VISVESVARAYA TECHNOLOGICAL UNIVERSITY

BELAGAVI – 590018,
Karnataka

INTERNSHIP REPORT

ON

“Predictive Sentiment Analysis”

Submitted in partial fulfillment for the award of degree(18ECI85)

BACHELOR OF ENGINEERING IN ELECTRONICS AND COMMUNICATION

Submitted by:

Name: Bhuvaneshwar Reddy J C

USN: 1BY19EC031

Conducted at
Varcons Technologies Pvt Ltd

BMS INSTITUTE OF TECHNOLOGY & MANAGEMENT

Department of Electronics and Communication

Yelahanka, Bengaluru-560064

BMS INSTITUTE OF TECHNOLOGY & MANAGEMENT

Department of Electronics and Communication

Yelahanka, Bengaluru-560064

CERTIFICATE

This is to certify that the Internship titled **"Predictive Sentiment Analysis"** carried out by **Bhuvaneshwar Reddy J C,** a bonafide student of BMS Institute of Technology & Management, in partial fulfillment for the award of **Bachelor of Engineering**, in **Electronics and Communication** under Visvesvaraya Technological University, Belagavi, during the year 2022-2023. It is certified that all corrections/suggestions indicated have been incorporated in the report.

The project report has been approved as it satisfies the academic requirements in respect of Internship prescribed for the course Internship / Professional Practice (18ECI85)

Signature of Guide **Signature of HOD** **Signature of Principal**

External Viva:

Name of the Examiner Signature with Date

1)__________________

2)__________________

D E C L A R A T I O N

I, Bhuvaneshwar Reddy J C, final year student of department Electronics and Communication Engineering, BMS Institute of Technology and Management - 560 064, declare that the Internship has been successfully completed, in **Varcons Technology Pvt Ltd**. This report is submitted in partial fulfillment of the requirements for award of Bachelor Degree in Electronics and Communication Engineering, during the academic year 2022-2023.

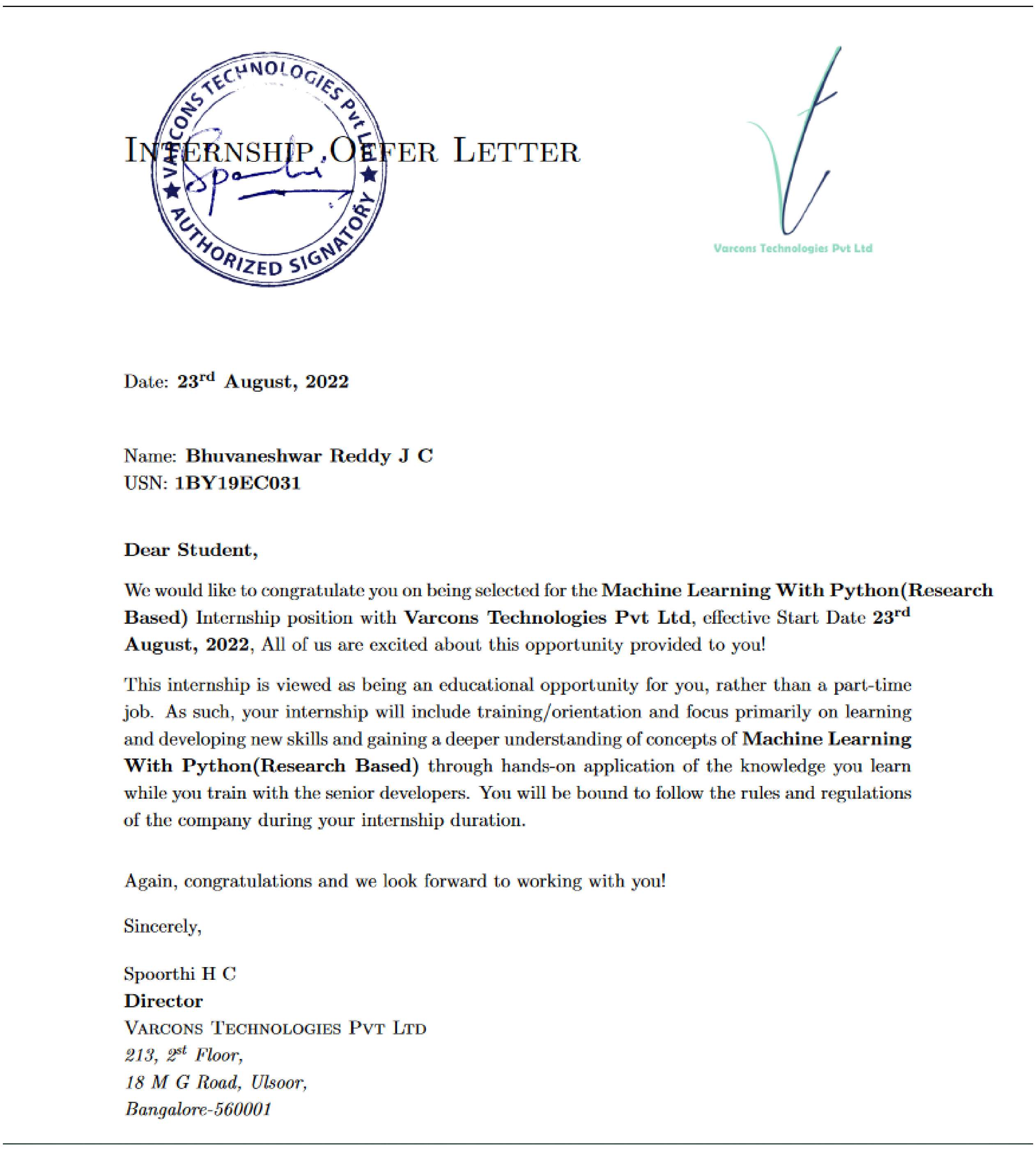
Date: 22-09-2022

Place: Bengaluru

USN: 1BY19EC031

NAME: Bhuvaneshwar Reddy J C

OFFER LETTER



VARCONS TECHNOLOGIES Pvt Ltd

AUTHORIZED SIGNATORY

[illegible]

Internship Offer Letter

Varcons Technologies Pvt Ltd

Date: **23rd August, 2022**

Name: **Bhuvaneshwar Reddy J C**
USN: **1BY19EC031**

Dear Student,

We would like to congratulate you on being selected for the **Machine Learning With Python(Research Based)** Internship position with **Varcons Technologies Pvt Ltd**, effective Start Date **23rd August, 2022**, All of us are excited about this opportunity provided to you!

This internship is viewed as being an educational opportunity for you, rather than a part-time job. As such, your internship will include training/orientation and focus primarily on learning and developing new skills and gaining a deeper understanding of concepts of **Machine Learning With Python(Research Based)** through hands-on application of the knowledge you learn while you train with the senior developers. You will be bound to follow the rules and regulations of the company during your internship duration.

Again, congratulations and we look forward to working with you!

Sincerely,

Spoorthi H C
Director
Varcons Technologies Pvt Ltd
213, 2st Floor,
18 M G Road, Ulsoor,
Bangalore-560001

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

This Internship is a result of accumulated guidance, direction and support of several important persons. We take this opportunity to express our gratitude to all who have helped us to complete the Internship.

We express our sincere thanks to our Principal, for providing us adequate facilities to undertake this Internship.

We would like to thank our Head of Dept – branch code, for providing us an opportunity to carry out Internship and for his valuable guidance and support.

We would like to thank our Varcons Software Services for guiding us during the period of internship.

We express our deep and profound gratitude to our guide, Assistant/Associate Prof, for their keen interest and encouragement at every step in completing the Internship.

We would like to thank all the faculty members of our department for the support extended during the course of Internship.

We would like to thank the non-teaching members of our dept, for helping us during the Internship.

Last but not the least, we would like to thank our parents and friends without whose constant help, the completion of Internship would have not been possible.

NAME:
Bhuvaneshwar
Reddy J C
USN: 1BY19EC031

ABSTRACT

Stock market prediction has been an active area of research for a considerable period. Mass psychology's effects may not be the only factor driving the markets, but it's unquestionably significant

This fascinating quality is something that we can measure and use to predict market movement with surprising accuracy levels.

With the real-time information available to us on massive social media platforms like Twitter, we have all the data we could ever need to create these predictions.

That is where sentiment analysis comes in. Sentiment analysis is a particularly interesting branch of Natural Language Processing (NLP), which is used to rate the language used in a body of text.

Through sentiment analysis, we can take thousands of tweets about a company and judge whether they are generally positive or negative (the sentiment) in real-time!

Arrival of computing, followed by Machine Learning has upgraded the speed of research as well as opened new avenues. As part of this research study, we aimed to predict the future stock movement of shares using the historical prices aided with availability of sentiment data.

In this paper, we apply sentiment analysis and machine learning principles to find the correlation between" public sentiment" and" market sentiment"

Table of Contents

<u>CHAPTER 1</u>

COMPANY PROFILE

1. COMPANY PROFILE

Varcons Technologies Pvt Ltd

A Brief History of Varcons Technologies

Varcons Technologies, was incorporated with a goal" To provide high quality and optimal Technological Solutions to business requirements of our clients". Every business is different and has a unique business model and so are the technological requirements. They understand this and hence the solutions provided to these requirements are different as well. They focus on clients' requirements and provide them with tailor made technological solutions. They also understand that Reach of their Product to its targeted market or the automation of the existing process into an e-client and simple process are the key features that our clients desire from the Technological Solution they are looking for and these are the features that we focus on while designing the solutions for their clients.

Sarvamoola Software Services. is a Technology Organization providing solutions for all web design and development, MYSQL, PYTHON Programming, HTML, CSS, ASP.NET and LINQ. Meeting the ever-increasing automation requirements, Sarvamoola Software Services. specialize in ERP, Connectivity, SEO Services, Conference Management, effective web promotion and tailor-made software products, designing solutions best suiting clients' requirements.

Varcons Technologies, strive to be the front runner in creativity and innovation in software development through their well-researched expertise and establish it as an out of the box software development company in Bangalore, India. As a software development company, they translate this software development expertise into value for their customers through their professional solutions.

They understand that the best desired output can be achieved only by understanding the client's demand better. Varcons Technologies work with their clients and help them to define their exact solution requirement. Sometimes even they wonder if they have completely redefined their solution or new application requirement during the brainstorming session, and here they position themselves as an IT solutions consulting group composed of high caliber consultants.

They believe that Technology when used properly can help any business to scale and achieve new heights of success. It helps Improve its efficiency, profitability, reliability; to put it in one sentence" Technology helps you Delight your customers" and that is what we want to achieve.

<u>CHAPTER 2</u>

ABOUT THE COMPANY

2. ABOUT THE COMPANY

Varcons Technologies is a Technology Organization providing solutions for all web design and development, MYSQL, PYTHON Programming, HTML, CSS, ASP.NET and LINQ. Meeting the ever-increasing automation requirements, Varcons Technologies specialize in ERP, Connectivity, SEO Services, Conference Management, effective web promotion and tailor-made software products, designing solutions best suiting clients' requirements. The organization where they have a right mix of professionals as stakeholders to help us serve our clients with best of our capability and with at-par industry standards. They have young, enthusiastic, passionate and creative Professionals to develop technological innovations in the field of Mobile technologies, Web applications as well as Business and Enterprise solutions. Motto of our organization is to "Collaborate with our clients to provide them with the best Technological solution hence creating a Good Present and Better Future for our client which will bring a cascading positive effect in their business shape as well". Providing a Complete suite of technical solutions is not just our tagline, it is Our Vision for Our Clients and for Us, We strive hard to achieve it.

Products of Varcons Technologies.

Android Apps

It is the process by which new applications are created for devices running the Android operating system. Applications are usually developed in Java (and/or Kotlin; or other such option) programming language using the Android software development kit (SDK), but other development environments are also available, some such as Kotlin support the exact same Android APIs (and bytecode), while others such as Go have restricted API access.

The Android software development kit includes a comprehensive set of development tools. These include a debugger, libraries, a handset emulator based on QEMU, documentation, sample code, and tutorials. Currently supported development platforms include computers running Linux (any modern desktop Linux distribution), Mac OS X 10.5.8 or later, and Windows 7 or later. As of March 2015, the SDK is not available on Android itself, but software development is possible by using specialized Android applications.

Web Application

It is a client–server computer program in which the client (including the user interface and client- side logic) runs in a web browser. Common web applications include web mail, online retail sales, online auctions, wikis, instant messaging services and many other functions. web applications use web documents written in a standard format such as HTML and JavaScript, which are supported by a variety of web browsers. Web applications can be considered as a specific variant of client–server software where the client software is downloaded to the client machine when visiting the relevant web page, using standard procedures such as HTTP. The Client web software updates may happen each time the web page is visited. During the session, the web browser interprets and displays the pages, and acts as the universal client for any web application. The use of web application frameworks can often reduce the number of errors in a program, both by making the code simpler, and by allowing one team to concentrate on the framework while another focuses on a specified use case. In applications which are exposed to constant hacking attempts on the Internet, security- related problems can be caused by errors in the program.

Frameworks can also promote the use of best practices such as GET after POST. There are some who view a web application as a two-tier architecture. This can be a “smart” client that performs all the work and queries a “dumb” server, or a “dumb” client that relies on a “smart” server. The client would handle the presentation tier, the server would have the database (storage tier), and the business logic (application tier) would be on one of them or on both. While this increases the scalability of the applications and separates the display and the database, it still doesn’t allow for true specialization of layers, so most applications will outgrow this model. An emerging strategy for application software companies is to provide web access to software previously distributed as local applications. Depending on the type of application, it may require the development of an entirely different browser-based interface, or merely adapting an existing application to use different presentation technology. These programs allow the user to pay a monthly or yearly fee for use of a software application without having to install it on a local hard drive. A company which follows this strategy is known as an application service provider (ASP), and ASPs are currently receiving much attention in the software industry.

Security breaches on these kinds of applications are a major concern because it can involve both enterprise information and private customer data. Protecting these assets is an important part of any web application and there are some key operational areas that must be included in the development process. This includes processes for authentication, authorization, asset handling, input, and logging and auditing. Building security into the applications from the beginning can be more effective and less disruptive in the long run.

Web design

It encompasses many different skills and disciplines in the production and maintenance of websites. The different areas of web design include web graphic design; interface design; authoring, including standardized code and proprietary software; user experience design; and search engine optimization. The term web design is normally used to describe the design process relating to the front-end (client side) design of a website including writing mark up. Web design partially overlaps web engineering in the broader scope of web development. Web designers are expected to have an awareness of usability and if their role involves creating markup then they are also expected to be up to date with web accessibility guidelines. Web design partially overlaps web engineering in the broader scope of web development.

Departments and services offered

Varcons Technologies plays an essential role as an institute, the level of education, development of student's skills are based on their trainers. If you do not have a good mentor then you may lag in many things from others and that is why we at Varcons Technologies gives you the facility of skilled employees so that you do not feel unsecured about the academics. Personality development and academic status are some of those things which lie on mentor's hands. If you are trained well then you can do well in your future and knowing its importance of Varcons Technologies always tries to give you the best.

They have a great team of skilled mentors who are always ready to direct their trainees in the best possible way they can and to ensure the skills of mentors we held many skill development programs as well so that each and every mentor can develop their own skills with the demands of the companies so that they can prepare a complete packaged trainee.

Services provided by Varcons Technologies.

- Core Java and Advanced Java
- Web services and development
- Dot Net Framework
- Python
- Selenium Testing
- Conference / Event Management Service
- Academic Project Guidance
- On The Job Training
- Software Training

CHAPTER 3

INTRODUCTION

3. INTRODUCTION

Introduction to ML

Machine learning (ML) is a branch of artificial intelligence (AI) that enables computers to "self-learn" from training data and improve over time, without being explicitly programmed. Machine Learning Algorithms are able to detect patterns in data and learn from them, in order to make their own predictions. In short, machine learning algorithms and models learn through experience.

The term machine learning was first coined in the 1950s when Artificial Intelligence Pioneer Arthur Samuel built the first self-learning system for playing checkers. He noticed that the more the system played, the better it performed.

Fueled by advances in statistics and computer science, as well as better datasets and the growth of neural networks, machine learning has truly taken off in recent years.

Today, whether you realize it or not, machine learning is everywhere – automated translation, image recognition, voice search technology, self-driving cars, and beyond.

Types of Machine Learning algorithms:

- Supervised
- Unsupervised
- Reinforcement

Problem Statement

In this project, Stockport-Predictive Sentiment Analysis, a model had to be designed for real-time Twitter Sentiment Analysis for stocks based on which the future movement of the market is predicted.
Goal was to understand the working of Sentiment analysis and Improve the accuracy.
Where the current accuracy rate was 92/100

CHAPTER 4

SYSTEM ANALYSIS

4. SYSTEM ANALYSIS

1. Existing System

We have different models for sentiment prediction:
RNN model-Recurrent Neural Network Models is one of the models of neural networks that do not depend on size about a window when a natural language processing scenario is considered. RNN can keep the record of all the input values that are viewed by the network and also current input, which is the value hidden at each layer network depending on all the previously seen inputs.
VADER model-VADER (Valence Aware Dictionary and sEntiment Reasoner) is a lexicon and rule-based sentiment analysis tool that is specifically attuned to sentiments expressed in social media. VADER uses a combination of A sentiment lexicon is a list of lexical features (e.g., words) which are generally labeled according to their semantic orientation as either positive or negative. VADER not only talks about the Positivity and Negativity score but also tells us about how positive or negative a sentiment is.

2. Proposed System

LSTM model is used in this project.
Long short-term memory networks (LSTMs) are an extension for recurrent neural networks, which basically extends the memory. LSTMs enable RNNs to remember inputs over a long period of time. This is because LSTMs contain information in a memory, much like the memory of a computer. The LSTM can read, write and delete information from its memory. This memory can be seen as a gated cell, with gated meaning the cell decides whether or not to store or delete information (i.e., if it opens the gates or not), based on the importance it assigns to the information. The assigning of importance happens through weights, which are also learned by the algorithm. This simply means that it learns over time what information is important and what is not.
In an LSTM you have three gates: input, forget and output gate. These gates determine whether or not to let new input in (input gate), delete the information because it isn't important (forget gate), or let it impact the output at the current timestep (output gate).
The gates in an LSTM are analog in the form of sigmoid, meaning they range from zero to one. The fact that they are analog enables them to do backpropagation.
The problematic issue of vanishing gradients is solved through LSTM because it keeps the gradients steep enough, which keeps the training relatively short and the accuracy high.

3. Objective of the System

The objective of this project is to conduct Real-Time Twitter Sentiment Analysis for stocks based on which the future movement of the
the market is predicted.
Through this project, we understand the working of sentiment analysis and aim at improving the accuracy.

CHAPTER 5

REQUIREMENT ANALYSIS

5. REQUIREMENT ANALYSIS

Software Requirement Specification

The software requirements for this project is:

- Python
- Google Colab
- LSTM model

❖ Python

Python is an interpreted, object-oriented, high-level programming language with dynamic semantics. Its high-level built in data structures, combined with dynamic typing and dynamic binding, make it very attractive for Rapid Application Development, as well as for use as a scripting or glue language to connect existing components together. Python's simple, easy to learn syntax emphasizes readability and therefore reduces the cost of program maintenance. Python supports modules and packages, which encourages program modularity and code reuse. The Python interpreter and the extensive standard library are available in source or binary form without charge for all major platforms, and can be freely distributed

❖ Google Colab

Colaboratory, or "Colab" for short, is a product from Google Research. Colab allows anybody to write and execute arbitrary python code through the browser, and is especially well suited to machine learning, data analysis and education. More technically, Colab is a hosted Jupyter notebook service that requires no setup to use, while providing access free of charge to computing resources including GPUs.

As a programmer, using Google Colab you can, Write and execute code in python, document your code that supports mathematical equations and so on.

❖ LSTM model

LSTM or long short-term memory models are used to solve these problems. LSTMs are a special kind of RNN model that has the capability of learning long term dependencies. They are made to remember long term data. What makes LSTM models special is the addition of a memory cell, which is an extra recurrent state and each cell has multiple gates that control the flow of information in and out of the memory cell.

There are three types of gates in an LSTM: an input gate, an output gate and a forget gate. The input gate is used to update the cell status, the forget state decides which information is to be stored and which needs to be discarded. The output gate determines the values for the next hidden gates.

CHAPTER 6

DESIGN ANALYSIS

6. DESIGN & ANALYSIS

After preprocessing the data and splitting the dataset, the LSTM model is used to train our model. LSTM layer is created with 100 neurons and also added a dense layer with sigmoid activation function. The process is introduced as follows:

- Due to their emotional marks, all training data is divided into three groups, positive, negative and neutral. The LSTM models are then trained in each data category and with multiple LSTM models resulting in them as well. This is done for equal ratings.
- To get a new input review, the LSTM models are available in the training phase evaluated on the new input review. The model which is giving the smallest error value is assigned to the new input review.

Figure 7 is showing the structure of the processing of the training phase: This model could overcome the vanishing gradient problem completely than the conventional RNN model. It also performs better in many experiments, like structure with conjunctions, such as, ‘not only…but also…,’ ‘However’, ‘in addition,’ etc.

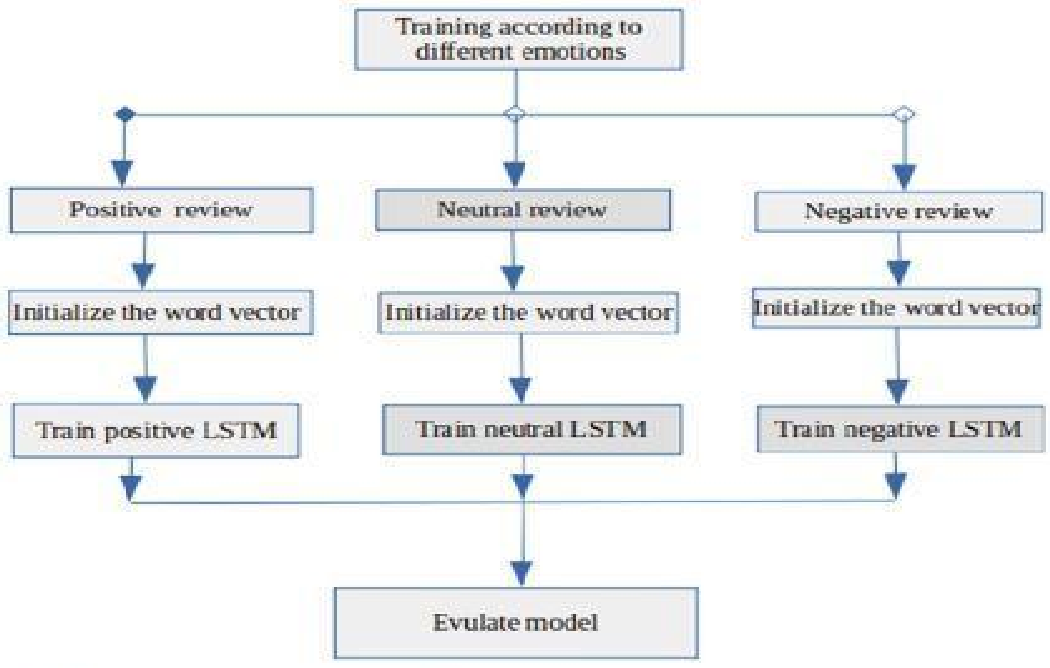

Training according to different emotions
Positive review
Neutral review
Negative review
Initialize the word vector
Initialize the word vector
Initialize the word vector
Train positive LSTM
Train neutral LSTM
Train negative LSTM
Evulate model

Fig. 7 Business sentimental emotion classification

Testing the Model

After training the model, new text data is used for testing our model. When a new product review is coming, this model has classified the new review as negative, positive or neutral.

Figure 8 describes the working process of our RNN-LSTM model in business sentiment analysis.

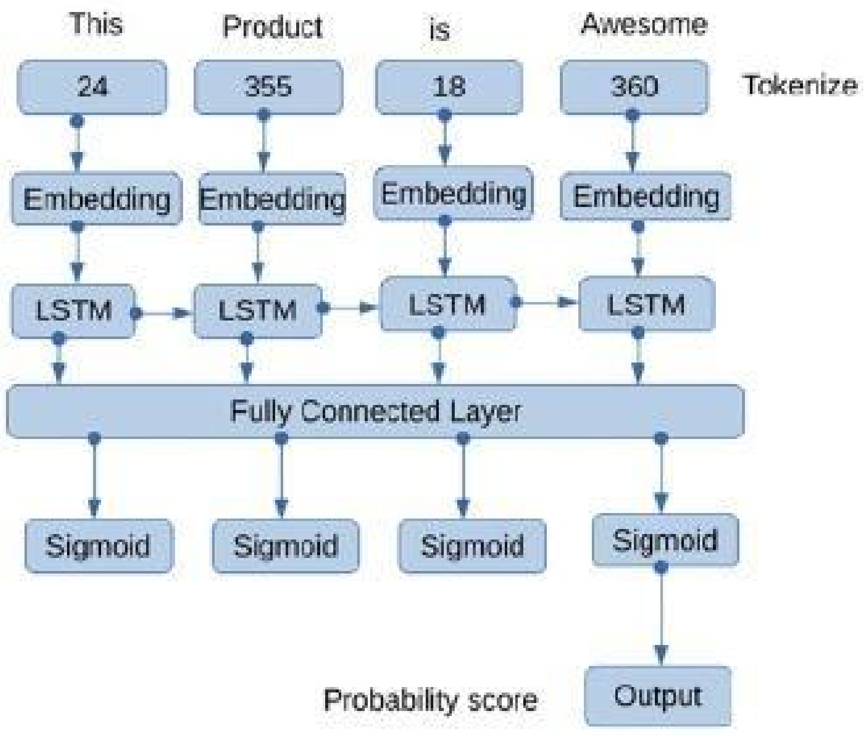


Fig. 8 RNN-LSTM working process in business sentiment analysis

<u>CHAPTER 7</u>

IMPLEMENTATION

7. IMPLEMENTATION

Implementation is the stage where the theoretical design is turned into a working system. The most crucial stage in achieving a new successful system and in giving confidence on the new system for the users that it will work efficiently and effectively.

The system can be implemented only after thorough testing is done and if it is found to work according to the specification. It involves careful planning, investigation of the current system and its constraints on implementation, design of methods to achieve the changeover and an evaluation of change over methods apart from planning.

Two major tasks of preparing the implementation are education and training of the users and testing of the system. The more complex the system being implemented, the more involved will be the system analysis and design effort required just for implementation.

The implementation phase comprises several activities. The required hardware and software acquisition is carried out. The system may require some software to be developed. For this, programs are written and tested. The user then changes over to his new fully tested system and the old system is discontinued.

TESTING

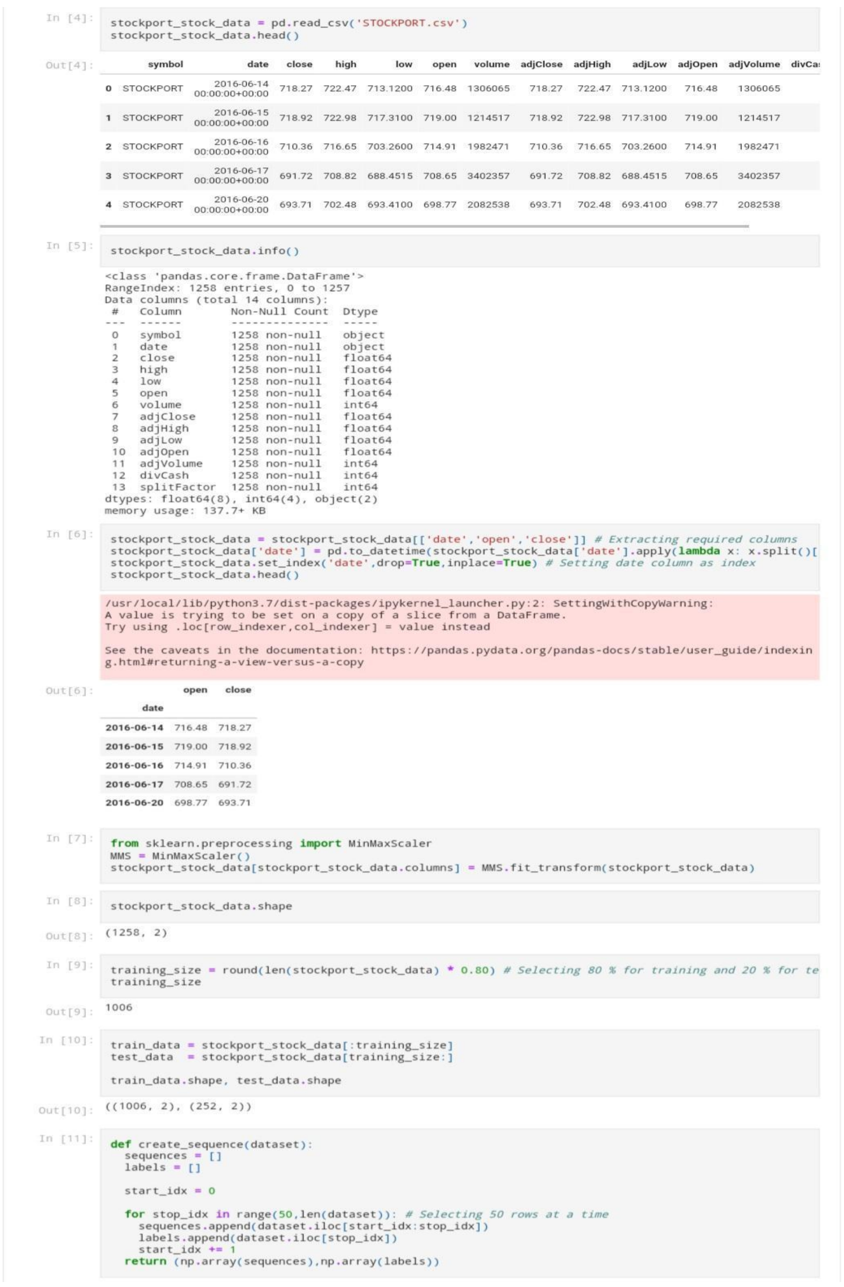
The testing phase is an important part of software development. The Information zed system will help in automating the process of finding errors and missing operations and also a complete verification to determine whether the objectives are met and the user requirements are satisfied. Software testing is carried out in three steps:

1. The first includes unit testing, where each module is tested to provide its correctness, validity and also determine any missing operations and to verify whether the objectives have been met. Errors are noted down and corrected immediately.
2. Unit testing is the important and major part of the project. So, errors are rectified easily in particular modules and program clarity is increased. In this project the entire system is divided into several modules and is developed individually. So, unit testing is conducted to individual modules.
3. The second step includes Integration testing. It need not be the case, the software whose modules when run individually and showing perfect results, will also show perfect results when run as a whole.

CHAPTER 8

SNAPSHOTS

8. SNAPSHOTS



```
In [4]:  stockport_stock_data = pd.read_csv('STOCKPORT.csv')
         stockport_stock_data.head()
```

Out[4]:

	symbol	date	close	high	low	open	volume	adjClose	adjHigh	adjLow	adjOpen	adjVolume	divCa
0	STOCKPORT	2016-06-14 00:00:00+00:00	718.27	722.47	713.1200	716.48	1306065	718.27	722.47	713.1200	716.48	1306065	
1	STOCKPORT	2016-06-15 00:00:00+00:00	718.92	722.98	717.3100	719.00	1214517	718.92	722.98	717.3100	719.00	1214517	
2	STOCKPORT	2016-06-16 00:00:00+00:00	710.36	716.65	703.2600	714.91	1982471	710.36	716.65	703.2600	714.91	1982471	
3	STOCKPORT	2016-06-17 00:00:00+00:00	691.72	708.82	688.4515	708.65	3402357	691.72	708.82	688.4515	708.65	3402357	
4	STOCKPORT	2016-06-20 00:00:00+00:00	693.71	702.48	693.4100	698.77	2082538	693.71	702.48	693.4100	698.77	2082538	

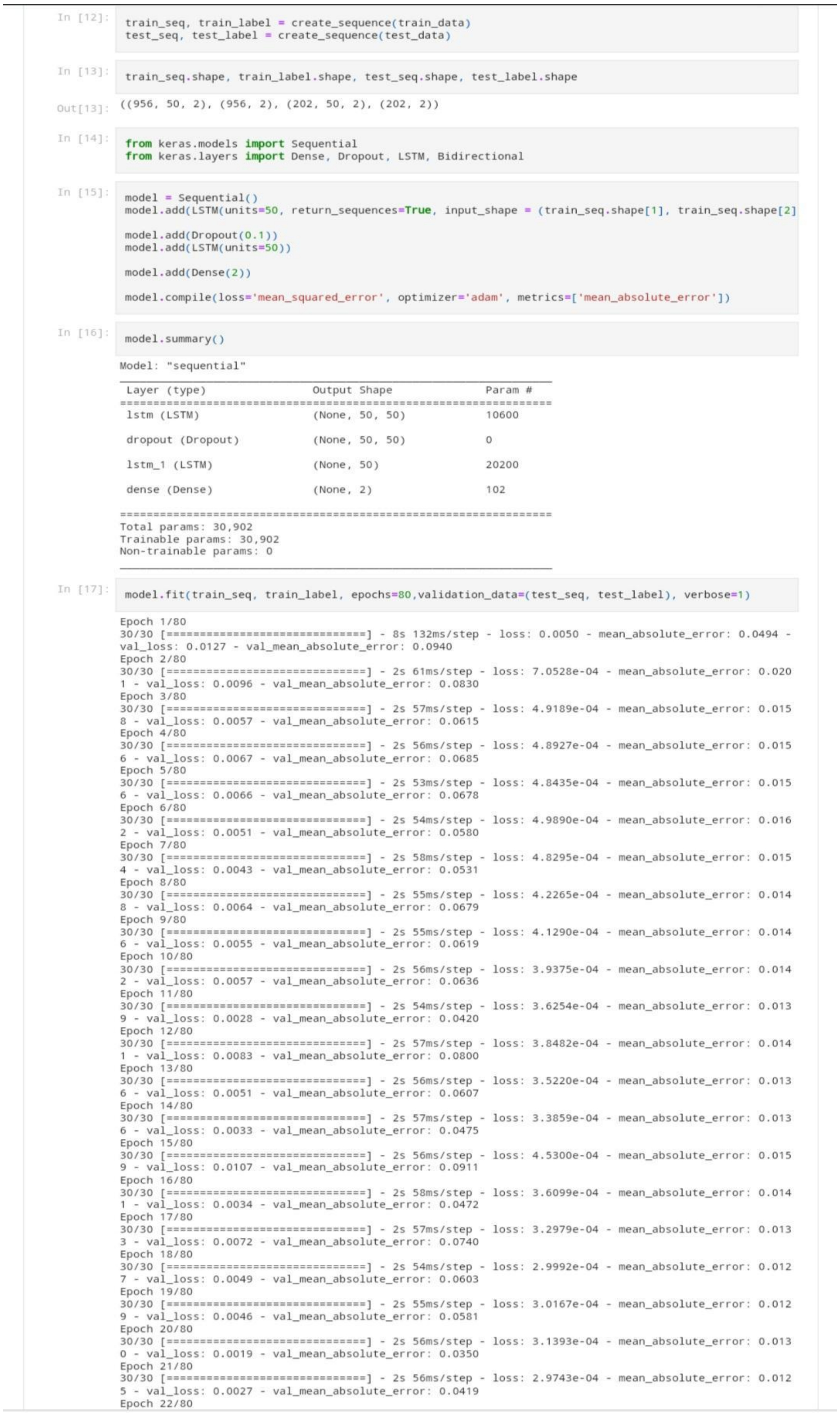
```
In [5]:  stockport_stock_data.info()

<class 'pandas.core.frame.DataFrame'>
RangeIndex: 1258 entries, 0 to 1257
Data columns (total 14 columns):
 #   Column       Non-Null Count  Dtype
---  ------       --------------  -----
 0   symbol       1258 non-null   object
 1   date         1258 non-null   object
 2   close        1258 non-null   float64
 3   high         1258 non-null   float64
 4   low          1258 non-null   float64
 5   open         1258 non-null   float64
 6   volume       1258 non-null   int64
 7   adjClose     1258 non-null   float64
 8   adjHigh      1258 non-null   float64
 9   adjLow       1258 non-null   float64
 10  adjOpen      1258 non-null   float64
 11  adjVolume    1258 non-null   int64
 12  divCash      1258 non-null   int64
 13  splitFactor  1258 non-null   int64
dtypes: float64(8), int64(4), object(2)
memory usage: 137.7+ KB
```

```
In [6]:  stockport_stock_data = stockport_stock_data[['date','open','close']] # Extracting required columns
         stockport_stock_data['date'] = pd.to_datetime(stockport_stock_data['date'].apply(lambda x: x.split()[
         stockport_stock_data.set_index('date',drop=True,inplace=True) # Setting date column as index
         stockport_stock_data.head()

/usr/local/lib/python3.7/dist-packages/ipykernel_launcher.py:2: SettingWithCopyWarning:
A value is trying to be set on a copy of a slice from a DataFrame.
Try using .loc[row_indexer,col_indexer] = value instead

See the caveats in the documentation: https://pandas.pydata.org/pandas-docs/stable/user_guide/indexin
g.html#returning-a-view-versus-a-copy
```

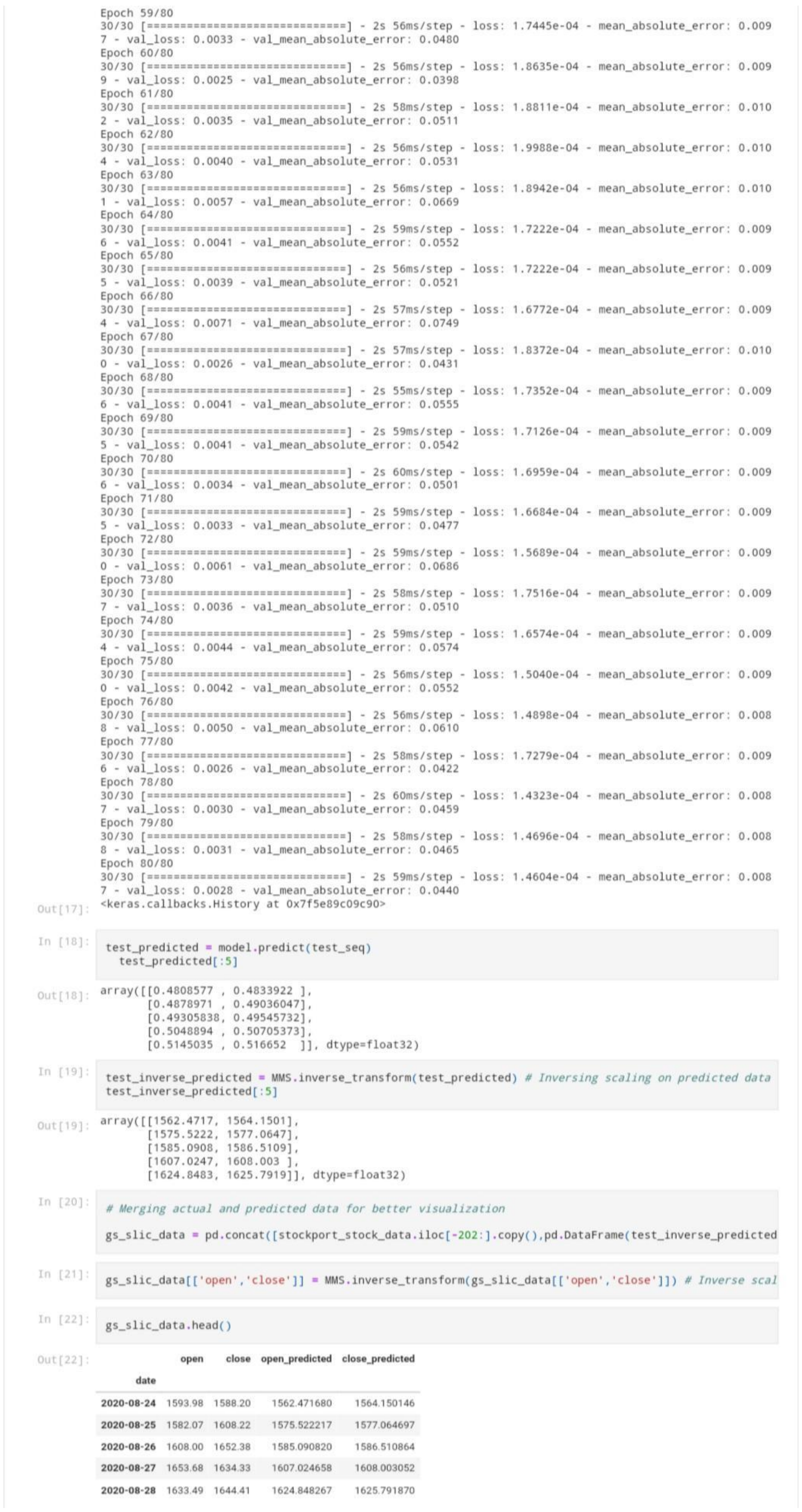
Out[6]:

	open	close
date		
2016-06-14	716.48	718.27
2016-06-15	719.00	718.92
2016-06-16	714.91	710.36
2016-06-17	708.65	691.72
2016-06-20	698.77	693.71

```
In [7]:  from sklearn.preprocessing import MinMaxScaler
         MMS = MinMaxScaler()
         stockport_stock_data[stockport_stock_data.columns] = MMS.fit_transform(stockport_stock_data)

In [8]:  stockport_stock_data.shape

Out[8]:  (1258, 2)

In [9]:  training_size = round(len(stockport_stock_data) * 0.80) # Selecting 80 % for training and 20 % for te
         training_size

Out[9]:  1006

In [10]: train_data = stockport_stock_data[:training_size]
         test_data  = stockport_stock_data[training_size:]

         train_data.shape, test_data.shape

Out[10]: ((1006, 2), (252, 2))

In [11]: def create_sequence(dataset):
           sequences = []
           labels = []

           start_idx = 0

           for stop_idx in range(50,len(dataset)): # Selecting 50 rows at a time
             sequences.append(dataset.iloc[start_idx:stop_idx])
             labels.append(dataset.iloc[stop_idx])
             start_idx += 1
           return (np.array(sequences),np.array(labels))
```



```
In [12]:  train_seq, train_label = create_sequence(train_data)
          test_seq, test_label = create_sequence(test_data)

In [13]:  train_seq.shape, train_label.shape, test_seq.shape, test_label.shape

Out[13]: ((956, 50, 2), (956, 2), (202, 50, 2), (202, 2))

In [14]:  from keras.models import Sequential
          from keras.layers import Dense, Dropout, LSTM, Bidirectional

In [15]:  model = Sequential()
          model.add(LSTM(units=50, return_sequences=True, input_shape = (train_seq.shape[1], train_seq.shape[2]

          model.add(Dropout(0.1))
          model.add(LSTM(units=50))

          model.add(Dense(2))

          model.compile(loss='mean_squared_error', optimizer='adam', metrics=['mean_absolute_error'])

In [16]:  model.summary()

         Model: "sequential"
         _________________________________________________________________
          Layer (type)                Output Shape              Param #
         =================================================================
          lstm (LSTM)                 (None, 50, 50)            10600

          dropout (Dropout)           (None, 50, 50)            0

          lstm_1 (LSTM)               (None, 50)                20200

          dense (Dense)               (None, 2)                 102

         =================================================================
         Total params: 30,902
         Trainable params: 30,902
         Non-trainable params: 0
         _________________________________________________________________

In [17]:  model.fit(train_seq, train_label, epochs=80,validation_data=(test_seq, test_label), verbose=1)

         Epoch 1/80
         30/30 [==============================] - 8s 132ms/step - loss: 0.0050 - mean_absolute_error: 0.0494 -
         val_loss: 0.0127 - val_mean_absolute_error: 0.0940
         Epoch 2/80
         30/30 [==============================] - 2s 61ms/step - loss: 7.0528e-04 - mean_absolute_error: 0.020
         1 - val_loss: 0.0096 - val_mean_absolute_error: 0.0830
         Epoch 3/80
         30/30 [==============================] - 2s 57ms/step - loss: 4.9189e-04 - mean_absolute_error: 0.015
         8 - val_loss: 0.0057 - val_mean_absolute_error: 0.0615
         Epoch 4/80
         30/30 [==============================] - 2s 56ms/step - loss: 4.8927e-04 - mean_absolute_error: 0.015
         6 - val_loss: 0.0067 - val_mean_absolute_error: 0.0685
         Epoch 5/80
         30/30 [==============================] - 2s 53ms/step - loss: 4.8435e-04 - mean_absolute_error: 0.015
         6 - val_loss: 0.0066 - val_mean_absolute_error: 0.0678
         Epoch 6/80
         30/30 [==============================] - 2s 54ms/step - loss: 4.9890e-04 - mean_absolute_error: 0.016
         2 - val_loss: 0.0051 - val_mean_absolute_error: 0.0580
         Epoch 7/80
         30/30 [==============================] - 2s 58ms/step - loss: 4.8295e-04 - mean_absolute_error: 0.015
         4 - val_loss: 0.0043 - val_mean_absolute_error: 0.0531
         Epoch 8/80
         30/30 [==============================] - 2s 55ms/step - loss: 4.2265e-04 - mean_absolute_error: 0.014
         8 - val_loss: 0.0064 - val_mean_absolute_error: 0.0679
         Epoch 9/80
         30/30 [==============================] - 2s 55ms/step - loss: 4.1290e-04 - mean_absolute_error: 0.014
         6 - val_loss: 0.0055 - val_mean_absolute_error: 0.0619
         Epoch 10/80
         30/30 [==============================] - 2s 56ms/step - loss: 3.9375e-04 - mean_absolute_error: 0.014
         2 - val_loss: 0.0057 - val_mean_absolute_error: 0.0636
         Epoch 11/80
         30/30 [==============================] - 2s 54ms/step - loss: 3.6254e-04 - mean_absolute_error: 0.013
         9 - val_loss: 0.0028 - val_mean_absolute_error: 0.0420
         Epoch 12/80
         30/30 [==============================] - 2s 57ms/step - loss: 3.8482e-04 - mean_absolute_error: 0.014
         1 - val_loss: 0.0083 - val_mean_absolute_error: 0.0800
         Epoch 13/80
         30/30 [==============================] - 2s 56ms/step - loss: 3.5220e-04 - mean_absolute_error: 0.013
         6 - val_loss: 0.0051 - val_mean_absolute_error: 0.0607
         Epoch 14/80
         30/30 [==============================] - 2s 57ms/step - loss: 3.3859e-04 - mean_absolute_error: 0.013
         6 - val_loss: 0.0033 - val_mean_absolute_error: 0.0475
         Epoch 15/80
         30/30 [==============================] - 2s 56ms/step - loss: 4.5300e-04 - mean_absolute_error: 0.015
         9 - val_loss: 0.0107 - val_mean_absolute_error: 0.0911
         Epoch 16/80
         30/30 [==============================] - 2s 58ms/step - loss: 3.6099e-04 - mean_absolute_error: 0.014
         1 - val_loss: 0.0034 - val_mean_absolute_error: 0.0472
         Epoch 17/80
         30/30 [==============================] - 2s 57ms/step - loss: 3.2979e-04 - mean_absolute_error: 0.013
         3 - val_loss: 0.0072 - val_mean_absolute_error: 0.0740
         Epoch 18/80
         30/30 [==============================] - 2s 54ms/step - loss: 2.9992e-04 - mean_absolute_error: 0.012
         7 - val_loss: 0.0049 - val_mean_absolute_error: 0.0603
         Epoch 19/80
         30/30 [==============================] - 2s 55ms/step - loss: 3.0167e-04 - mean_absolute_error: 0.012
         9 - val_loss: 0.0046 - val_mean_absolute_error: 0.0581
         Epoch 20/80
         30/30 [==============================] - 2s 56ms/step - loss: 3.1393e-04 - mean_absolute_error: 0.013
         0 - val_loss: 0.0019 - val_mean_absolute_error: 0.0350
         Epoch 21/80
         30/30 [==============================] - 2s 56ms/step - loss: 2.9743e-04 - mean_absolute_error: 0.012
         5 - val_loss: 0.0027 - val_mean_absolute_error: 0.0419
         Epoch 22/80
```



```
Epoch 59/80
30/30 [==============================] - 2s 56ms/step - loss: 1.7445e-04 - mean_absolute_error: 0.009
7 - val_loss: 0.0033 - val_mean_absolute_error: 0.0480
Epoch 60/80
30/30 [==============================] - 2s 56ms/step - loss: 1.8635e-04 - mean_absolute_error: 0.009
9 - val_loss: 0.0025 - val_mean_absolute_error: 0.0398
Epoch 61/80
30/30 [==============================] - 2s 58ms/step - loss: 1.8811e-04 - mean_absolute_error: 0.010
2 - val_loss: 0.0035 - val_mean_absolute_error: 0.0511
Epoch 62/80
30/30 [==============================] - 2s 56ms/step - loss: 1.9988e-04 - mean_absolute_error: 0.010
4 - val_loss: 0.0040 - val_mean_absolute_error: 0.0531
Epoch 63/80
30/30 [==============================] - 2s 56ms/step - loss: 1.8942e-04 - mean_absolute_error: 0.010
1 - val_loss: 0.0057 - val_mean_absolute_error: 0.0669
Epoch 64/80
30/30 [==============================] - 2s 59ms/step - loss: 1.7222e-04 - mean_absolute_error: 0.009
6 - val_loss: 0.0041 - val_mean_absolute_error: 0.0552
Epoch 65/80
30/30 [==============================] - 2s 56ms/step - loss: 1.7222e-04 - mean_absolute_error: 0.009
5 - val_loss: 0.0039 - val_mean_absolute_error: 0.0521
Epoch 66/80
30/30 [==============================] - 2s 57ms/step - loss: 1.6772e-04 - mean_absolute_error: 0.009
4 - val_loss: 0.0071 - val_mean_absolute_error: 0.0749
Epoch 67/80
30/30 [==============================] - 2s 57ms/step - loss: 1.8372e-04 - mean_absolute_error: 0.010
0 - val_loss: 0.0026 - val_mean_absolute_error: 0.0431
Epoch 68/80
30/30 [==============================] - 2s 55ms/step - loss: 1.7352e-04 - mean_absolute_error: 0.009
6 - val_loss: 0.0041 - val_mean_absolute_error: 0.0555
Epoch 69/80
30/30 [==============================] - 2s 59ms/step - loss: 1.7126e-04 - mean_absolute_error: 0.009
5 - val_loss: 0.0041 - val_mean_absolute_error: 0.0542
Epoch 70/80
30/30 [==============================] - 2s 60ms/step - loss: 1.6959e-04 - mean_absolute_error: 0.009
6 - val_loss: 0.0034 - val_mean_absolute_error: 0.0501
Epoch 71/80
30/30 [==============================] - 2s 59ms/step - loss: 1.6684e-04 - mean_absolute_error: 0.009
5 - val_loss: 0.0033 - val_mean_absolute_error: 0.0477
Epoch 72/80
30/30 [==============================] - 2s 59ms/step - loss: 1.5689e-04 - mean_absolute_error: 0.009
0 - val_loss: 0.0061 - val_mean_absolute_error: 0.0686
Epoch 73/80
30/30 [==============================] - 2s 58ms/step - loss: 1.7516e-04 - mean_absolute_error: 0.009
7 - val_loss: 0.0036 - val_mean_absolute_error: 0.0510
Epoch 74/80
30/30 [==============================] - 2s 59ms/step - loss: 1.6574e-04 - mean_absolute_error: 0.009
4 - val_loss: 0.0044 - val_mean_absolute_error: 0.0574
Epoch 75/80
30/30 [==============================] - 2s 56ms/step - loss: 1.5040e-04 - mean_absolute_error: 0.009
0 - val_loss: 0.0042 - val_mean_absolute_error: 0.0552
Epoch 76/80
30/30 [==============================] - 2s 56ms/step - loss: 1.4898e-04 - mean_absolute_error: 0.008
8 - val_loss: 0.0050 - val_mean_absolute_error: 0.0610
Epoch 77/80
30/30 [==============================] - 2s 58ms/step - loss: 1.7279e-04 - mean_absolute_error: 0.009
6 - val_loss: 0.0026 - val_mean_absolute_error: 0.0422
Epoch 78/80
30/30 [==============================] - 2s 60ms/step - loss: 1.4323e-04 - mean_absolute_error: 0.008
7 - val_loss: 0.0030 - val_mean_absolute_error: 0.0459
Epoch 79/80
30/30 [==============================] - 2s 58ms/step - loss: 1.4696e-04 - mean_absolute_error: 0.008
8 - val_loss: 0.0031 - val_mean_absolute_error: 0.0465
Epoch 80/80
30/30 [==============================] - 2s 59ms/step - loss: 1.4604e-04 - mean_absolute_error: 0.008
7 - val_loss: 0.0028 - val_mean_absolute_error: 0.0440
Out[17]: <keras.callbacks.History at 0x7f5e89c09c90>
```

```
In [18]: test_predicted = model.predict(test_seq)
           test_predicted[:5]
```

```
Out[18]: array([[0.4808577 , 0.4833922 ],
                [0.4878971 , 0.49036047],
                [0.49305838, 0.49545732],
                [0.5048894 , 0.50705373],
                [0.5145035 , 0.516652  ]], dtype=float32)
```

```
In [19]: test_inverse_predicted = MMS.inverse_transform(test_predicted) # Inversing scaling on predicted data
         test_inverse_predicted[:5]
```

```
Out[19]: array([[1562.4717, 1564.1501],
                [1575.5222, 1577.0647],
                [1585.0908, 1586.5109],
                [1607.0247, 1608.003 ],
                [1624.8483, 1625.7919]], dtype=float32)
```

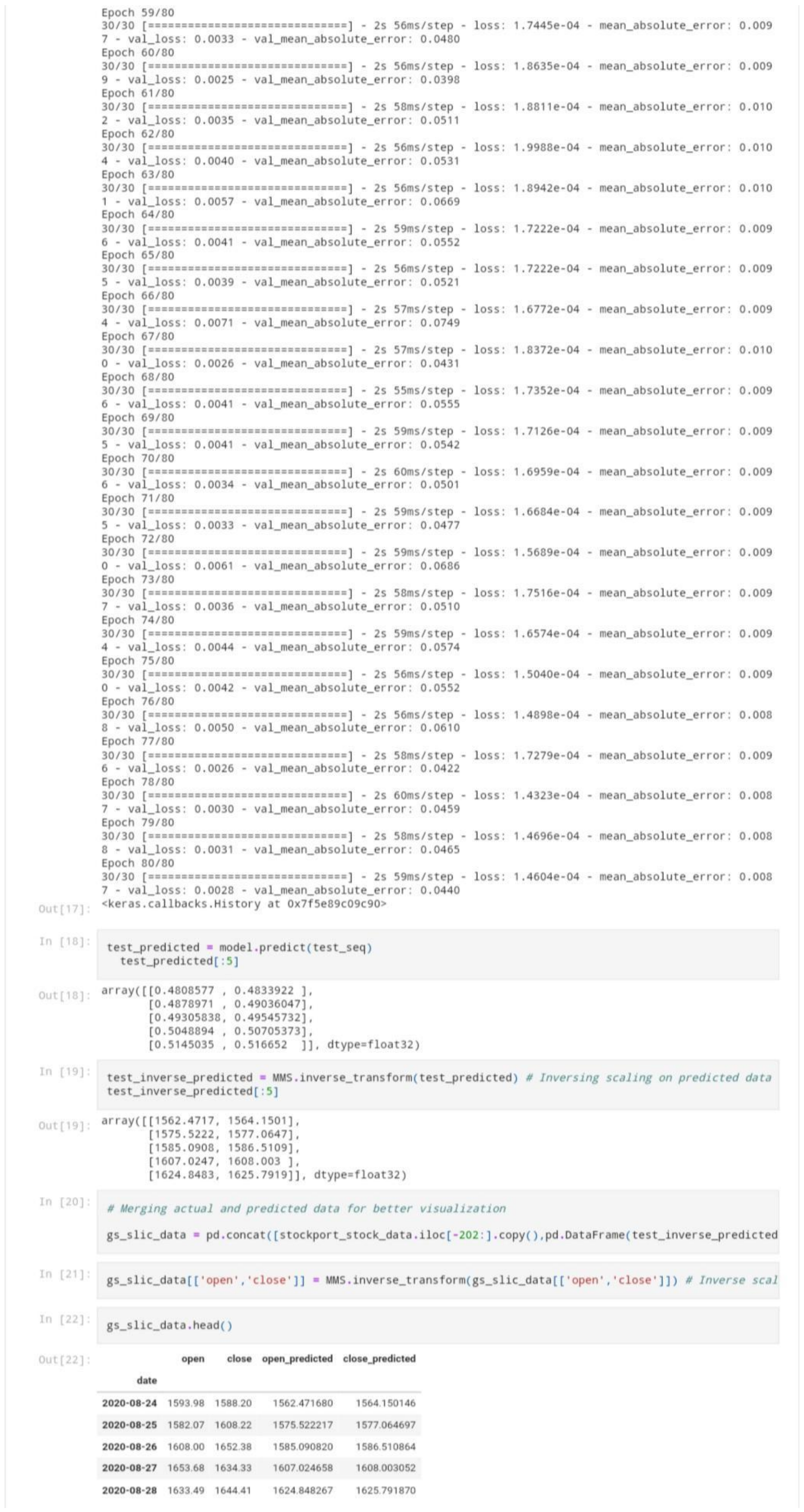
```
In [20]: # Merging actual and predicted data for better visualization

         gs_slic_data = pd.concat([stockport_stock_data.iloc[-202:].copy(),pd.DataFrame(test_inverse_predicted
```

```
In [21]: gs_slic_data[['open','close']] = MMS.inverse_transform(gs_slic_data[['open','close']]) # Inverse scal
```

```
In [22]: gs_slic_data.head()
```

Out[22]:

	open	close	open_predicted	close_predicted
date				
2020-08-24	1593.98	1588.20	1562.471680	1564.150146
2020-08-25	1582.07	1608.22	1575.522217	1577.064697
2020-08-26	1608.00	1652.38	1585.090820	1586.510864
2020-08-27	1653.68	1634.33	1607.024658	1608.003052
2020-08-28	1633.49	1644.41	1624.848267	1625.791870



```
Epoch 59/80
30/30 [==============================] - 2s 56ms/step - loss: 1.7445e-04 - mean_absolute_error: 0.009
7 - val_loss: 0.0033 - val_mean_absolute_error: 0.0480
Epoch 60/80
30/30 [==============================] - 2s 56ms/step - loss: 1.8635e-04 - mean_absolute_error: 0.009
9 - val_loss: 0.0025 - val_mean_absolute_error: 0.0398
Epoch 61/80
30/30 [==============================] - 2s 58ms/step - loss: 1.8811e-04 - mean_absolute_error: 0.010
2 - val_loss: 0.0035 - val_mean_absolute_error: 0.0511
Epoch 62/80
30/30 [==============================] - 2s 56ms/step - loss: 1.9988e-04 - mean_absolute_error: 0.010
4 - val_loss: 0.0040 - val_mean_absolute_error: 0.0531
Epoch 63/80
30/30 [==============================] - 2s 56ms/step - loss: 1.8942e-04 - mean_absolute_error: 0.010
1 - val_loss: 0.0057 - val_mean_absolute_error: 0.0669
Epoch 64/80
30/30 [==============================] - 2s 59ms/step - loss: 1.7222e-04 - mean_absolute_error: 0.009
6 - val_loss: 0.0041 - val_mean_absolute_error: 0.0552
Epoch 65/80
30/30 [==============================] - 2s 56ms/step - loss: 1.7222e-04 - mean_absolute_error: 0.009
5 - val_loss: 0.0039 - val_mean_absolute_error: 0.0521
Epoch 66/80
30/30 [==============================] - 2s 57ms/step - loss: 1.6772e-04 - mean_absolute_error: 0.009
4 - val_loss: 0.0071 - val_mean_absolute_error: 0.0749
Epoch 67/80
30/30 [==============================] - 2s 57ms/step - loss: 1.8372e-04 - mean_absolute_error: 0.010
0 - val_loss: 0.0026 - val_mean_absolute_error: 0.0431
Epoch 68/80
30/30 [==============================] - 2s 55ms/step - loss: 1.7352e-04 - mean_absolute_error: 0.009
6 - val_loss: 0.0041 - val_mean_absolute_error: 0.0555
Epoch 69/80
30/30 [==============================] - 2s 59ms/step - loss: 1.7126e-04 - mean_absolute_error: 0.009
5 - val_loss: 0.0041 - val_mean_absolute_error: 0.0542
Epoch 70/80
30/30 [==============================] - 2s 60ms/step - loss: 1.6959e-04 - mean_absolute_error: 0.009
6 - val_loss: 0.0034 - val_mean_absolute_error: 0.0501
Epoch 71/80
30/30 [==============================] - 2s 59ms/step - loss: 1.6684e-04 - mean_absolute_error: 0.009
5 - val_loss: 0.0033 - val_mean_absolute_error: 0.0477
Epoch 72/80
30/30 [==============================] - 2s 59ms/step - loss: 1.5689e-04 - mean_absolute_error: 0.009
0 - val_loss: 0.0061 - val_mean_absolute_error: 0.0686
Epoch 73/80
30/30 [==============================] - 2s 58ms/step - loss: 1.7516e-04 - mean_absolute_error: 0.009
7 - val_loss: 0.0036 - val_mean_absolute_error: 0.0510
Epoch 74/80
30/30 [==============================] - 2s 59ms/step - loss: 1.6574e-04 - mean_absolute_error: 0.009
4 - val_loss: 0.0044 - val_mean_absolute_error: 0.0574
Epoch 75/80
30/30 [==============================] - 2s 56ms/step - loss: 1.5040e-04 - mean_absolute_error: 0.009
0 - val_loss: 0.0042 - val_mean_absolute_error: 0.0552
Epoch 76/80
30/30 [==============================] - 2s 56ms/step - loss: 1.4898e-04 - mean_absolute_error: 0.008
8 - val_loss: 0.0050 - val_mean_absolute_error: 0.0610
Epoch 77/80
30/30 [==============================] - 2s 58ms/step - loss: 1.7279e-04 - mean_absolute_error: 0.009
6 - val_loss: 0.0026 - val_mean_absolute_error: 0.0422
Epoch 78/80
30/30 [==============================] - 2s 60ms/step - loss: 1.4323e-04 - mean_absolute_error: 0.008
7 - val_loss: 0.0030 - val_mean_absolute_error: 0.0459
Epoch 79/80
30/30 [==============================] - 2s 58ms/step - loss: 1.4696e-04 - mean_absolute_error: 0.008
8 - val_loss: 0.0031 - val_mean_absolute_error: 0.0465
Epoch 80/80
30/30 [==============================] - 2s 59ms/step - loss: 1.4604e-04 - mean_absolute_error: 0.008
7 - val_loss: 0.0028 - val_mean_absolute_error: 0.0440
Out[17]: <keras.callbacks.History at 0x7f5e89c09c90>
```

```
In [18]: test_predicted = model.predict(test_seq)
           test_predicted[:5]
```

```
Out[18]: array([[0.4808577 , 0.4833922 ],
                [0.4878971 , 0.49036047],
                [0.49305838, 0.49545732],
                [0.5048894 , 0.50705373],
                [0.5145035 , 0.516652  ]], dtype=float32)
```

```
In [19]: test_inverse_predicted = MMS.inverse_transform(test_predicted) # Inversing scaling on predicted data
         test_inverse_predicted[:5]
```

```
Out[19]: array([[1562.4717, 1564.1501],
                [1575.5222, 1577.0647],
                [1585.0908, 1586.5109],
                [1607.0247, 1608.003 ],
                [1624.8483, 1625.7919]], dtype=float32)
```

```
In [20]: # Merging actual and predicted data for better visualization

         gs_slic_data = pd.concat([stockport_stock_data.iloc[-202:].copy(),pd.DataFrame(test_inverse_predicted
```

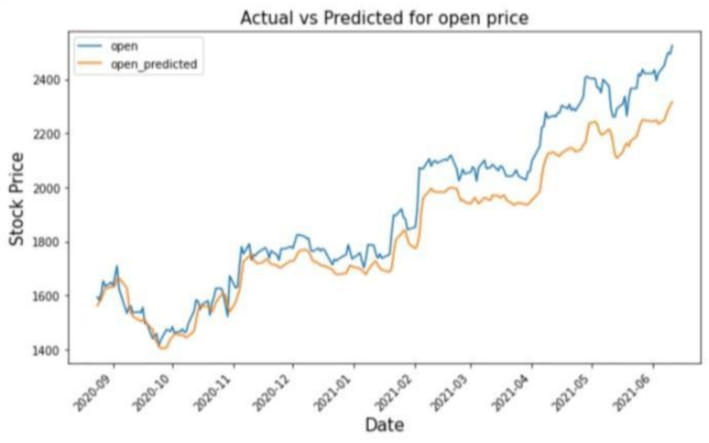
```
In [21]: gs_slic_data[['open','close']] = MMS.inverse_transform(gs_slic_data[['open','close']]) # Inverse scal
```

```
In [22]: gs_slic_data.head()
```

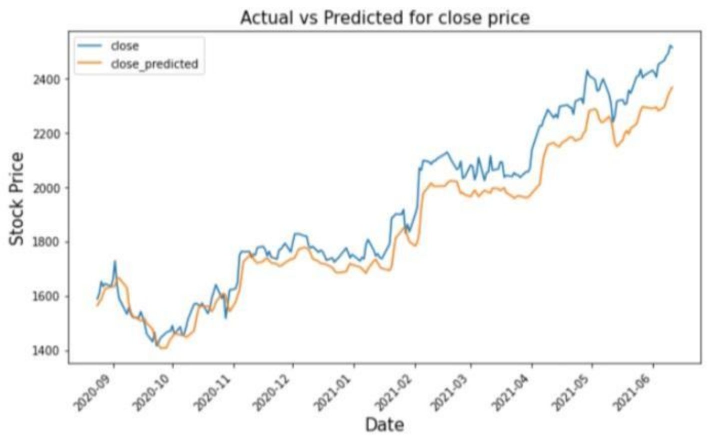
Out[22]:

	open	close	open_predicted	close_predicted
date				
2020-08-24	1593.98	1588.20	1562.471680	1564.150146
2020-08-25	1582.07	1608.22	1575.522217	1577.064697
2020-08-26	1608.00	1652.38	1585.090820	1586.510864
2020-08-27	1653.68	1634.33	1607.024658	1608.003052
2020-08-28	1633.49	1644.41	1624.848267	1625.791870

```
In [23]:
gs_slic_data[['open','open_predicted']].plot(figsize=(10,6))
plt.xticks(rotation=45)
plt.xlabel('Date',size=15)
plt.ylabel('Stock Price',size=15)
plt.title('Actual vs Predicted for open price',size=15)
plt.show()
```



```
In [24]:
gs_slic_data[['close','close_predicted']].plot(figsize=(10,6))
plt.xticks(rotation=45)
plt.xlabel('Date',size=15)
plt.ylabel('Stock Price',size=15)
plt.title('Actual vs Predicted for close price',size=15)
plt.show()
```



```
In [25]:
# Creating a dataframe and adding 10 days to existing index

gs_slic_data = gs_slic_data.append(pd.DataFrame(columns=gs_slic_data.columns,index=pd.date_range(star
```

```
In [26]:
gs_slic_data['2021-06-09          ':'2021-06-16']
```

Out[26]:

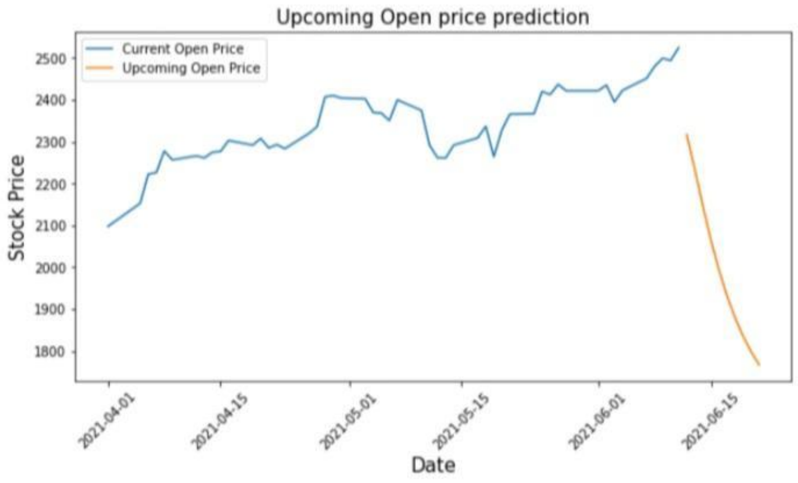
	open	close	open_predicted	close_predicted
2021-06-09	2499.50	2491.40	2289.421143	2338.256836
2021-06-10	2494.01	2521.60	2304.060791	2353.885254
2021-06-11	2524.92	2513.93	2316.642090	2367.072754
2021-06-12	NaN	NaN	NaN	NaN
2021-06-13	NaN	NaN	NaN	NaN
2021-06-14	NaN	NaN	NaN	NaN
2021-06-15	NaN	NaN	NaN	NaN
2021-06-16	NaN	NaN	NaN	NaN

```
In [27]:
upcoming_prediction = pd.DataFrame(columns=['open','close'],index=gs_slic_data.index)
upcoming_prediction.index=pd.to_datetime(upcoming_prediction.index)
```

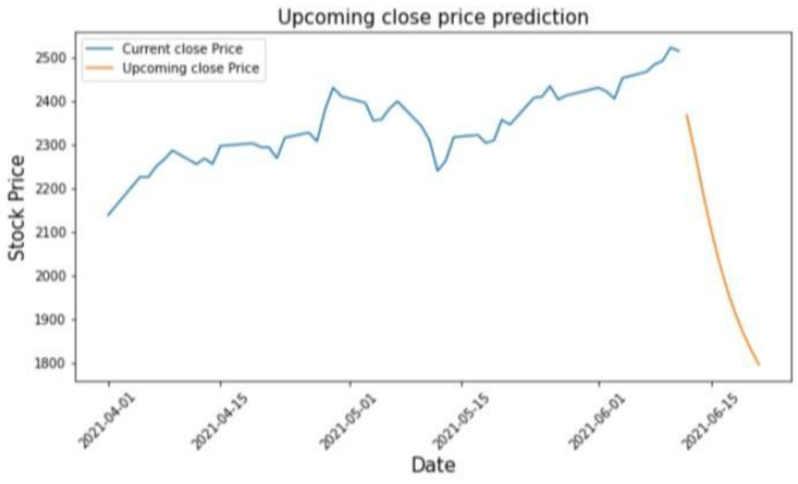
```
In [28]:
curr_seq = test_seq[-1:]

for i in range(-10,0):
  up_pred = model.predict(curr_seq)
  upcoming_prediction.iloc[i] = up_pred
  curr_seq = np.append(curr_seq[0][1:],up_pred,axis=0)
  curr_seq = curr_seq.reshape(test_seq[-1:].shape)
```

```
In [29]:
upcoming_prediction[['open','close']] = MMS.inverse_transform(upcoming_prediction[['open','close']])
```

```
In [30]:
fg,ax=plt.subplots(figsize=(10,5))
ax.plot(gs_slic_data.loc['2021-04-01':,'open'],label='Current Open Price')
ax.plot(upcoming_prediction.loc['2021-04-01':,'open'],label='Upcoming Open Price')
plt.setp(ax.xaxis.get_majorticklabels(), rotation=45)
ax.set_xlabel('Date',size=15)
ax.set_ylabel('Stock Price',size=15)
ax.set_title('Upcoming Open price prediction',size=15)
ax.legend()
fg.show()
```



```
In [31]:
fg,ax=plt.subplots(figsize=(10,5))
ax.plot(gs_slic_data.loc['2021-04-01':,'close'],label='Current close Price')
ax.plot(upcoming_prediction.loc['2021-04-01':,'close'],label='Upcoming close Price')
plt.setp(ax.xaxis.get_majorticklabels(), rotation=45)
ax.set_xlabel('Date',size=15)
ax.set_ylabel('Stock Price',size=15)
ax.set_title('Upcoming close price prediction',size=15)
ax.legend()
fg.show()
```



CHAPTER 9

CONCLUSION

9. CONCLUSION

We have successfully developed a python sentiment analysis model based on LSTM technique that is pretty robust and highly accurate. As discussed earlier, sentiment analysis has many use-cases based on requirements we can use. We can similarly train it on any other kind of data just by changing the dataset according to our needs. We can use this sentiment analysis model in all different ways possible. In this project we have used sentiment analysis to predict stock market

and we have achieved the accuracy of

The package was designed in such a way that future modifications can be done easily. The following conclusions can be deduced from the development of the project:

- Automation of the entire system improves the efficiency
- It provides a friendly graphical user interface which proves to be better when compared to the existing system.
- It gives appropriate access to the authorized users depending on their permissions.
- It effectively overcomes the delay in communications.
- Updating information becomes so easy.
- System security, data security and reliability are the striking features.
- The System has adequate scope for modification in future if it is necessary.

10. REFERENCE

- J. Bollen and H. Mao. Twitter mood as a stock market predictor. IEEE Computer, 44(10):91–94
- A. Lapedes and R. Farber. Nonlinear signal processing using neural network: Prediction and system modeling. In Los Alamos National Lab Technical Report
- Anshul Mittla and Arpit Goel. Stock Prediction Using Twitter Sentiment Analysis