A
Minor Project Report on
"STOCK MARKET PREDICTION"

In partial fulfillment of requirements for the degree of

Bachelor of Technology (B. Tech.)

in

Computer Science and Engineering

Submitted by

Ms. Riya Sinha (170385)

Under the Guidance of

Dr. Anand Sharma

SCHOOL OF ENGINEERING AND TECHNOLOGY

Mody University and Science and Technology
Lakshmangarh, Distt. Sikar-332311

December 2020

A C K N O W L E D G E M E N T

I have taken efforts in this training. However, it would not have been possible without the kind support and help of many of the individuals. I would like to extend my sincere gratitude to all of them.

I'm highly obliged to A Senthil , Assistant Dean SET and HOD (SET deptt.), Mody University of science and technology, for the amenities provided to accomplish this training.

I am obliged to Dr. Anand Sharma for his direction and steady oversight just as giving fundamental data with respect to the mechanical preparation. I might likewise want to communicate my appreciation towards my folks and individuals from Mody University for their caring collaboration and consolation which caused me in consummation of this mechanical preparation. My thanks and gratefulness goes to every one of those individuals who have acutely assisted me with their capacities.

Riya Sinha (170385)

CERTIFICATE

This is to certify that the minor project report entitled "Stock Market Analysis " submitted by Ms. Riya Sinha, as a partial fulfillment for the requirement of B. Tech. VII Semester examination of the School of Engineering and Technology, Mody University of Science and Technology, Lakshmangarh for the academic session 2020-2021 is an original project work carried out under the supervision and guidance of Dr. Anand Sharma has undergone the requisite duration as prescribed by the institution for the project work.

PROJECT GUIDE:

Approval Code: AUT_20_CSE_F4_06

Name: Dr. Anand Sharma

Date: 25th December 2020

HEAD OF DEPARTMENT

Signature:

Name: Dr. A. Senthil

Date: 25th December 2020

EXAMINER-I:

Name: Mr. Suneet Gupta

Dept: SET

EXAMINER-II

Name: Dr. Vikas Raina

Dept: SET

ABSTRACT

The route toward checking the stock expenses has been a problematic task for countless trained professionals, specialists and new examiners. Surely, theorists are uncommonly charmed by the assessment area of stock worth desire. For a respectable and productive theory, various examiners are sharp in knowing the future condition of the protection trade. Incredible and feasible estimate structures for the protections trade help intermediaries, monetary subject matter experts, and examiners by giving consistent information like the future heading of the protections trade. Foreseeing protections trade cost is an erratic task that usually incorporates wide human-PC affiliation. There are various desire techniques for share esteem deciding. Time Series Forecasting is major for share esteem envisioning and other money related model checks. As offer expenses are more nonlinear, more keen time game plan figure systems are required. Existing systems' accuracy are not capable enough in foreseeing. In this we propose to use LSTM Machine Learning Algorithm for gainful foreseeing of stock expense. This will give more correct results when appeared differently in relation to existing stock value expectation calculations.

In this project I'll carry out these tasks:

- To bring the ongoing information of the organization from yahoo finance and google finance, load it and refine the information to deliver a stock examination report.
- Examination to decide the supply of a specific organization, how the stock did as for term of time (week by week, month to month, yearly) lastly discovering which stock is more steady and which is more unstable.

Table of Contents

List of Figures

Chapter 1: Introduction

1. INTRODUCTION

The specialty of estimating stock costs has been a troublesome errand for a significant number of scientists and experts; indeed, speculators are profoundly keen on examination in the region of stock value decreases. For a decent and fruitful venture, numerous financial specialists are sharp in knowing the future circumstance of the securities exchange. In quite a situation a successful expectation framework for securities exchange helped brokers, financial specialists, and examiner by offering help with data like the future estimation of specific stocks. By utilizing stock examination, speculators and dealers show up at value purchasing and selling choices. Examining and assessing past and current information encourages financial specialists and brokers to pick up an edge in the business sectors to settle on educated choices.

STOCKS: A stock (otherwise called value) is a security that speaks to the responsibility for part of a partnership. This entitles the proprietor of the stock to an extent of the organization's resources and benefits equivalent to how much stock they own. Units of stock are designated "shares." Stock possession suggests that the investor claims a cut of the organization equivalent to the quantity of offers held as an extent of the organization's absolute exceptional offers. For example, an individual or substance that claims 100,000 portions of an organization with 1,000,000 remarkable offers would have a 10% possession stake in it.

BONDS: Bonds are units of corporate obligation given by organizations and securitized as tradable resources. Security is alluded to as a fixed pay instrument since securities customarily paid a fixed financing cost (coupon) to debtholders. Variable or skimming loan fees are likewise now very normal. Security costs are contrarily associated with loan fees: when rates go up, security costs fall and the other way around. Bonds have development dates so, all things considered, the chief sum should be taken care of in full or danger default. The prominence of financial exchange exchanging is developing incredibly quickly, which is urging specialists to discover new strategies for the expectation utilizing new procedures and the gauging strategy isn't simply useful to analysts, yet in addition, helps speculators or any person dealing with the protections trade to help envision the stock records. A deciding model with fair accuracy is required. In this work, I will utilize quite possibly the most exact

determining innovation RNNs and LSTM units, which help financial specialists investigators or any person who is keen on putting resources into the securities exchange by giving them correct information on the future circumstance of the securities exchange.

1.1. PRESENT SYSTEM

In this current framework, a sliding window calculation has been readied which prompts wastage of time and memory space. The desire for future stock expense by Sliding Window Algorithm is less gainful by virtue of taking care of bothersome data. The Sliding Window Algorithm which is used in the current structure isn't excessively much amazing in taking care of non-straight information. In this way, our proposed future stock worth figure is done using LSTM (Long Short-Term Memory) which is more successful than Sliding Window Algorithm. At the point when you analyze various kinds of long-haul speculation methodologies, there are not many that can coordinate the profits acknowledged by stocks. Indeed, even land has not stayed aware of the 10% normal pace of return seen by huge stocks yearly since World War II.

Playing out an examination prior to making a speculation is an unquestionable requirement. It is simply after a careful examination that you can make a few suppositions into the worth and future execution of a venture. Regardless of whether you are following stock exchanging tips, it is ideal to do some exploration, just to guarantee that you are making a speculation that is required to get you most extreme returns. When you put resources into value, you buy a few bits of a business hoping to bring in cash upon increment in the estimation of the business. Prior to purchasing anything, be it a vehicle or telephone, you do some level of exploration about its exhibition and quality. A venture is the same. It is your well-deserved cash that you are going to contribute, so you should have a reasonable information on the thing you are putting resources into.

Alongside the potential for more prominent income, notwithstanding, comes the danger of dynamic market influences that can make stock costs vacillate significantly. Potential securities exchange financial specialists should consistently know that there is no ensured return on their venture. So it's truly critical to have full information and examination behind any stock trade to make it productive.

1.2. PROPOSED SYSTEM

The point of the proposed game plan is to help theorists in their decisions and recommend buying the assets which give the best advantages. In our paper, we propose to use LSTM (Long Short-Term Memory) count to give capable stock worth desire. Long transient memory (LSTM) is a phony neural association (RNN) plan used in the field of significant learning. In contrast to standard feed forward neural organizations, LSTM has input associations that make it a "universally useful PC". It cannot just cycle single information focuses, (for example, pictures), yet in addition whole groupings of information. In view of the dropout cycle which happens in the LSTM calculation, it is relatively quicker than the Sliding window calculation. LSTM calculation is more productive in foreseeing the future stock cost than the Sliding Window calculation in light of eliminating the undesirable information. The time and memory utilization is likewise decreased when contrasted with the energizing framework because of the dropout process. LSTM calculation is more reasonable in taking care of non-straight information.

In this work, I present a "LSTM" approach to manage foresee monetary trade records. As of now there's a lot of obfuscated financial pointers and moreover the change of the protections trade is unfathomably, horrible. Regardless, as the development is getting advanced, the event to get a predictable fortune from the monetary trade is extended. Moreover, it in like manner makes experts find the most illuminating pointers to improve a conjecture.

Besides, by and by, the estimate of the market regard is essential to help you in boosting the advantage of your venture opportunity purchase while keeping the peril low. Furthermore, this is significant on the grounds that you need to put your cash in a stock which will increment in incentive over the long haul and not lessen. So RNNs or intermittent neural organizations have demonstrated to be perhaps the most remarkable models for handling consecutive information, the long momentary memory is quite possibly the best RNN structures.

LSTM show the memory cell, a unit of estimation that replaces the customary fake neurons in the hid layer of the association. With these memory cells, organizations can effectively relate memories and data distantly as expected. Henceforth, the suit to get a handle on the structure of the information progressively over the long run with high forecast limit.

In what manner will my framework help?

1. Performing examination or investigation prior to making a speculation is an unquestionable requirement. It is simply after intensive examination that you can make a few suspicions into the worth and future execution of a speculation.

2. Regardless of whether you are following stock exchange tips, it is ideal to do some exploration, just to guarantee that you are making a venture that is required to get you the most extreme returns.

3. At the point when you put resources into value, you buy a few segments of a business hoping to bring in cash upon an expansion in the estimation of the business. Prior to purchasing anything, be it a vehicle or telephone, you do some level of examination about its exhibition and quality. Venture is the same.

4. It is your well deserved cash that you are going to contribute, so you should have reasonable information on the thing you are putting resources into.

Chapter 2: System Design

2.1. SYSTEM FLOWCHART

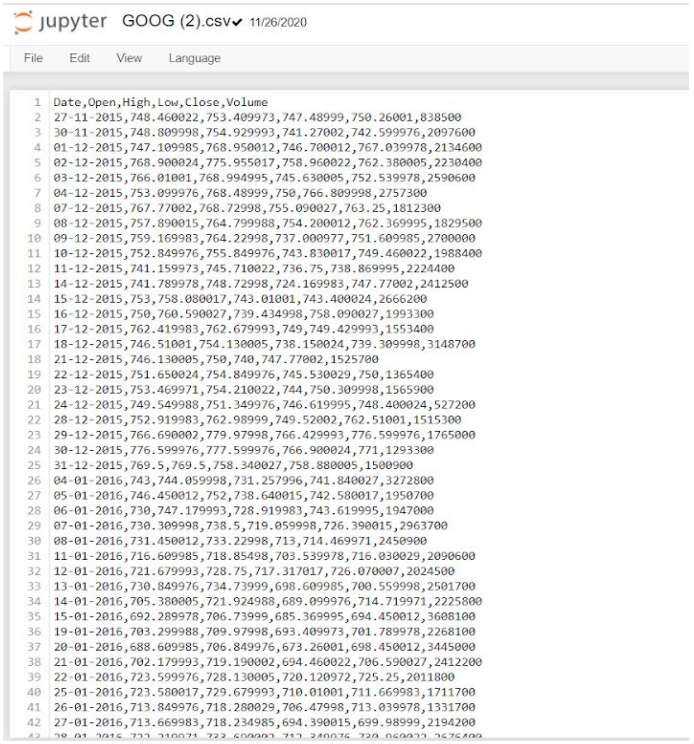
The system flow chart is shown in figure 2.2 that contains the following steps:

2.1.1. DATASET CREATION (Download share price data)

An offer cost is the cost of a single part of different saleable heaps of an association, subordinate, or other financial assets. In layman's terms, the stock expense is the most raised total someone is glad to pay for the stock or the least aggregate that it will, in general, be bought for. We download the offer information of a specific organization utilizing the Yahoo finance source. The information, for example, date, open offer value, high worth, low worth, last offer worth, close offer value, absolute exchange and turn over qualities are utilized.

2.1.2. DATA SETS

A dataset is a grouping of data. Most consistently an educational list identifies with the substance of a lone database table, or a single real data network, where each fragment of the table addresses a particular variable, and each line analyzes to guarantee individuality from the informational index being referred to. The informational index records esteems for every one of the factors, for example, the tallness furthermore, weight of an article, for each person from the instructive assortment. Every value is known as a datum. The educational assortment may include data for in any event one individual, contrasted with the amount of segments. Here we keep all our data as CSV archives. In enlisting, a comma-separated characteristics (CSV) record is a delimited substance archive that uses a comma to disengage values. CSV records store even data (numbers and text) in plain substance. Each line of the archive is a data record. Each record contains at any rate one field, disconnected by commas. The use of the comma as a field separator is the wellspring of the name for this record plan. Our dataset is kept in even arrangement to dominate with qualities, for example, date, open, high, low, close, volume.


jupyter GOOG (2).csv 11/26/2020

File Edit View Language

```
Date,Open,High,Low,Close,Volume
27-11-2015,748.460022,753.409973,747.48999,750.26001,838500
30-11-2015,748.809998,754.929993,741.27002,742.599976,2097600
01-12-2015,747.109985,768.950012,746.700012,767.039978,2134600
02-12-2015,768.900024,775.955017,758.960022,762.380005,2230400
03-12-2015,766.01001,768.994995,745.630005,752.539978,2590600
04-12-2015,753.099976,768.48999,750,766.809998,2757300
07-12-2015,767.77002,768.72998,755.090027,763.25,1812300
08-12-2015,757.890015,764.799988,754.200012,762.369995,1829500
09-12-2015,759.169983,764.22998,737.000977,751.609985,2700000
10-12-2015,752.849976,755.849976,743.830017,749.460022,1988400
11-12-2015,741.159973,745.710022,736.75,738.869995,2224400
14-12-2015,741.789978,748.72998,724.169983,747.77002,2412500
15-12-2015,753,758.080017,743.01001,743.400024,2666200
16-12-2015,750,760.590027,739.434998,758.090027,1993300
17-12-2015,762.419983,762.679993,749,749.429993,1553400
18-12-2015,746.51001,754.130005,738.150024,739.309998,3148700
21-12-2015,746.130005,750,740,747.77002,1525700
22-12-2015,751.650024,754.849976,745.530029,750,1365400
23-12-2015,753.469971,754.210022,744,750.309998,1565900
24-12-2015,749.549988,751.349976,746.619995,748.400024,527200
28-12-2015,752.919983,762.98999,749.52002,762.51001,1515300
29-12-2015,766.690002,779.97998,766.429993,776.599976,1765000
30-12-2015,776.599976,777.599976,766.900024,771,1293300
31-12-2015,769.5,769.5,758.340027,758.880005,1500900
04-01-2016,743,744.059998,731.257996,741.840027,3272800
05-01-2016,746.450012,752,738.640015,742.580017,1950700
06-01-2016,730,747.179993,728.919983,743.619995,1947000
07-01-2016,730.309998,738.5,719.059998,726.390015,2963700
08-01-2016,731.450012,733.22998,713,714.469971,2450900
11-01-2016,716.609985,718.85498,703.539978,716.030029,2090600
12-01-2016,721.679993,728.75,717.317017,726.070007,2024500
13-01-2016,730.849976,734.73999,698.609985,700.559998,2501700
14-01-2016,705.380005,721.924988,689.099976,714.719971,2225800
15-01-2016,692.289978,706.73999,685.369995,694.450012,3608100
19-01-2016,703.299988,709.97998,693.409973,701.789978,2268100
20-01-2016,688.609985,706.849976,673.26001,698.450012,3445000
21-01-2016,702.179993,719.190002,694.460022,706.590027,2412200
22-01-2016,723.599976,728.130005,720.120972,725.25,2011800
25-01-2016,723.580017,729.679993,710.01001,711.669983,1711700
26-01-2016,713.849976,718.280029,706.47998,713.039978,1331700
27-01-2016,713.669983,718.234985,694.390015,699.98999,2194200
```

Figure 2.1 Dataset of stock prices

2.1.3. DATA PRE-PROCESSING

Data / Information preprocessing is a critical headway in AI adventures. Data gathering methodologies are routinely estimatedly controlled, coming to fruition in out-of-range regards missing characteristics, etc Examining data that has not been purposely screened for such issues can make misleading outcomes. Thus, the depiction and nature of data is in particular preceding running an examination. Routinely, data preprocessing is the primary time of an AI adventure, especially in computational data. If there is a ton of unessential and abundance information present or disorderly and tricky data, by then data divulgence during the readiness stage is more inconvenient. Data course of action and filtering steps can take a great deal of taking care of time. Data preprocessing joins cleaning, model assurance, normalization, change, incorporate extraction and decision and so on The consequence of data preprocessing is the last getting ready set.

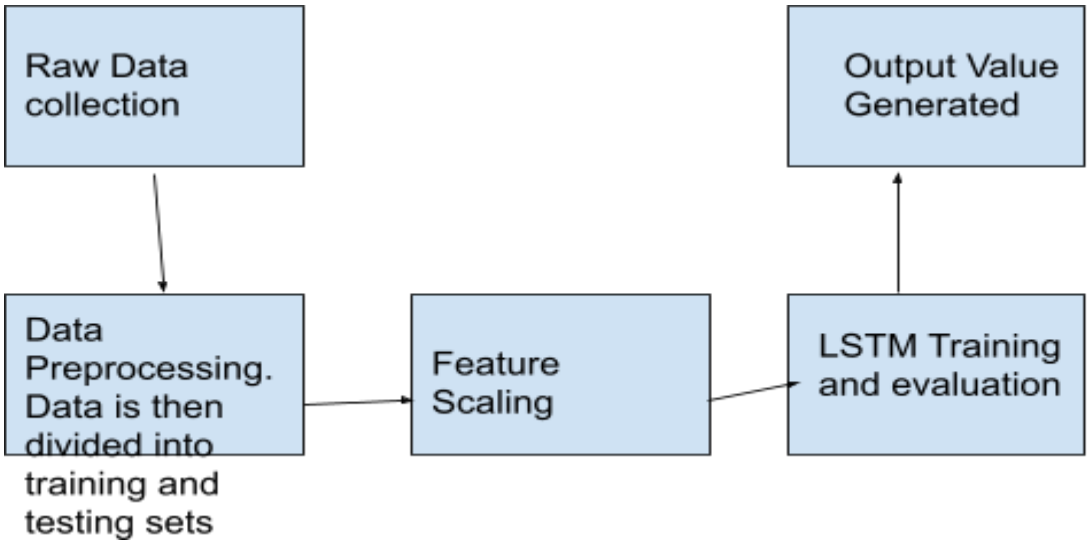


Figure 2.2 System Flowchart

2.1.4. FEATURE SCALING

Feature / Highlight scaling is a technique used to standardize the extent of self-ruling variables or features of data. In data preprocessing, it is generally called data normalization and is overall performed during the data preprocessing step. Since the extent of assessments of unrefined data varies comprehensively, target limits won't work properly without normalization. Consequently, the extension, in light of everything, should be normalized so every part contributes around proportionately to the last distance. Another inspiration driving why feature scaling is applied is that angle drop meets a lot quicker and includes scaling. Also known as min-max scaling, is the least complex technique and comprises in rescaling the scope of highlights to scale the reach in [0, 1] or [−1, 1]. Choosing the objective reach relies upon the idea of the information. The overall equation is given as

$$x' = z - \min(x) / \max(x)-\min(x)$$

where x is an original value, x’ is the scaled value.

2.1.5. LSTM TRAINING AND EVALUATION.

Long momentary memory (LSTM) is a fake repetitive neural organization (RNN) designing used in the field of significant learning. Not at all like standard analysis neural associations,

LSTM has input affiliations that make it a "all around valuable PC". It can handle single data centers, yet likewise entire courses of action of data. Long Short-Term Memory (LSTM) networks are an enlargement for irregular neural associations, which basically widens their memory. Thusly it is suitable to pick up from critical experiences that have amazingly drawn out time interval slacks in the center. The units of a LSTM are used as building units for the layers of a RNN, which is then regularly called a LSTM association. LSTM engages RNN's to remember their commitments all through a broad timespan. This is because LSTM contains their information in a memory, that is a ton like the memory of a PC considering the way that the LSTM can scrutinize, create and eradicate information from its memory.

LSTM memory cell, has the accompanying three parts:

1. Dismissal portal: the neglect to recall entryway picks when express fragments of the phone state are to be replaced with later information. It yields gauges close to 1 for parts of the cell express that should be held, and zero for values that should be ignored.
2. Information door: considering the data (i.e., past yield o(t1), input x(t), and past cell state c(t-1)), this portion of the association learns the conditions under which any information should be taken care of (or revived) in the phone state.
3. Yield passage: dependent upon the information and cell express, this spot picks what information is incited forward (i.e., yields o(t) and cell state c(t)) to the accompanying center in the association.

Thusly, LSTM networks are ideal for exploring how assortment in one stock's expense can impact the expenses of a couple of various stocks all through a critical time interval. They can moreover pick (in an amazing plan) for how long information about express past examples in stock value development should be held to all the more precisely anticipate future designs of stock costs.

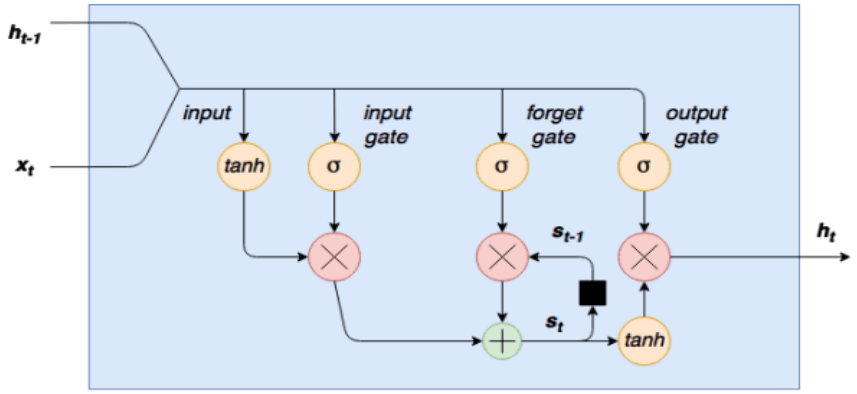


Figure 2.3 Overview of LSTM

2.1.6. VISUALISING THE PREDICTION

We use matplotlib which is a plotting library to see the desire for future stock expense. Matplotlib is a 2-D plotting library that helps in envisioning figures. Matplotlib copies Matlab like outlines and portrayals. Matlab isn't free, is hard proportional and as a programming language is dull. Thusly, matplotlib in Python is used as it is a good, free and basic library for data insight. StockPrice data will be envisioned using matplotlib as 2D graphs. Current information and Predicted information will be introduced for examination. Exactness will be introduced depending on forecast and existing information.

Chapter 3: Hardware and Software Details

3.1. HARDWARE REQUIREMENT

- **Operating System:** Windows 10/8.1/8/7
- **Processors:** Any Intel 64-x86 processor
- **RAM**: 4 GB is recommended

3.2. SOFTWARE REQUIREMENT

- **Jupyter Notebook**

  Jupyter Notebook is an open-source web application that grants us to make and share codes and records. It gives an atmosphere, where you can chronicle your code, run it, look at the outcome, imagine data, and see the results without leaving the atmosphere. This makes it an advantageous contraption for performing beginning to end data science work measures – data cleaning, quantifiable showing, assembling and planning AI models, imagining data, and many, various occupations. Jupyter Notebooks truly sparkle when you are as yet in the prototyping stage. This is on the grounds that your code is written in free cells, which are executed exclusively. This permits the client to test a particular square of code in a task without executing the code from the beginning of the content. Numerous other IDE conditions (like RStudio) additionally do this severally, yet I have by and by discovered Jupyter's individual cell structure to be the best of the part.

What Jupyter notebook offers?

- As a developer, you can play out the accompanying utilizing Notebook.
- Compose and execute code in Python.
- Document your code that bolsters mathematical equations.
- Make/Upload notebooks.
- Import/Save notebooks from/to your device.
- This allows the user to test a specific block of code in a project without having to execute the code from the start of the script.
- Coordinate PyTorch, TensorFlow, Keras, OpenCV.
- Since they are more interactive than an IDE platform, they are widely used to display codes in a more pedagogical manner.

Creating notebook in jupyter notebook

Step 1:

To open a new Jupyter notebook, click on the 'New' option on the right-hand side of the page. Here, you get four options to choose from:

- Python 3
- Text File
- Folder
- Terminal

In a Text File, you are given a blank slate. Add whatever alphabets, words and numbers you wish. It basically works as a text editor (similar to the application on Ubuntu). You also get the option to choose a language (there are a plethora of them given to you) so you can write a script in that. You also have the ability to find and replace words in the file.In the Folder option, it does what the name suggests. You can create a new folder to put your documents in, rename it and delete it, whatever your requirement.The Terminal works exactly like the terminal on your Mac or Linux machine (cmd on Windows). It does a job of supporting terminal sessions within your web browser. Type *python* in this terminal and voila! Your python script is ready to be written.

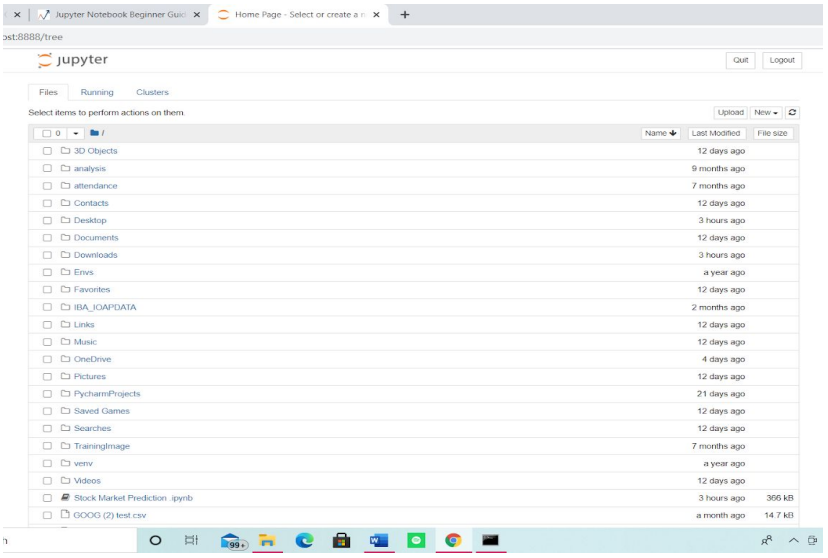


Figure 3.1 Opening Jupyter

Step 2:

We will select the Python 3 option from the 'New' option. You will get the below screen:

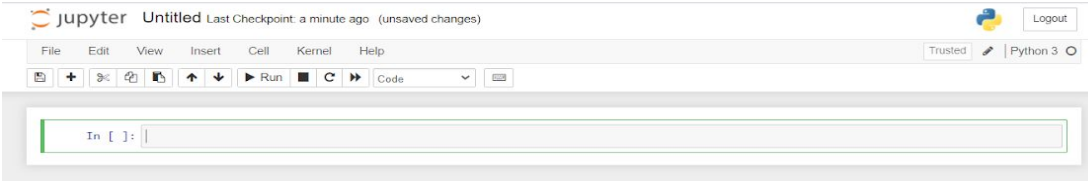


Figure 3.2 Creating Python Notebook

- **Setting Notebook Name:**

By default, the notebook uses the naming convention UntitledXX.ipynb.

To rename the notebook, click on this name and type in the desired name in the edit box as shown here –

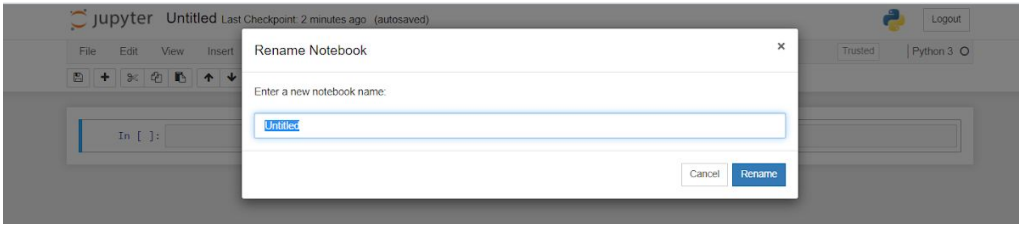


Figure 3.3 Setting Notebook Name

- **Entering Code:**

You will now enter a trivial Python code in the code window and execute it.

Enter the following two Python statements in the code window −

```
import time
print(time.ctime())
```

Figure 3.4 Entering code

- **Executing Code:**

To execute the code, click on the arrow on the left side of the code window.

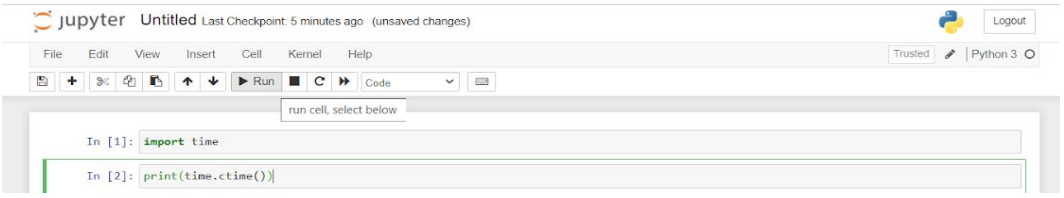


Figure 3.5 Executing code

After a while, you will see the output underneath the code window, as shown here —

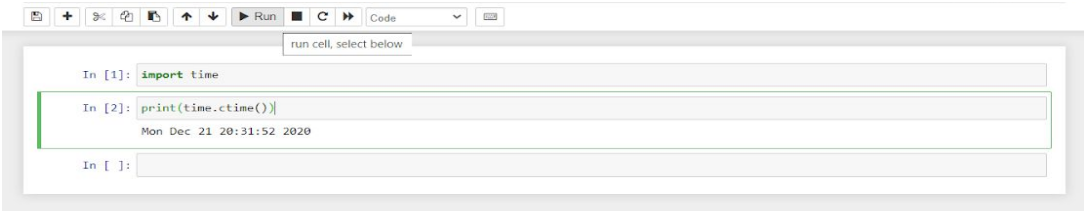


Figure 3.6 Result of code executed

You can clear the output anytime by clicking the icon on the left side of the display i.e. cut option.

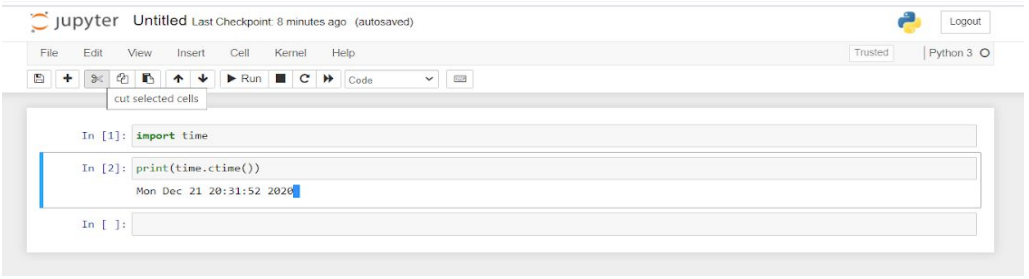


Figure 3.7 Clearing output by cut option

●Adding Code Cells:

To add more code to your notebook, select the following **menu** options –

```
Insert / Code Cell
```

Figure 3.8 Adding Code Cells

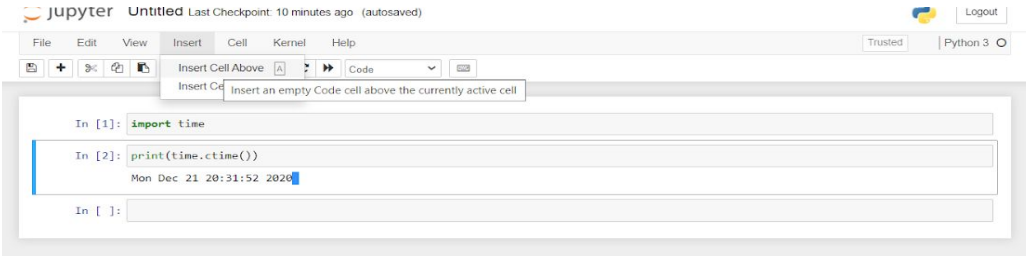


Figure 3.9 Add code cell

A new code cell will be added underneath the current cell.

●Saving Work:

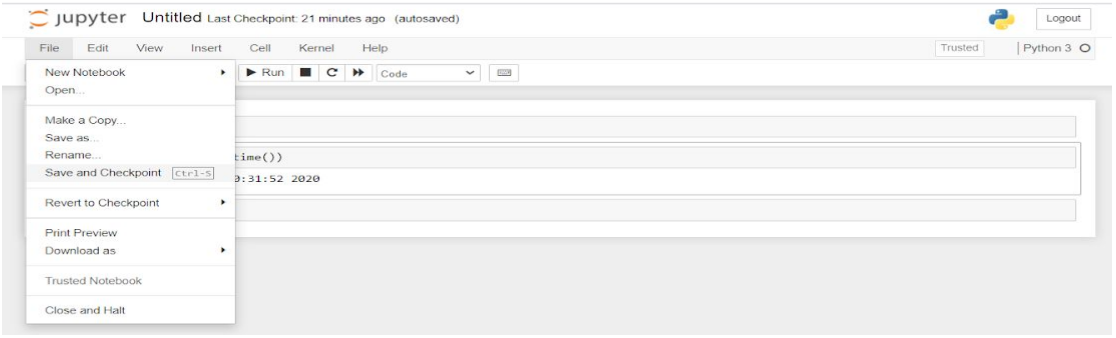


Figure 3.10 Saving work using ctrl+s

3.3. TECHNOLOGY USED

- **PYTHON**

Python is a translator, object-situated, elevated level programming language with dynamic semantics. It raises level certain data structures, gotten together with powerful making and dynamic authority; make it especially charming for Rapid Application Development, similarly concerning use as a scripting or glue language to relate existing parts together. Python's direct, easy to learn language structure underscores clarity and in this manner diminishes the cost of program upkeep. Python maintains modules and groups, which supports program estimated quality and code reuse. The Python interpreter and the wide standard library are available in source or twofold structure without charge for each huge stage, and can be transparently scattered. As often as possible, designers experience energetic affections for Python because of the extended productivity it gives. Since there is no collection step, the change test-investigate cycle is unimaginably snappy. Examining Python programs is basic: a bug or horrendous data will never cause a division issue. In light of everything, when the interpreter finds a slip-up, it raises an exception. Right when the program doesn't get the uncommon case, the go between prints a stack follow. A source level debugger grants survey of neighborhood and overall elements, evaluation of self-confident enunciations, setting breakpoints, wandering through the code a line at a without a moment's delay, on. The debugger is written in Python itself,vouching for Python's insightful power. On the other hand, routinely the speediest strategy to explore a program is to add a few print declarations to the source: the brisk modify test-investigate cycle makes this direct system extraordinarily convincing.

3.3.1 LIBRARIES USED:

- **PANDAS:**

In computer programming, a panda is a software library composed for the Python programming language for information control and examination. Specifically, it offers information structures and tasks for controlling mathematical tables and time arrangement. It is free programming delivered under the three-clause BSD permit.

- **NUMPY:**

It is a general-purpose array-processing package It gives a superior multidimensional cluster item and devices for working with these exhibits. It is the major bundle for logical processing with Python. Other than its undeniable logical uses, Numpy can likewise be utilized as a proficient multi-dimensional compartment of conventional information.

- **MATPLOTLIB:**

Matplotlib is a plotting library for the Python programming language and its mathematical science expansion NumPy. It gives an article situated API to installing plots into applications utilizing broadly useful GUI toolkits.

- **SKLEARN:**

SciKit library provides a tool, called the Model Selection library. There's a class in the library which is, aptly, named train_test_split. Using this we can easily split the dataset into the training and the testing datasets in various proportions.
There are a few parameters that we need to understand before we use the class:

- **train_size** - You have to specify this parameter only if you're not specifying the test_size. This is the same as test_size, but instead you tell the class what percent of the dataset you want to split as the training set.
- **random_state** - Here you pass an integer, which will act as the seed for the random number generator during the split. Or, you can also pass an instance of the RandomState class, which will become the number generator. If you don't pass anything, the RandomState instance used by np.random will be used instead.

Chapter 4: Implementation Work Details

4.1 DATA IMPLEMENTATION AND PROGRAM EXECUTION

The project is divided into the following steps:

1. Collection of data / Downloading share price data.
2. Dataset as CSV files.
3. Data preprocessing.
4. Feature Scaling.
5. LSTM training and evaluation
6. Making the predictions and visualizing the results

4.1.1. DATASET COLLECTION

An offer cost is the cost of a singular segment of different saleable supplies of an association, auxiliary or other money related asset. In layman's terms, the stock expense is the most imperative total someone is anxious to pay for the stock, or the least aggregate that it might be bought for. In this paper, we download the offer information of a specific organization utilizing the yahoo finance information source. The information, for example, date, open offer value, high worth, low worth, last offer worth, close offer value, complete exchange and turn over qualities are utilized.

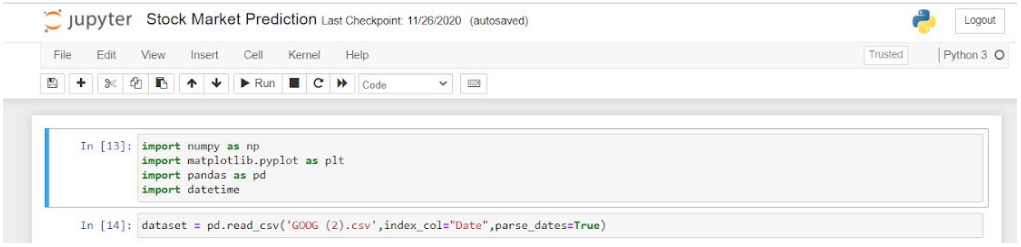


Figure 4.1.1 Reading dataset

```
In [15]: dataset.head()
```

Out[15]:

Date	Open	High	Low	Close	Volume
2015-11-27	748.460022	753.409973	747.489990	750.260010	838500
2015-11-30	748.809998	754.929993	741.270020	742.599976	2097600
2015-01-12	747.109985	768.950012	746.700012	767.039978	2134600
2015-02-12	768.900024	775.955017	758.960022	762.380005	2230400
2015-03-12	766.010010	768.994995	745.630005	752.539978	2590600

Figure 4.1.2 Creating dataset

4.1.2 DATASET AS CSV FILES

A dataset is an assortment of information. Most usually an informational collection relates to the substance of a solitary information base table, or a solitary measurable information network, where each segment of the table speaks to a specific variable, and each line compares to a given individual from the informational index being referred to. The educational assortment records regards for all of the elements, for instance, stature and weight of a thing, for each person from the enlightening assortment. Every value is known as a datum. The enlightening assortment may contain data for at any rate one individual, contrasted with the amount of lines. Here we keep all our data as csv records. In handling, a comma confined characteristics (CSV) record is a delimited substance archive that uses a comma to disconnect esteems. A CSV record stores even data (numbers and text) in plain substance. Each line of the archive is a data record. Each record contains in any event one field, confined by commas. The usage of the comma as a field separator is the wellspring of the name for this record plan. Our dataset is kept in indeed, even plan to rule with characteristics, for instance, date, open, high, low, close volume.

Date	Open	High	Low	Close	Volume
27-11-2015	748.46	753.41	747.49	750.26	838500
30-11-2015	748.81	754.93	741.27	742.6	2097600
01-12-2015	747.11	768.95	746.7	767.04	2134600
02-12-2015	768.9	775.955	758.96	762.38	2230400
03-12-2015	766.01	768.995	745.63	752.54	2590600
04-12-2015	753.1	768.49	750	766.81	2757300
07-12-2015	767.77	768.73	755.09	763.25	1812300
08-12-2015	757.89	764.8	754.2	762.37	1829500
09-12-2015	759.17	764.23	737.001	751.61	2700000
10-12-2015	752.85	755.85	743.83	749.46	1988400
11-12-2015	741.16	745.71	736.75	738.87	2224400
14-12-2015	741.79	748.73	724.17	747.77	2412500
15-12-2015	753	758.08	743.01	743.4	2666200
16-12-2015	750	760.59	739.435	758.09	1993300
17-12-2015	762.42	762.68	749	749.43	1553400
18-12-2015	746.51	754.13	738.15	739.31	3148700
21-12-2015	746.13	750	740	747.77	1525700
22-12-2015	751.65	754.85	745.53	750	1365400
23-12-2015	753.47	754.21	744	750.31	1565900
24-12-2015	749.55	751.35	746.62	748.4	527200
28-12-2015	752.92	762.99	749.52	762.51	1515300
29-12-2015	766.69	779.98	766.43	776.6	1765000
30-12-2015	776.6	777.6	766.9	771	1293300
31-12-2015	769.5	769.5	758.34	758.88	1500900
04-01-2016	743	744.06	731.258	741.84	3272800
05-01-2016	746.45	752	738.64	742.58	1950700
06-01-2016	730	747.18	728.92	743.62	1947000
07-01-2016	730.31	738.5	719.06	726.39	2963700
08-01-2016	731.45	733.23	713	714.47	2450900
11-01-2016	716.61	718.855	703.54	716.03	2090600
12-01-2016	721.68	728.75	717.317	726.07	2024500
13-01-2016	730.85	734.74	698.61	700.56	2501700
14-01-2016	705.38	721.925	689.1	714.72	2225800
15-01-2016	692.29	706.74	685.37	694.45	3608100
19-01-2016	703.3	709.98	693.41	701.79	2268100
20-01-2016	688.61	706.85	673.26	698.45	3445000
21-01-2016	702.18	719.19	694.46	706.59	2412200
22-01-2016	723.6	728.13	720.121	725.25	2011800
25-01-2016	723.58	729.68	710.01	711.67	1711700
26-01-2016	713.85	718.28	706.48	713.04	1331700
27-01-2016	713.67	718.235	694.39	699.99	2194200
28-01-2016	722.22	733.69	712.35	730.96	2676400
29-01-2016	731.53	744.99	726.8	742.95	3474300
01-02-2016	750.46	757.86	743.27	752	5139200
02-02-2016	784.5	789.87	764.65	764.65	6348100
03-02-2016	770.22	774.5	720.5	726.95	6171000

Figure 4.1.3 Google stock market dataset (Training dataset)

	Date	Open	High	Low	Close	Volume
2	02-01-2020	1341.55	1368.14	1341.55	1367.37	1406600
3	03-01-2020	1347.86	1372.5	1345.54	1360.66	1186400
4	06-01-2020	1350	1396.5	1350	1394.21	1732300
5	07-01-2020	1397.94	1402.99	1390.38	1393.34	1502700
6	08-01-2020	1392.08	1411.58	1390.84	1404.32	1528000
7	09-01-2020	1420.57	1427.33	1410.27	1419.83	1500900
8	10-01-2020	1427.56	1434.93	1418.35	1429.73	1820700
9	13-01-2020	1436.13	1440.52	1426.02	1439.23	1652300
10	14-01-2020	1439.01	1441.8	1428.37	1430.88	1558900
11	15-01-2020	1430.21	1441.4	1430.21	1439.2	1282700
12	16-01-2020	1447.44	1451.99	1440.92	1451.7	1173700
13	17-01-2020	1462.91	1481.3	1458.22	1480.39	2396200
14	21-01-2020	1479.12	1491.85	1471.2	1484.4	2036700
15	22-01-2020	1491	1503.21	1484.93	1485.95	1610800
16	23-01-2020	1487.64	1495.52	1482.1	1486.65	1351200
17	24-01-2020	1493.59	1495.49	1465.25	1466.71	1784600
18	27-01-2020	1431	1438.07	1421.2	1433.9	1755200
19	28-01-2020	1443	1456	1432.47	1452.56	1577400
20	29-01-2020	1458.8	1465.43	1446.74	1458.63	1077700
21	30-01-2020	1439.96	1457.28	1436.4	1455.84	1339400
22	31-01-2020	1468.9	1470.13	1428.53	1434.23	2417200
23	03-02-2020	1462	1490	1458.99	1485.94	3055200
24	04-02-2020	1457.07	1469.5	1426.3	1447.07	3933000
25	05-02-2020	1462.42	1463.84	1430.56	1448.23	1986200
26	06-02-2020	1450.33	1482	1449.57	1476.23	1679400
27	07-02-2020	1467.3	1485.84	1466.35	1479.23	1172300
28	10-02-2020	1474.32	1509.5	1474.32	1508.68	1419900
29	11-02-2020	1511.81	1529.63	1505.64	1508.79	1344600
30	12-02-2020	1514.48	1520.69	1508.11	1518.27	1167600
31	13-02-2020	1512.69	1527.18	1504.6	1514.66	929500
32	14-02-2020	1515.6	1520.74	1507.34	1520.74	1197800

Figure 4.1.4 Google stock market dataset (Testing dataset)

4.1.3. DATA PREPROCESSING

Data preprocessing is a significant advance in AI ventures. Information gathering strategies are frequently approximately controlled, coming about in out-of-range esteems missing qualities, and so forth Breaking down information that has not been deliberately screened for such issues can deliver deceiving results. Along these lines, the portrayal and nature of information is most importantly prior to running an examination. Regularly, information preprocessing is the main period of an AI venture, particularly in computational information. On the off chance that there is a lot of immaterial and repetitive data present or boisterous and questionable information, at that point information revelation during the preparation stage is more troublesome. Information readiness and separating steps can take a lot of preparing time. Information preprocessing incorporates cleaning, occurrence determination, standardization, change, highlight extraction and choice and so forth The result of information preprocessing is the last preparing set.

```
In [16]: dataset.isna().any()
         #function detect missing values in the given series object.
         #It return a boolean same-sized object indicating if the values are NA.
         #Missing values gets mapped to True and non-missing value gets mapped to False.

Out[16]: Open      False
         High      False
         Low       False
         Close     False
         Volume    False
         dtype: bool

In [17]: dataset.info()

         <class 'pandas.core.frame.DataFrame'>
         DatetimeIndex: 1030 entries, 2015-11-27 to 2019-12-31
         Data columns (total 5 columns):
          #   Column  Non-Null Count  Dtype
         ---  ------  --------------  -----
          0   Open    1030 non-null   float64
          1   High    1030 non-null   float64
          2   Low     1030 non-null   float64
          3   Close   1030 non-null   float64
          4   Volume  1030 non-null   int64
         dtypes: float64(4), int64(1)
         memory usage: 48.3 KB
```

Figure 4.1.5 Cleaning dataset

4.1.4. FEATURE SCALING

Highlight scaling is a system used to standardize the extent of free factors or features of data. In data preprocessing, it is generally called data normalization and is overall performed during the data preprocessing step. Since the extent of assessments of rough data varies comprehensively, target limits won't work suitably without normalization. Along these lines, the extension, things being what they are, should be normalized so every segment contributes around proportionately to the last distance. Another inspiration driving why feature scaling is applied is that slant plunge blends much speedier with incorporate scaling than without it. In any case called min-max scaling or min-max normalization, is the most effortless procedure and involves rescaling the extent of features to scale the scope in [0, 1] or [−1, 1]. Choosing the objective reach relies upon the idea of the information.

```
In [29]: # Feature Scaling
         from sklearn.preprocessing import MinMaxScaler
         sc = MinMaxScaler(feature_range = (0, 1))
         training_set_scaled = sc.fit_transform(training_set)
```

Figure 4.1.6 Feature scaling

4.1.5. LSTM TRAINING AND EVALUATION

LSTM are very stunning in gathering estimate issues since they're prepared to store past information. This is huge for our circumstance because the previous expense of a stock is

basic in anticipating its future expense. We will at first start by acquiring Numpy for consistent count, Matplotlib for plotting graphs, and Pandas to help in stacking and controlling our datasets. The resulting stage is to stack in our arrangement dataset and select the Open and High sections that we'll use in our showing. We check the highest point of our dataset to give us a concise investigate such a dataset we're working with. The open fragment is the starting expense while the Close segment is the last expense of a stock on a particular trading day. The High and Low sections address the most significant and smallest expenses for a particular day. From past association in significant learning models, we understand that we need to scale our data for ideal execution. For our circumstance, we'll use Scikit Learn's MinMaxScaler and scale our dataset to numbers some place in the scope of zero and one. LSTMs foresee that our data ought to be in a specific plan, regularly a 3D display. We start by making data in 60 timesteps and changing over it into a show using NumPy. Next, we convert the data into a 3D estimation bunch with X_train tests, 60 timestamps, and one component at every movement. To manufacture the LSTM, we need to import a couple of modules from Keras :

1.Sequential for instating the neural association

2.Dense for adding a thickly related neural association layer

3.LSTM for adding the Long Short-Term Memory layer

4.Dropout for adding dropout layers that hinder overfitting

We add the LSTM layer and later add a few Dropout layers to prevent overfitting. We add the LSTM layer with the going with disputes:

1.50 units which is the dimensionality of the yield space.

2.return_sequences=True which chooses if to reestablish the last yield in the yield gathering or the full progression.

3.input_shape as the condition of our arrangement set. While describing the Dropout layers, we decide 0.2, suggesting that 20% of the layers will be dropped. Starting there, we add the Dense layer that decides the yield of 1 unit. After this, we assemble our model using the standard adam analyzer and set the disaster as the mean_squarred_error. This will calculate the mean of the squared bumbles. Next, we fit the model to run on 100 ages with a group size of 32.

```
In [31]: # Part 2 - Building the RNN

         # Importing the Keras libraries and packages
         from keras.models import Sequential
         from keras.layers import Dense
         from keras.layers import LSTM
         from keras.layers import Dropout

In [32]: # Initialising the RNN
         regressor = Sequential()

In [33]: # Adding the first LSTM layer and some Dropout regularisation
         regressor.add(LSTM(units = 50, return_sequences = True, input_shape = (X_train.shape[1], 1)))
         regressor.add(Dropout(0.2))

         # Adding a second LSTM layer and some Dropout regularisation
         regressor.add(LSTM(units = 50, return_sequences = True))
         regressor.add(Dropout(0.2))

         # Adding a third LSTM layer and some Dropout regularisation
         regressor.add(LSTM(units = 50, return_sequences = True))
         regressor.add(Dropout(0.2))

         # Adding a fourth LSTM layer and some Dropout regularisation
         regressor.add(LSTM(units = 50))
         regressor.add(Dropout(0.2))

         # Adding the output layer
         regressor.add(Dense(units = 1))

In [34]: # Compiling the RNN
         regressor.compile(optimizer = 'adam', loss = 'mean_squared_error')

         # Fitting the RNN to the Training set
         regressor.fit(X_train, y_train, epochs = 100, batch_size = 32)

         Epoch 1/100
         31/31 [==============================] - 5s 156ms/step - loss: 0.0572
         Epoch 2/100
         31/31 [==============================] - 3s 110ms/step - loss: 0.0082
```

Fig 4.1.7 LSTM Training and Evaluation

4.1.6. PREDICTION AND VISUALIZATION

To anticipate future stock costs we need to do a few things ensuing to stacking in the test set:

1.Merge the readiness set and the test set on the 0 centers.

2.Set the time adventure as 60 (as noticed in advance)

3.Use MinMaxScaler to change the new dataset

4.Reshape the dataset as done already

Ensuing to making the desires we use inverse_transform to get back the stock expenses in a customary clear association. Finally, we use Matplotlib to imagine the delayed consequence of the foreseen stock expense and the certified stock expense. From the plot, we can see that the veritable stock cost went up while our model is moreover foreseen that the expense of the stock would go up. This clearly shows how historic LSTMs are for inspecting time game

plan and progressive data.

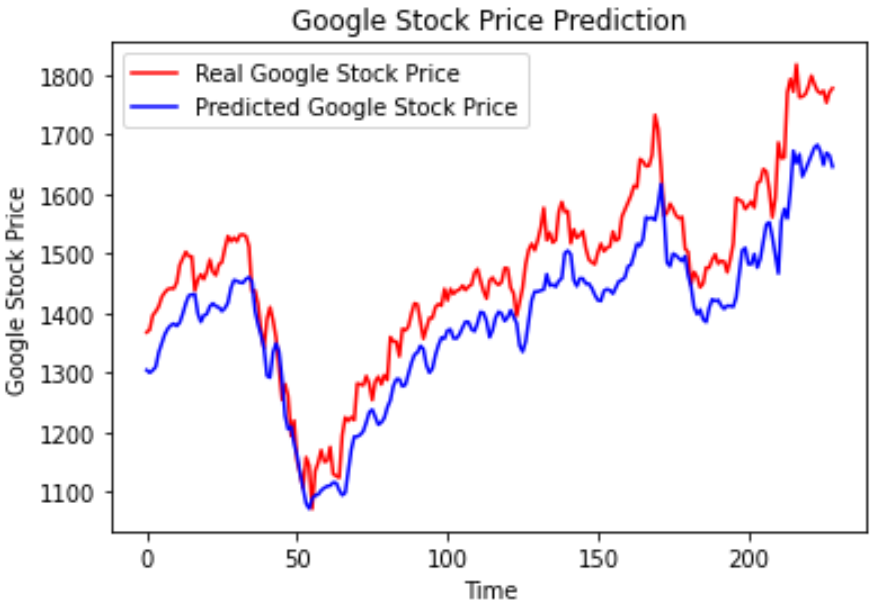


Figure 4.1.8 Prediction

Chapter 5: Source Code/ Simulation Code

```
import numpy as np
import matplotlib.pyplot as plt
import pandas as pd
import datetime

dataset = pd.read_csv('GOOG (2).csv',index_col="Date",parse_dates=True)
dataset.head()

dataset.isna().any()
#function detect missing values in the given series object.
#It return a boolean same-sized object indicating if the values are NA.
#Missing values gets mapped to True and non-missing value gets mapped to False.

dataset.info()
dataset['Open'].plot(figsize=(16,6))

# 7 day rolling mean
dataset.rolling(7).mean().head(20)

dataset['Open'].plot(figsize=(16,6))
dataset.rolling(window=30).mean()['Close'].plot()

dataset['Close: 30 Day Mean'] = dataset['Close'].rolling(window=30).mean()
dataset[['Close','Close: 30 Day Mean']].plot(figsize=(16,6))

# Optional specify a minimum number of periods
dataset['Close'].expanding(min_periods=1).mean().plot(figsize=(16,6))

training_set=dataset['Open']
training_set=pd.DataFrame(training_set)

# Feature Scaling
from sklearn.preprocessing import MinMaxScaler
sc = MinMaxScaler(feature_range = (0, 1))
training_set_scaled = sc.fit_transform(training_set)

# Creating a data structure with 60 timesteps and 1 output
X_train = []
y_train = []
for i in range(60, 1030):
    X_train.append(training_set_scaled[i-60:i, 0])
    y_train.append(training_set_scaled[i, 0])
X_train, y_train = np.array(X_train), np.array(y_train)

# Reshaping
X_train = np.reshape(X_train, (X_train.shape[0], X_train.shape[1], 1))
# Part 2 - Building the RNN
```

```
# Importing the Keras libraries and packages
from keras.models import Sequential
from keras.layers import Dense
from keras.layers import LSTM
from keras.layers import Dropout

# Initialising the RNN
regressor = Sequential()

# Adding the first LSTM layer and some Dropout regularisation
  regressor.add(LSTM(units = 50, return_sequences = True,
  input_shape =   (X_train.shape[1], 1)))
regressor.add(Dropout(0.2))

# Adding a second LSTM layer and some Dropout regularisation
regressor.add(LSTM(units = 50, return_sequences = True))
regressor.add(Dropout(0.2))

# Adding a third LSTM layer and some Dropout regularisation
regressor.add(LSTM(units = 50, return_sequences = True))
regressor.add(Dropout(0.2))

# Adding a fourth LSTM layer and some Dropout regularisation
regressor.add(LSTM(units = 50))
regressor.add(Dropout(0.2))

# Adding the output layer
regressor.add(Dense(units = 1))

# Compiling the RNN
regressor.compile(optimizer = 'adam', loss = 'mean_squared_error')

# Fitting the RNN to the Training set
regressor.fit(X_train, y_train, epochs = 100, batch_size = 32)


# Part 3 - Making the predictions and visualising the results

# Getting the real stock price of 2020
dataset_test = pd.read_csv('GOOG (2) test.csv',index_col="Date",parse_dates=True)

real_stock_price = dataset_test.iloc[:, 1:2].values
dataset_test.head()
test_set=dataset_test['Open']
test_set=pd.DataFrame(test_set)
test_set.info()

# Getting the predicted stock price of 2020
dataset_total = pd.concat((dataset['Open'], dataset_test['Open']), axis = 0)
inputs = dataset_total[len(dataset_total) - len(dataset_test) - 60:].values
```

```
inputs = inputs.reshape(-1,1)
inputs = sc.transform(inputs)
X_test = []
for i in range(60, 289):
    X_test.append(inputs[i-60:i, 0])
X_test = np.array(X_test)
X_test = np.reshape(X_test, (X_test.shape[0], X_test.shape[1], 1))
predicted_stock_price = regressor.predict(X_test)
predicted_stock_price = sc.inverse_transform(predicted_stock_price)

predicted_stock_price=pd.DataFrame(predicted_stock_price)
predicted_stock_price.info()

# Visualising the results
plt.plot(real_stock_price, color = 'red', label = 'Real Google Stock Price')
plt.plot(predicted_stock_price, color = 'blue', label = 'Predicted Google Stock Price')
plt.title('Google Stock Price Prediction')
plt.xlabel('Time')
plt.ylabel('Google Stock Price')
plt.legend()
plt.show()
```

Chapter 6: Input/output Screens/ Model's Photograph

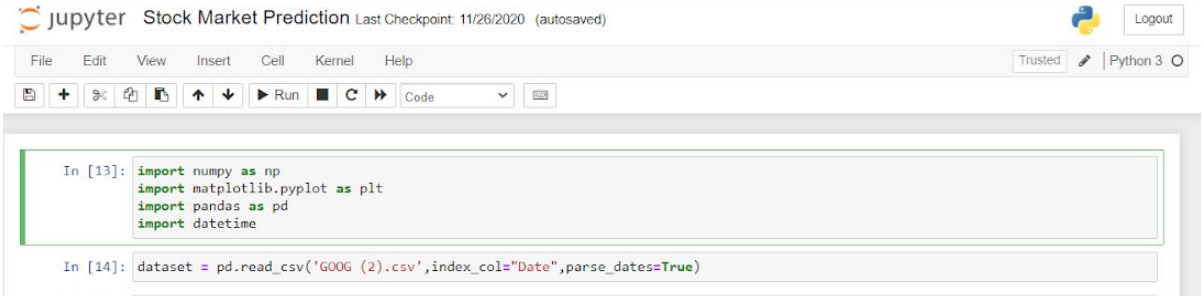


Figure 6.1 Reading final dataset

In [15]:
```
dataset.head()
```

Out[15]:

Date	Open	High	Low	Close	Volume
2015-11-27	748.460022	753.409973	747.489990	750.260010	838500
2015-11-30	748.809998	754.929993	741.270020	742.599976	2097600
2015-01-12	747.109985	768.950012	746.700012	767.039978	2134600
2015-02-12	768.900024	775.955017	758.960022	762.380005	2230400
2015-03-12	766.010010	768.994995	745.630005	752.539978	2590600

Figure 6.2 Showing final dataset

In [16]:
```
dataset.isna().any()
#function detect missing values in the given series object.
#It return a boolean same-sized object indicating if the values are NA.
#Missing values gets mapped to True and non-missing value gets mapped to False.
```

Out[16]:
```
Open      False
High      False
Low       False
Close     False
Volume    False
dtype: bool
```

In [17]:
```
dataset.info()
```

```
<class 'pandas.core.frame.DataFrame'>
DatetimeIndex: 1030 entries, 2015-11-27 to 2019-12-31
Data columns (total 5 columns):
 #   Column  Non-Null Count  Dtype
---  ------  --------------  -----
 0   Open    1030 non-null   float64
 1   High    1030 non-null   float64
 2   Low     1030 non-null   float64
 3   Close   1030 non-null   float64
 4   Volume  1030 non-null   int64
dtypes: float64(4), int64(1)
memory usage: 48.3 KB
```

Figure 6.3 Deleting unwanted values

```
In [18]: dataset['Open'].plot(figsize=(16,6))
Out[18]: <AxesSubplot:xlabel='Date'>
```

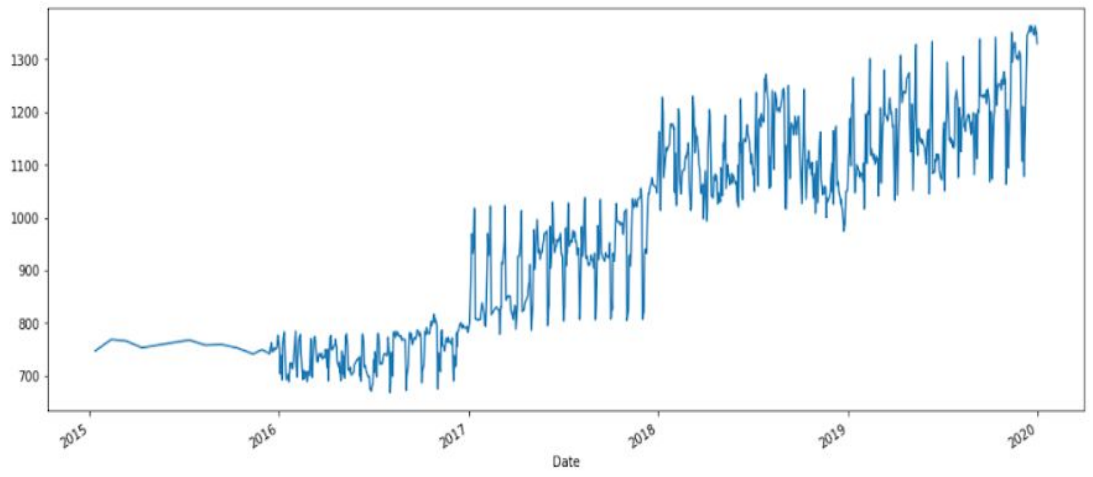


Figure 6.4 Plotting graph of open

```
In [19]: # 7 day rolling mean
         dataset.rolling(7).mean().head(20)
```

Out[19]:

Date	Open	High	Low	Close	Volume
2015-11-27	NaN	NaN	NaN	NaN	NaN
2015-11-30	NaN	NaN	NaN	NaN	NaN
2015-01-12	NaN	NaN	NaN	NaN	NaN
2015-02-12	NaN	NaN	NaN	NaN	NaN
2015-03-12	NaN	NaN	NaN	NaN	NaN
2015-04-12	NaN	NaN	NaN	NaN	NaN
2015-07-12	757.165719	765.637137	749.305725	757.839992	2.065900e+06
2015-08-12	758.512861	767.264282	750.264300	759.569990	2.207471e+06
2015-09-12	759.992859	768.592852	749.654436	760.857134	2.293529e+06
2015-10-12	760.812858	766.721418	749.244437	758.345712	2.272643e+06
2015-11-12	756.849993	762.400704	746.071577	754.987139	2.271786e+06
2015-12-14	753.389989	759.505702	743.005859	754.305716	2.246343e+06
2015-12-15	753.375706	758.018563	742.007289	750.961434	2.233329e+06
2015-12-16	750.837132	756.855713	739.770857	750.224295	2.259186e+06
2015-12-17	751.484270	756.552856	739.027998	748.375724	2.219743e+06
2015-12-18	749.675703	755.110003	739.192147	746.618583	2.283843e+06
2015-12-21	748.715707	754.274292	738.645002	746.377154	2.217743e+06
2015-12-22	750.214286	755.580000	739.899292	747.967155	2.095029e+06
2015-12-23	751.882856	756.362863	742.732152	748.330009	1.974086e+06
2015-12-24	751.389997	755.401428	743.247864	749.044294	1.668514e+06

Figure 6.5 Finding 7 day rolling mean

```
In [20]: dataset['Open'].plot(figsize=(16,6))
         dataset.rolling(window=30).mean()['Close'].plot()

Out[20]: <AxesSubplot:xlabel='Date'>
```

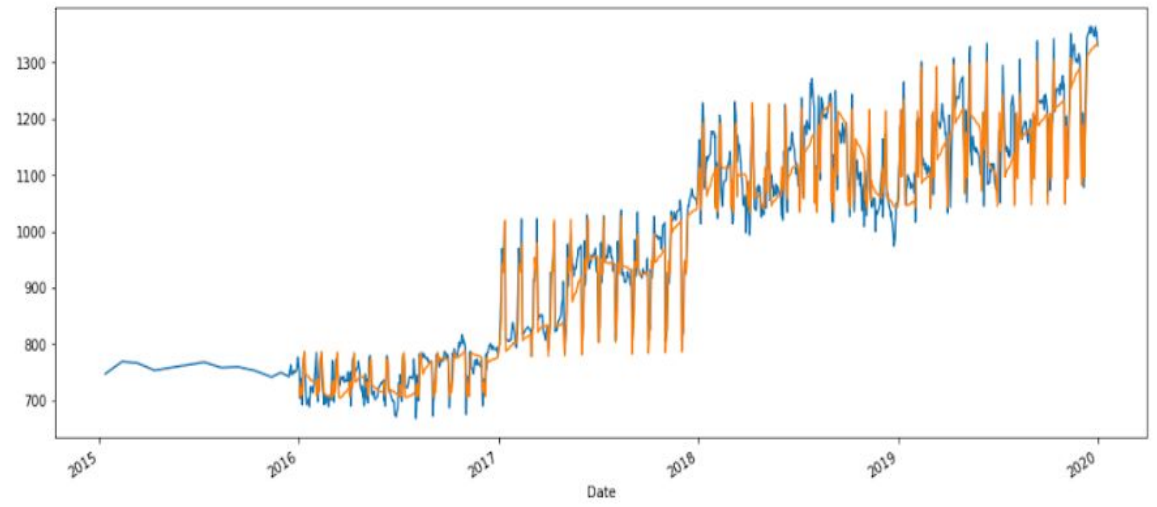


Figure 6.6 Comparing the plots of open and rolling mean close value.

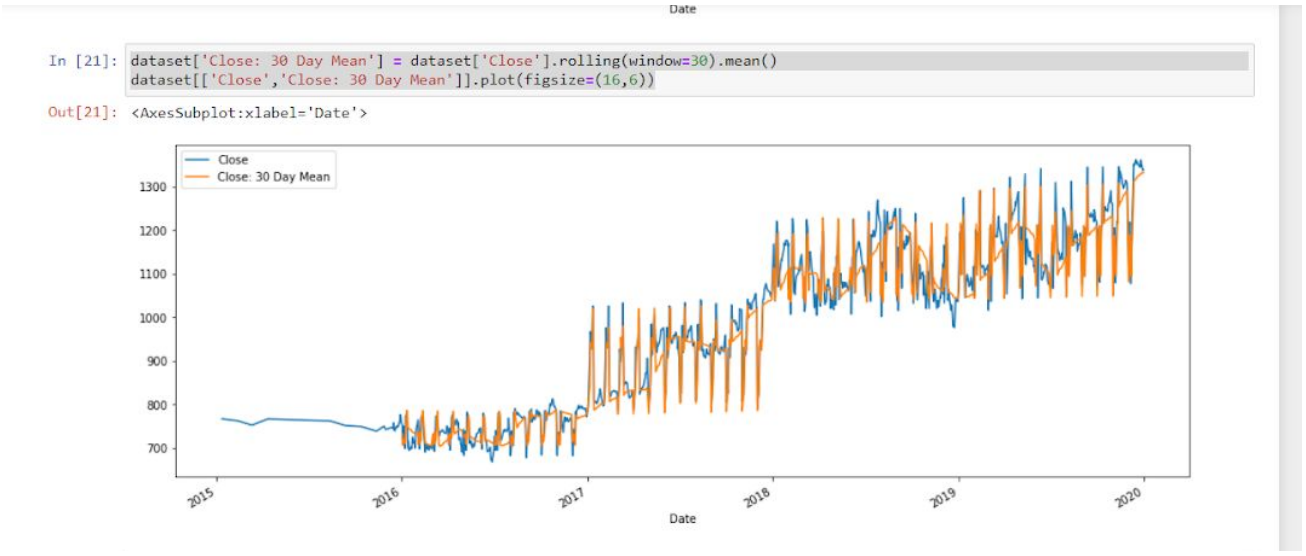


Figure 6.7 comparing the plot of close: 30 day mean and close.

```
In [22]: # Optional specify a minimum number of periods
         dataset['Close'].expanding(min_periods=1).mean().plot(figsize=(16,6))

Out[22]: <AxesSubplot:xlabel='Date'>
```

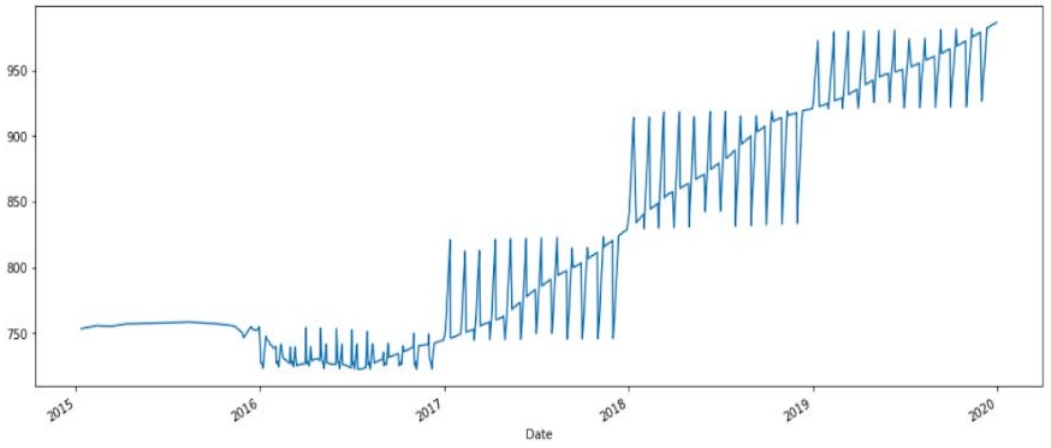


Figure 6.8 minimum number of periods

```
In [29]: # Feature Scaling
         from sklearn.preprocessing import MinMaxScaler
         sc = MinMaxScaler(feature_range = (0, 1))
         training_set_scaled = sc.fit_transform(training_set)

In [30]: # Creating a data structure with 60 timesteps and 1 output
         X_train = []
         y_train = []
         for i in range(60, 1030):
             X_train.append(training_set_scaled[i-60:i, 0])
             y_train.append(training_set_scaled[i, 0])
         X_train, y_train = np.array(X_train), np.array(y_train)

         # Reshaping
         X_train = np.reshape(X_train, (X_train.shape[0], X_train.shape[1], 1))
```

Figure 6.9 Feature scaling and creating a data structure with 60 timestamps.

```
In [31]: # Part 2 - Building the RNN

         # Importing the Keras libraries and packages
         from keras.models import Sequential
         from keras.layers import Dense
         from keras.layers import LSTM
         from keras.layers import Dropout

In [32]: # Initialising the RNN
         regressor = Sequential()

In [33]: # Adding the first LSTM layer and some Dropout regularisation
         regressor.add(LSTM(units = 50, return_sequences = True, input_shape = (X_train.shape[1], 1)))
         regressor.add(Dropout(0.2))

         # Adding a second LSTM layer and some Dropout regularisation
         regressor.add(LSTM(units = 50, return_sequences = True))
         regressor.add(Dropout(0.2))

         # Adding a third LSTM layer and some Dropout regularisation
         regressor.add(LSTM(units = 50, return_sequences = True))
         regressor.add(Dropout(0.2))

         # Adding a fourth LSTM layer and some Dropout regularisation
         regressor.add(LSTM(units = 50))
         regressor.add(Dropout(0.2))

         # Adding the output layer
         regressor.add(Dense(units = 1))
```

Figure 6.10 Building the RNN and initializing the RNN

```
In [34]: # Compiling the RNN
regressor.compile(optimizer = 'adam', loss = 'mean_squared_error')

# Fitting the RNN to the Training set
regressor.fit(X_train, y_train, epochs = 100, batch_size = 32)
```

```
Epoch 1/100
31/31 [==============================] - 5s 156ms/step - loss: 0.0572
Epoch 2/100
31/31 [==============================] - 3s 110ms/step - loss: 0.0082
Epoch 3/100
31/31 [==============================] - 3s 84ms/step - loss: 0.0074
Epoch 4/100
31/31 [==============================] - 3s 84ms/step - loss: 0.0062
Epoch 5/100
31/31 [==============================] - 4s 118ms/step - loss: 0.0060
Epoch 6/100
31/31 [==============================] - 4s 140ms/step - loss: 0.0071
Epoch 7/100
31/31 [==============================] - 5s 154ms/step - loss: 0.0067
Epoch 8/100
31/31 [==============================] - 5s 163ms/step - loss: 0.0057
Epoch 9/100
31/31 [==============================] - 5s 145ms/step - loss: 0.0057
Epoch 10/100
31/31 [==============================] - 5s 152ms/step - loss: 0.0051
Epoch 11/100
31/31 [==============================] - 5s 158ms/step - loss: 0.0055
Epoch 12/100
31/31 [==============================] - 4s 131ms/step - loss: 0.0065
Epoch 13/100
31/31 [==============================] - 5s 154ms/step - loss: 0.0067
Epoch 14/100
31/31 [==============================] - 4s 125ms/step - loss: 0.0049
Epoch 15/100
31/31 [==============================] - 4s 139ms/step - loss: 0.0049
Epoch 16/100
31/31 [==============================] - 5s 154ms/step - loss: 0.0046
Epoch 17/100
31/31 [==============================] - 5s 169ms/step - loss: 0.0049
Epoch 18/100
31/31 [==============================] - 5s 177ms/step - loss: 0.0048
Epoch 19/100
```

Figure 6.11 Compiling and fitting the RNN to the training set

```
In [35]: # Part 3 - Making the predictions and visualising the results

# Getting the real stock price of 2020
dataset_test = pd.read_csv('GOOG (2) test.csv',index_col="Date",parse_dates=True)
```

```
In [36]: real_stock_price = dataset_test.iloc[:, 1:2].values
dataset_test.head()
test_set=dataset_test['Open']
test_set=pd.DataFrame(test_set)
test_set.info()
```

```
<class 'pandas.core.frame.DataFrame'>
DatetimeIndex: 229 entries, 2020-02-01 to 2020-11-25
Data columns (total 1 columns):
 #   Column  Non-Null Count  Dtype
---  ------  --------------  -----
 0   Open    229 non-null    float64
dtypes: float64(1)
memory usage: 3.6 KB
```

Figure 6.12 Making the predictions and visualizing the results

In [37]:
```
# Getting the predicted stock price of 2020
dataset_total = pd.concat((dataset['Open'], dataset_test['Open']), axis = 0)
inputs = dataset_total[len(dataset_total) - len(dataset_test) - 60:].values
inputs = inputs.reshape(-1,1)
inputs = sc.transform(inputs)
X_test = []
for i in range(60, 289):
    X_test.append(inputs[i-60:i, 0])
X_test = np.array(X_test)
X_test = np.reshape(X_test, (X_test.shape[0], X_test.shape[1], 1))
predicted_stock_price = regressor.predict(X_test)
predicted_stock_price = sc.inverse_transform(predicted_stock_price)
```

In [38]:
```
predicted_stock_price=pd.DataFrame(predicted_stock_price)
predicted_stock_price.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 229 entries, 0 to 228
Data columns (total 1 columns):
 #   Column  Non-Null Count  Dtype
---  ------  --------------  -----
 0   0       229 non-null    float32
dtypes: float32(1)
memory usage: 1.0 KB
```

Figure 6.13 Getting the predicted stock price of 2020

In [39]:
```
# Visualising the results
plt.plot(real_stock_price, color = 'red', label = 'Real Google Stock Price')
plt.plot(predicted_stock_price, color = 'blue', label = 'Predicted Google Stock Price')
plt.title('Google Stock Price Prediction')
plt.xlabel('Time')
plt.ylabel('Google Stock Price')
plt.legend()
plt.show()
```

Figure 6.14 Visualizing the results

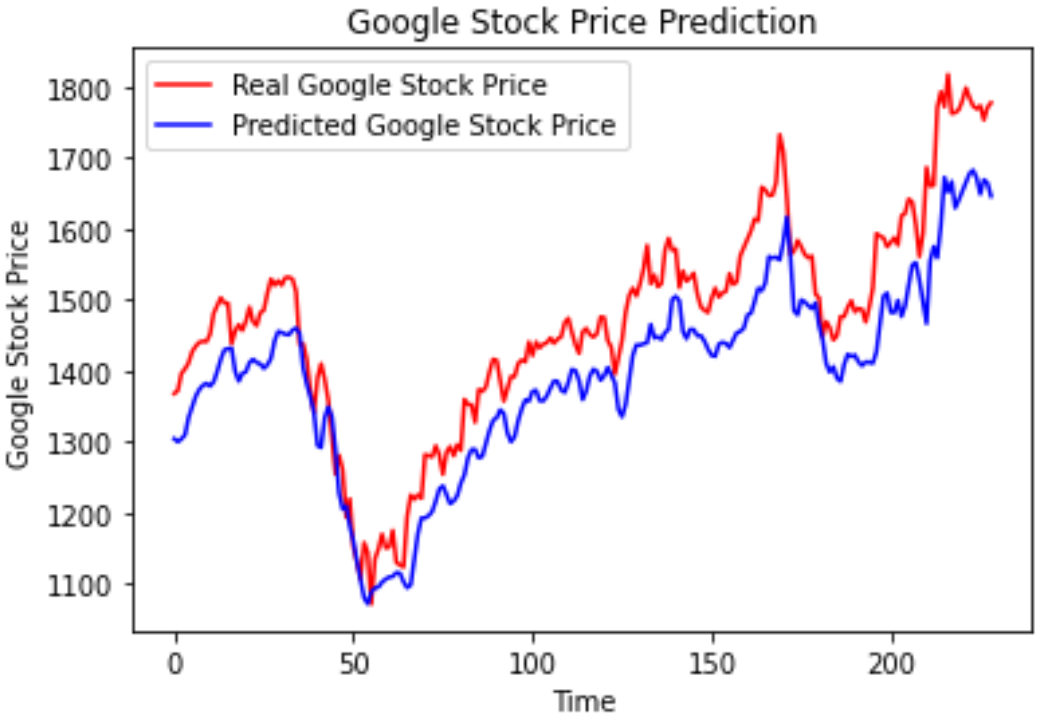


Figure 6.15 Comparing the actual and predicted result

Chapter 7: System Testing

```
In [31]: # Part 2 - Building the RNN

         # Importing the Keras libraries and packages
         from keras.models import Sequential
         from keras.layers import Dense
         from keras.layers import LSTM
         from keras.layers import Dropout

In [32]: # Initialising the RNN
         regressor = Sequential()

In [33]: # Adding the first LSTM layer and some Dropout regularisation
         regressor.add(LSTM(units = 50, return_sequences = True, input_shape = (X_train.shape[1], 1)))
         regressor.add(Dropout(0.2))

         # Adding a second LSTM layer and some Dropout regularisation
         regressor.add(LSTM(units = 50, return_sequences = True))
         regressor.add(Dropout(0.2))

         # Adding a third LSTM layer and some Dropout regularisation
         regressor.add(LSTM(units = 50, return_sequences = True))
         regressor.add(Dropout(0.2))

         # Adding a fourth LSTM layer and some Dropout regularisation
         regressor.add(LSTM(units = 50))
         regressor.add(Dropout(0.2))

         # Adding the output layer
         regressor.add(Dense(units = 1))

In [34]: # Compiling the RNN
         regressor.compile(optimizer = 'adam', loss = 'mean_squared_error')

         # Fitting the RNN to the Training set
         regressor.fit(X_train, y_train, epochs = 100, batch_size = 32)

         Epoch 1/100
         31/31 [==============================] - 5s 156ms/step - loss: 0.0572
         Epoch 2/100
         31/31 [==============================] - 3s 110ms/step - loss: 0.0082
         Epoch 3/100
```

Figure 7.1 Building RNN for testing further

```
In [35]: # Part 3 - Making the predictions and visualising the results

         # Getting the real stock price of 2020
         dataset_test = pd.read_csv('GOOG (2) test.csv',index_col="Date",parse_dates=True)
```

Figure 7.2 Test data

```
In [36]: real_stock_price = dataset_test.iloc[:, 1:2].values
         dataset_test.head()
         test_set=dataset_test['Open']
         test_set=pd.DataFrame(test_set)
         test_set.info()

         <class 'pandas.core.frame.DataFrame'>
         DatetimeIndex: 229 entries, 2020-02-01 to 2020-11-25
         Data columns (total 1 columns):
          #   Column  Non-Null Count  Dtype
         ---  ------  --------------  -----
          0   Open    229 non-null    float64
         dtypes: float64(1)
         memory usage: 3.6 KB
```

Figure 7.3 Comparing the actual and predicted result

```
In [37]: # Getting the predicted stock price of 2020
dataset_total = pd.concat((dataset['Open'], dataset_test['Open']), axis = 0)
inputs = dataset_total[len(dataset_total) - len(dataset_test) - 60:].values
inputs = inputs.reshape(-1,1)
inputs = sc.transform(inputs)
X_test = []
for i in range(60, 289):
    X_test.append(inputs[i-60:i, 0])
X_test = np.array(X_test)
X_test = np.reshape(X_test, (X_test.shape[0], X_test.shape[1], 1))
predicted_stock_price = regressor.predict(X_test)
predicted_stock_price = sc.inverse_transform(predicted_stock_price)
```

```
In [38]: predicted_stock_price=pd.DataFrame(predicted_stock_price)
predicted_stock_price.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 229 entries, 0 to 228
Data columns (total 1 columns):
 #   Column  Non-Null Count  Dtype
---  ------  --------------  -----
 0   0       229 non-null    float32
dtypes: float32(1)
memory usage: 1.0 KB
```

Figure 7.4 Comparing the actual and predicted result (1)

In [39]:
```
# Visualising the results
plt.plot(real_stock_price, color = 'red', label = 'Real Google Stock Price')
plt.plot(predicted_stock_price, color = 'blue', label = 'Predicted Google Stock Price')
plt.title('Google Stock Price Prediction')
plt.xlabel('Time')
plt.ylabel('Google Stock Price')
plt.legend()
plt.show()
```

Figure 7.5 Comparing the actual and predicted result (2)

Chapter 8: Individual Contribution

For completion of project I have performed the following steps:

1. Collection of data / Downloading share price data.
2. Dataset as CSV files.
3. Data preprocessing.
4. Feature Scaling.
5. LSTM training and evaluation
6. Making the predictions and visualising the results

8.1. DATASET COLLECTION

An offer cost is the cost of a single part of different saleable heaps of an association, subordinate, or other financial assets. In layman's terms, the stock expense is the most raised total someone is glad to pay for the stock or the least aggregate that it will, in general, be bought for. We download the offer information of a specific organization utilizing the Yahoo finance source. The information, for example, date, open offer value, high worth, low worth, last offer worth, close offer value, absolute exchange and turn over qualities are utilized.

8.2 DATASET AS CSV FILES

A dataset is a grouping of data. Most consistently an educational list identifies with the substance of a lone database table, or a single real data network, where each fragment of the table addresses a particular variable, and each line analyzes to guarantee individuality from the informational index being referred to. The informational index records esteems for every one of the factors, for example, the tallness furthermore, weight of an article, for each person from the instructive assortment. Every value is known as a datum. The educational assortment may include data for in any event one individual, contrasted with the amount of segments. Here we keep all our data as CSV archives. In enlisting, a comma-separated characteristics (CSV) record is a delimited substance archive that uses a comma to disengage values. CSV records store even data (numbers and text) in plain substance. Each line of the archive is a data record. Each record contains at any rate one

field, disconnected by commas. The use of the comma as a field separator is the wellspring of the name for this record plan. Our dataset is kept in even arrangement to dominate with qualities, for example, date, open, high, low, close, volume.

8.3 DATA PREPROCESSING

Data preprocessing is a significant advance in AI ventures. Information gathering strategies are frequently approximately controlled, coming about in out-of-range esteems missing qualities, and so forth Breaking down information that has not been deliberately screened for such issues can deliver deceiving results. Along these lines, the portrayal and nature of information is most importantly prior to running an examination. Regularly, information preprocessing is the main period of an AI venture, particularly in computational information. On the off chance that there is a lot of immaterial and repetitive data present or boisterous and questionable information, at that point information revelation during the preparation stage is more troublesome. Information readiness and separating steps can take a lot of preparing time. Information preprocessing incorporates cleaning, occurrence determination, standardization, change, highlight extraction and choice and so forth The result of information preprocessing is the last preparing set.

8.4 FEATURE SCALING

Feature / Highlight scaling is a system used to standardize the extent of free factors or features of data. In data preprocessing, it is generally called data normalization and is overall performed during the data preprocessing step. Since the extent of assessments of rough data varies comprehensively, target limits won't work suitably without normalization. Along these lines, the extension, things being what they are, should be normalized so every segment contributes around proportionately to the last distance. Another inspiration driving why feature scaling is applied is that slant plunge blends much speedier with incorporate scaling than without it. In any case called min-max scaling or min-max normalization, is the most effortless procedure and involves rescaling the extent of features to scale the scope in [0, 1] or [−1, 1]. Choosing the objective reach relies upon the idea of the information.

8.5 LSTM TRAINING AND EVALUATION

Long momentary memory (LSTM) is a fake repetitive neural organization (RNN)

designing used in the field of significant learning. Not at all like standard analysis neural associations, LSTM has input affiliations that make it a "all around valuable PC". It can handle single data centers, yet likewise entire courses of action of data. Long Short-Term Memory (LSTM) networks are an enlargement for irregular neural associations, which basically widens their memory. Thusly it is suitable to pick up from critical experiences that have amazingly drawn out time interval slacks in the center. The units of a LSTM are used as building units for the layers of a RNN, which is then regularly called a LSTM association. LSTM engages RNN's to remember their commitments all through a broad timespan. This is because LSTM contains their information in a memory, that is a ton like the memory of a PC considering the way that the LSTM can scrutinize, create and eradicate information from its memory.

8.6 PREDICTION AND VISUALIZATION

To anticipate future stock costs we need to do a few things ensuing to stacking in the test set:

1.Merge the readiness set and the test set on the 0 centers.

2.Set the time adventure as 60 (as noticed in advance)

3.Use MinMaxScaler to change the new dataset

4.Reshape the dataset as done already

Ensuing to making the desires we use inverse_transform to get back the stock expenses in a customary clear association. Finally, we use Matplotlib to imagine the delayed consequence of the foreseen stock expense and the certified stock expense.

Chapter 9: Conclusion

The fame of financial exchange exchanging is developing quickly, which is urging scientists to discover new strategies for the forecast utilizing new methods. The deciding technique isn't simply supporting the trained professionals at this point it moreover helps theorists and any individual dealing with the monetary trade. To help envision the stock documents, a deciding model with incredible exactness is required. In this work, we have used maybe the most definite envisioning development using Long Short-Term Memory unit which helps monetary subject matter experts, inspectors or any individual enthused about placing assets into the protections trade by giving them a good data on the future situation of the monetary trade. When stood out from the ARIMA (Auto Regressive Integrated Moving Average) count, it is exhibited that ARIMA estimation understands the past data and doesn't focus in on an intermittent part. As such exactness is less. LSTM gives more definite results than ARIMA count. From the plot, you can see that the veritable stock expense went up, while our model similarly foreseen that the expense of the stock will go up. Moreover, this obviously shows how astonishing LSTMs are for examining the time plan and the progressive data. And all the assessment here has been executed with no trouble, because of keras and the down to earth API. Considering, I'd want to express that the conspicuousness of protections trade trading is growing incredibly rapidly, which is asking researchers to find new techniques for the desire using new strategies and the assessing methodology isn't just valuable to subject matter experts, yet furthermore helps examiners or any individual dealing with the protections trade to help envision the stock records. An assessing model with a good exactness is required. In this work I have used perhaps the most precise foreseeing advancement is using RNNs and LSTM units, which help monetary experts specialists or any individual enthusiastic about placing assets into the protections trade by giving them a good data on the future situation of the monetary trade.

9.1 FUTURE SCOPE

As a future direction, this research would like to perform a comparative analysis with other deep learning classifiers and extraordinary learning classifiers with the assistance of an element decrease calculation dependent on the boundaries utilized for securities exchange expectation. The future improvement incorporates contrasting the accuracy of LSTM and other forecast calculations. LSTM is more precise than some other forecast calculations.

Chapter 10: Bibliography

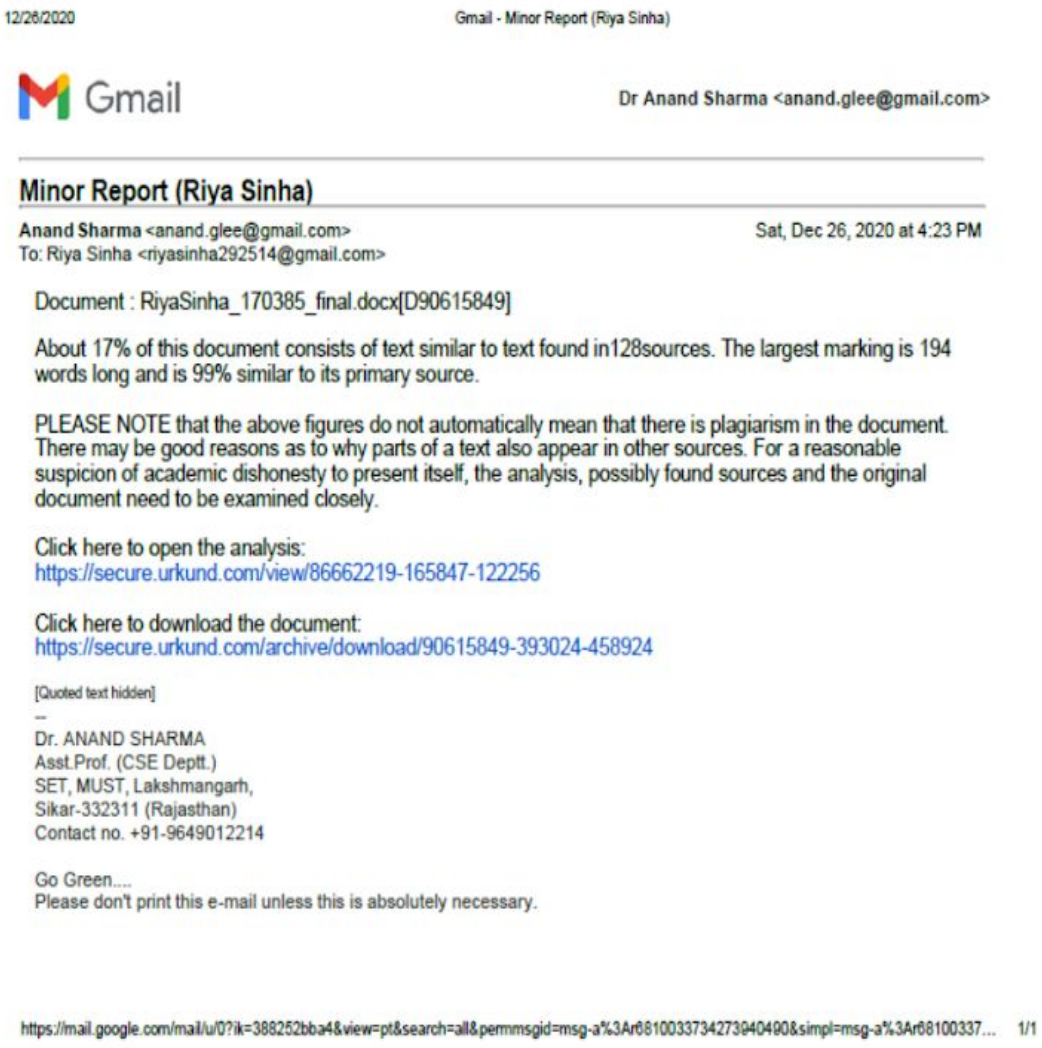
[1] Salah Bouktif, Ali Fiaz, Mamoun Awad, "Stock Market Movement Prediction using Disparate Text Features with Machine Learning", *Intelligent Computing in Data Sciences (ICDS) 2019 Third International Conference on*, pp. 1-6, 2019.

[2] Puneet Misra, Siddharth Chaurasia, "Data-Driven Trend Forecasting in Stock Market Using Machine Learning Techniques", *Journal of Information Technology Research*, vol. 13, pp. 130, 2020.

[3] Meghna Misra, Ajay Prakash Yadav, Harkiran Kaur, "Stock Market Prediction using Machine Learning Algorithms: A Classification Study", *Recent Innovations in Electrical Electronics & Communication Engineering (ICRIEECE) 2018 International Conference on*, pp. 2475-2478, 2018.

[4]https://towardsdatascience.com/predicting-the-stock-market-with-machine-learning-introduction-310cd6069ffa : Accessed on : 27/10/2020

Chapter 11: Plagiarism Report



12/26/2020 Gmail - Minor Report (Riya Sinha)

Gmail Dr Anand Sharma <anand.glee@gmail.com>

Minor Report (Riya Sinha)

Anand Sharma <anand.glee@gmail.com> Sat, Dec 26, 2020 at 4:23 PM
To: Riya Sinha <riyasinha292514@gmail.com>

Document : RiyaSinha_170385_final.docx[D90615849]

About 17% of this document consists of text similar to text found in128sources. The largest marking is 194 words long and is 99% similar to its primary source.

PLEASE NOTE that the above figures do not automatically mean that there is plagiarism in the document. There may be good reasons as to why parts of a text also appear in other sources. For a reasonable suspicion of academic dishonesty to present itself, the analysis, possibly found sources and the original document need to be examined closely.

Click here to open the analysis:
https://secure.urkund.com/view/86662219-165847-122256

Click here to download the document:
https://secure.urkund.com/archive/download/90615849-393024-458924

[Quoted text hidden]
--
Dr. ANAND SHARMA
Asst.Prof. (CSE Deptt.)
SET, MUST, Lakshmangarh,
Sikar-332311 (Rajasthan)
Contact no. +91-9649012214

Go Green....
Please don't print this e-mail unless this is absolutely necessary.

https://mail.google.com/mail/u/0?ik=388252bba4&view=pt&search=all&permmsgid=msg-a%3Ar6810033734273940490&simpl=msg-a%3Ar68100337... 1/1