 64: Validation Report of APPLE stock price.

The figure 64 shows the Validation report of APPLE stock price. Here we import metrics from Sklearn for calculating the errors between the predicted model and actual model. We also calculated RMSE , MAE, and MSE. After all we print all the values.

RMSE of APPLE stock price = 2.41.

7.4.3. GOOGLE

```
from sklearn import metrics
print('Mean Absolute Error:', metrics.mean_absolute_error(y_test, y_pred_test_LSTM))
print('Mean Squared Error:', metrics.mean_squared_error(y_test,y_pred_test_LSTM ))
print('Root Mean Squared Error:', np.sqrt(metrics.mean_squared_error(y_test,y_pred_test_LSTM )))

Mean Absolute Error: 22.13527312480053
Mean Squared Error: 883.420178773488
Root Mean Squared Error: 29.72238514610643
```

Figure 65: Validation Report of GOOGLE stock price.

The figure 65 shows the Validation report of GOOGLE stock price. Here we import mean squared error and mean absolute error from Sklearn for calculating the errors between the predicted model and actual model. We also calculated RMSE , MAE, and MSE. After all we print all the values.

RMSE of GOOGLE stock price = 29.72.

The above figures clearly show the testing and evaluation of time series models of di fferent banks. Now we are comparing the models in bar diagrams

7.5.   Comparison of Forecasting Models

We drafted the data in the form of a bar diagram in Jupyter Notebook. For the comparison of RMSE value we import NumPy and Matplotlib libraries. NumPy is python library that deals with arrays. Matplotlib is a visualizing library in python.

Comparison of Forecasting Models Using RMSE value

```
import numpy as np
import matplotlib.pyplot as plt
%matplotlib inline
```

Figure 66: Importing libraries.

```
Models=['ARIMA','PROPHET','KERAS with LSTM']
RMSE=[85.78,128.95,74.9]
graph = np.arange(len(Models))
plt.style.use('fivethirtyeight')
plt.figure(figsize=(10,6))
plt.bar(graph,RMSE, label="RMSE")
plt.xticks(graph,Models)
plt.ylabel("Value of RMSE")
plt.title('Comparison Report Graph - Federal Bank')
plt.legend()
```

Figure 67: Code for Developing Bar Diagram.

The figure 67 describes the code for the developing the Comparison bar diagram. First, we initialize the model names in a variable called Models. Then we store RMSE values in an array variable called RMSE. We use the plot style as five thirty eight. Figure size of the graph is 10,6. Title of the graph is given using plt.title.
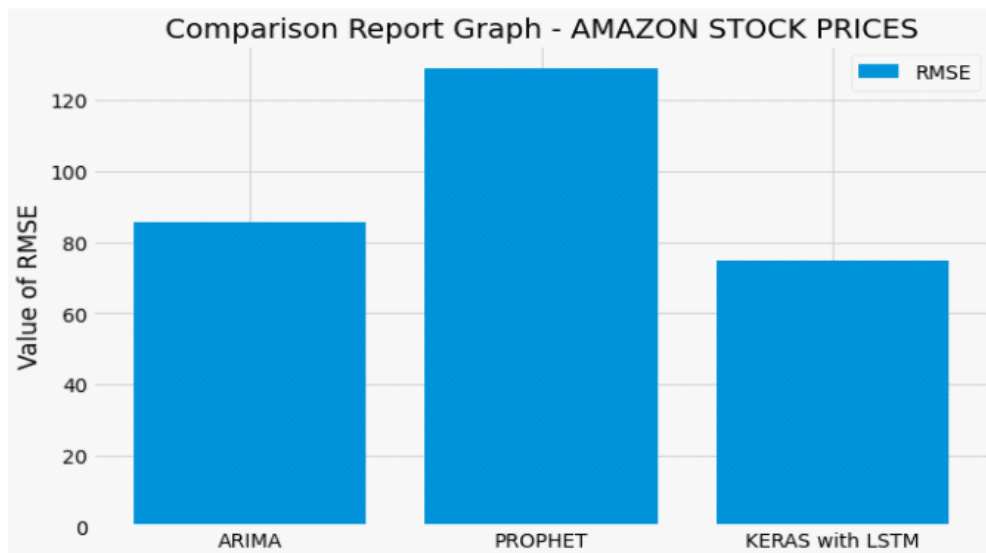
Figure 68: Comparison Report Graph - AMAZON stock price.
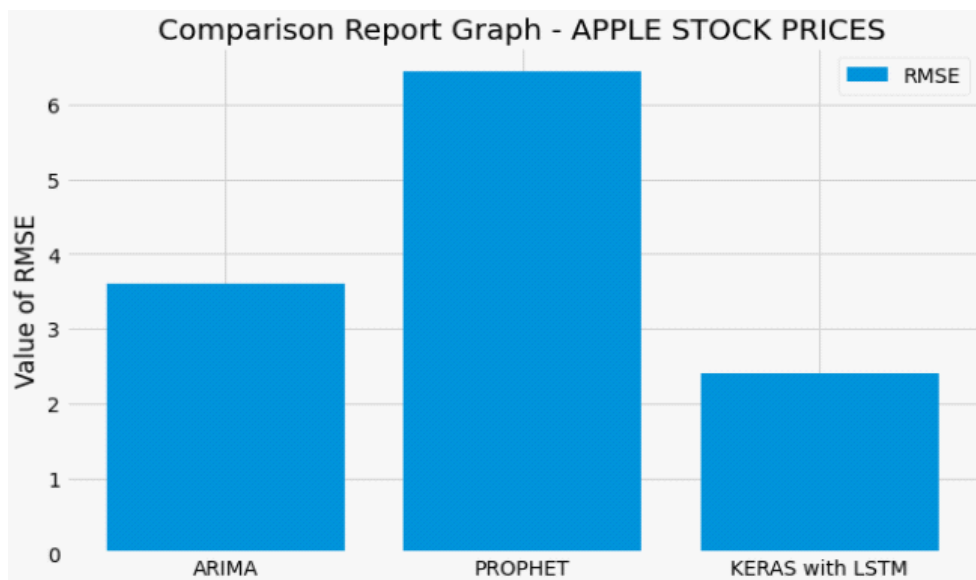
Figure 69: Comparison Report Graph – APPLE stock price.
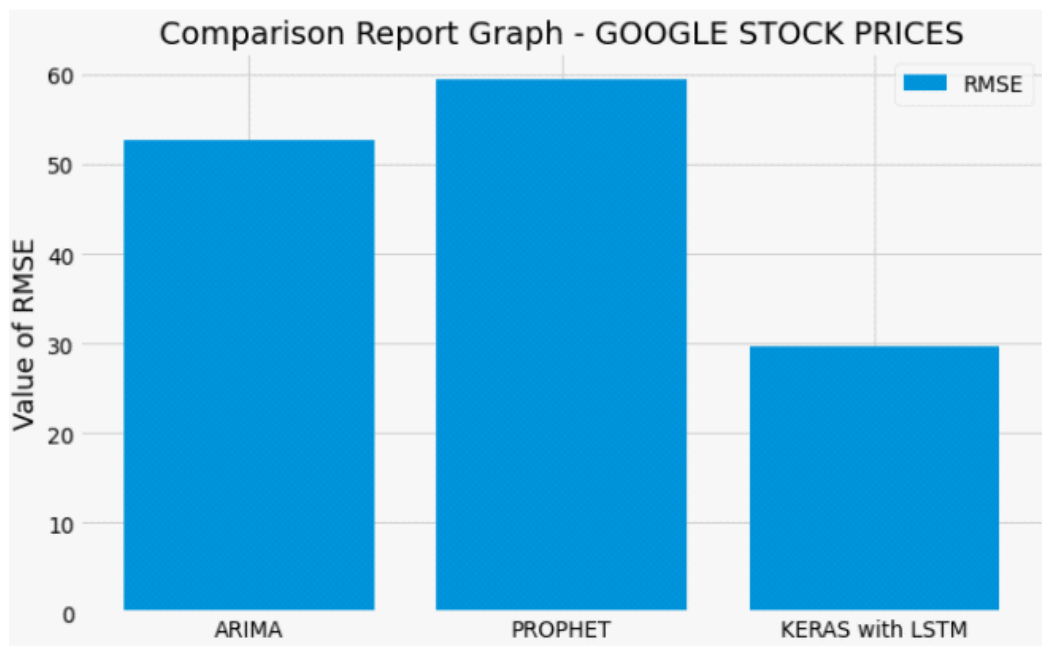
Figure 70: Comparison Report Graph - GOOGLE
stock price.

In all above graphs it is clearly identified that KERAS with LSTM time series model having the least RMSE value. Lower RMSE value have the better performance model.

Chapter 8. Conclusion and Future Work

8.1. Conclusion

In this study, it was a question of predicting the future value of stock prices using different models of machine learning predictions. The prediction of share prices was successful, the main goal of this work was completed. A comparative analysis was made between three ARIMA, PROPHET AND LSTM models applied to historical data over a 10-year period (from April 2011 to April 2021) of three large companies that are AMAZON, APPLE and GOOGLE, data from the famous yahoo! Finance. After applying each algorithm to our different data sets, the results of the predictions were satisfactory for our 3 models but comparing the different RMSE values obtained for each KERAS model with LSTM gives the best time series model for the prediction of the stock price. ARIMA and PROPHET also generated a strong predicted graph with the stock price. By the increase of investors in the stock market the economy of a country can be consequently developed, the different methods of predicting stock market shares that machine learning offers us can be a great help to investors in order to know when and how to invest and which are the different companies that offer better forecasts for stock prices.

8.2. Future Work

The future scope of the project is listed below

- Setting up a website allow the forecast of stock prices

- The accuracy of the stock prediction depends on marketplace, interest rates, dividends, economic fluctuations, political climate. So, need to focus on the external factors as well.

- Introduce more algorithms for the stock price prediction.

References

[1]	J. Chen, "Investopedia," 28 February 2020. [Online]. Available: https://www.investopedia.com/terms/s/stockmarket.asp.
[2]	K. Amadeo, "The Balance," 15 May 2020. [Online]. Available: https://www.thebalance.com/how-does-the-stock-market-work-3306244].
[3]	17 March 2020. [Online]. Available: https://capital.com/stock-market-prediction-definition.
[4]	"Valamis," [Online]. Available: https://www.valamis.com/hub/predictive-analytics.
[5]	"Yahoofinance," 02 June 2020. [Online]. Available: https://finance.yahoo.com/chart/FEDERALBNK.BO#eyJpbnRlcnZhbCI6Im1vbnRoIiwicGVyaW9kkaWNpdHkiOjEsInRpbWVVbml0IjpudWxsLCJjYW5kbGVXaWR0aCI6OC40NzQ1NzYyNzExODY0NDEsImZsaXBwZWQiOmZhbHNlLCJ2b2x1bWVVbmRlcmxheSI6dHJ1ZSwiYWRqIjp0cnVlLCJjcm9zc2hhaXIiOnRydWUsImNoYXJ0.
[6]	R. Das, "PhysicsPI," 13 February 2019. [Online]. Available: https://thephysicspi.blogspot.com/2019/02/advantages-and-disadvantages-of-time.html.
[7]	"advisorymandi," 17 November 2018. [Online]. Available: http://www.advisorymandi.com/blog/advantages-and-disadvantages-of-investing-in- banking-sector/.
[8]	L. Bramble. [Online]. Available: https://smallbusiness.chron.com/stock-market-started-whom-14745.html.
[9]	T. Kimoto, K. Asakawa and M. Yoda, 15 October 2012. [Online]. Available:https://ieeexplore.ieee.org/abstract/document/5726498.
[10]	A. A. Ariyo, A. O. Adewumi and C. K. Ayo, "IEEE Xplore," 23 February 2015. [Online]. Available:https://ieeexplore.ieee.org/abstract/document/7046047.
[11]	S. Selvin, R. Vinayakumar, E. A. Gopalakrishnan, V. K. Menon and K. P. Soman, 04 December2017. [Online]. Available: https://ieeexplore.ieee.org/abstract/document/8126078.
[12]	M. Mazed, "BracU," 2019. [Online]. Available: http://dspace.bracu.ac.bd/xmlui/handle/10361/12818.

[14]	K. K. T. I. Y. F. Y. NAKAMURA, "Wiley Online Library," 4 December 1998. [Online]. Available: https://onlinelibrary.wiley.com/doi/abs/10.1002/(SICI)1099-1174(199703)6:1%3C11::AID-ISAF115%3E3.0.CO;2-3.
[15]	M. M. Ali, "Hindawi," 05 March 2015. [Online]. Available: https://www.hindawi.com/journals/jam/2014/614342/.
[16]	P. R. Charkha, 29 July 2008. [Online]. Available: https://ieeexplore.ieee.org/abstract/document/4579969.
[17]	L. D. Persio, "IRIS.it," [Online]. Available: https://iris.univr.it/retrieve/handle/11562/955101/60620/Artificial%20Neural%20Networks%20architectures%20for%20stock%20price%20prediction%20comparisons%20and%20applications.pdf.
[18]	M. Roondiwal, "Semantic Scholar," 02 June 2020. [Online]. Available: https://pdfs.semanticscholar.org/3f5a/cb5ce4ad79f08024979149767da6d35992ba.pdf.
[19]	"Datascience," 2012. [Online]. Available: http://www.datascience-pm.com/crisp-dm-2/.
[20]	V. Dsouza, "Research gate," July 2018. [Online]. Available: https://www.researchgate.net/figure/CRISP-DM-Model-Taylor-2017_fig1_326235288.
[21]	adishesha. [Online]. Available: https://www.slideshare.net/adisesha12/python-programming-168124234.
[22]	G. McIntire. [Online]. Available: https://www.learndatasci.com/tutorials/python-pandas-tutorial-complete-introduction-for-beginners/.
[23]	"edpresso," [Online]. Available: https://www.educative.io/edpresso/what-is-pandas-in-python.
[24]	DataFlair, 31 May 2019. [Online]. Available: https://data-flair.training/blogs/pandas-HYPERLINK "https://data-flair.training/blogs/pandas-%20dataframe/"HYPERLINK "https://data-flair.training/blogs/pandas-%20dataframe/"dataframe/.
[25]	"W3 Schools and UCF," [Online]. Available: https://webcourses.ucf.edu/courses/1249560/pages/python-lists-vs-numpy-arrays-what-is-the-difference, https://www.w3schools.com/python/numpy_intro.asp.
[26]	"ScikitLearn," [Online]. Available: https://scikit-learn.org/stable/.
[27]	J. T. Point. [Online]. Available: https://www.javatpoint.com/advantage-and-disadvantage-of-tensorflow.
[28]	DATA FLAIR TEAM, "dataflair," 9 May 2020. [Online]. Available: https://data-flair.training/blogs/python-keras-advantages-and-limitations/.

[29]	P. Dar, "Analytics Vidhya," 24 May 2018. [Online]. Available: https://www.analyticsvidhya.com/blog/2018/05/starters-guide-jupyter-notebook/.
[33]	Anaconda Documentation, "Anaconda," [Online]. Available: https://docs.anaconda.com/anaconda/user-guide/getting-started/.
[34]	S. Prabhakaran, "Machine learning plus," [Online]. Available: https://www.machinelearningplus.com/time-series/arima-model-time-series-forecasting- python/.
[35]	A. Choudhary, "Analytics Vidhya," 10 May 2018. [Online]. Available: https://www.analyticsvidhya.com/blog/2018/05/generate-accurate-forecasts-facebook- prophet-python-r/.
[36]	adventuresinmachinelearning, "Adventures in machine learning," [Online]. Available:https://adventuresinmachinelearning.com/keras-lstm-tutorial/.
[37]	colah's blog, "colah's blog," 27 August 2015. [Online]. Available:https://colah.github.io/posts/2015-08-Understanding-LSTMs/.
[38]	"Classic.d2.ai," [Online]. Available: https://classic.d2l.ai/chapter_recurrent-neural-networks/lstm.html.
[39]	A. Singh, "Analytics Vidhya," 25 October 2018. [Online]. Available: https://www.analyticsvidhya.com/blog/2018/10/predicting-stock-price-machine-learningnd- deep-learning-techniques-python/.
[40]	M. Rouse, "TechTarget," [Online]. Available: https://whatis.techtarget.com/definition/time-series-forecasting.
[41]	J. Brownlee, "Machine Learning," 2 December 2016. [Online]. Available:https://machinelearningmastery.com/time-series-forecasting/.