



# **Chapter: 1**

# 1. Introduction

## 1.1 Stock price prediction

Due to the high profit of the stock market, it is one of the most popular investments. People investigated for methods and tools that would increase their gains while minimizing the risk, as the level of trading and investing grew. Two stock exchanges namely- the Dhaka Stock Exchange (DSE) and the Chittagong Stock Exchange (CSE), which are the most of the trading in Bangladesh Stock Market takes place. Since the prices in the stock market are dynamic, the stock market prediction is complicated.

The actual owners of an establishment sell their stocks to others to get financial strength through investment and in turn to make company grow. The buyer of these stocks owns a portion of the company. Stock prices change as a result of supply and demand. Suppose, when more people want to shop for stock, the fee is going up as there is a greater need. When more people are willing to let go stock, the fee is going down as there may be greater delivery than demand. Or understanding the offer and demand is simple, to determining the contribution of factors that cause to the increase in demand or supply is difficult. These factors will often depend on social factors such as market behaviour, price rises, trends and most importantly, what is good about the corporation in the news and what is harmful. [1,2]

Investing in stocks is a complex task and requires deep knowledge and study of the market to obtain high returns. With the advancement in computer techniques like ML and AI predicting the stock prices has fascinated the programmers. Stock price prediction isn't best accurate with linear models, it's a complex process depending on the dynamic nature of stocks as well the public sentiments. With proper in-depth study humans have managed to predict the prices. But it's not everyone's cup of tea to devote so much extensive time and energy. The question arises can we build something that can think like humans. This brings Neural Networks in picture. The Traditional NN cannot store memory which is required for the better stock price prediction. Then RNN can be used but they too suffer from Vanishing Gradient and fail to store long term dependencies. [2]

The vital part of machine learning is the dataset used. The dataset should be as concrete as possible because a little change in the data can perpetuate massive changes in the outcome. In this project, supervised machine learning is employed on a dataset obtained from Dhaka Stock Exchange. This dataset comprises of following five variables: open, close, low, high and volume. Open, close, low and high are different bid prices for the stock at separate times with nearly direct names. The volume is the number of shares that passed from one owner to another during the time period. The model is then tested on the test data. [3]

Recurrent networks have a memory state that can store background information and can store details about past inputs, but the amount of period of time that it can remember this is not fixed. This time depends on its weights, context and on the provided input data. LSTM tackled the problem of long-term dependencies that RNN lacked by which RNN could not use information

passed to it in early stages. As a result, RNN model could not always provide accurate results for long term, but can provide more accurate outcome based on recent data. [2]

## **1.2 Applications**

- Business
- Companies
- Insurance company
- Government Agency

This application is helpful for stock investors, sellers, buyers, brokers.

## **1.3 Objectives**

A stock market prediction is described as an action of attempting to classify the future value of the company stock or other financial investment traded on the stock exchange. The forthcoming price of a stock of the successful estimation is called the Yield significant profit. This helps you to invest wisely for making good profits.[1]

## **1.4 Motivation**

The future price of a stock is the main motivation behind the stock price prediction. In various cases like business and industry, environmental science, finance and economics motivation can be useful. The future value of the company's stock can be determining. [1]

## 1.5 Literature Review

### **1. Forecasting stock price in two ways based on LSTM neural network**

**Publication year:** 2019

**Author:** Jingyi Du, Qingli Liu, Kang Chen, Jiacheng Wang

**Journal Name:** 2019 IEEE

**Summary:** The LSTM neural network is used to predict Apple stocks by consuming single feature input variables and multi-feature input variables to verify the forecast effect of the model on stock time series. The experimental results show that the model has a high accuracy of 0.033 for the multivariate input and is accurate, that is in line with the actual demand. For the univariate feature input, the predicted squared absolute error is 0.155, which is inferior to the multi-feature variable input. [3]

### **2. Stock Price Prediction Based on Information Entropy and Artificial Neural Network**

**Publication Year:** 2019

**Author:** Zang Yeze, Wang Yiyang

**Journal Name:** 2019 IEEE

**Summary:** One of the most important components of the financial system is the stock market. For supporting the activity and evolvement, money is directed by the investors of the associated firm. Along with information theory and Artificial Neural Network (ANN) the combination of machine learning framework is formed. Information entropy for non-linear causality and stock relevance also to facilitate ANN time series modelling are creatively used by this method. The feasibility of this machine learning framework is analysed with Amazon, Apple, Google and Facebook prices. A time series analysis method based on information theory as well as LSTM to model the stock price dynamics are outlined in this paper. The transfer entropy between relevant variables to help LSTM time series prediction is merged in this modelling infrastructure, thus the accuracy of the assumption outcome is broadly granted. Modelled and real stock price is highly correlated while differ slightly in terms of Mean Absolute Error (MAE) and Root Mean Square Error (RMSE) which are investigated by the outcomes. [4]

### **3. Stock Market Prediction Using Machine Learning**

**Publication Year:** 2018

**Author:** Ishita Parmar, Ridam Arora, Lokesh Chouhan, Navanshu Agarwal, Shikhin Gupta, Sheirsh Saxena, Himanshu Dhiman

**Journal Name:** 2018 IEEE

**Summary:** In this paper studies, the use of Regression and LSTM based Machine learning to forecast stock prices. Factors measured are open, close, low, high and volume. This paper was an attempt to determine the future prices of the stocks of a company with improved accuracy and reliability using machine learning techniques. LSTM algorithm resulted in a positive outcome with more accuracy in predicting stock prices.

### **4. Share Price Prediction using Machine Learning Technique**

**Publication Year:** 2018

**Author:** Jeevan B, Naresh E, Vijaya kumar B P, Prashanth Kambli

**Journal Name:** 2018 IEEE

**Summary:** This paper is mostly based on the approach of predicting the share price using Long Short Term Memory (LSTM) and Recurrent Neural Networks (RNN) to forecast the stock value on NSE data using various factors such as current market price, price-earning ratio, base value and other anonymous events. The efficiency of the model is analysed by comparing the true data and the predicted data using an RNN graph. Machine learning to predict stock price as see the model is able to predict the stock price very close to the actual price where this model captures the detailed feature and uses different strategies to make a prediction. The model train for all the NSE data from the internet and recognize the input and group them and provide input according to the user configuration this RNN based architecture proved very efficient in forecasting the stock price by changing the configuration accordingly which also use backpropagation mechanism while gathering and grouping data to avoid mixing of data. [5]

For investors, this type of analysis of trends and cycles will obtain more profit. We must use networks like LSTM as they rely on the current information to analyse various information.

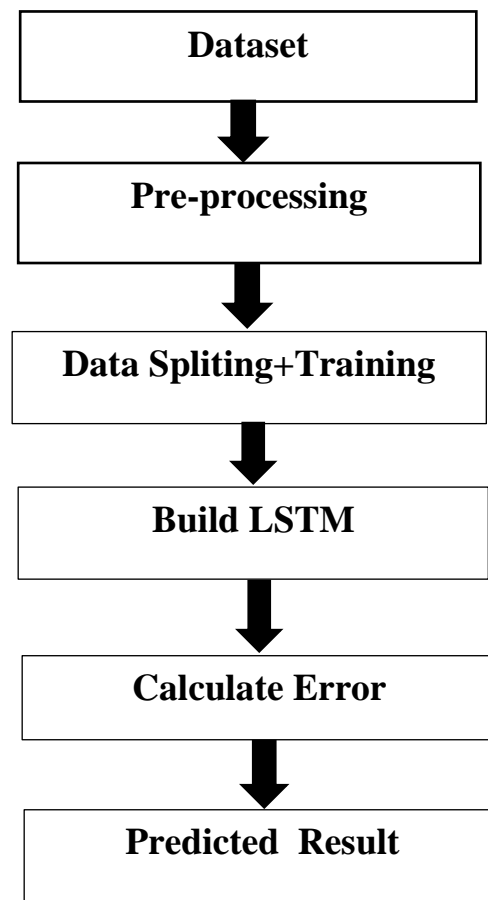


## **Chapter: 2**

## 2. Methodology

The Open, Close, High, Low, Adjusted Closing price, and Volume are all included in several data sets used for price prediction. The maximum and minimum prices of a certain stock on a given day are referred to as high and low, respectively. Adjusted Closing refers to the closing price after any corporate actions have been taken into account, as opposed to the raw closing price. Finally, volume refers to the number of stocks that are traded and bought each day. Earnings per share (EPS) is a key metric that measures a company's profitability. The Price to Earnings Ratio (P/E) is the ratio of a concern's present stock price to its earnings per share (EPS) . Deep learning is a subset of machine learning algorithms that are based on learning data representations. Deep learning models employ a network of multi-layered nonlinear processing units known as neurons, which can automatically extract and transform features. An Artificial Neural Network is a network of such neurons. ANN layers are made up of clusters of nodes that are coupled to produce a view of the human brain. The yield of one layer is fed into the next layer as input. These system models have a proclivity for learning from their training data. [2]

### 2.1 Proposed Methodology



## 2.2 Recurrent Neural Network (RNN)

A Recurrent Neural Network (RNN) is a kind of Artificial Neural Network (ANN). RNNs are a kind of neural network that is designed to deal with sequential input, and are made up of ordinary recurrent cells, in which the networks between the neurons form a directed graph, or, to put it another way, the hidden layers have a self-loop. This allows RNNs to learn current state by using the hidden neurons' previous state, as well as RNNs use the information they've learned before in time and the current input. This allows them to learn a wide range of skills. Handwriting recognition, speech recognition, and other activities are among them. [2]

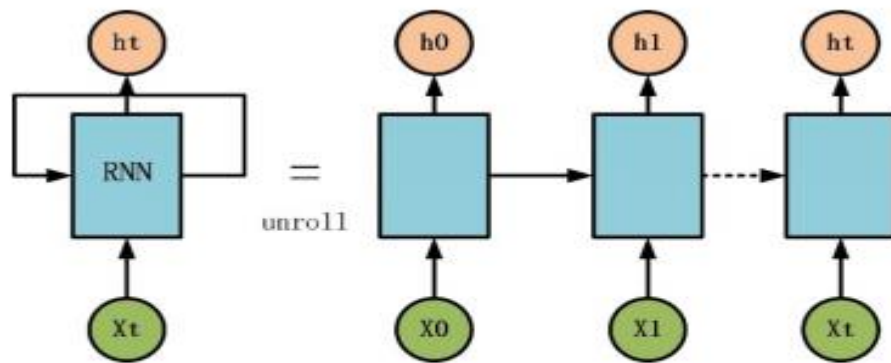


Fig. 1. An unrolled recurrent neural network

Initially (at time step  $t$ ), the RNN produces an output of  $h_t$  for some input  $x_t$ . The RNN uses two inputs,  $x_{t+1}$  and  $h_t$ , to generate the output  $h_{t+1}$  in the next time step ( $t+1$ ). Information can be transmitted from one network step to the next via a loop. All input vector units in a typical neural network are presumed to be not reliant on other. As a result, in a typical neural network, sequential information cannot be used. In the RNN model, sequential data from a time series produces a hidden state, which is then combined with the network output that is reliant on the hidden state. Because stock trends are a type of time series numbers, RNNs are the ideal fit for this application. Although training an RNN is difficult due to the RNN's structure's reverse reliance over time. As a result, as the learning period lengthens, RNNs get more complex. As demonstrated in a study addressed by Yashoua et al, RNNs lack the ability to learn long-term dependencies. While the fundamental objective of employing an RNN is to understand long-term addictions but it fails at it due to Vanishing Gradient problem that occurs when weights are getting updated during back propagation using chain rule, the weights become too small. As a result, Hochreiter and Schmidhuber presented a method called Long Short-Term Memory (LSTM) in 1997 to deal with these long-term dependencies. [2]



## 2.3 Long Short Time Memory (LSTM)

Long Short Term Memory Network is an advanced RNN, a sequential network, that allows information to persist. It is capable of handling the vanishing gradient problem faced by RNN. A recurrent neural network is also known as RNN is used for persistent memory. [1]

Since the change from state to state of RNN is indicated, the learning model just has the same dimensionality of the data. At each time step, the architecture also employs the same changeover function with the no different new arguments. The RNN has a structure of recurring modules of NN, as indicated above. In typical RNNs this module that is repetitive has a single layer of simple tanh. [2]

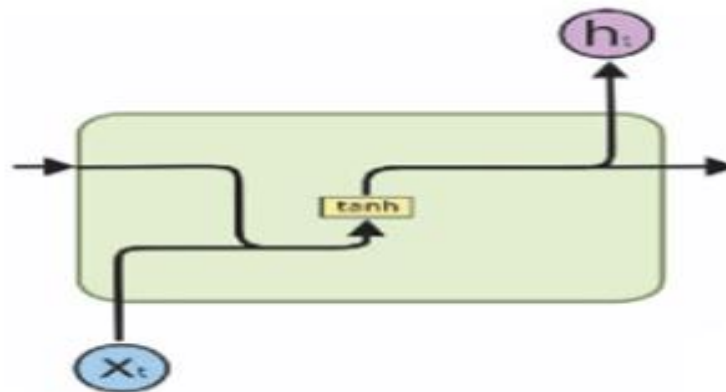


Fig. 2. RNN's single layer

Since LSTM is an advance version of RNN, it too follows chain-like structure of RNN, but has four neural network layers instead of a single, interacting in a special way. [2]

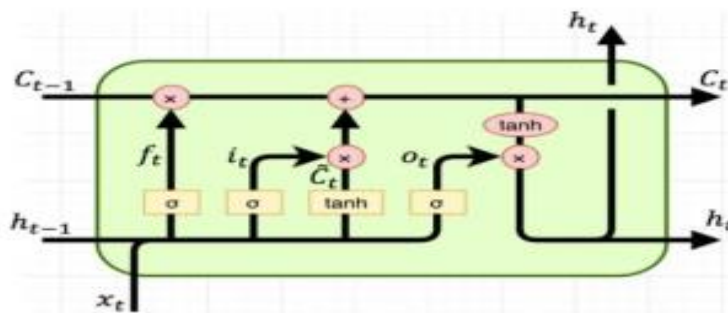


Fig. 3. LSTM's four layers

LSTM consists of memory cells that are responsible to store memory, and gates. These gates control the flow of data and act as controlling knobs which are responsible for the mixing of data. The data in LSTM is manipulated by these gates. The memory unit can keep track of the certain amount of training data. It is the memory unit consisting of cell states that makes LSTM capable of remembering long term dependencies. The three gates are forget gate, input gate

and output gate. Memory Cells: They store both short- and long-term memory. It is rather analogous to a conveyor belt. It runs through the entire chain with some trivial interactions with the gates in form of pointwise operations.[2,7]

**Hidden State:** Hidden state is the output of a LSTM layer that is fed an input to next layer. The sigmoid layer generates values ranging from zero to one, indicating how much of each element should be allowed to pass. The tanh layer creates a new vector that further makes an addition to the state. [2,7]

**Forget Gate:** The forget gate is responsible to forget some information from the memory cell. If there is change in context or something then the forget gate generates zeroes which are pointwise multiplied with the memory cell and the corresponding information is forgotten. It generates a vector with values between 0 and 1 with the sigmoid layer. In a cell of the LSTM network, the first step is to decide whether we should keep the information from the previous time stamp or forget it. Here is the equation for forget gate. [2,7]

**Forget Gate:**

$$\bullet \quad f_t = \sigma(x_t * U_f + H_{t-1} * W_f)$$

Where,

- $X_t$  = input to the current timestamp
- $U_f$  = weight associated with the input
- $H_{t-1}$  = The hidden state of the previous timestamp
- $W_f$  = It is the weight matrix associated with hidden state

Later, a sigmoid function is applied over it. That will make  $f_t$  a number between 0 and 1. This  $f_t$  is later multiplied with the cell state of the previous timestamp as shown below.

$$C_{t-1} * f_t = 0 \quad \dots \text{if } f_t = 0 \text{ (forget everything)}$$

$$C_{t-1} * f_t = C_{t-1} \quad \dots \text{if } f_t = 1 \text{ (forget nothing)}$$

If  $f_t$  is 0 then the network will forget everything and if the value of  $f_t$  is 1 it will forget nothing.

**Input Gate:** The input gate examines the conditions under which any information should be added (or updated) in the state cell based on the input (e.g., previous output  $o_{t-1}$ , input  $x_t$ , and the prior state of cell  $c_{t-1}$ ). This is then pointwise added to memory cell. Input gate is used to

quantify the importance of the new information carried by the input. Here is the equation of the input gate. [2,7]

#### Input Gate:

$$\bullet \quad i_t = \sigma(x_t * U_i + H_{t-1} * W_i)$$

Where,

- $X_t$  = input at the current timestamp
- $U_i$  = weight matrix of input
- $H_{t-1}$  = A hidden state at the previous timestamp
- $W_i$  = Weight matrix of input associated with hidden state

**Output Gate:** This output will be based on our cell state, but will be a filtered version. First, we run a sigmoid layer which decides what parts of the cell state we're going to output. Then, we put the cell state through tanh (to push the values to be between -1-1 and 11) and multiply it by the output of the sigmoid gate, so that we only output the parts we decided to. [2,7]

Here is the equation of the output gate, which is pretty similar to the two previous gates.

#### Output Gate:

$$\bullet \quad o_t = \sigma(x_t * U_o + H_{t-1} * W_o)$$

The updation of the memory cell is defined by the following equations,

$$ft = \sigma(Wf.[ht-1, xt] + bf) \quad (1)$$

$$it = \sigma(Wi.[ht-1, xt] + bi) \quad (2)$$

$$ct = \tanh(Wc.[ht-1, xt] + bc) \quad (3)$$

$$ot = \sigma(Wo[ht-1, xt] + bo) \quad (4)$$

$$ht = ot * \tanh(ct) \quad (5)$$

where,

$ft$  = forget gate

$ht-1$  = previous hidden state output.

$xt$  = current input

$Wx$  = weight for respective gate

$it$  = input gate

$ot$  = output gate

$\sigma$  = sigmoid function

$ct$  = memory state.

$ht$  = current hidden state output.

$bx$  = biases for respective gates

## 2.4 Evaluating Performance for Stock Price Prediction

Before putting the algorithms into practice, let's clarify the metric to measure the performance of our models. Stock price prediction being a fundamental regression problem, we can use RMSE (Root Mean Squared Error) to measure how close or far off our price predictions are from the real world. [7]

Looking closely at the formula of RMSE, we can see how we will be able to consider the difference (or error) between the actual ( $A_t$ ) and predicted ( $F_t$ ) price values for all  $N$  timestamps and get an absolute measure of error. [7,9]

$$RMSE = \sqrt{\frac{1}{N} * \sum_{t=1}^N (A_t - F_t)^2}$$



## **Chapter: 3**

## 3 Dataset, Implementation and Results

### 3.1 Dataset detail

For this project, we used to csv file historical data source of Titas Gas Transmission and Distribution Company from (<https://www.investing.com/equities/titas-gas-transmission-distribution-historical-data>) from August 28, 2019 to August 28, 2022.

### 3.2 Tools and Techniques

#### 3.2.1 Python

The language of select for this project was Python. This was a straightforward call for many reasons.

- Python as a language has a vast community behind it. Any problems which may be faced is simply resolved with visit to Stack Overflow. Python is the foremost standard language on the positioning that makes it is very straight answer to any question. [1,11]
- Python is an abundance of powerful tools ready for scientific computing Packages. The packages like NumPy, Pandas and SciPy area unit freely available and well documented. These Packages will intensely scale back, and variation the code necessary to write a given program. This makes repetition fast. [1,11]
- Python is a language as forgiving and permits for the program that appear as if pseudo code. This can be helpful once pseudo code give in tutorial papers should be required and verified. Using python this step is sometimes fairly trivial. [1,11]

However, Python is not without its errors. The python is dynamically written language and packages are area unit infamous for Duck writing. This may be frustrating once a package technique returns one thing that, for instance, looks like an array instead of being an actual array. Plus the standard Python documentation did not clearly state the return type of a method, this can't lead without a lot of trials and error testing otherwise happen in a powerfully written language. This is a problem that produces learning to use a replacement Python package or library more difficult than it otherwise may be. [1,11]

#### 3.2.2 Numpy

Numpy is python package which provide scientific and higher level mathematical abstractions wrapped in python. It is the core library for scientific computing, that contains a provide tools for integrating C, strong n-dimensional array object, C++ etc. It is also useful in random number capability, linear algebra etc. Numpy's array type augments the Python language with an efficient data structure used for numerical work, e.g., manipulating matrices. Numpy additionally provides basic numerical routines, like tools for locating Eigenvectors. [1,12]

Using NumPy, a developer can perform the following operations –

- Mathematical and logical operations on arrays.
- Fourier transforms and routines for shape manipulation.
- Operations related to linear algebra. NumPy has in-built functions for linear algebra and random number generation.

### 3.2.3 Pandas

Pandas is a Python library used for working with data sets. It has functions for analyzing, cleaning, exploring, and manipulating data. Pandas allows us to analyze big data and make conclusions based on statistical theories. Pandas can clean messy data sets, and make them readable and relevant. Relevant data is very important in data science. Pandas are also able to delete rows that are not relevant, or contains wrong values, like empty or NULL values. This is called *cleaning* the data. [14]

### 3.2.4 Matplotlib

Matplotlib is a low level graph plotting library in python that serves as a visualization utility. Matplotlib is open source and we can use it freely. Matplotlib is mostly written in python, a few segments are written in C, Objective-C and Javascript for Platform compatibility. Most of the Matplotlib utilities lies under the pyplot submodule, and are usually imported under the plt alias. Now the Pyplot package can be referred to as plt.

### 3.2.5 SCIKIT-LEARN

Scikit-learn could be a free machine learning library for Python. It features numerous classification, clustering and regression algorithms like random forests, k-neighbours, support vector machine, and it furthermore supports Python scientific and numerical libraries like SciPy and NumPy. In Python Scikit-learn is specifically written, with the core algorithms written in Cython to get the performance. Support vector machines are enforced by a Cython wrapper around LIBSVM .i.e., linear support vector machines and logistic regression by a similar wrapper around LIBLINEAR. [1,13]

### 3.2.6 Keras

Keras is a high-level neural networks API, it is written in Python and also capable of running on top of the Theano, CNTK, or. TensorFlow. It was developed with attention on enabling quick experimentation. having the ability to travel from plan to result with the smallest amount doable delay is key to doing great research. Keras permits for straightforward and quick prototyping (through user-friendliness, modularity, and extensibility). Supports each recurrent networks and convolutional networks, also as combinations of the 2. Runs seamlessly on GPU and CPU. The library contains numerous implementations of generally used neural network building blocks like optimizers, activation functions, layers, objectives and a number of tools to create operating with text and image data easier. The code is hosted on GitHub, and community support forums embody the GitHub issues page, a Gitter channel and a Slack channel. [1,13]

### 3.2.7 Compiler option

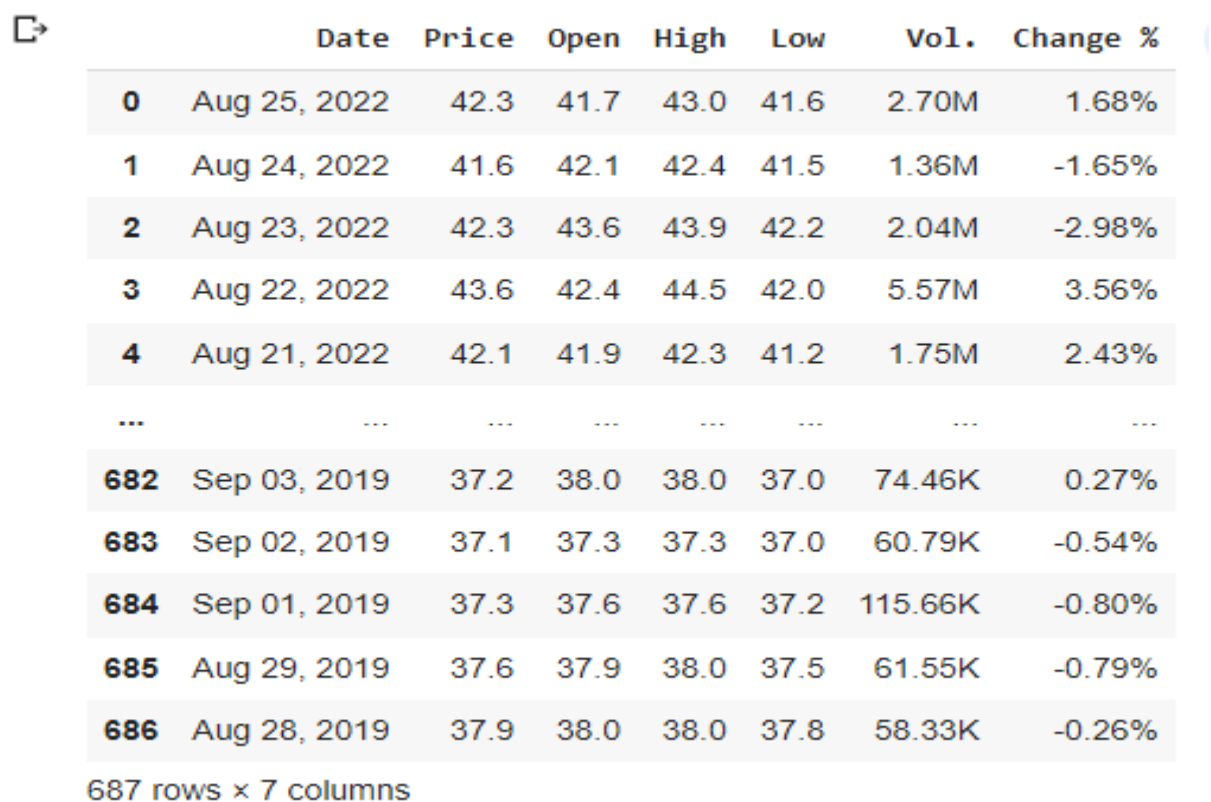
Anaconda is free premium open-source distribution of the R and Python programming languages for scientific computing, predictive analytics, and large-scale process that aim is to modify package managing and deployment. Package versions unit managed by the package management system conda.

### 3.2.8 Jupyter notebook

The Jupyter Notebook is an open-source web application that enables to making and sharing documents that contain visualizations, narrative text, live code and equations. Uses include: data, data visualization, data transformation, statistical modelling, machine learning, numerical simulation, data cleaning and much more.

## 3.3 Implementation results

### Step 1: Read Dataset



	Date	Price	Open	High	Low	Vol.	Change %
0	Aug 25, 2022	42.3	41.7	43.0	41.6	2.70M	1.68%
1	Aug 24, 2022	41.6	42.1	42.4	41.5	1.36M	-1.65%
2	Aug 23, 2022	42.3	43.6	43.9	42.2	2.04M	-2.98%
3	Aug 22, 2022	43.6	42.4	44.5	42.0	5.57M	3.56%
4	Aug 21, 2022	42.1	41.9	42.3	41.2	1.75M	2.43%
...	...	...	...	...	...	...	...
682	Sep 03, 2019	37.2	38.0	38.0	37.0	74.46K	0.27%
683	Sep 02, 2019	37.1	37.3	37.3	37.0	60.79K	-0.54%
684	Sep 01, 2019	37.3	37.6	37.6	37.2	115.66K	-0.80%
685	Aug 29, 2019	37.6	37.9	38.0	37.5	61.55K	-0.79%
686	Aug 28, 2019	37.9	38.0	38.0	37.8	58.33K	-0.26%

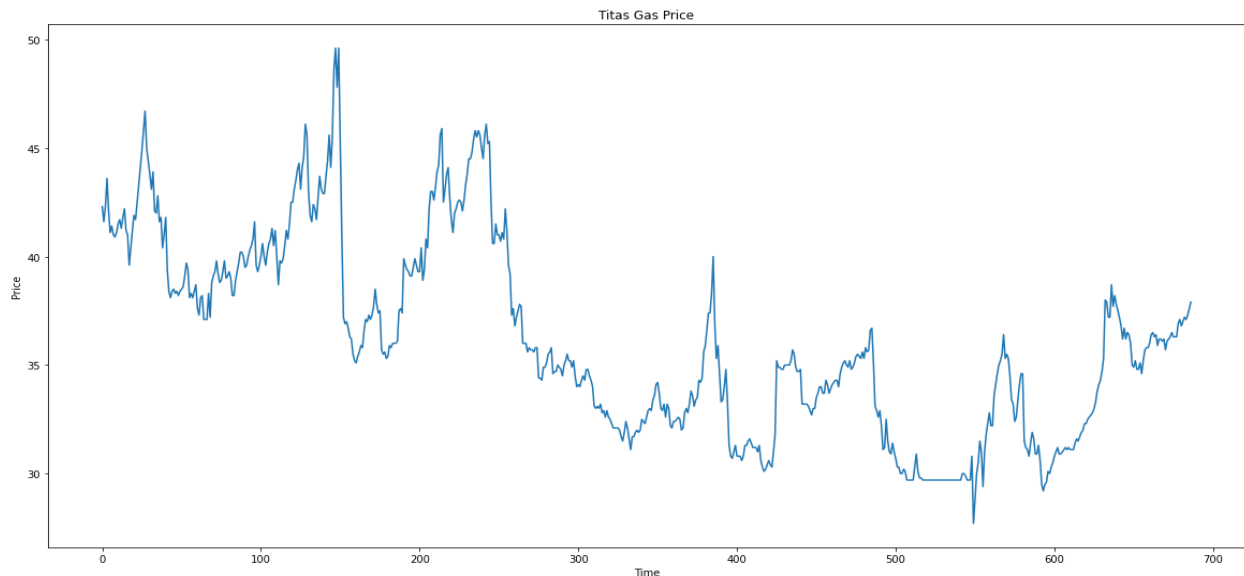
687 rows x 7 columns

Figure 1: Read Dataset



After performing data analysis, I have read the dataset. It shows the dataset information table starting from the tail. There are 687 data are available for this companies dataset.

## Step 2: Graph of price history



**Figure 2: Graph of price history**

## Step 3: Normalize the data

LSTM is very sensitive to the scale of the data, Here the scale of the Close value is in a kind of scale, we should always try to transform the value. Here we will use min-max scalar to transform the values from 0 to 1. We should reshape so that we can use fit transform.

### Code:

```
normalizer = MinMaxScaler(feature_range=(0,1))  
ds_scaled = normalizer.fit_transform(np.array(ds).reshape(-1,1))  
len(ds_scaled), len(ds)
```

### Output:

```
(687, 687)
```

## Step 4: Train and Test Split

Whenever training timeseries data we should divide the data differently we should train the data with the respective date. Always remember that in time-series data the one data is dependent on other data. The training size should be 70% of the total length of the data frame, the test size should be the difference between the length of the dataset and the training size.

### Code:

```
train_size = int(len(ds_scaled)*0.70)
test_size = len(ds_scaled) - train_size
train_size, test_size
ds_train, ds_test = ds_scaled[0:train_size,:], ds_scaled[train_size:len
(ds_scaled),:1]
len(ds_train), len(ds_test)
```

## Step 5: Data preprocessing

Now the timestep value will be 100. Let's split the data X, Y. In the 0th iteration the first 100 elements goes as your first record and the 101 elements will be put up in the X. The 100 elements will be put up in the Y.

### Code:

```
def create_ds(dataset, step):
    Xtrain, Ytrain = [], []
    for i in range(len(dataset)-step-1):
        a = dataset[i:(i+step), 0]
        Xtrain.append(a)
        Ytrain.append(dataset[i + step, 0])
    return np.array(Xtrain), np.array(Ytrain)
time_stamp = 100
X_train, y_train = create_ds(ds_train, time_stamp)
X_test, y_test = create_ds(ds_test, time_stamp)
X_train.shape, y_train.shape
```

### Output:

```
((379, 100), (379,))
```

## Step 6: Build LSTM

LSTMs are widely used for sequence prediction problems and have proven to be extremely effective. The reason they work so well is that LSTM can store past important information and forget the information that is not. While Implementing any LSTM, we should always reshape our X train in 3-D, add 1 the reason behind is the time step and the 1 is given to the LSTM.

### Code:

```
X_train = X_train.reshape(X_train.shape[0],X_train.shape[1] , 1)
X_test = X_test.reshape(X_test.shape[0],X_test.shape[1] , 1)

model = Sequential()
model.add(LSTM(units=50,return_sequences=True,input_shape=(X_train.shape[1],1)))
model.add(LSTM(units=50,return_sequences=True))
model.add(LSTM(units=50))
model.add(Dense(units=1,activation='linear'))
model.summary()
```

### Output:

Model: "sequential"

Layer (type)	Output Shape	Param #
lstm (LSTM)	(None, 100, 50)	10400
lstm_1 (LSTM)	(None, 100, 50)	20200
lstm_2 (LSTM)	(None, 50)	20200
dense (Dense)	(None, 1)	51

=====  
Total params: 50,851  
Trainable params: 50,851  
Non-trainable params: 0  
=====

Figure 3: LSTM

## Step 7: Estimate RMSE

Stock price prediction being a fundamental regression problem, we use here RMSE (Root Mean Squared Error) to measure how close or far off our price predictions are from the real world.

### Code:

```
model.compile(loss='mean_squared_error',optimizer='adam')
```

```
model.fit(X_train,y_train,validation_data=(X_test,y_test),epochs=100,batch_size=64)
```

### Step 8: Plot the error

```
loss = model.history.history['loss']  
plt.plot(loss)
```

### Output:

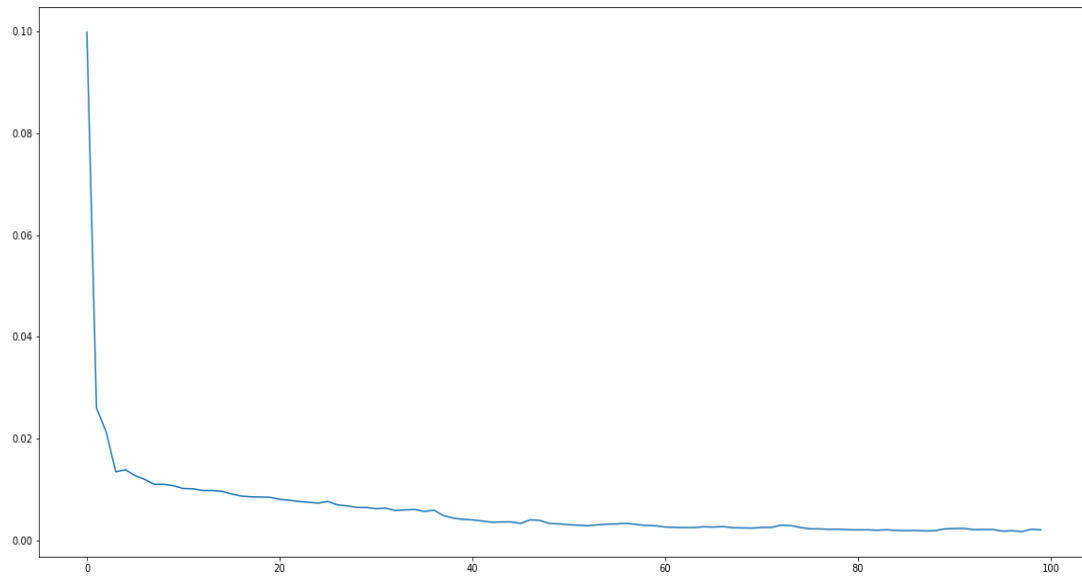


Figure 4: Estimate Error

### Step 9: Prediction

Predict both the  $X_{train}$  and the  $X_{test}$ , now let's scaler inverse transform because I want to see the root mean square performance.

### Code:

```
train_predict = model.predict(X_train)  
test_predict = model.predict(X_test)  
train_predict = normalizer.inverse_transform(train_predict)  
test_predict = normalizer.inverse_transform(test_predict)
```

### Step 10: Plot the train and test predict:

```
plt.plot(normalizer.inverse_transform(ds_scaled))  
plt.plot(train_predict)  
plt.plot(test_predict)
```

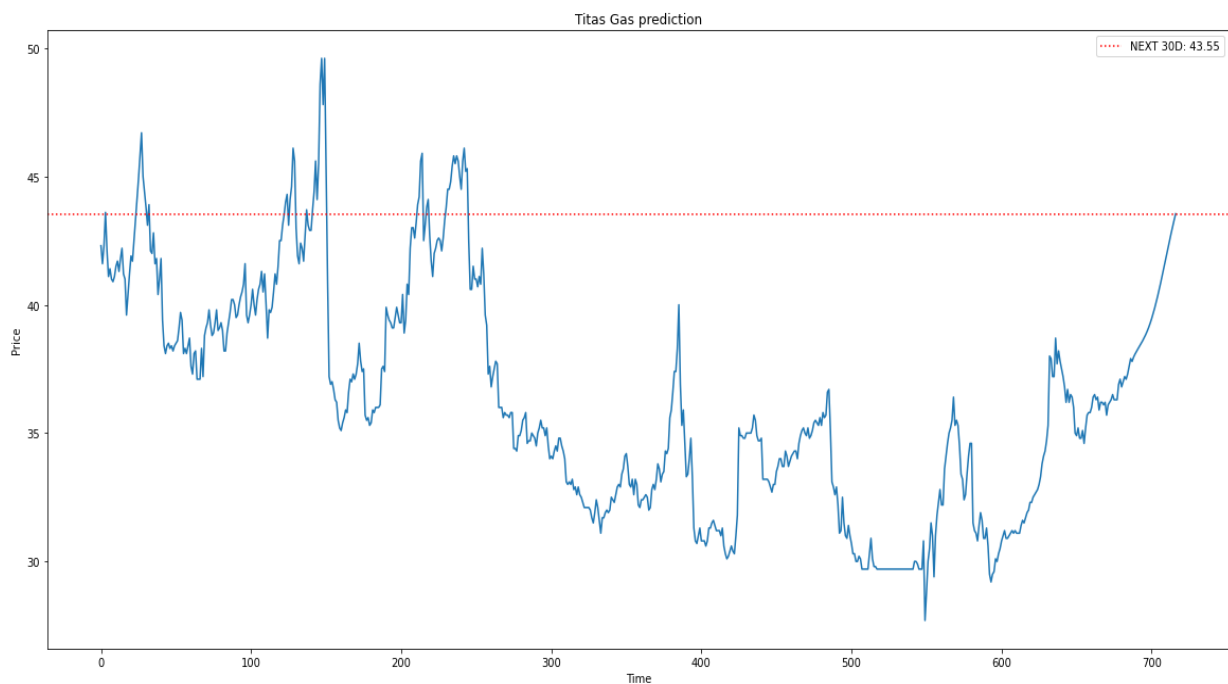


**Figure 5: Train and Test predict**

## Step 11: Final Result

```
final_graph = normalizer.inverse_transform(ds_new).tolist()
plt.plot(final_graph,)
plt.ylabel("Price")
plt.xlabel("Time")
plt.title('Titas Gas prediction')
plt.axhline(y=final_graph[len(final_graph)-1], color = 'red', linestyle = ':', label = 'NEXT 30D: {0}'.format(round(float(*final_graph[len(final_graph)-1]),2)))
plt.legend()
```

### Output:



**Figure 6: Final predicted result**

As we can see above, the model can predict the trend of the actual stock prices very closely. The accuracy of the model can be enhanced by training with more data and increasing the LSTM layers.



## **Chapter: 4**

## 4 Conclusion and Future Work

The stock market plays a remarkable role in our daily lives. It is a significant factor in a country's GDP growth. In this project, we learned the basics of the stock market and how to perform stock price prediction using machine learning.

In the project, we looked at how we can address the problem of predicting stock market prices by considering stock market data as a time series. We also used real-life stock data to predict prices of the Titas Gas Distribution Company stock using these methods and conducted a comparative performance analysis using RMSE.

However, it is quite impossible to predict the actual values but it can be possible to predict the upward and downward trends.

The changes in the stock market is not always be in a regular pattern or not always follow the continuous cycle. Based on the companies and sectors, the existence of the trends and the period of their existence will differ. The analysis of this type of cycles and trends can offer a more profit to the investors.

In future work, we add more stock market data and compare more model to improve accuracy of predicted stock price. In the future, for better accuracy model can be trained with more varied and detailed data. Also, other algorithms along with proposed can be used to create a new hybrid model.



## 5 References

- [1] “A Survey Paper on Stock Price Prediction Using Machine Learning Technique by 18CP80: Maithili Patel, July 2020” in the International Journal of Advance Engineering and Research Development
- [2] Shanket Chaudhari, K. Rajeswari, Sushma Vispute “A Review on using Long-Short Term Memory for prediction of Stock Price ” in IJERT Vol. 10 Issue 11, November 2021.
- [3] K. Raza, "Prediction of Stock Market performance by using machine learning techniques," 2017 International Conference on Innovations in Electrical Engineering and Computational Technologies (ICIEECT), Karachi, 2017,
- [4] J. Du, Q. Liu, K. Chen, and J. Wang, “Forecasting stock prices in two ways based on LSTM neural network,” in *2019 IEEE 3rd Information Technology, Networking, Electronic and Automation Control Conference (ITNEC)*, Mar. 2019
- [5] Z. Yeze and W. Yiyang, “Stock Price Prediction Based on Information Entropy and Artificial Neural Network,” in *2019 5th International Conference on Information Management (ICIM)*, Cambridge, United Kingdom, Mar. 2019
- [6] B. Jeevan, E. Naresh, B. P. V. kumar, and P. Kambli, “Share Price Prediction using Machine Learning Technique,” in *2018 3rd International Conference on Circuits, Control, Communication and Computing (I4C)*, Bangalore, India, Oct. 2018
- [7] Forecasting Stock Prices Prediction Using Neural Networks  
<https://www.andrew.cmu.edu/user/wyliec/project.pdf>
- [8] Stock price prediction and forecasting using LSTM  
<https://www.analyticsvidhya.com/blog/2021/05/stock-price-prediction-and-forecasting-using-stacked-lstm/>
- [9] Stock price prediction using python  
<https://www.askpython.com/python/examples/stock-price-prediction-python>
- [10] An easy guide to stock price prediction using machine learning  
<https://www.simplilearn.com/tutorials/machine-learning-tutorial/stock-price-prediction-using-machine-learning>
- [11] A. Tipirisetty, “Stock Price Prediction using Deep Learning,” p. 60.
- [12] “Python Numpy Tutorial | Learn Numpy Arrays with Examples,” *Edureka*, Jul. 14, 2017.  
<https://www.edureka.co/blog/python-numpy-tutorial/>
- [13] “Scikit-learn Tutorial: Machine Learning in Python – Dataquest.”  
<https://www.dataquest.io/blog/sci-kit-learn-tutorial/>
- [14] <https://www.w3schools.com/python/pandas/default.asp>