Project Report

On

Stock Price Prediction using LSTM for IT Companies

Submitted for partial fulfillment of requirement for the degree of

BACHELOR OF ENGINEERING

(Computer Science and Engineering)

Submitted By

Rushikesh L. Chaudhari

Ritika G. Belsare

Sahil S. Saundale

Sanjana G. Thakare

Under the Guidance of

Prof. Pratik S. Deshmukh

Department of Computer Science & Engineering,

Prof. Ram Meghe Institute of Technology & Research, Badnera

2022-2023

Department of Computer Science & Engineering
Prof. Ram Meghe Institute of Technology & Research, Badnera
2022 - 2023

CERTIFICATE

This is to certify that the Project (8KS07) entitled

Stock Price Prediction using LSTM for IT Companies

is a bonafide work and it is submitted to the

Sant Gadge Baba Amravati University, Amravati

By

Rushikesh L. Chaudhari
Ritika G. Belsare
Sahil S. Saundale
Sanjana G. Thakare

in the partial fulfillment of the requirement for the degree of ***Bachelor of Engineering*** *in* ***Computer Science & Engineering****, during the academic year* **2022-2023** *under my guidance.*

Prof Pratik S. Deshmukh	***Dr. M. A. Pund***
Guide	*Head,*
Department of Computer Sci.& Engg.	*Department of Computer Sci.& Engg.*
Prof. Ram Meghe Institute Of Technology & Research, Badnera	*Prof. Ram Meghe Institute Of Technology & Research, Badnera*

External Examiner

____

Acknowledgment

With great pleasure we hereby acknowledge the help given to us by various individuals throughout the project. This Project itself is an acknowledgement to the inspiration, drive and technical assistance contributed by many individuals. This project would have never seen the light of this day without the help and guidance we have received.

We would like to express our profound thanks to our Project Guide **Prof. P. S. Deshmukh**, Assistant Professor Department of Computer Science & Engineering for their guidance and constant supervision as well as for providing necessary information regarding the project & also for their support in completing the project. We would also like to express our gratitude towards **Dr. G. R. Bamnote**, Principal Prof. Ram Meghe Institute of Technology and Research-Badnera and **Dr. M. A. Pund**, Head Department of Computer Science and Engineering for encouraging us for our betterment. We owe an incalculable debt to all staffs of the Department of Computer Science & Engineering for their direct and indirect help.

Our thanks and appreciations also go to our colleague in developing the project and people who have willingly helped us out with their abilities.

We extend our heartfelt thanks to our parents, friends and well-wishers for their support and timely help. Last but not the least; we thank the God Almighty for guiding us in every step of the way.

Abstract

Stock market Recommendation is the scene of trying to complete the long-run value of company stock. Analysis of the stock price we take the price. By using the neural network, we develop a model. Within the neural network, we use a recurrent neural network that remembers each and each information through time .In this project we attempt to implement a machine learning approach to predict stock prices. Machine learning is effectively implemented in forecasting stock prices. The objective is to predict the stock prices in order to make more informed and accurate investment decisions. We propose a stock price Recommendation system, machine learning, and other external factors for the purpose of achieving better Stock Recommendation accuracy and issuing profitable trades.
Keywords: Machine Learning, Stock Price Prediction, Long ShortTerm Memory, Stock Market

Contents

List of Figure

List of Screenshot

1 Introduction

The stock price of a company is changed every day based on their exchange of products and raw goods. A company provides the shares to increase the production of their company. To predict the value of the share is not an essay because it will change based on many factors. Many investors try to invest their amount in a company to get profit. Here we have a problem with finding the company share value is going to increase or decrease for that day end. In this case, we are trying to use technology for attaining perfect value. Technology is evolving swiftly. There are many feathers to be acknowledged while we try to build a machine learning model. In a day the stock market has three major values: high, low, open and close. Scrutinize the historical data to predict whether the value is going to increase or decrease. Many machine learning algorithms are used to predict the value but as these types of time forecasting problems, we can apply a Recurrent Neural Network. Stock markets have been studied over and over again to extract useful patterns and predict their movements. Stock market prediction has always had a certain appeal for researchers and financial investors. The reason is that who can beat the market, can gain excess profit. Financial analysts who invest in stock markets usually are not aware of the stock market behavior. If they can predict the future behavior of stock prices, they can act immediately upon it and make profit. The more accurate the system predicts the stock price movement, the more profit one can gain from the prediction model. Stock price trend forecasting based solely on the technical and fundamental data analysis enjoys great popularity. But numeric time series data only contain the event and not the cause why it happened. Textual data such as news articles have richer information, hence exploiting textual information especially in addition to numeric time series data increases the quality of the input and improved predictions are expected from this kind of input rather than only numerical data. Without a doubt, human behaviors are always influenced by their environment. One of the most significant impacts that affect human behavior comes from the mass media or to be more specific, from the news articles.

1.1 Motivation

In the existing system, SVM and Backpropagation Algorithm there is no which won't do dropout process. Because of this, unwanted data has been processed which leads to wastage of time and memory space. The Recommendation of future stock price by SVM and Backpropagation Algorithm is less efficient because of processing unwanted data. The SVM and Backpropagation Algorithm which is used in the existing system is not that effective in handling non-linear data. So, in our proposed future stock price Recommendation is done using LSTM (Long Short Term Memory) which is a higher accurate value for the next day than SVM and Backpropagation Algorithm.ket and earn money.

1.2 EXISTING TECHNOLOGY

In the existing system, SVM and Backpropagation Algorithm there is no which won't do dropout process. Because of this, unwanted data has been processed which leads to wastage of time and memory space. The Recommendation of future stock price by SVM and Backpropagation Algorithm is less efficient because of processing unwanted data. The SVM and Backpropagation Algorithm which is used in the existing system is not that effective in handling non-linear data. So, in our proposed future stock price Recommendation is done using LSTM (Long Short Term Memory) which is a higher accurate value for the next day than SVM and Backpropagation Algorithm.

1.3 PROBLEM STATEMENT

Stock prices are volatile in nature and price depends on various factors. The main aim of this project is to predict stock prices using LSTM. Develop a stock prediction model using LSTM algorithm to forecast future stock prices for a given stock. The model should accurately capture the temporal dependencies, non-linearity, and volatility of stock prices, even with limited data availability. Evaluation of the model should be based on appropriate time-series forecasting metrics. The goal is to provide meaningful insights to investors or traders for making informed decisions.

1.4 OBJECTIVE

1 The objective of stock prediction using LSTM algorithm is to leverage the historical data of stock prices and capture long-term dependencies in sequential data to generate accurate forecasts or insights about future stock prices, trends, volatility, and portfolio optimization.

2 Price prediction: LSTM can be used to forecast the future prices of stocks, which can be helpful for traders and investors in making informed decisions on buying or selling stocks.

1.5 SCOPE OF PROJECT

1 Stock Market- scope of undertaking traits within the stock market and elements affecting the inventory expenses.

2 Data mining strategies- scope of task is in the to be had gear and techniques for information mining and then selecting those that are great in shape to resolve the trouble

1.6 FEASIBILITY STUDY

Significance of our venture or want of the undertaking is all about the accuracy in Recommendation of the inventory values. Stock marketplace Recommendation pursuits to determine the future movement of the inventory value of an economic change. The correct Recommendation of percentage charge motion will result in more income traders could make. The Accuracy is finished via distinctive techniques like Using ANN, SVM etc. But we used the LSTM model , because there higher accuracy than different approach

2 LITERATURE REVIEW

The current stock price of a publicly traded company reflects the company's past operation, current timing and future profitability prospects. To obtain a more accurate projection of the stock price, several types of studies have already been carried out. The most classic ones focus on the financial data of the target companies, added to micro and macroeconomic aspects. However, the non-linear and non-stationary features of financial data sequences make predictions more challenging [11]. Some recurrent neural networks—RNN performed better in predicting financial data and became popular, such as Long Short-Term Memory [23]. LSTM (Long Short-Term Memory) is a popular deep learning algorithm that has shown promising results in the field of stock prediction. In recent years, many researchers have used LSTM to develop models for predicting stock prices. In this literature review, we will discuss some of the notable research works on stock prediction using LSTM[23]. focuses on the effect of the indices in the stock price prediction. The model identifies the variables and relationship between the indices and overcomes the limitations of the traditional linear model and uses LSTM to understand the dynamics of the S&P 500 Index. The paper also analyses the sensitivity of internal memory of LSTM modelling. However, the study has some limitations, the difference between the predictive value and actual value becomes large after a certain point and thus cannot be used to develop a system to give a profitable trading strategy.

Proposes a system that would recommend stock purchases to the buyers. The approach opted by the authors combines the prediction from historical and real-time data using LSTM for predicting. In the RNN model, latest trading data and technical indicators are given as input in the first layer, followed by the LSTM, a compact layer and finally the output layer gives the predicted value. These predicted values are further integrated with the summarized data which is collected from the news analytics to generate a report showing the percentage in change. these studies suggest that LSTM can be a powerful algorithm for stock prediction, achieving high levels of prediction accuracy on a range of stock market indices. However, it's important to note that stock prediction is inherently challenging and requires careful feature selection, model tuning, and risk management to minimize potential losses.

"Stock price prediction using LSTM, RNN and CNN-SVM model" by H. Yu et al. (2019): This study compared the performance of LSTM, RNN, and CNN-SVM models for stock price prediction on the Shanghai Stock Exchange Composite Index. The authors found that the LSTM model outperformed the other models, achieving a prediction accuracy of 61.73

"Stock price prediction using LSTM and MLP with Bayesian optimization" by Y. Xu et al. (2019): This study proposed a hybrid LSTM-MLP model for stock price prediction on the S&P 500 index. The authors used Bayesian optimization to tune the hyperparameters of the model and achieved a prediction accuracy of 65.24

"Stock price prediction using LSTM and attention-based feature fusion" by J. Zhou et al. (2020): This study proposed a novel attention-based feature fusion approach for stock price prediction using LSTM. The authors used a combination of technical indicators and news sentiment as input features and achieved a prediction accuracy of 61.15

”An LSTM model with attention mechanism for stock price prediction” by S. Liu et al. (2020): This study proposed an LSTM model with attention mechanism for stock price prediction on the S&P 500 index. The authors found that the attention mechanism improved the performance of the model, achieving a prediction accuracy of 64.19

”An empirical comparison of LSTM, ARIMA, and random walk models for stock price prediction” by C. Wang et al. (2019): This study compared the performance of LSTM, ARIMA, and random walk models for stock price prediction on the S&P 500 index. The authors found that the LSTM model outperformed the other models, achieving a prediction accuracy of 59.82Stock prediction is a challenging task due to its complex and dynamic nature. However, the Long Short-Term Memory (LSTM) algorithm has shown promising results in stock prediction. In this literature review, we summarize some of the recent studies that have used LSTM for stock prediction.

”Stock price prediction using LSTM, RNN and CNN-sliding window model” by Zhang et al. (2021) The authors proposed a hybrid model that combines LSTM, RNN, and CNN to predict stock prices. The model uses a sliding window approach to capture the temporal dependencies in the data. The study showed that the proposed model outperforms other models in terms of prediction accuracy. ”A Deep Learning Model for Stock Price Prediction Using LSTM Neural Networks” by Ferreira et al. (2021) The authors proposed a deep learning model that uses LSTM neural networks for stock price prediction. The model is trained on historical stock price data and uses technical indicators as input features. The study showed that the LSTM-based model outperforms traditional time series models like ARIMA in terms of prediction accuracy.

”Predicting the Stock Market Using LSTM Recurrent Neural Networks” by Fischer et al. (2020). The authors proposed an LSTM-based model that predicts stock prices based on news sentiment analysis. The study showed that incorporating news sentiment data into the LSTM model improves the prediction accuracy. ”Stock price prediction using LSTM-based hybrid approach with feature selection and parameter optimization” by Singh et al. (2020) The authors proposed a hybrid approach that combines LSTM with feature selection and parameter optimization techniques to predict stock prices. The study showed that the proposed model outperforms other models in terms of prediction accuracy. ”Stock Price Prediction Using LSTM and Attention Mechanism” by Gao et al. (2019) The authors proposed an LSTM-based model that uses an attention mechanism to improve the prediction accuracy. The study showed that incorporating attention mechanism into the LSTM model improves the model’s ability to capture important features in the data.

Overall, the reviewed literature suggests that LSTM-based models have shown promising results in stock prediction. The studies have used various approaches such as hybrid models, incorporating news sentiment data, and attention mechanisms to improve the prediction accuracy of the LSTM models. The future research could focus on addressing the interpretability and explainability of LSTM models and exploring the integration of LSTM models with other advanced technologies such as NLP and machine vision.

AUTHER	PROBLEM	SOLUTION
Manoj S Hegde, Ganesh Krishna,Dr. RamamoorthySrinath.[17]	In this comparison of algorithms like ANN,LSTM,RNN will be done based on the maximum stock price and classification	LSTM, RNN algorithm
Ryo akita, Akira yoshihara, Takashi matsubara, Kuniaki uehara. [18]	In this time series fore casting will be taken place and LSTM classification problem will also raised.	Time analysis will be take and single company dataset will be taken
Saurav Kumar and Dhruba Ningombam [19]	Here RNN with LSTM will be taken and sentiment analysis will be taken and over ridding will taken place.	Linear scaling will be done.
Ashwin S. Ravi,Akshay Sarvesh, Koshy George [21].	Time series prediction using Sequential algorithm and and minimizing the oil & gas exchange of Bombay stock.	Sequential algorithm and time stamp analysis.
Ashwin S. ravi, Akshay sarvesh and Koshygeoe[22].	Time complexity two types of NN network based algorithms.	ANN, Multiple models, Online sequential learning.

3 RESEARCH METHODOLOGY

3.1 PROPOSED SYSTEM

We propose to use the LSTM (Long Short Term Memory) algorithm for stock price prediction.

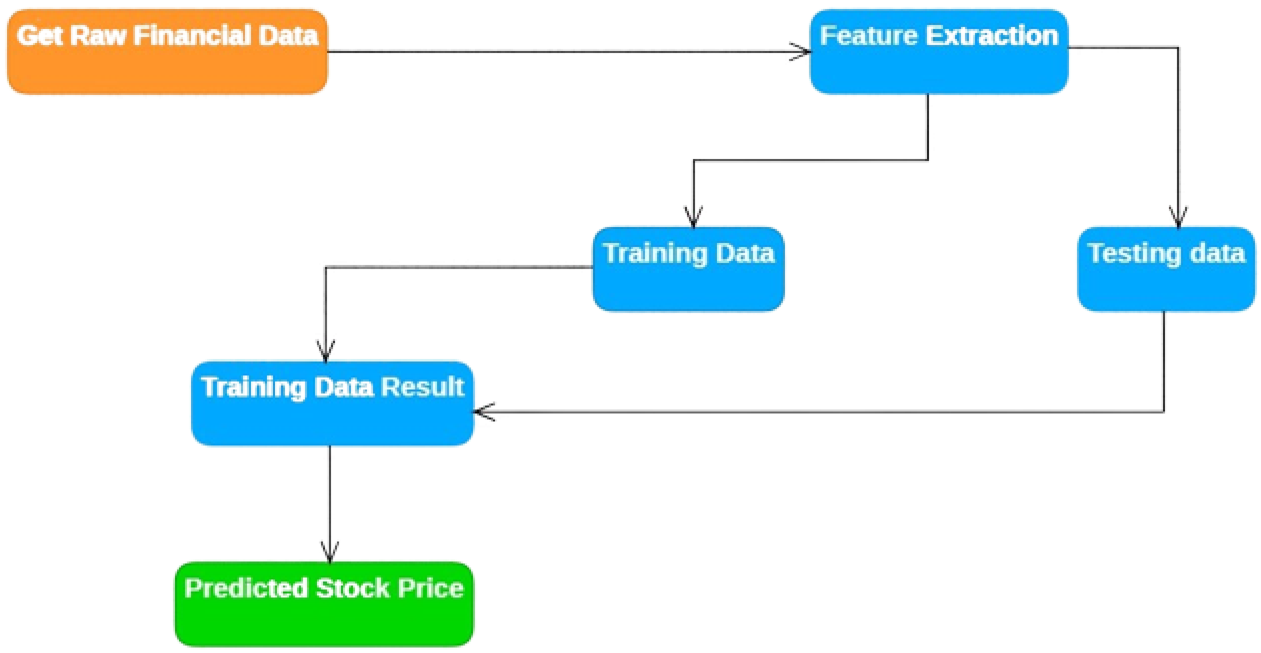


Figure 3.1: Proposed System

3.1.1 Long Short Term Memory

Long Short-Term Memory (LSTM) is one type of recurrent neural network which is used to learn order dependence in sequence prediction problems. Due to its capability of storing past information, LSTM is very useful in predicting stock prices. This is because the prediction of a future stock price is dependent on the previous prices.

LSTM, which stands for Long Short Term Memory, is a type of neural network which is particularly useful in the case of time series forecasting. According to an article by Srivastava on LSTM's and essentials of deep learning, an LSTM network is the most effective solution to time series analysis and thus stock market prediction. With the recent breakthroughs that have been happening in data science, it is found that for almost all of these sequence prediction problems, long short Term Memory networks have been observed as the most effective solution. LSTMs have an edge over conventional feed-forward neural networks and Recurrent Neural Networks in many ways. This is because of their property of selectively remembering patterns for long durations of time

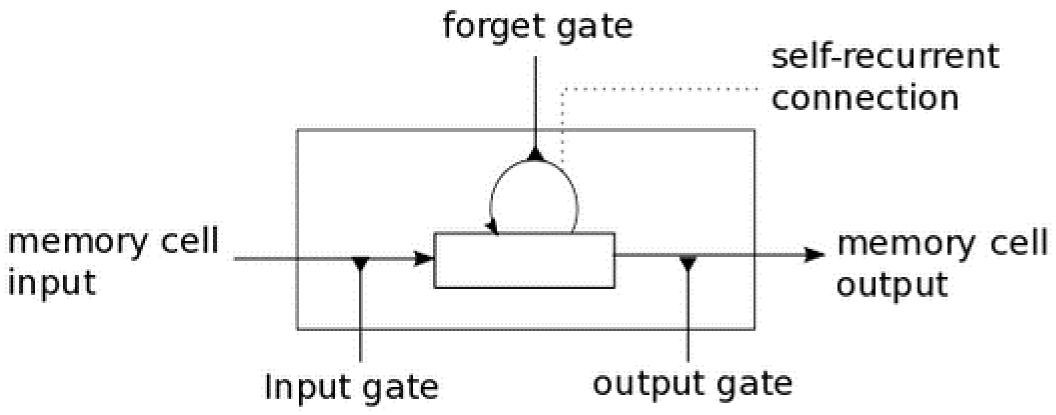


Figure 3.2: LSTM Cell

In the case of a basic neural network, in order to add new information, it transforms the existing information completely by applying a sigmoid function. Because of this, the entire information is modified as a whole. Thus, there is no consideration for 'important' information and 'not so important' information. LSTMs on the other hand, make small modifications to the information by multiplications and additions. With LSTMs, the information flows through a mechanism known as cell states. This way, LSTMs can selectively remember or forget things. The following figure, represents a more detailed view at the internal architecture of an LSTM network.

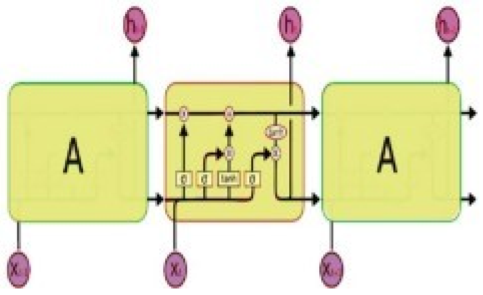
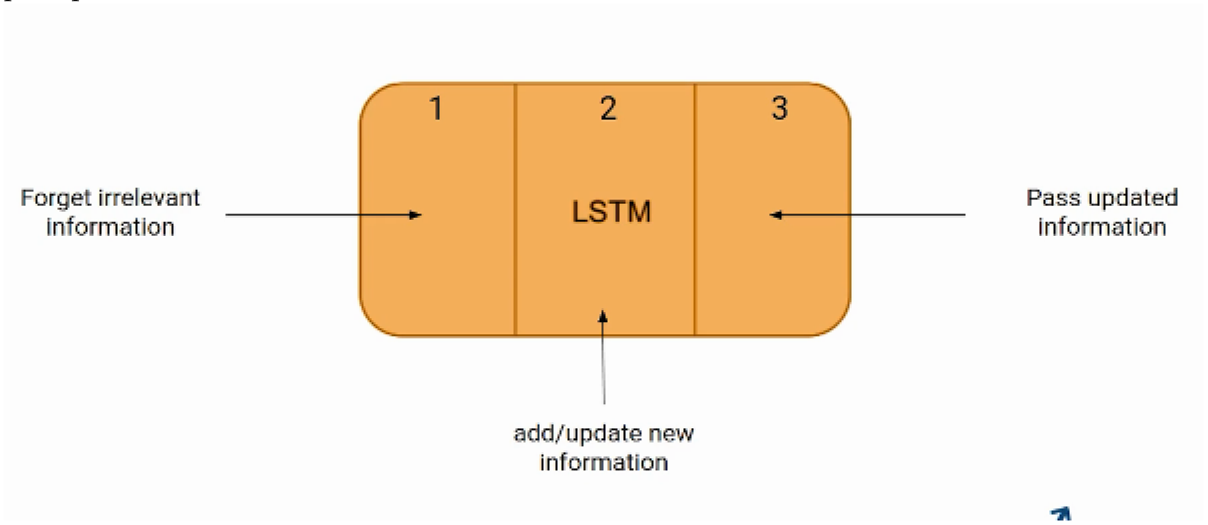


Figure 3.3: LSTM Cell Architecture

A typical LSTM network is comprised of different memory blocks called cells. There are two states that are being transferred to the next cell; the cell state and the hidden state. The memory blocks are responsible for remembering things, and manipulations to this memory is done through three major mechanisms called gates.

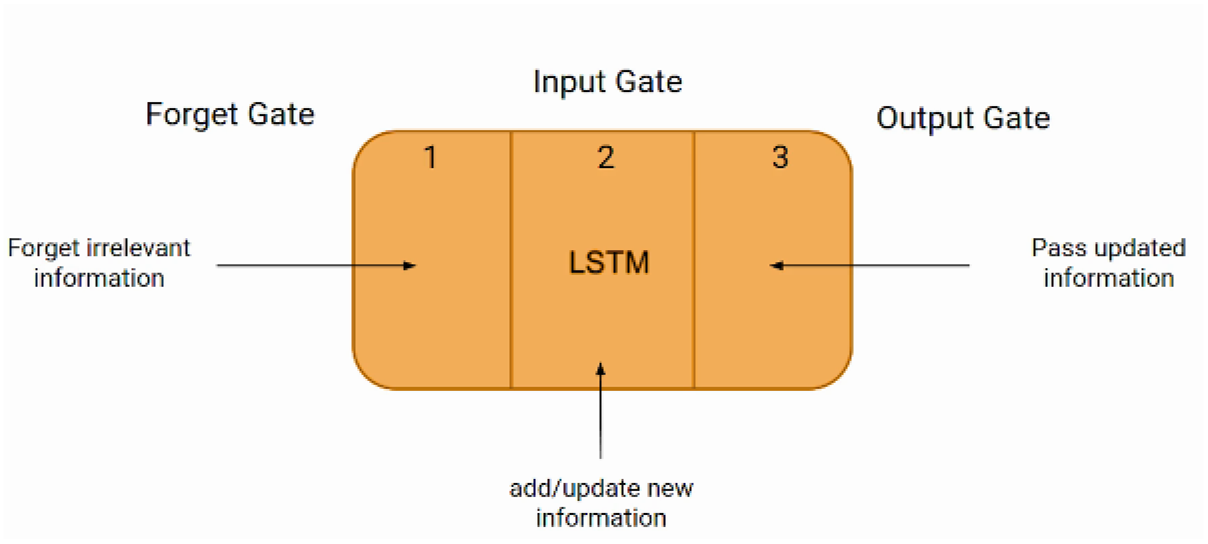
Step By Step Algorithm

There are several steps while building the LSTM model. In the introduction to long short-term memory, we learned that it resolves the vanishing gradient problem faced by RNN, so now, in this section, we will see how it resolves this problem by learning the architecture of the LSTM. At a high level, LSTM works very much like an RNN cell. Here is the internal functioning of the LSTM network. The LSTM network architecture consists of three parts, as shown in the image below, and each part performs an individual function.

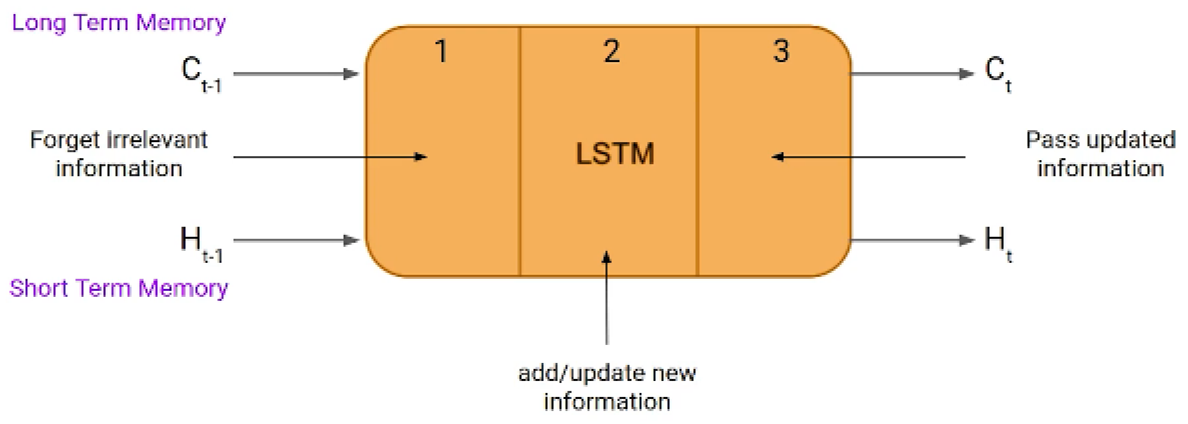


The first part chooses whether the information coming from the previous timestamp is to be remembered or is irrelevant and can be forgotten. In the second part, the cell tries to learn new information from the input to this cell. At last, in the third part, the cell passes the updated information from the current timestamp to the next timestamp. This one cycle of LSTM is considered a single-time step.

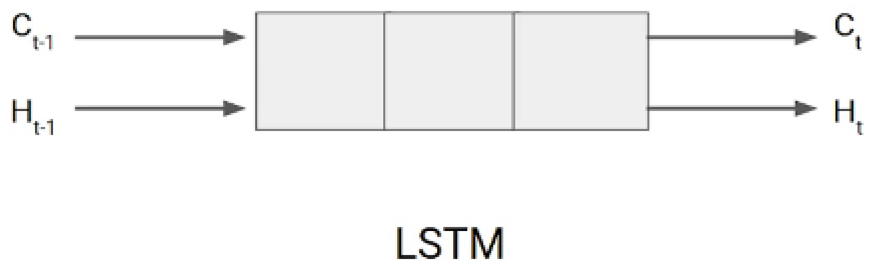
These three parts of an LSTM unit are known as gates. They control the flow of information in and out of the memory cell or lstm cell. The first gate is called Forget gate, the second gate is known as the Input gate, and the last one is the Output gate. An LSTM unit that consists of these three gates and a memory cell or lstm cell can be considered as a layer of neurons in traditional feedforward neural network, with each neuron having a hidden layer and a current state.



Just like a simple RNN, an LSTM also has a hidden state where H(t-1) represents the hidden state of the previous timestamp and Ht is the hidden state of the current timestamp. In addition to that, LSTM also has a cell state represented by C(t-1) and C(t) for the previous and current timestamps, respectively.
Here the hidden state is known as Short term memory, and the cell state is known as Long term memory. Refer to the following image.



It is interesting to note that the cell state carries the information along with all the timestamps.



Bob is a nice person. Dan on the other hand is evil.

Let’s take an example to understand how LSTM works. Here we have two sentences separated by a full stop. The first sentence is “Bob is a nice person,” and the second sentence is “Dan, on the Other hand, is evil”. It is very clear, in the first sentence, we are talking about Bob, and as soon as we encounter the full stop(.), we started talking about Dan.

As we move from the first sentence to the second sentence, our network should realize that we are no more talking about Bob. Now our subject is Dan. Here, the Forget gate of the network allows it to forget about it. Let’s understand the roles played by these gates in LSTM architecture.

Forget

In a cell of the LSTM neural network, the first step is to decide whether we should keep the information from the previous time step or forget it. Here is the equation for forget gate.
forget everything

$$f_t = \sigma(X_t * U_f + H_{t-1} * W_f) \tag{3.1}$$

Let’s try to understand the equation, here
Xt: input to the current timestamp.
Uf: weight associated with the input
Ht-1: The hidden state of the previous timestamp
Wf: It is the weight matrix associated with the hidden state

Later, a sigmoid function is applied to it. That will make ft a number between 0 and 1. This ft is later multiplied with the cell state of the previous timestamp, as shown below.
forget nothing

$$C_{t-1} * f_t = C_{t-1} \tag{3.2}$$

Input Gate

Let's take another example. "Bob knows swimming. He told me over the phone that he had served the navy for four long years."

So, in both these sentences, we are talking about Bob. However, both give different kinds of information about Bob. In the first sentence, we get the information that he knows swimming. Whereas the second sentence tells, he uses the phone and served in the navy for four years.

Now just think about it, based on the context given in the first sentence, which information in the second sentence is critical? First, he used the phone to tell, or he served in the navy. In this context, it doesn't matter whether he used the phone or any other medium of communication to pass on the information. The fact that he was in the navy is important information, and this is something we want our model to remember for future computation. This is the task of the Input gate.

The input gate is used to quantify the importance of the new information carried by the input. Here is the equation of the input gate.
Input Gate

$$i_t = \sigma(X_t * U_i + H_{t-1} * W_i) \tag{3.3}$$

Here,
Xt: Input at the current timestamp t
Ui: weight matrix of input
Ht-1: A hidden state at the previous timestamp
Wi: Weight matrix of input associated with hidden state
Again we have applied the sigmoid function over it. As a result, the value of I at timestamp t will be between 0 and 1.
new Information

$$N_t = tanh(X_t * U_c + H_{t-1} * W_c) \tag{3.4}$$

Now the new information that needed to be passed to the cell state is a function of a hidden state at the previous timestamp t-1 and input x at timestamp t. The activation function here is tanh. Due to the tanh function, the value of new information will be between -1 and 1. If the value of Nt is negative, the information is subtracted from the cell state, and if the value is positive, the information is added to the cell state at the current timestamp.
However, the Nt won't be added directly to the cell state. Here comes the updated equation
Updating cell state

$$C_t = f_t * C_{t-1} + i_t * N_t \tag{3.5}$$

Here, Ct-1 is the cell state at the current timestamp, and the others are the values we have calculated previously.
Output Gate

Now consider this sentence. "Bob single-handedly fought the enemy and died for his country. For his contributions, brave__."

During this task, we have to complete the second sentence. Now, the minute we see the word brave, we know that we are talking about a person. In the sentence, only Bob is brave, we can not say the enemy is brave, or the country is brave. So based on the current expectation, we have to give a relevant word to fill in the blank. That word is our output, and this is the function of our Output gate. Here is the equation of the Output gate, which is pretty similar to the two previous gates. **Output Gate**

$$o_t = \sigma(X_t * U_o + H_{t-1} * W_o) \tag{3.6}$$

Its value will also lie between 0 and 1 because of this sigmoid function. Now to calculate the current hidden state, we will use Ot and tanh of the updated cell state. As shown below.

$$H_t = o_t * tanh(C_t) \tag{3.7}$$

It turns out that the hidden state is a function of Long term memory (Ct) and the current output. If you need to take the output of the current timestamp, just apply the SoftMax activation on hidden state Ht.

$$Output = Softmax(H_t) \tag{3.8}$$

Here the token with the maximum score in the output is the prediction.
This is the More intuitive diagram of the LSTM network. **Building the LSTM**

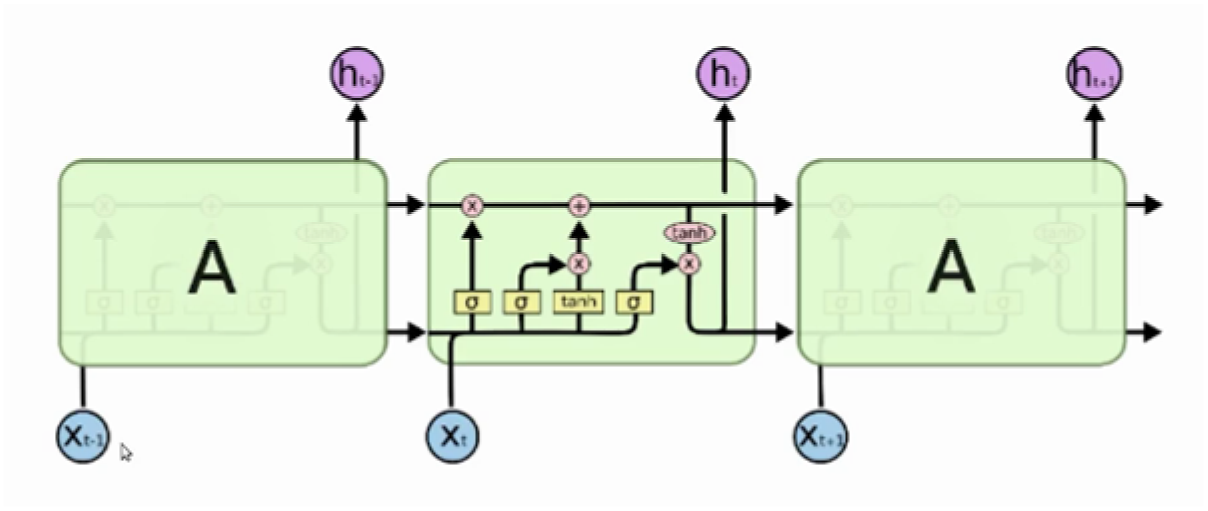


Figure 3.4: LSTM Network

model Follow several steps
In order to build the LSTM, we need to import a couple of modules from Keras:

1 Sequential for initializing the neural network

2 Dense for adding a densely connected neural network layer

3 LSTM for adding the Long Short-Term Memory layer

4 Dropout for adding dropout layers that prevent overfitting.

3.1.2 EDA -Exploratory data analysis

EDA is an approach to analyzing the data using visual techniques. It is used to discover trends, and patterns, or to check assumptions with the help of statistical summaries and graphical representations.

While performing the EDA of the TCS Stock Price data we will analyze how prices of the stock have moved over the period of time and how the end of the quarters affects the prices of the stock. EDA, or to check assumptions with the help of statistical summaries and graphical representations.

While performing the EDA of the TCS Stock Price data we will analyze how prices of the stock have moved over the period of time and how the end of the quarters affects the prices of the stock.

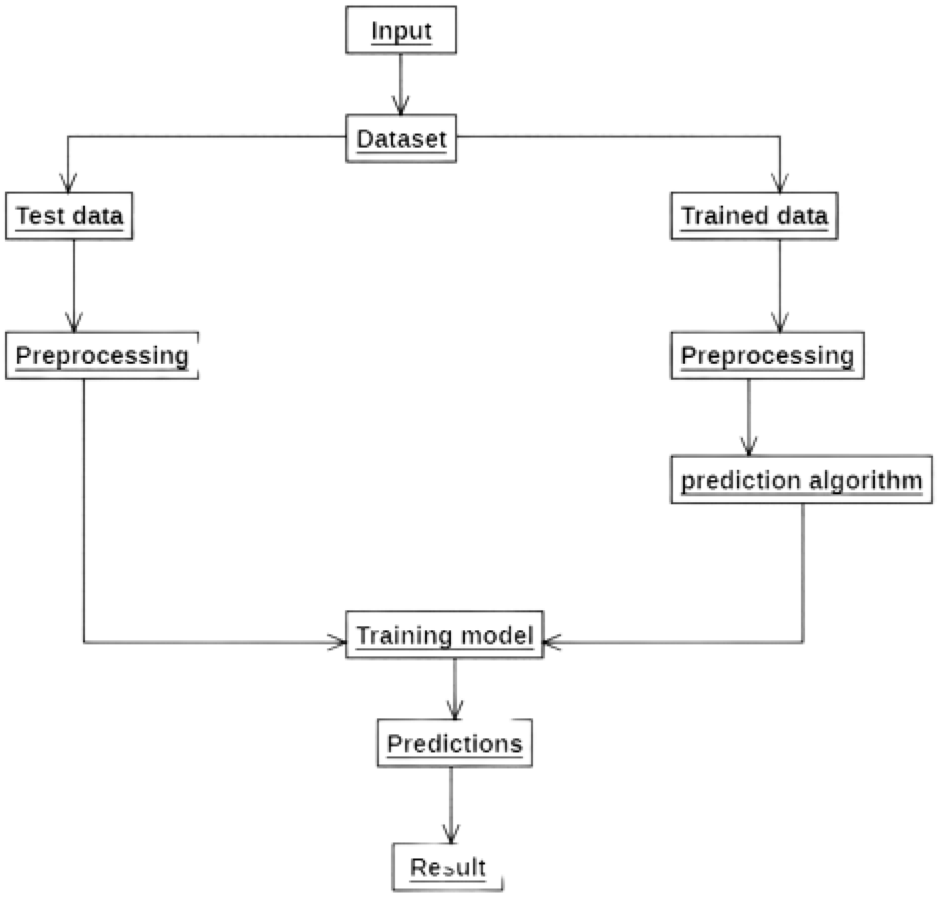


Figure 3.5: Architecture of Deep learning model

3.2 Dataset information

We here used the dataset from the yahoo website in the CSV format it is upto the 1000+ . The dataset also contains a date-wise price of stock with open, close, high, and low prices along with volume traded as well as turnover on that day.
The obtained data contained five features:

1 Date: Date of stock price.

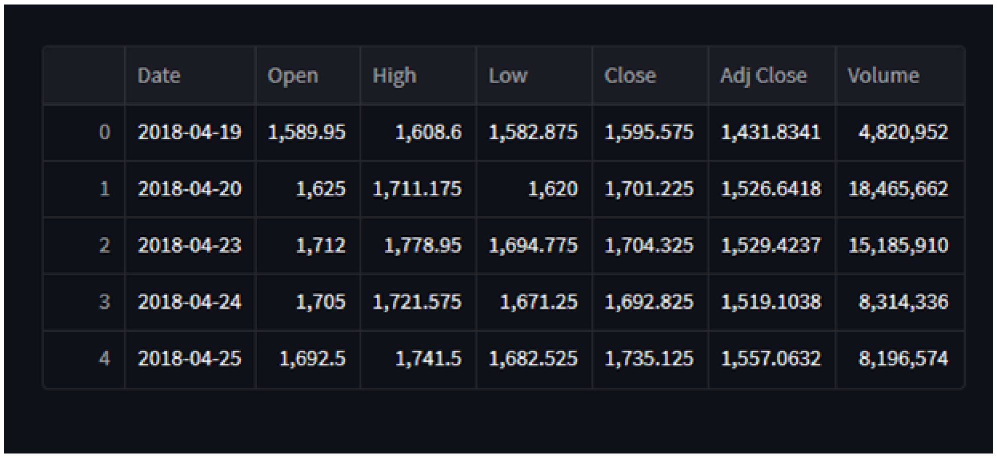
2 Opening price: When trading begins each day this is the opening price of stock.

3 High: The highest price at which the stock was traded during a period(day).

4 Low: The Lowest price at which the stock was traded during a period(day).

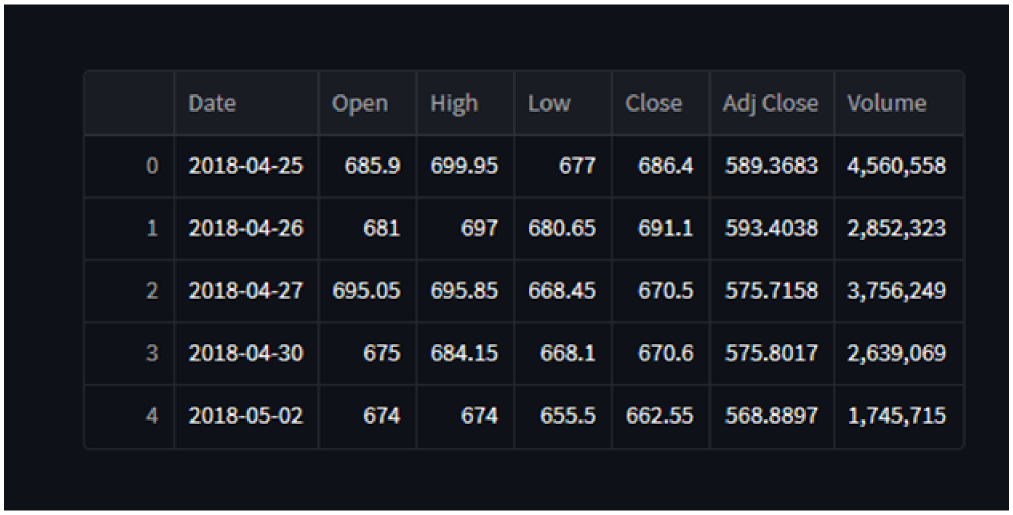
5 Volume: How much of a given financial asset has traded in a period of time.

6 Close Interest: The last price at which a particular stock traded for the trading session



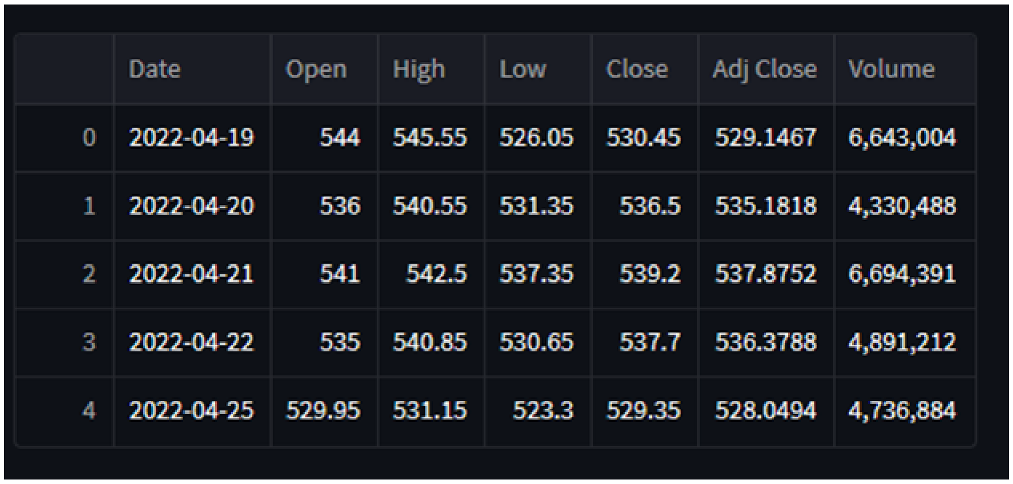
	Date	Open	High	Low	Close	Adj Close	Volume
0	2018-04-19	1,589.95	1,608.6	1,582.875	1,595.575	1,431.8341	4,820,952
1	2018-04-20	1,625	1,711.175	1,620	1,701.225	1,526.6418	18,465,662
2	2018-04-23	1,712	1,778.95	1,694.775	1,704.325	1,529.4237	15,185,910
3	2018-04-24	1,705	1,721.575	1,671.25	1,692.825	1,519.1038	8,314,336
4	2018-04-25	1,692.5	1,741.5	1,682.525	1,735.125	1,557.0632	8,196,574

Figure 3.6: TCS Dataset



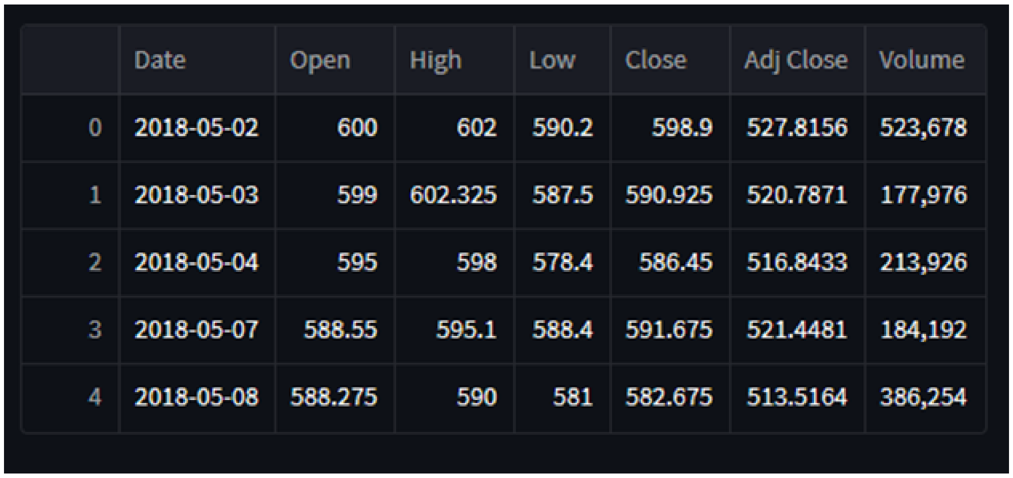
	Date	Open	High	Low	Close	Adj Close	Volume
0	2018-04-25	685.9	699.95	677	686.4	589.3683	4,560,558
1	2018-04-26	681	697	680.65	691.1	593.4038	2,852,323
2	2018-04-27	695.05	695.85	668.45	670.5	575.7158	3,756,249
3	2018-04-30	675	684.15	668.1	670.6	575.8017	2,639,069
4	2018-05-02	674	674	655.5	662.55	568.8897	1,745,715

Figure 3.7: TechMahindra (TechMeh)dataset



	Date	Open	High	Low	Close	Adj Close	Volume
0	2022-04-19	544	545.55	526.05	530.45	529.1467	6,643,004
1	2022-04-20	536	540.55	531.35	536.5	535.1818	4,330,488
2	2022-04-21	541	542.5	537.35	539.2	537.8752	6,694,391
3	2022-04-22	535	540.85	530.65	537.7	536.3788	4,891,212
4	2022-04-25	529.95	531.15	523.3	529.35	528.0494	4,736,884

Figure 3.8: Wipro dataset



	Date	Open	High	Low	Close	Adj Close	Volume
0	2018-05-02	600	602	590.2	598.9	527.8156	523,678
1	2018-05-03	599	602.325	587.5	590.925	520.7871	177,976
2	2018-05-04	595	598	578.4	586.45	516.8433	213,926
3	2018-05-07	588.55	595.1	588.4	591.675	521.4481	184,192
4	2018-05-08	588.275	590	581	582.675	513.5164	386,254

Figure 3.9: Infosys (Infy) dataset

4 SYSTEM REQUIREMENT

4.1 Hardware Requirement

4.1.1 CPU I3 processor

A central processing unit (CPU), also called a central processor, main processor or just processor, is the electronic circuitry that executes instructions comprising a computer program. The CPU performs basic arithmetic, logic, controlling, and input/output (I/O) operations specified by the instructions in the program. This contrasts with external components such as main memory and I/O circuitry, and specialized processors such as graphics processing units (GPUs).

4.1.2 RAM

Random-access memory (RAM; /ræm/) is a form of computer memory that can be read and changed in any order, typically used to store working data and machine code. A random-access memory device allows data items to be read or written in almost the same amount of time irrespective of the physical location of data inside the memory, in contrast with other direct-access data storage media (such as hard disks, CD-RWs, DVD-RWs and the older magnetic tapes and drum memory), where the time required to read and write data items varies significantly depending on their physical locations on the recording medium, due to mechanical limitations such as media rotation speeds and arm movement.

4.1.3 ROM

Read-only memory (ROM) is a type of non-volatile memory used in computers and other electronic devices. Data stored in ROM cannot be electronically modified after the manufacture of the memory device. Read-only memory is useful for storing software that is rarely changed during the life of the system, also known as firmware. Software applications (like video games) for programmable devices can be distributed as plug-in cartridges containing ROM.Python Software 3.8 version

Python is a popular programming language. It was created by Guido van Rossum, and released in 1991.

4.2 Application

1 Python can be used on a server to create web applications.

2 Python can be used alongside software to create workflows.

3 Python can connect to database systems. It can also read and modify files.

4 Python can be used to handle big data and perform complex mathematics.

5 Python can be used for rapid prototyping, or for production-ready software development

4.3 Features

1 Python works on different platforms (Windows, Mac, Linux, Raspberry Pi, etc).

2 Python has a simple syntax similar to the English language.

3 Python has syntax that allows developers to write programs with fewer lines than some other programming languages.

4 Python runs on an interpreter system, meaning that code can be executed as soon as it is written. This means that prototyping can be very quick.

5 Python can be treated in a procedural way, an object-oriented way or a functional way.

4.4 Modification

1 The most recent major version of Python is Python 3, which we shall be using in this tutorial. However, Python 2, although not being updated with anything other than security updates, is still quite popular.

2 In this tutorial Python will be written in a text editor. It is possible to write Python in an Integrated Development Environment, such as Thonny, Pycharm, Netbeans or Eclipse which are particularly useful when managing large collections of Python files.

4.5 Advantages

1 It helps you to invest wisely to make good profits.

2 Probability of high returns over the short-term- The biggest advantage of share market investment is that it has the potential to generate inflation-beating returns within a short period of time as compared to other investment avenues like bank FDs, saving accounts etc.

3 Ownership stake in the company.-When you buy shares of a public listed company, no matter how small your share size is, it gives you proportionate control over the company. This ownership of shares will in turn grant you the voting rights and you will receive dividends, bonus, etc.

4 High liquidity -Unlike other investment avenues, shares do not have any lock-in period. Investors can buy and sell shares through the stock exchanges within seconds.

5 Your rights are well protected by SEBI- The stock market is regulated by the Securities and Exchange Board of India (SEBI). SEBI strictly monitors market participants like brokers, sub-brokers, advisors and stock exchanges to safeguard the interest of the shareholders.

4.6 Disadvantages

1 VolatilityInvestments in the share market are considered risky since the markets are volatile and shares can fluctuate and even hit lower circuits.

2 Risk Risk is the possibility of an investor experiencing losses due to factors that affect the overall performance of the financial markets. Risks are of two types:

3 Systematic risk Systematic risk tends to influence the overall market and it cannot be eliminated through diversification. Example: Natural calamities, political turmoil, terrorist attacks, etc.

4 Unsystematic risk -Unsystematic risk is unique to a specific industry or a company and it can be diversified.

5 Stockholders are paid last. -When a company is winded up, shareholders are the last one to get paid whereas bondholders and creditors of the company get paid first.

6 Emotional Roller Coaster -The stock prices rise and fall frequently due to volatility. Many investors tend to buy a share at a high price out of greed and sell at a low price out of fear. Hence, coffee- can investing is the best strategy to avoid roller coaster investing.

5 IMPLEMENTION & RESULT

5.1 Data Preprocessing

1 Load the historical stock price data for the chosen stock(s) using Pandas DataReader.

2 Preprocess the data using Pandas functions, such as fillna, dropna, and resample, to ensure that the data is clean and consistent.

3 Scale the data using NumPy functions, such as min and max, to prevent vanishing gradients in the LSTM model.

5.2 Splitting the Data

Split the data into training and testing sets using Pandas functions, such as train_test_split, to evaluate the performance of the LSTM model. **Creating a LSTM Model:**

1 Build a LSTM model using Keras library. The architecture of LSTM consists of an input layer, one or more LSTM layers, and an output layer. You can add dropout layers to avoid overfitting.

2 Choose an optimizer such as Adam, RMSprop, or SGD to compile the model.

3 Fit the model to the training data and evaluate its performance on the testing data.

5.3 Making Predictions

1 Use the trained model to make predictions on the testing data.

2 Unscale the predicted values to obtain the actual stock prices.

3 Visualize the predicted stock prices against the actual stock prices using Matplotlib.

This is just a general workflow for implementing an LSTM model for stock prediction. You may need to experiment with different parameters, such as the number of LSTM layers, the size of the input sequence, the learning rate, and the number of epochs, to obtain the best results for your specific dataset. Additionally, you may want to consider adding other features such as news sentiment analysis to improve the accuracy of the model.

5.4 Implementation

The code you provided reads a CSV file named 'tc.csv' into a Pandas DataFrame called 'df', and then displays the first five rows of the DataFrame using the 'head()' method drops two columns named 'Date' and 'Adj Close' from the DataFrame 'df' using the drop() method, and then displays the first five rows of the resulting DataFrame using the head() method.plots a line chart of the 'Close' column in the DataFrame 'df' using Matplotlib's plot() function.

100MA

Calculates the 100-day moving average of the 'Close' column in the DataFrame 'df' using the rolling() and mean() methods in Pandas. creates a line chart of the 'Close' column in the DataFrame 'df' and overlays it with a line chart of the 100-day moving average of the 'Close' column. The figure() function in Matplotlib is used to specify the size of the figure.

200MA

Creates a line chart of the 'Close' column in the DataFrame 'df' and overlays it with line charts of the 100-day and 200-day moving averages of the 'Close' column. The figure() function in Matplotlib is used to specify the size of the figure. The code first reads the data from the 'tc.csv' file and calculates the 100-day and 200-day moving averages of the 'Close' column using the rolling() and mean() methods in Pandas. The figure() function is used to create a new figure with the specified size (in this case, 12 inches wide by 6 inches tall). We then use the plot() function to create three line charts: one for the 'Close' column of the DataFrame 'df', another for the 100-day moving average of the 'Close' column, and a third for the 200-day moving average of the 'Close' column.

Shape

df.shape returns the number of rows and columns in the DataFrame 'df' as a tuple of integers in the format (rows, columns). Split into 2 70 and 30 75 for training and 25 for prediction creates two new DataFrames, 'data_training' and 'data_testing', which are subsets of the 'Close' column in the original DataFrame 'df'. 'data_training' contains the first 75% of the 'Close' column in 'df', while 'data_testing' contains the last 25% of the 'Close' column in 'df'. The int() function is used to convert the result of the multiplication to an integer, which is required to slice the DataFrame. Assuming that the 'tc.csv' file is in the same directory as the script, this code reads the data from the 'tc.csv' file and creates two new DataFrames: 'data_training' and 'data_testing'. 'data_training' contains the first 75% of the 'Close' column in 'df', which is obtained by slicing the 'Close' column from index 0 to 75% of the length of 'df' using the int() and len() functions.

Similarly, 'data_testing' contains the last 25% of the 'Close' column in 'df', which is obtained by slicing the 'Close' column from 25% of the length of 'df' to the end of the 'Close' column using the same functions.

Min max scaller

MinMaxScaler is a method from the sklearn.preprocessing module that is used for feature scaling. Feature scaling is a method of transforming numerical features to have a similar scale, typically between 0 and 1, to improve the performance of machine learning models. The code you provided applies the fit_transform() method of the MinMaxScaler object named 'scaler' to the 'data_training' DataFrame, which scales the data between 0 and 1 using the minimum and maximum values of the data in 'data_training'. data_training_array.shape returns the shape of the NumPy array 'data_training_array'. The shape of a NumPy array is a tuple that represents the size of each dimension of the array.

X train y train

The code you provided creates the training data for a time series prediction model by iterating through the scaled training data 'data_training_array' and creating input sequences and corresponding output values. The for loop iterates through each row of the training data starting from the 100th row (since the first 100 rows are used as input for the first output). For each row, it creates an input sequence of 100 data points by selecting the 100 rows preceding the current row (i.e., data_training_array[i-100: i]). It also creates the corresponding output value, which is the 101st data point (i.e., data_training_array[i, 0]). The input sequences and output values are stored in separate lists (x_train and y_train, respectively). Finally, these lists are converted to NumPy arrays using the np.array() function. The resulting arrays are used to train a time series prediction model, where the input sequences are used to predict the corresponding output values.

Layer

Code defines a Sequential model in Keras with several LSTM layers and dropout regularization. The first layer added to the model is an LSTM layer with 50 units, ReLU activation, and a return_sequences parameter set to True. The input_shape parameter is set to (x_train.shape[1], 1) to match the input shape of the training data. After the first LSTM layer, a Dropout layer is added to the model with a dropout rate of 0.2. This helps to prevent overfitting of the model. Next, three more LSTM layers are added to the model with 60, 80, and 120 units, respectively, each with ReLU activation and a return_sequences parameter set to True. A Dropout layer is added after each of these LSTM layers with dropout rates of 0.3, 0.3, and 0.5, respectively. Finally, a fully connected Dense layer with a single output unit is added to the model. This layer will output the predicted value for the next time step in the time series.

Optimiser Adam

The model using the Adam optimizer and the mean squared error (MSE) loss function. The Adam optimizer is a popular choice for training deep neural networks, and the MSE loss function is commonly used for regression problems. The model using the training data x_train and y_train for 50 epochs. During training, the model will attempt to minimize the mean squared error between its predicted values and the true values of the training data. The fit() method updates the weights of the model by backpropagating the error through the layers and applying the Adam optimizer.

File save

model.save('keras_model.h5') saves the trained LSTM model to a file named "keras_model.h5" in the current working directory. This saved model can be loaded later using the load_model() function in Keras, and used to make predictions on new data without having to retrain the model from scratch. This is useful when working with large datasets or when deploying the model in production environments where training may not be feasible.

Show testing data

data_testing is a Pandas DataFrame that contains the closing stock prices of a particular company for the testing period. The method data_testing.head() is used to display the first 5 rows of the DataFrame data_testing, which provides a quick look at the structure and contents of the DataFrame. It allows us to check if the data is loaded and formatted correctly, and if there are any missing or invalid values. The output of data_testing.head() displays the first 5 rows of the data_testing DataFrame, along with the column headers. data_training is a Pandas DataFrame that contains the closing stock prices of a particular company for the training period. The method data_training.tail(100) is used to display the last 100 rows of the DataFrame data_training. This provides a quick look at the recent stock price trends during the training period, which can help in identifying any patterns or trends that the LSTM model might be able to learn from. The output of data_training.tail(100) displays the last 100 rows of the data_training DataFrame, along with the column headers. past_100_days is a Pandas DataFrame that contains the closing stock prices of a particular company for the last 100 days of the training period. The line past_100_days = data_training.tail(100) selects the last 100 rows of the data_training DataFrame using the tail() method and assigns them to a new DataFrame called past_100_days. This subset of the training data is then used as input to the LSTM model to predict the stock prices for the testing period. The past_100_days DataFrame can be used to visualize the recent trends in the stock prices and compare them with the predicted values obtained from the LSTM model. final_df is a Pandas DataFrame that combines the last 100 days of the training data and the testing data. The line final_df = past_100_days.append(data_testing, ignore_index = True) appends the data_testing DataFrame to the past_100_days DataFrame using the append() method, and assigns the result to a new DataFrame called final_df. The ignore_index parameter is set to True to ensure that the row indices of the appended DataFrame are reset to avoid duplicates. The resulting final_df DataFrame contains all the closing stock prices for the last 100 days of the training period, as well as the closing stock prices for the testing period, which can be used to evaluate the performance of the LSTM model in predicting the stock prices for the testing period. final_df.head() is a method call that displays the first five rows of the final_df DataFrame, which is a combination of the last 100 days of the training data and the testing data.

Scaler data for next 100 days

The code input_data = scaler.fit_transform(final_df) applies the fit_transform() method of the MinMaxScaler object to the final_df DataFrame, which scales the data to a range between 0 and 1. The fit_transform() method first fits the scaler object to the data by computing the minimum and maximum values of the data, and then transforms the data by scaling it to the range between 0 and 1 using these values.

The resulting input_data is a numpy array that contains the scaled values of the closing stock prices for the last 100 days of the training period and the testing period, which can be used as input to the LSTM model for making predictions. The shape of the input_data array is (n, 100, 1), where n is the total number of days (i.e., the number of rows in the final_df DataFrame). The first dimension represents the number of days, the second dimension represents the number of time steps in the input sequence (i.e., 100), and the third dimension represents the number of features in the input data (i.e., 1, since we are using only the closing stock prices as input).

Testing data

The code x_test = [] and y_test = [] initializes two empty lists for storing the input data and target values for the test set.

The loop for i in range(100, input_data.shape[0]): iterates over each row of the input_data array, starting from the 100th row. For each iteration, the code appends a slice of the input data containing the 100 previous closing stock prices to the x_test list, and the current closing stock price to the y_test list. This creates a set of input sequences with length 100, which can be used to make predictions for the corresponding target values.

After the loop completes, the resulting x_test and y_test lists are converted to numpy arrays using the np.array() function, and can be used to evaluate the performance of the LSTM model on the test set.

The code x_test, y_test = np.array(x_test), np.array(y_test) converts the x_test and y_test lists to numpy arrays, which can be used as input to the LSTM model for testing.
The resulting arrays have the following shapes:

x_test: A 2D array with shape (n_samples, n_timesteps, n_features), where n_samples is the number of test samples, n_timesteps is the length of the input sequences (100 in this case), and n_features is the number of input features (1 in this case, as we are using only the closing stock prices as input). In other words, each row of x_test represents a sequence of 100 previous closing stock prices.

y_test: A 1D array with shape (n_samples,), containing the corresponding target values for each input sequence. In other words, each element of y_test represents the closing stock price at the end of the corresponding input sequence

Predict

y_predicted contains the predicted values of the stock prices based on the test data x_test using the LSTM model model.

Testing

y_test contains the actual values of the stock prices for the test data. These are the values we will compare with the predicted values to evaluate the accuracy of the LSTM model.

Y predict

y_predicted contains the predicted values of the stock prices based on the test data x_test using the LSTM model model. These predicted values are obtained by feeding the test data into the trained LSTM model, and can be compared with the actual test values in y_test to evaluate the accuracy of the model. scaler.scale_ is an attribute of the MinMaxScaler object, and it represents the scale factors that were used to scale the training and testing data. These factors are based on the maximum and minimum values of the training data, and are used to transform the data into the specified feature range of 0 to 1. In other words, scaler.scale_ contains the scaling factors used to normalize the input data to the LSTM model.

scale_factor is calculated as the reciprocal of scaler.scale_[0], which is the scaling factor used to scale the input data. This value is then used to rescale the predicted and actual values of the stock prices back to their original values, since they were previously scaled to a range of 0 to 1 using MinMaxScaler. By multiplying the predicted and actual values by the scale factor, we can get the stock prices in their original units.

1 a plot of the original price values (in blue) and the predicted price values (in red) using the test data. This is a common visualization in time series analysis to compare the performance of the model with the actual data.

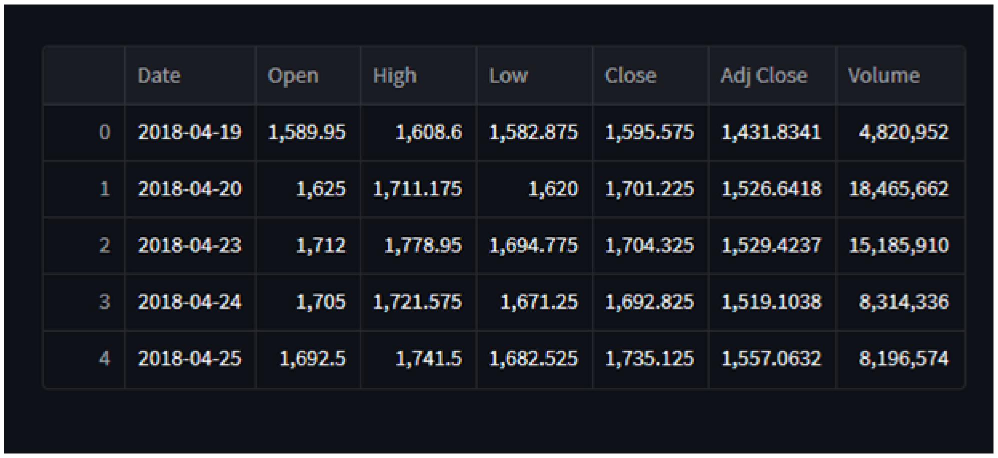
2 The plt.plot() function is used to plot the values. The x-axis represents time, and the y-axis represents price. The xlabel() and ylabel() functions set the labels for the x and y axes, respectively. The legend() function creates a legend for the plot to differentiate between the original and predicted price values.

3 The plot shows the trend of the original price data and the trend of the predicted price data. The goal is to have the predicted values as close to the actual values as possible.

5.5 RESULT

The implementation of the proposed LSTM model using Python which predicts the future price of TCS share based on its historical data. a. The below visualization figure shows the visualization of TCS prediction.In our paper the implementation of an algorithm which predicts the stock price of a share for given period of time, the below graph from our algorithm will show the predicted price of TCS share. In the result shown in the below graph is the plotted form our algorithm outcome by applying LSTM units for achieving the accuracy.

FIG 5.1 shows the TCS stock Price History using LSTM. the X axis shows the Days and Y axis shows the Orice Of stock. Fig 5.2. The X axis shows the days and



	Date	Open	High	Low	Close	Adj Close	Volume
0	2018-04-19	1,589.95	1,608.6	1,582.875	1,595.575	1,431.8341	4,820,952
1	2018-04-20	1,625	1,711.175	1,620	1,701.225	1,526.6418	18,465,662
2	2018-04-23	1,712	1,778.95	1,694.775	1,704.325	1,529.4237	15,185,910
3	2018-04-24	1,705	1,721.575	1,671.25	1,692.825	1,519.1038	8,314,336
4	2018-04-25	1,692.5	1,741.5	1,682.525	1,735.125	1,557.0632	8,196,574

Figure 5.1: TCS Dataset

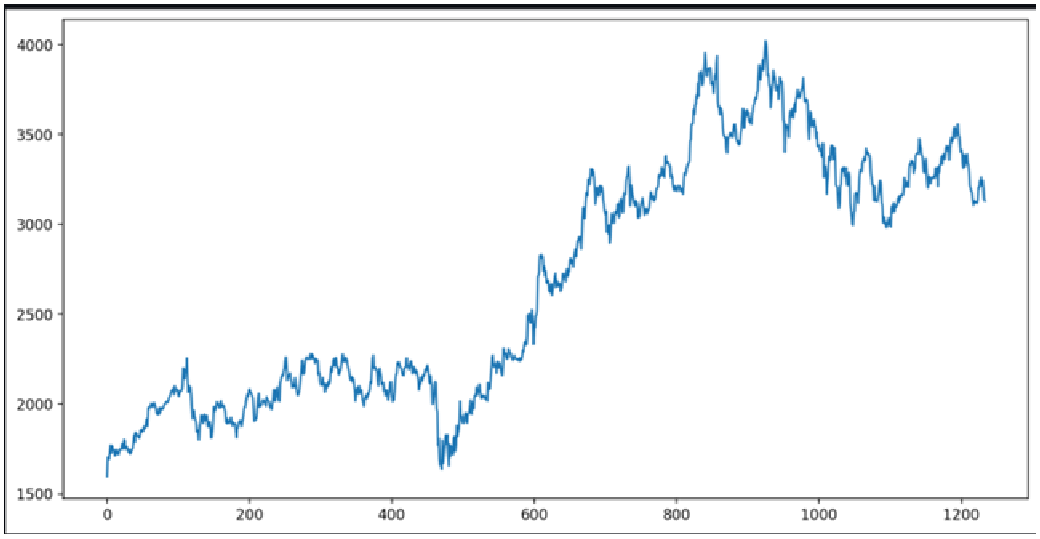


Figure 5.2: shows the TCS stock Price History

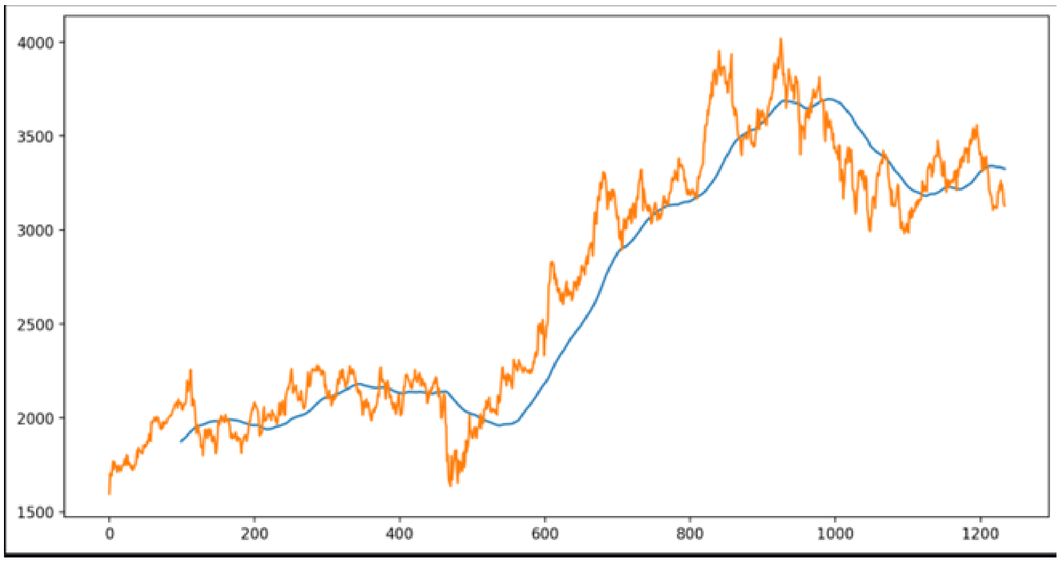


Figure 5.3: Closing price vs Time Chart with 100MA

the Y axis shows the Days. Fig 5.3 The X-axis shows the Date and the Y axis shows the close price. Fig 5.4 Shows the TCS Share Prices from 2018 to 2023. The X-axis

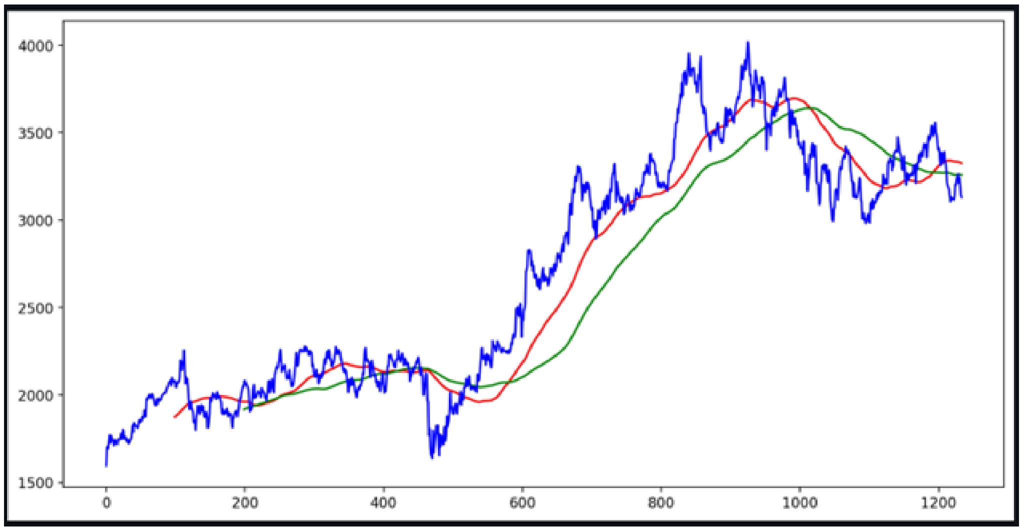


Figure 5.4: Closing price vs Time Chart with 200MA

shows the Date and the Y-axis shows the close price. Fig 5.5 Shows the average of the Dataset and close price as the predicted price for next day also with min , avg

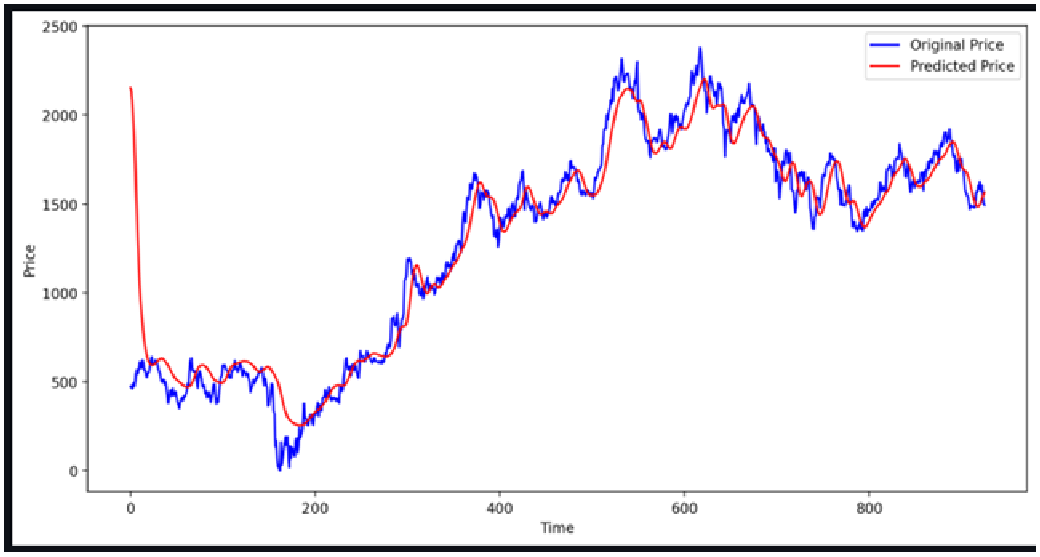
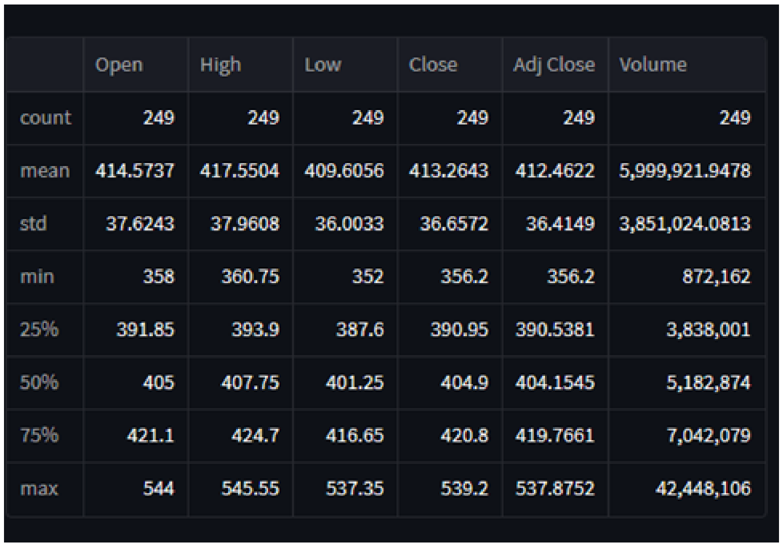


Figure 5.5: Prediction vs Original



	Open	High	Low	Close	Adj Close	Volume
count	249	249	249	249	249	249
mean	414.5737	417.5504	409.6056	413.2643	412.4622	5,999,921.9478
std	37.6243	37.9608	36.0033	36.6572	36.4149	3,851,024.0813
min	358	360.75	352	356.2	356.2	872,162
25%	391.85	393.9	387.6	390.95	390.5381	3,838,001
50%	405	407.75	401.25	404.9	404.1545	5,182,874
75%	421.1	424.7	416.65	420.8	419.7661	7,042,079
max	544	545.55	537.35	539.2	537.8752	42,448,106

and max value .

6 CONCLUSION & FUTURE SCOPE

6.1 CONCLUSION

The study of the share is carried out in this project and it can be carried out for several shares in the future. The prediction could be more reliable if the model trains a greater number of data sets using higher computing capacities, an increased number of layers, and an LSTM module. we Implement a system on TCS Company Stock Price. And proposed the model for the prediction of stock with prediction value and close value of the stock.

6.2 FUTURE SCOPE

In future enhancement the inclusion of sentiment analysis from social media to understand what the market thinks about the price variation for a particular share and it can be implemented by adding Twitter and Facebook API to our program as Facebook is a leading social media which has lots of market trend information posted by users.

References

[1] Reddy, V. K. (2021). Stock Market Prediction Using Machine Learning. International Research Journal of Engineering and Technology (IRJET), Vol. 5.

[2] Nti, I. K., Adekoya, A. F., & Weyori, B. A. (2020). A comprehensive evaluation of ensemble learning for stock-market prediction. Journal of Big Data, 7(1). https://doi.org/10.1186/s40537-020-00299-5

[3] Shen, J.,& Shafiq, M. O. (2020). Short-term stock market price trend prediction using a comprehensive deep learning system. Journal of Big Data, 7(1). https://doi.org/10.1186/s40537-020-00333-6

[4] Sandhu, O. (2021). Stock market trend prediction using regression errors. https://doi.org/10.32920/ryerson.14645169

[5] Harahap, L. A., Lipikorn, R., & Kitamoto, A. (2020b). Nikkei Stock Market Price Index Prediction Using Machine Learning. Journal of Physics: Conference Series, 1566, 012043. https://doi.org/10.1088/1742- 6596/1566/1/012043

[6] Harsh Panday, V. Vijayarajan, Anand Mahendran, A. Krishnamoorthy, V.B. Surya Prasath, “Stock Recommendation using Sentiment analysis and Long Short Term Memory”, European Journal of Molecular & Clinical Medicine, 7-February 2020.

[7] Ashish Pathak, Nisha P shetty, “Indian Stock Market Recommendation Using Machine Learning and Sentiment Analysis”, Conference paper, 4th July 2018.

[8] Saloni Mohan, Sahitya Mullapudi, Sudheer Sammeta, Parag Vijayvergia, David C. Anastasiu, “Stock Price Recommendation Using News Sentiment Analysis”, 4-9 April2019

[9] Faten Subhi Alzazah, “Recent Advances in Stock Market Recommendation Using Text Mining: A Survey”, June 1st 2020.

[10] Frasconi, P., Gori, M., Maggini, M. and Soda, G. (1995) Unified Integration of Explicit Knowledge and Learning by Example in Recurrent Networks. IEEE Transactions on Knowledge and Data Engineering, 7, 340-346. https://doi.org/10.1109/69.382304

[11] H. Yu, X. Chen, Y. Cheng, S. Ye, and H. Liu, ”Stock price prediction using LSTM, RNN and CNN-SVM model,” IEEE Access, vol. 7, pp. 10216-10223, 2019.

[12] Y. Xu, C. Wang, and J. Wang, ”Stock price prediction using LSTM and MLP with Bayesian optimization,” Expert Systems with Applications, vol. 131, pp. 240-254, 2019.

[13] J. Zhou, Y. Liu, L. Xie, and Z. Zhang, ”Stock price prediction using LSTM and attention-based feature fusion,” IEEE Access, vol. 8, pp. 11999-12008, 2020.

[14] S. Liu, Z. Zhang, and C. Wu, ”An LSTM model with attention mechanism for stock price prediction,” Expert Systems with Applications, vol. 152, pp. 113429, 2020.

[15] C. Wang, J. Wang, and Y. Xu, ”An empirical comparison of LSTM, ARIMA, and random walk models for stock price prediction,” IEEE Access, vol. 7, pp. 102618-102625, 2019

[16] M. S. Hegde, G. Krishna and R. Srinath, ”An Ensemble Stock Predictor and Recommender System,” 2018 International Conference on Advances in Computing, Communications and Informatics (ICACCI), Bangalore, 2018, pp. 1981-1985.

[17] R. Akita, A. Yoshihara, T. Matsubara and K. Uehara, ”Deep learning for stock prediction using numerical and textual information,” 2016 IEEE/ACIS 15th International Conference on Computer and Information Science (ICIS), Okayama, 2016, pp. 1-6.

[18] S. Kumar and D. Ningombam, ”Short-Term Forecasting of Stock Prices Using Long Short Term Memory,” 2018 International Conference on Information Technology (ICIT), Bhubaneswar, India, 2018, pp. 182-186.

[19] M. Alam, A. Al Galib and R. M. Rahman, ”Algorithms to predict opening price and trading decision of stocks in Dhaka Stock Exchange,” 14th International Conference on Computer and Information Technology (ICCIT 2011), Dhaka, 2011, pp. 213-218.

[20] A. S. Ravi, A. Sarvesh and K. George, ”Sequential ELM for financial markets,” 2016 Second International Conference on Cognitive Computing and Information Processing (CCIP), Mysore, 2016, pp. 1-6.

[21] A. S. Ravi, A. Sarvesh and K. George, ”Online multiple-model approach to prediction for financial markets,” 2016 2nd International Conference on Contemporary Computing and Informatics (IC3I), Noida, 2016, pp. 644-649

[22] Prediction Of Stock Market Exchange Using LSTM Algorithm K.Sai Sravani, Dr.P.RajaRajeswari.