Google Stock Price Prediction

A PROJECT REPORT

Submitted by

Vinay Poddar [RA2011032010061]

Under the guidance of

MRS. D. SAISANTHIYA

Assistant Professor, Department of Networking and Communications

In partial fulfillment of the requirements for the award of the degree of

BACHELOR OF TECHNOLOGY

in

COMPUTER SCIENCE ENGINEERING

with specialization in Internet of Things

SRM
INSTITUTE OF SCIENCE & TECHNOLOGY
Deemed to be University u/s 3 of UGC Act, 1956

DEPARTMENT OF NETWORKING AND COMMUNICATIONS

COLLEGE OF ENGINEERING AND TECHNOLOGY

SRM INSTITUTE OF SCIENCE ANDTECHNOLOGY

KATTANKULATHUR- 603203

MAY 2023

SRM INSTITUTE OF SCIENCE AND TECHNOLOGY
KATTANKULATHUR – 603 203

BONAFIDE CERTIFICATE

Certified that this B. Tech – 18CSC305J project titled “**Google Stock Price Prediction”** is the bonafide work of **Vinay Poddar[RA2011032010061]** who carried out the project work under my supervision. Certified further, that to the best ofmy knowledge the work reported herein does not form part of any other thesis or dissertation on the basis of which a degree or award was conferred on an earlier occasion for this or any other candidate.

MRS. D SAISATHIYA
SUPERVISOR
Assistant Professor
Department of Networking and
Communications

DR. ANNAPURANI PANAIYAPPAN K.
HEAD OF THE DEPARTMENT
Professor
Department of Networking and
Communications

SIGNATURE OF INTERNAL EXAMINER

SIGNATURE OF EXTERNAL EXAMINER

TABLE OF CONTENTS

ABSTRACT

Stock prices are driven by corporate earnings or profit expectations. If a trader thinks that the company's earnings are high or will rise further, they will raise the price of the stock. One way for shareholders to get a return on their investment is to buy low stocks and sell them at high prices. If the company performs poorly and the value of the stock declines, the shareholder will lose some or all of his investment at the time of sale. Therefore, accurate stock price information is important. In this work, we proposed a google stock price prediction model using Recurrent Neural Network (RNN). Previous works on Google stack prediction have used some important techniques and models. Such as deep learning models like Artificial Neural Network (ANN) and Convolutional Neural Network (CNN) has been used for google stock movement prediction. Stock movement prediction on social media data by introducing Stock-Net, Artificial Neural Network (ANN) has also been used with an accuracy score of 0.58. Most of the proposed solutions have limited accuracy. In this paper, we have used Kaggle data of google stock price from the year 2012 to 2016. To predict the stock price of the first two months of 2017 based on the last two months of 2016. For this purpose, we used the Recurrent Neural Network (RNN) as a deep learning model and obtained an accuracy of 87.32%.

CHAPTER 1.

Introduction

Stock prices are driven by corporate earnings or profit expectations. If a trader thinks that the company's earnings are high or will rise further, they will raise the price of the stock. One way for shareholders to get a return on their investment is to buy low stocks and sell them at high prices. If the company performs poorly and the value of the stock declines, the shareholder will lose some or all of the investment at the time of sale [1]. A second way for shareholders to make a profit is for the company to pay dividends. These are quarterly payments to shareholders based on each share. The company's board of directors pays dividends from the proceeds. This is a way to reward shareholders. They are the actual owners of the company to invest in. This is especially important for companies that are profitable but may not have rapid growth [2]. In this work, we want to predict Google's stock price, but there's a Brownian movement stipulates that future valuations of stock prices are independent of the past. Therefore, it is virtually impossible to accurately predict future stock prices. Otherwise, we will all be billionaires. But some trends actually may be predicted. Therefore, we will try to predict the upward and downward trends that exist in the stock price of Google.

1.1 Google Stock Price

The stock price or share price is the cost of buying one share of a company. Stock prices are not fixed but fluctuate according to market conditions. If the company performs well, it may increase, and if the company does not meet expectations, it may decrease. If traders believe that the company's earnings are high or will rise further, they will increase the stock price. One way for shareholders to obtain a return on their investment is to buy low-priced stocks and sell them at high prices. If the company's performance is poor and the value of the stock drops, shareholders will lose part or all of their investment when selling the stock. [1], The second way for shareholders to profit is for the company to pay dividends. These are payments to shareholders quarterly per share. The company's board of directors pays dividends from the proceeds. This is a way to reward shareholders. They are the actual owners of the company to invest in. This is especially important for companies that are profitable but may not have rapid growth [2]. Stock price dynamics play an important role in shaping the stability of the financial system and are therefore an important feature of financial markets. Also, price dynamics are very complicated. Many factors, such as politics, economics, and emergencies, can have a significant impact on stock prices. Stock price dynamics play an important role in the shaping process. Therefore, the stability of the financial system is crucial to the characteristics of financial markets. The price dynamics are very complicated. Many factors, such as governance, economics, and emergencies, can have a significant impact on stock prices.

1.2 Google Stock Split

Google stock is an important and active part of today's financial markets. Investors and speculators in the market want to get better profits by analyzing market information. Some stock splits will only lead to more stocks, which will result in drop-in stock prices per share. But it is not the truth. There are two different Google stock codes to choose from. Figure 1 shows Google stock splits. The main difference between the GOOG stock code and the GOOGL stock code is that GOOG stock owners do not have voting rights, while GOOGL stocks owners have voting rights. [12] The company issued two stocks in April 2014 to retain control of founders Larry Page and Sergey Brin. When a company publicly issues shares, the founder usually loses control of the company. Alphabet firmly believes in the mission of organizing world information and firmly believes in the vision of the founders.

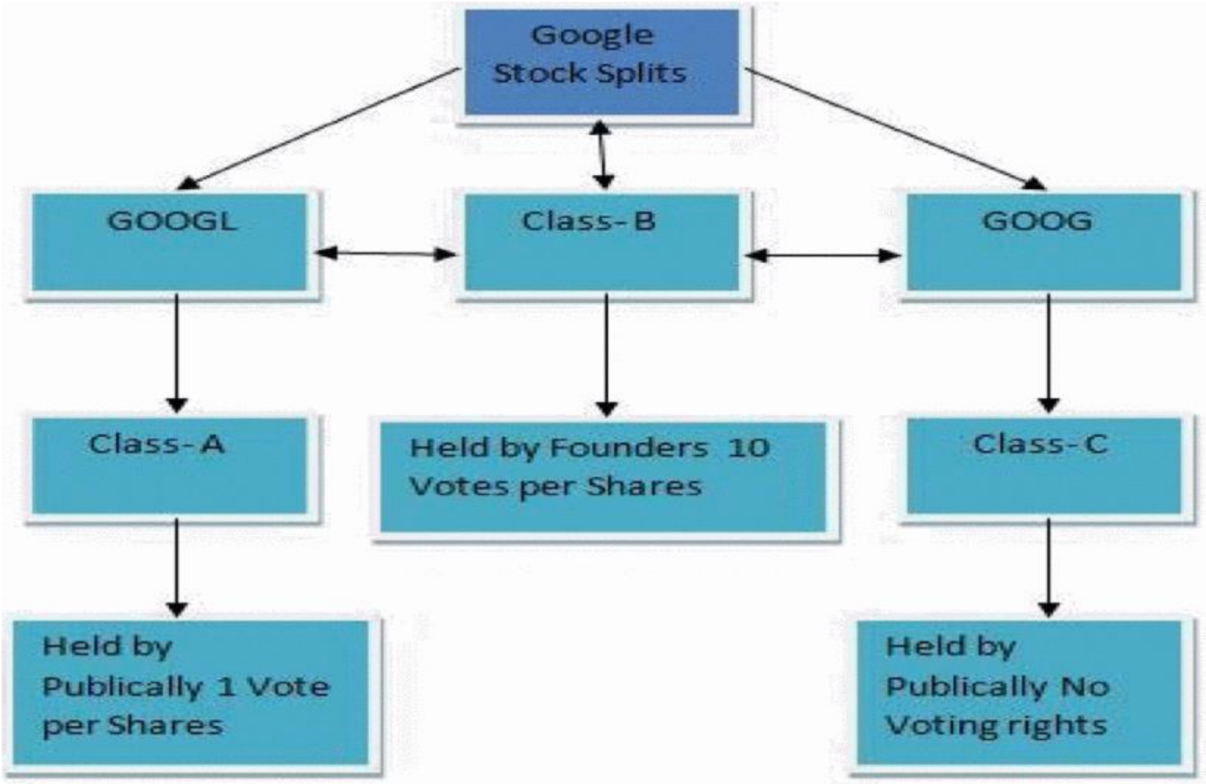

Google Stock Splits
GOOGL
Class-B
GOOG
Class-A
Held by Founders 10 Votes per Shares
Class-C
Held by Publically 1 Vote per Shares
Held by Publically No Voting rights

Figure 1:
Google stock split into classes

After the company's listing, the company's vision may be affected, because the vision is often forced to take second place for the benefit of shareholders. Markets and investors may turn a blind eye to immediate results, at the expense of long-term results. The share split is a way for Brin and Page to take advantage of the liquidity of the public market, while still retaining voting rights without losing control of the company.

Google split its stock in April 2014, creating A shares and C shares. As with any other one-to-one split, the number of shares has doubled and the price has fallen by half. However, there is a key difference. One share received one vote, C shares did not receive a vote, and B shares received 10 votes. Anyone holding A shares at the time of split receives the same number of C shares, but voting rights have not increased. There is a turning point in owning C shares. To allow some shareholders to oppose the initial spin-off, Google promised to compensate C shareholders when the share price of A-shares fell more than 1 % a year after its spin-off, which somewhat subsided Google's interest. Although the difference is not big, it does exist

1.3 Prediction

Prediction is a statement about a future event. Forecasts are generating information and guidance about the future. This is the fact or evidence that the guess usually supports. However, using a crystal ball for prediction is not always a fortune teller. Meteorologists use maps and scientific data to tell us about the possibility of rain, snow, or sunlight . Correct predictions of stocks can lead to huge profits for sellers and brokers. Often, it is brought out of predictions that are confusing rather than random, which means it can predict their stock market by carefully analyzing history. Machine learning is an effective way to represent such processes. It predicts that market value is close to tangible values, which improves accuracy. Bringing machine learning into the field of inventory prediction has attracted many types of research due to its predictive, efficient, and accurate measurements.

1.4 Deep Learning

Deep learning is a subfield of machine learning, involving algorithms that are inspired by human brain structure and function, called Artificial Neural Networks. It is used to advance qualitative and quantitative prediction analysis in many areas. In deep learning, computer model learning performs classification tasks directly from images, text, or sound. Deep learning models can achieve the highest level of accuracy, sometimes even exceeding human performance . The model is trained by using a large amount of tagged data and neural network architecture containing many layers. Deep learning is a machine learning technique that learns features and tasks directly from data. Data can be images, text, or sound.

1.5 Proposed Solution

Already different works have been done on Google stock price prediction and they find some important trends for prediction google stack price. Previously work had used different deep learning models like ANN and CNN for google stock movement prediction. they also used an approach for stock movement prediction on social media data by introducing Stock-Net, an ANN. They tested the model on a new dataset and showed that it has enhanced performance than other models. These datasets are publicly available on GitHub. Their proposed neural network architecture obtained an accuracy of 0.58 for stock movement prediction. In the existing research work, many approaches were proposed for the prediction of Google stock price and its movements. But they have limited accuracy for prediction.

This study aims to improve the calculation, method, and accuracy of Google stock prices and to propose an enhanced technique using Python Jupyter Notebook and to use a deep learning RNN model based on LSTM to predict the upward and downward movement of Google stocks with improved accuracy. In this paper, we proposed a simple technique for predicting Google stock price using a deep learning RNN for prediction and enhanced accuracy. The implementation of our proposed work will help the investors or stockholders to overcome the loss and get more profit. Form this research work stock shareholders or companies could take decisions on the time to invest or sell shares to reduce loss and getting more profit.

CHAPTER 2.

Literature Review

Ishita Parmar et al [1] focused on using regression and LSTM-based machine learning to predict stock value. The factors considered are open; close, Low, High, and volume. Zang Yeze et al [2] proposed a combined machine learning framework Information Theory and ANN. This method creatively uses information entropy to inform nonlinear causality and stock correlation and use it to Promote ANN time series modeling. The analysis of Google, Amazon, Facebook, and app stock prices proved the feasibility of this machine learning framework. [4] discussed dynamic correlation Change the algorithm to predict the volatility of the stock market. When an investor decides to buy or sell a stock, it depends on the rise or fall of the stock market price. This research use model-independent methods to reveal hidden dynamic stock market data using a variety of deep learning RNN, LSTM, and GRU technologies. Data from using the automobile and banking listed in the National Stock Exchange (NSEI) of India for this research. Sheikh Irfan Akbar et [8] analyzed some stock forecasting models. They also investigated the stock forecasting model. Jageshwer Shriwas et al [9] examine and analyzed the use of neural networks as predictive tools. Specifically, the ability of neural networks to predict future trends in the stock market index was tested. Compared the accuracy to traditional forecasting methods. Although it is a simple discussion of neural network theory, this study identifies the feasibility and practicality of using data mining as a predictor for individual investors. If the data mining algorithm has a reasonable amount of input data and the required output, it has a good ability to find hidden patterns and trends. As the number of input values increases, the quality of prediction increases. The extremely non-linear nature of stock market data makes it very difficult to design a system that can accurately predict the future direction of the stock market. Therefore, in this paper, they introduce a neural network-based financial predictor. Yumo Xu et al [10] done on Google stock price prediction and they find some important trends for google stack price. Previously has used some important techniques and model to predict google stock movement. They used different deep learning models like ANN and CNN for google stock movement prediction, they also used an approach for stock movement prediction on social media data by introducing Stock-Net, ANN. They tested the model on a new total dataset and showed its performance enhanced than other models, as well as the execution of previous research work. The total dataset is publicly available on GitHub.

CHAPTER 3.

Methodology and Tools

In this research work, we used Google Stock Price data from 2012 to 2016 from Kaggle. Kaggle is a scientific group of people where the data is openly accessible for solving the challenging research problem. Python supports great powerful libraries like Pandas, NumPy, Seaborn, Matplotlib, Keras, TensorFlow, etc. These are used for many supervised and unsupervised models. These libraries are used for all types of data, including numbers and categorical. The model that we implement LSTM using deep learning.

4.1. Google Stock Price Data Set

In data set Google Stock Price data set from the year 2012 to 2016 from Kaggle. This data set contains information about google stock price from the year 2012 to 2016 which contains open stock price, close stock price, high stock price, low stock price, and volume of the stock price. we are going to predict the next two months google stock price based on this data to use the previous two months (60 days). we try to predict the first two months of 2017 google stock price by using previous information for the last two months of 2016 and then to compare real google stock price with predicted google stock price and investigate the accuracy of our predicted results.

4.2. Overall Process Flow

Figure 2 shows the overall process flow of our proposed work. First, we imported the Dataset using python. Next, we are going to explain Figure 2 step by step.

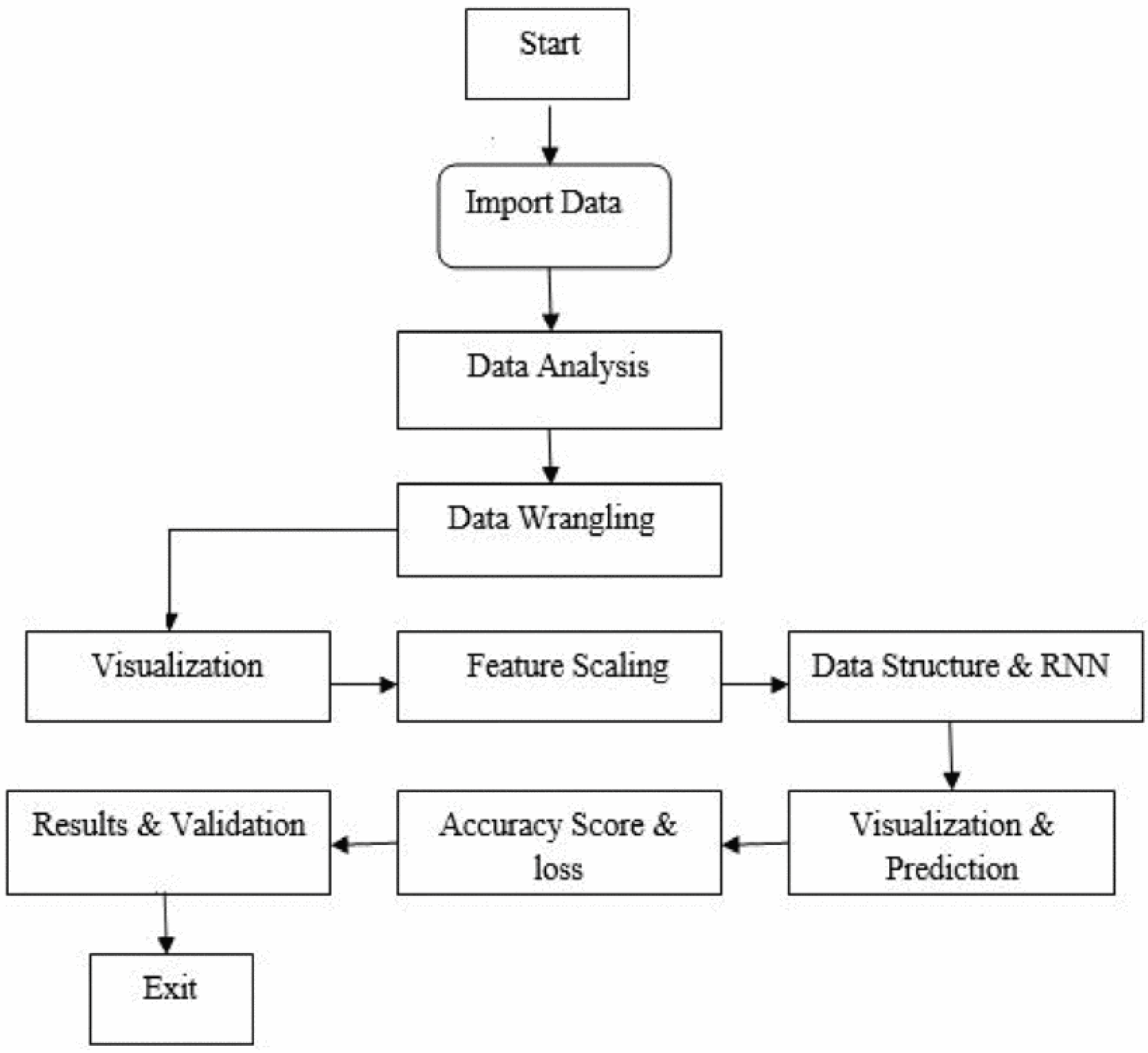
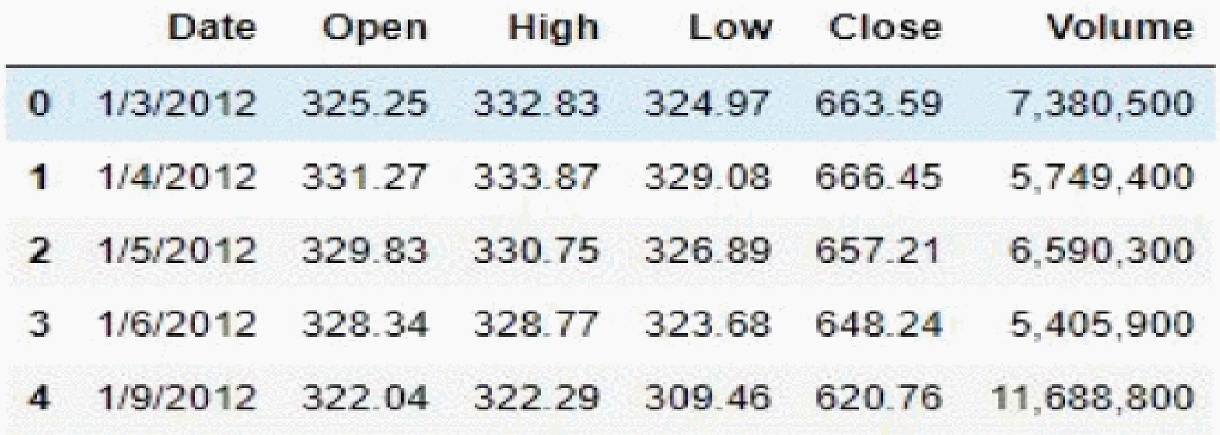


Figure 2:
Step of our mythology / over-all process flow

Step 1. Data Import

We imported the Dataset to Python Jupyter notebook by using libraries e.g. (libraries Pandas, NumPy, Matplotlib, Seaborn, and Math) Table 1 shows the imported data using python Jupyter notebook.

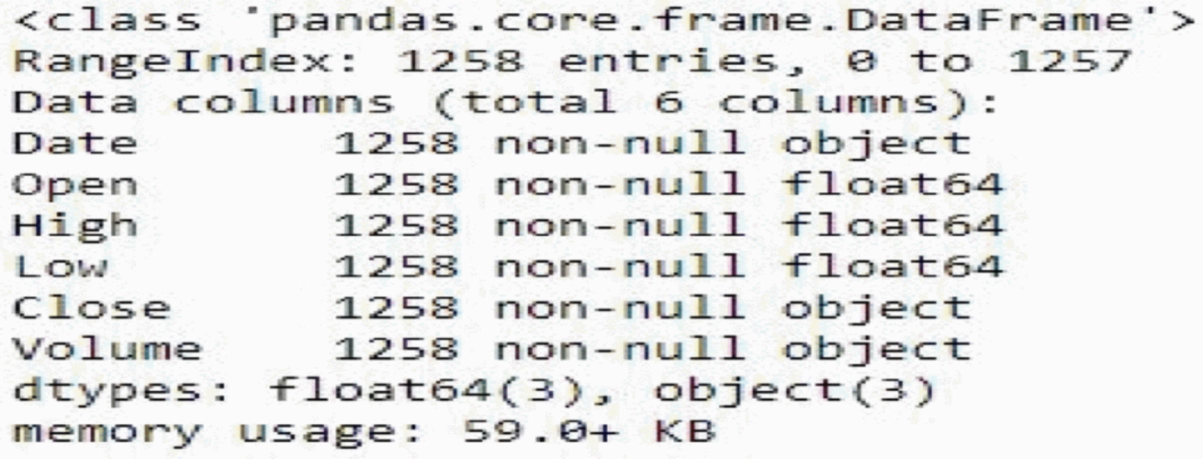
Table 1: Imported of data set in jupyter notebook



	Date	Open	High	Low	Close	Volume
0	1/3/2012	325.25	332.83	324.97	663.59	7,380,500
1	1/4/2012	331.27	333.87	329.08	666.45	5,749,400
2	1/5/2012	329.83	330.75	326.89	657.21	6,590,300
3	1/6/2012	328.34	328.77	323.68	648.24	5,405,900
4	1/9/2012	322.04	322.29	309.46	620.76	11,688,800

Step 2. Data Analysis and Dataset Description

In this step, we analyzed the data and collected some statistics of the dataset, we calculated the mean, max, and min values of the dataset. This data also tells us about the total number of rows, columns, and data types in the date set. The following figure shows the description of google stock price data.



```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 1258 entries, 0 to 1257
Data columns (total 6 columns):
Date        1258 non-null object
Open        1258 non-null float64
High        1258 non-null float64
Low         1258 non-null float64
Close       1258 non-null object
Volume      1258 non-null object
dtypes: float64(3), object(3)
memory usage: 59.0+ KB
```

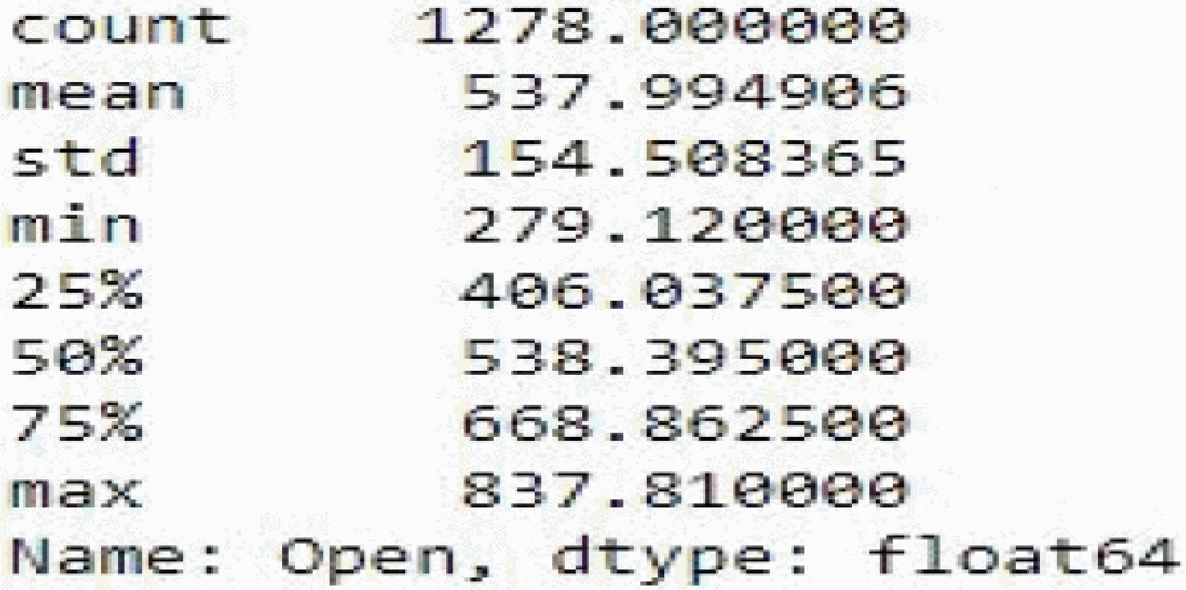
Figure 3:
Dataset description in a visual format

Figure 3 shows that this data set contains total numbers of 1258 entries of rows and 6 columns. Also, have three float and three objects of data types. It is the Google stock price data set contains six columns as following.

1. Date: The date of the recorded data
2. Open: The price when the stock market open
3. High: The highest price on that date
4. Low: The lowest price point on that date
5. Close: The close price on that date
6. Volume: Total Sale of stock on that date

Table 2 shows the statistical description of our data set.

Table 2: Dataset statistical description



```
count    1278.000000
mean      537.994906
std       154.508365
min       279.120000
25%       406.037500
50%       538.395000
75%       668.862500
max       837.810000
Name: Open, dtype: float64
```

The count value shows the total number of entries in our data set. The next is the mean value of our google stock price which is 537.99. Next is the standard deviation of 154.51. The minimum value of google stock price is 279.12 25% values are 406.03 50% values are 538.39 75% values are 668.86 and the maximum values are 837.81.

Step 3. Data Wrangling

Data wrangling is a data preprocessing technique used for data cleaning. In this step, we removed unnecessary columns, null rows, and unnecessary values, like month id. First, we generated a heatmap that represents graphically missing values in the data set as shown in Figure 4.

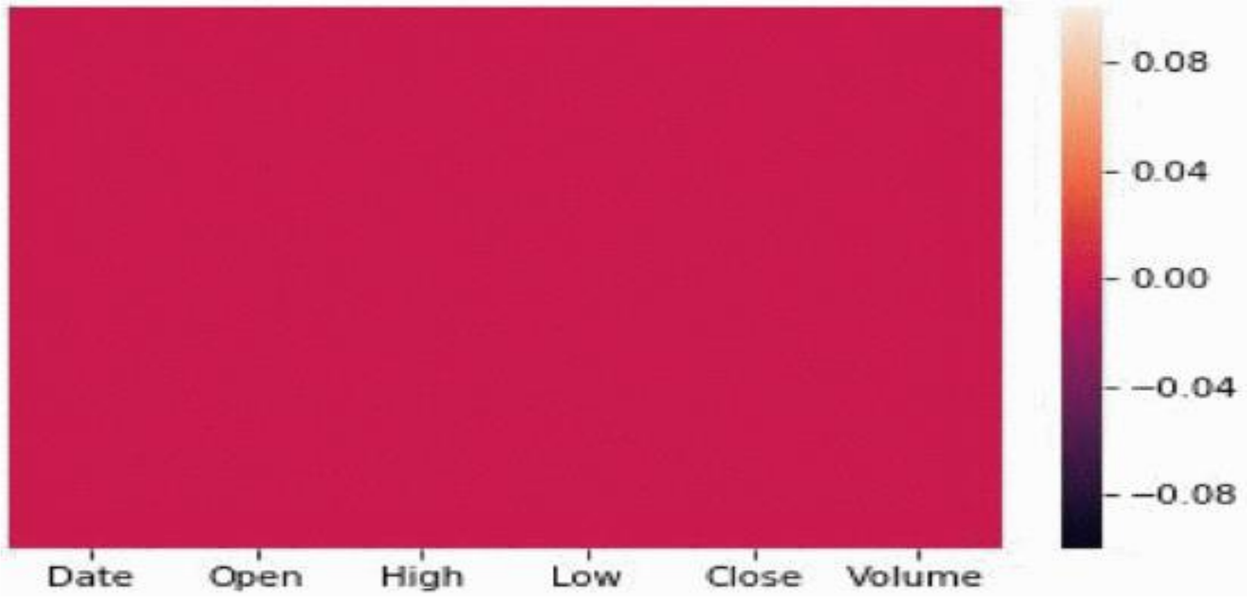


Figure 4:
Heat map of the dataset

Heatmap shows all columns of our data set. But there is no change in red color which means there are no missing values present in the heatmap therefore, it is very good for our model implementation.

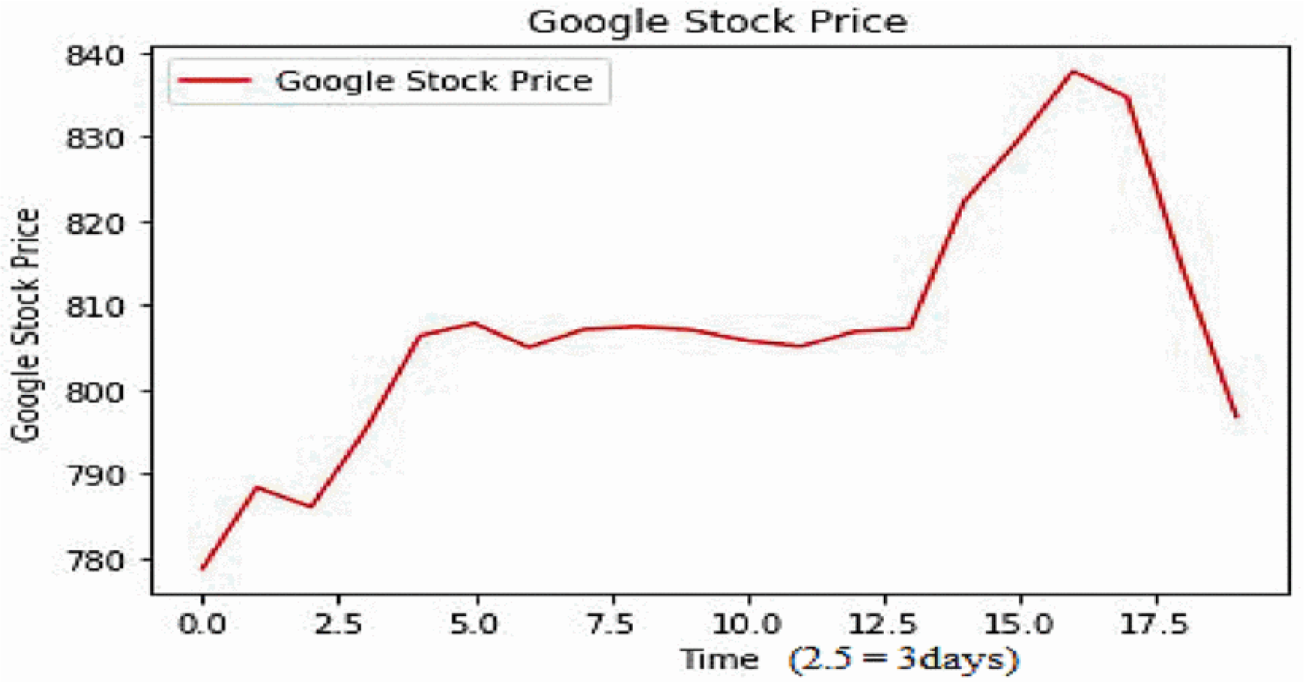


Figure 5:
Real google stock prices for nov and dec 2016.

Figure 5 shows the actual google stock price of the last two months of 2016 i.e Nov and Dec based on which we are going to predict google stock price for the first two months of 2017(Jan, Feb) by using our proposed model. The y-axis presents the prices of google stock. The x-axis shows the number of times the Google stock prices changed in two months. After obtaining prediction results, we will be comparing it with this real google stock price for the same period (Jan, Feb 2017). The next step is the feature scaling.

Step 5. Feature Scaling

Feature scaling is a technique for normalizing the existence of independent features in data over a fixed range. It is performed during data preprocessing to handle the magnitude or value or unit of height variation. If feature scaling is not done, the machine learning algorithm will weigh larger values, larger values, and treat smaller values as lower values, regardless of the unit of value. There are two ways for applying feature scaling. the first one is standardization and the second is normalization. In our proposed work we used the Normalization feature scaling technique for feature scaling. The formula in Equation 1 for normalization.

$$Xnew = Xi - \min(X)/\max(x) - \min(x) \quad ..(1)$$

Step 6. Data Structure Rnn

In this step, we created a special data structure for our RNN [3]. This can also be called the heart of RNN [3] and defines the total number of inputs to the RNN Data structure. That can understand the correlation of the data to try predicting some future trends based on these input data structure. RNN is a popular and powerful Supervised Deep Learning algorithm. The learning model will make a prediction and calculate the average forecast to better estimate the actual value. It is used for time series analysis like regression. RNN [3] has proven to be one of the most powerful models for processing sequential data. Long and short-term memory is one of the most successful RNN [3] architectures. LSTM introduces a storage unit, which is a computing unit that can replace traditional artificial neurons in the hidden layer of the network. Using these storage units, the network can effectively associate the memory and make remote input in time, so it is suitable for dynamically grasping the structure of data over time with high predictive ability.

Step 7. Accuracy Score or Loss

In this step, we calculate the accuracy of our proposed solution with the help of the RMSE formula. RMSE is the standard deviation of the residual (prediction error). Residuals are a measure of how far away are the regression line data points from the actual line. This low RMSE value means high prediction accuracy. Equation 2 represents the RMSE formula.

Step 8. Visualization and Prediction

The predicated Google stock prices were visualized and compared with actual Google stock prices for the same period.

4.3 Proposed Algorithm

In this work, we propose the RNN Algorithm with LSTM to predict google stock prices. It is the most popular and powerful deep learning supervised algorithm used to perform many decisions. When it needs to predict new data, each input layer of the RNN model makes a prediction and calculates an average prediction to better estimate the actual output value. RNN is used for time series analysis, such as regression analysis. Figure 6 shows the overall concept of RNN[3].

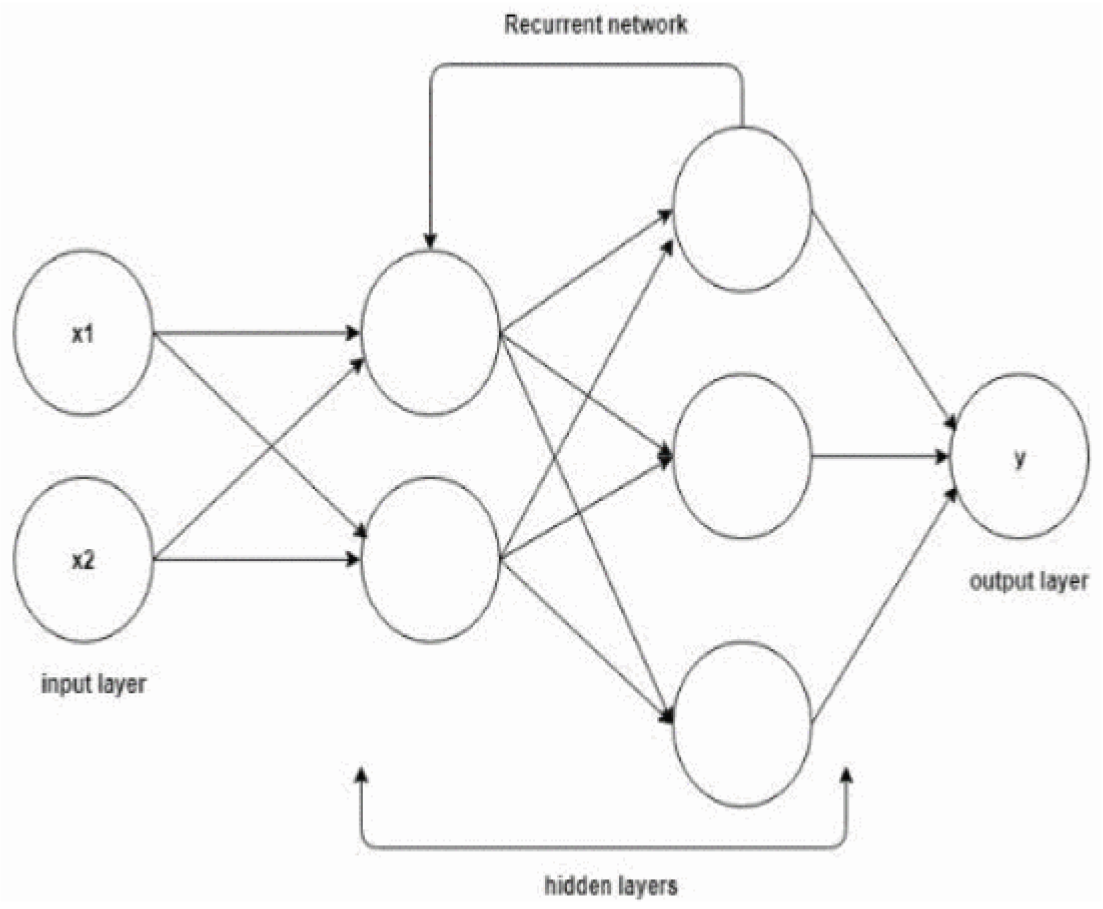
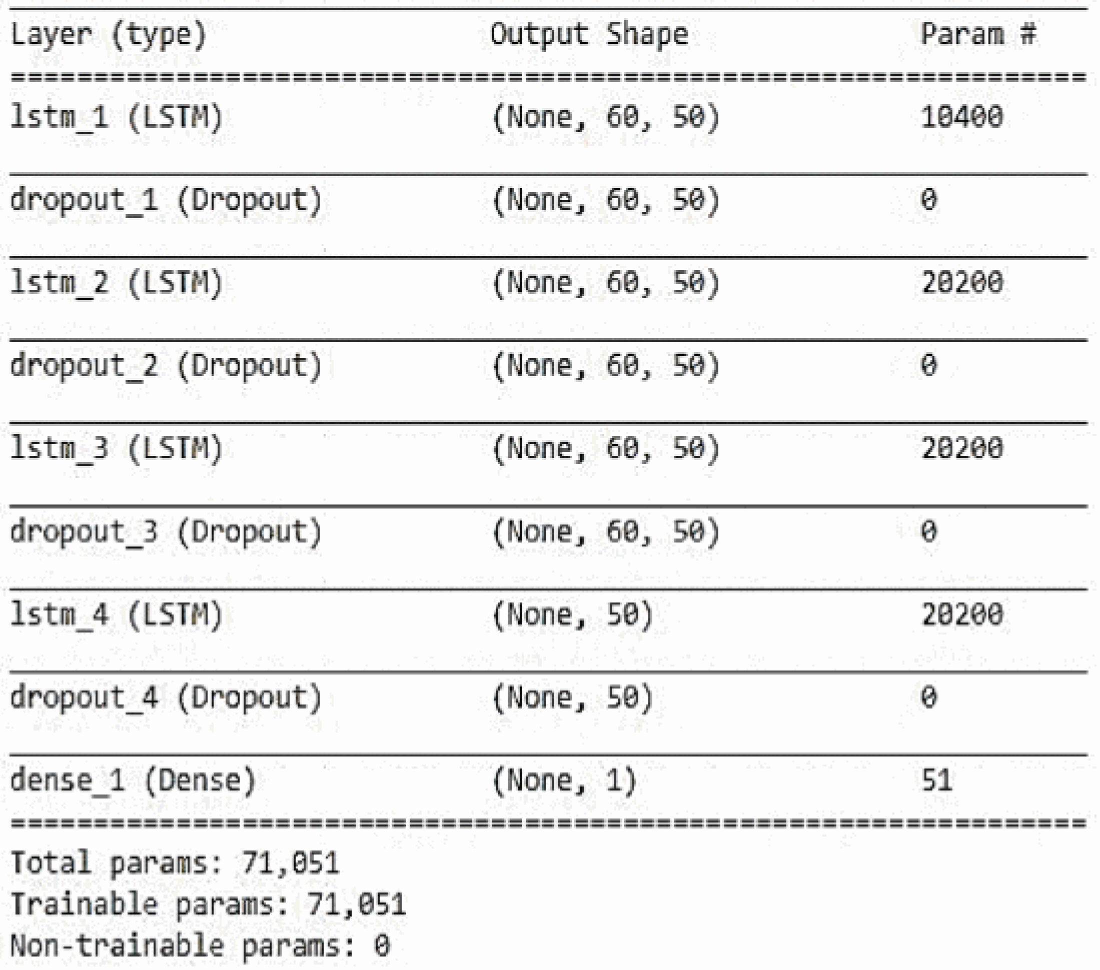


Figure 6:
RNN input-output layers

Figure 6 shows the input, output, and hidden layer. In hidden layer is looping networks and uses LSTM. The Long-Term and Short-Term Memory network is a modified version of the RNN that makes it easier to remember past data. The gradient problem of RNN disappearance is solved here. LSTM is ideal for classifying, processing, and predicting time series for a given time lag with unknown duration. There are many types of input and output methods using the RNN model. For example, one to many and many to many, etc. the following figure contains the execution of an RNN and its hidden layers LSTM.



```
_________________________________________________________________
Layer (type)                 Output Shape              Param #
=================================================================
lstm_1 (LSTM)                (None, 60, 50)            10400
_________________________________________________________________
dropout_1 (Dropout)          (None, 60, 50)            0
_________________________________________________________________
lstm_2 (LSTM)                (None, 60, 50)            20200
_________________________________________________________________
dropout_2 (Dropout)          (None, 60, 50)            0
_________________________________________________________________
lstm_3 (LSTM)                (None, 60, 50)            20200
_________________________________________________________________
dropout_3 (Dropout)          (None, 60, 50)            0
_________________________________________________________________
lstm_4 (LSTM)                (None, 50)                20200
_________________________________________________________________
dropout_4 (Dropout)          (None, 50)                0
_________________________________________________________________
dense_1 (Dense)              (None, 1)                 51
=================================================================
Total params: 71,051
Trainable params: 71,051
Non-trainable params: 0
```

Figure 7:
LSTM layers and its input/output

Figure 7 shows the total number 4 hidden layers of LSTM and one output layer that means it is many to one method which we have used in our RNN [3]. The total number of 60 output generates every LSTM layer because we going to predict 60 days of future trends of google stock price. Neri Kiml used formulas for training RNN data structure [11]. The total number of 71,051 parameters is train and 0 non-train that means we build our RNN [3] model very well. If the model not sufficient that us a non-trainable parameter.

CHAPTER 5

IMPLEMENTATION

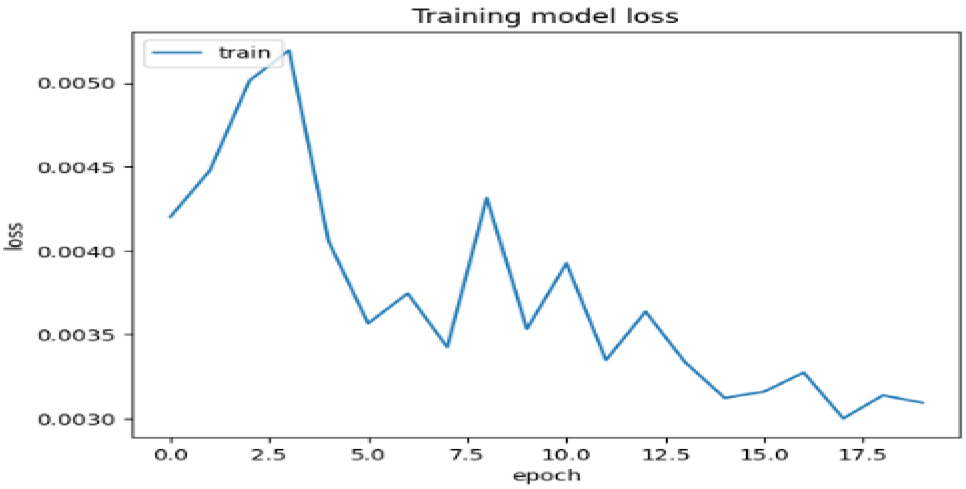
```
#imporitng Libraries
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
from sklearn.preprocessing import MinMaxScaler
from keras.models import Sequential
from keras.layers import Dense,LSTM,Dropout

#Mount google drive to google colab
from google.colab import drive
drive.mount("/content/gdrive")

#import training dataset from google drive
StockData = pd.read_csv("/content/gdrive/My Drive/Google stock price dataset/Google_train_data.csv")
StockData.head()  #Gives first five rows of the dataset

#data Visualization
StockData.info()
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 1258 entries, 0 to 1257
Data columns (total 6 columns):
 #   Column  Non-Null Count  Dtype
---  ------  --------------  -----
 0   Date    1258 non-null   object
 1   Open    1258 non-null   float64
 2   High    1258 non-null   float64
 3   Low     1258 non-null   float64
 4   Close   1258 non-null   object
 5   Volume  1258 non-null   object
dtypes: float64(3), object(3)
memory usage: 59.1+ KB
StockData["Close"]=pd.to_numeric(StockData.Close,errors='coerce')
StockData = StockData.dropna()
trainData = StockData.iloc[:,4:5].values
#data Visualization after process
StockData.info()
```

```
<class 'pandas.core.frame.DataFrame'>
Int64Index: 1149 entries, 0 to 1257
Data columns (total 6 columns):
 #   Column  Non-Null Count  Dtype  
---  ------  --------------  -----  
 0   Date    1149 non-null   object 
 1   Open    1149 non-null   float64
 2   High    1149 non-null   float64
 3   Low     1149 non-null   float64
 4   Close   1149 non-null   float64
 5   Volume  1149 non-null   object 
dtypes: float64(4), object(2)
memory usage: 62.8+ KB
```

```
sc = MinMaxScaler(feature_range=(0,1)) # MinMaxScaler
object called "sc". The "feature_range" parameter is set
to (0, 1), which means that the data will be scaled to
values between 0 and 1.
trainData = sc.fit_transform(trainData)
trainData.shape
X_train = []
y_train = []

for i in range (60,1149): #60 : timestep // 1149 : length
of the data
    X_train.append(trainData[i-60:i,0])
    y_train.append(trainData[i,0])

X_train,y_train = np.array(X_train),np.array(y_train)
X_train =
np.reshape(X_train,(X_train.shape[0],X_train.shape[1],1))
#adding the batch_size axis
#1 represents the number of features in the input data (in
this case, there is only one feature).
X_train.shape
#The model consists of four LSTM layers and a dense output
layer.
model = Sequential()

model.add(LSTM(units=100, return_sequences = True,
input_shape =(X_train.shape[1],1)))
model.add(Dropout(0.2))
model.add(LSTM(units=100, return_sequences = True))
model.add(Dropout(0.2))

model.add(LSTM(units=100, return_sequences = True))
model.add(Dropout(0.2))

model.add(LSTM(units=100, return_sequences = False))
model.add(Dropout(0.2))

model.add(Dense(units =1))
model.compile(optimizer='adam',loss="mean_squared_error")
#epochs: The number of times to iterate over the entire
training dataset.
#batch_size: The number of samples to use in each batch
for training. The model weights are updated after each
batch.
#verbose: The level of logging during training. A value of
2 means that progress bars will be displayed for each
epoch.
train_model = model.fit(X_train, y_train, epochs = 20,
batch_size = 32, verbose=2)
```

Output exceeds the size limit. Open the full output data in a text editor

```
plt.plot(train_model.history['loss'])
plt.title('Training model loss')
plt.ylabel('loss')
plt.xlabel('epoch')
plt.legend(['train'], loc='upper left')
plt.show()
```



```
#import test dataset from google drive
```

```
testData = pd.read_csv("/content/gdrive/My Drive/Google 
stock price dataset/Google_test_data.csv")
testData["Close"]=pd.to_numeric(testData.Close,errors='coe
rce')
testData = testData.dropna()
testData = testData.iloc[:,4:5]

y_test = testData.iloc[60:,0:].values
#input array for the model
inputClosing = testData.iloc[:,0:].values
inputClosing_scaled = sc.transform(inputClosing)
inputClosing_scaled.shape
X_test = []
length = len(testData)
timestep = 60
for i in range(timestep,length):
    X_test.append(inputClosing_scaled[i-timestep:i,0])
X_test = np.array(X_test)
X_test = 
np.reshape(X_test,(X_test.shape[0],X_test.shape[1],1))
X_test.shape
#stock price prediction
y_pred = model.predict(X_test)
y_pred
#This line of code transforms the predicted output values 
from their scaled range back to their original scale.
predicted_price = sc.inverse_transform(y_pred)
#compare the visualization of the predicted stock price 
and actual stock price graph by using matplotlib
plt.plot(y_test, color = 'red', label = 'Actual Stock 
Price')
plt.plot(predicted_price, color = 'green', label = 
'Predicted Stock Price')
plt.title('Google stock price prediction')
plt.xlabel('Time')
plt.ylabel('Stock Price')
plt.legend()
plt.show()
```

6. Results

After applying an RNN to make some prediction we have converted this prediction to graphical form to compare the results of the prediction with the real result. The following Figure 8 shows our final prediction and comparison with real google stock price.

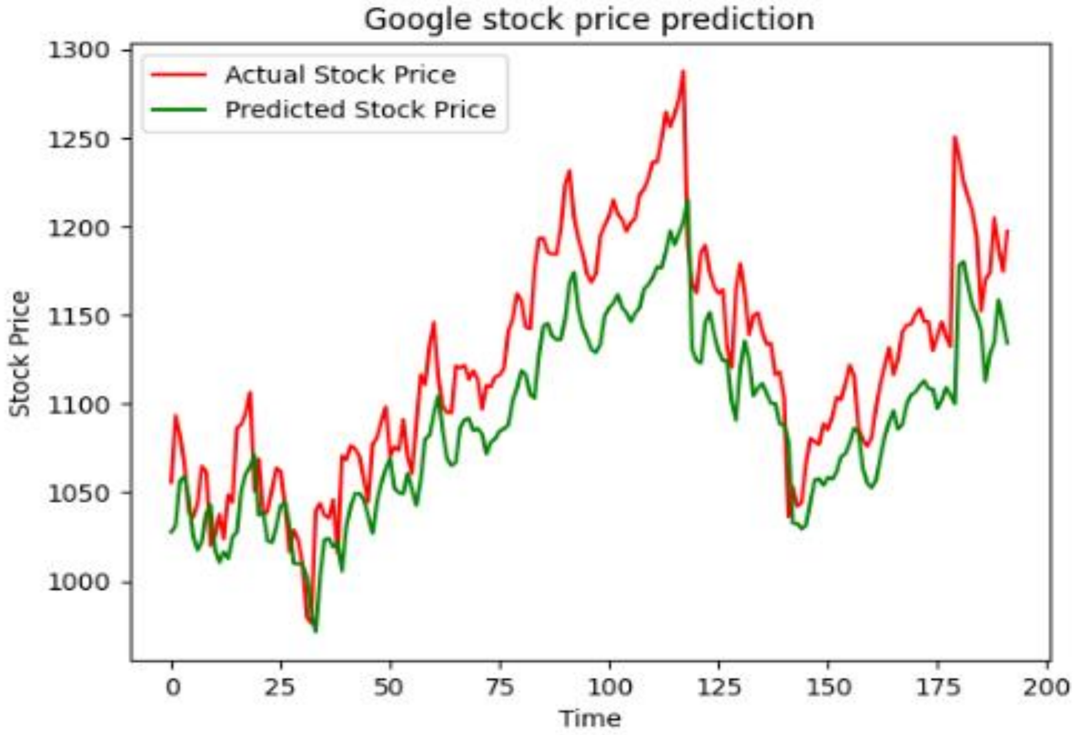
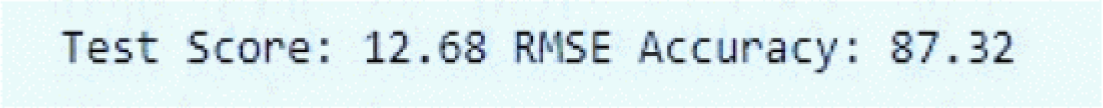


Figure8:
Real google stock price vs predicted google stock price

Figure 8 shows both the real google stock price and predicted Google stock price. The red line shows the real google stock price while the blue line shows the predicted Google stock price. The y-axis presents the prices of google stock. The x-axis shows the number of times of Google stock prices changed in two months.

6.1 Model Performance

In this section, we report the performance of our model by calculating its accuracy score and overfitting loss. The following figure shows the test score error and accuracy.



```
Test Score: 12.68 RMSE Accuracy: 87.32
```

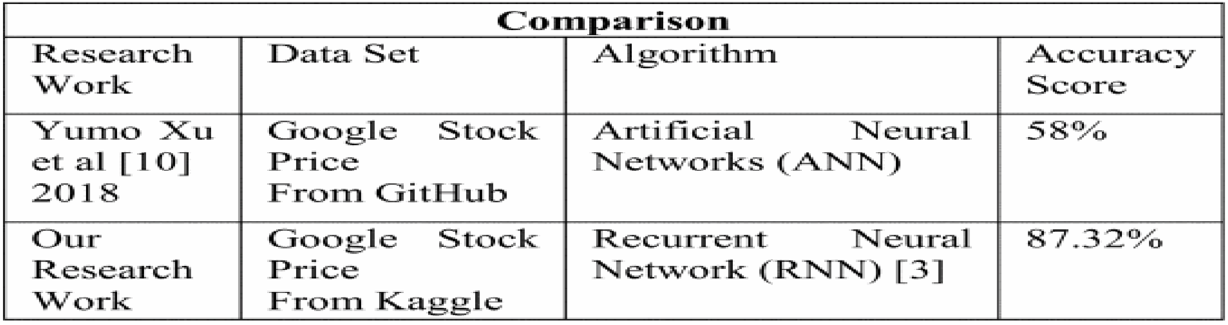
Figure 9:
Accuracy score of the model

The RMSE tells us how the data is focused on what fits best. Figure 9 shows that there is a total of 12.68 % percent Root Mean Square Error in prediction and thus the accuracy of our proposed work is 87.32 % percent. which means that our predicted results are 87.32 % percent similar to the actual google stock prices of Jan, Feb 2017.

6.2 Validation and Comparison

We evaluated and compared our results with existing research work and for validation and comparison. Table 3 provides a comparison of our approach with the work of Yumo Xu et al [10]2018 on Google Stock Predication has an accuracy score of 58%. This means that our predictions are 29.32% more accurate compared to previous work on Google stock price prediction.

Table 3: Comparison with existing research



Comparison			
Research Work	Data Set	Algorithm	Accuracy Score
Yumo Xu et al [10] 2018	Google Stock Price From GitHub	Artificial Neural Networks (ANN)	58%
Our Research Work	Google Stock Price From Kaggle	Recurrent Neural Network (RNN) [3]	87.32%

CHAPTER 7.

Conclusion

There is Brownian's movement in Google stock price which stipulates the future valuations of stock prices are independent of the past. Therefore, it is virtually impossible to accurately predict future stock prices. But some trends can be predicted. Therefore, we have tried to predict the upward and downward trends that exist in the stock price of Google. In our proposed work we used google stock price data from Kaggle from the year 2012 to 2016. We predicted the first two months Jan, Feb 2017 of google stock price by using previous information. The tool we used for this proposed work is the python Jupyter- notebook. Through data visualization, we made our predictions. We used normalized feature scaling techniques for feature scaling. We create a special data structure for the RNN and define the total number of input structure steps for it. The relevance of the data can be understood in an attempt to predict certain future trends based on these input data structures. In the data structure, we create a data structure with 60 steps and 1 output. Then build an RNN for the features we selected. This is the most popular and powerful deep learning supervised algorithm and especially used for time series data, such as regression analysis. After building the model, we generate predictions and present the results graphically. We calculate the RSME and the accuracy of the proposed work. The results show an RMSE of 12.68% for prediction with an accuracy of 87.32%. which means that our predictions are 87.32% similar to actual Google's stock prices.

SECTION 8.

Future Research Directions

In our research, we used advanced methods and RNN to predict Google's stock price [3] and achieved good results. We also proposed an advanced step-by-step method to predict Google's stock price. But in the future, more effort may lead to higher accuracy. It is important to test different models in Google stock prediction and try to reduce the RMSE to get more accurate results. It is also important to implement different deep learning models, find more reliable models for time series data, or use the latest algorithms or combinations of algorithms to obtain more accurate predictions.

References

1.Rami, C. (2022, September 14). Could google trade lower? exploring the bear case arguments. Retrieved September 17, 2022, from https://seekingalpha.com/article/4541035-could-google-trade-lowerexploring-bear-case

2. Ali, R. (2020, October 15). The Why Behind Financial Forecasting. Retrieved September 16, 2022, from https://www.netsuite.com/portal/resource/articles/financial-management/importance-financialforecasting.shtml

3. King, T. (2020). Detailed in Linear Regression model. Retrieved September 17, 2022, from https://blog.csdn.net/iqdutao/article/details/109402570

4. https://www.researchgate.net/publication/366296303_Stock_Price_Prediction_of_Google_based_on_Machine_Learning

5. https://ieeexplore.ieee.org/document/9265146