INVESTING GURU

A STEP TOWARDS FINANCIAL LITERACY

Amartya Vibhu 201413

Akash Rathore 201425

Table of Content

1. Acknowledgement 2

2. Project Overview 3

3. Goals 4

Analysis

---

4. Data Exploration 5

5. Algorithms and Techniques 7

Methodology

---

6. Implementation 8

Outputs

---

7. Results 13

8. Conclusion 15

9. Improvement 16

10. Reference 17

Acknowledgement

I would like to express my special thanks to our mentor Dr. Aman Sharma, who gave us the golden opportunity to do this wonderful project on the topic Investing Guru: A Step towards Financial Literacy, which also helped me in doing a lot of Research and I came to know about so many new thing, for his time and efforts he provided throughout the semester. Your useful advice and suggestions were really helpful to me during the project's completion. In this aspect, I am eternally grateful to you.

Project Overview

Investment firms and individuals have been using financial models to better understand market behavior and make profitable investments and trades. A wealth of information is available in the form of historical stock prices and company performance data, suitable for machine learning algorithms to process. Even countries' economics depends on the situation of the market. If the market is volatile the economy will see inflation affects in the coming days or if the market is non-volatile then the economy will have a substancial growth in all the sectors of profit. Studying the Stock market and investing in market is one a great skill that every citizen should do. This not only makes their money more but also helps the companies to grow.

We have this idea of a trading floor being filled with adrenaline infused men with loose ties running around yelling something into a phone but these days they're more likely to see rows of machine learning experts quietly sitting in front of computer screens. In fact about 70% of all orders on Wall Street are now placed by software, we're now living in the age of the algorithm. Many applications such as zerodha, upstocks teach stock market analysis on the basis of algorithms only. About 25-30% of people who invest heavily in the stock market use old and orthodox methods of predicting the market prices, but the reality is that prices predicted by the algorithms are more accurate and precise.

This project seeks to utilize Deep Learning models, Long-Short Term Memory (LSTM) Neural Network algorithm, to predict stock prices. Unlike standard FFNN(feed-forward neural networks), the LSTM has feedback connections. It can not only process single data points, but also entire sequences of data. It is widely used for the problems of sequence prediction and has been very effective. Use of many libraries and functions made the project complete. The algorithms used in this made the prediction easy to predict the upcoming value of the stock price.

Specifications:

- Use of pandas_datareader
- Use of matplotlib
- Use of keras

Goals

1. Analyzing the stock prices:

   Analyzing the stock market means that we have to study the trends of the market. The market is more often a non-linear market but if we see the current situation of the market i.e market is too volatile so there is a lot of variation of prices in the market. It may fall or rise at any time. Using LSTM we can overcome this problem.

2. Implementation of LSTM using keras library:

   Keras is a free open source python library for developing and evaluating deep learning models. It may be called a high level API, providing essential abstraction and building blocks.

3. Comparison of the result with old market price:

   We not only predict the value of market prices of the future but also compare the prices with the old line of prices. We see very less deviation in the prediction. The predicted prices are all around the old prices and got them by using old prices.

4. To inform people of the updated market price:

   People should have the knowledge of the current market values of the company. The customer review also plays a major role in the stock market. If the current value of the company's market price is good then the quality of the product they sell will be good.

Analysis

Data Exploration

The data used in the project is of **YAHOO! FINANCE** from **January 1,2020 to January 1, 2022**. Data was taken on the day to day live value system. Yahoo! Finance provided all time stock price data. We got the listed prices of the company as well as the daily opening, closing values of the stock prices. Not only the opening and closing values but also the high and low values were provided. To predict the value of future stock prices we use the closing price of the day. Closing price is usually preferred because the upcoming stock price depends on how last day has ended, iff there is no casualty or anything that happened in the current world.

Date	High	Low	Open	Close	Volume	Adj Close
2010-01-04	540.427307	506.127411	540.427307	532.700500	35372156.0	484.265961
2010-01-05	569.551208	527.697937	569.551208	530.323059	9872785.0	482.104767
2010-01-06	542.111328	530.298279	534.879883	538.891846	10933743.0	489.894470
2010-01-07	552.265076	533.938782	538.891846	547.832092	12090336.0	498.021820
2010-01-08	551.670715	542.854309	548.797974	546.395691	6973331.0	496.716034

The image above tells the stock price date of **RELIANCE** company. The image shows the value of **High, Low, Open, Close** and **Volume**. With the following data we can calculate the mean, standard deviation, max and min using the mean(), MinMaxScaler() functions.

When it comes to check whether the values we need to check the graph too. Graph is easily accessible by the user. The certain variations can only be seen in the graphs. Graphical representation of the stock prices is as important as teaching algebra to students in mathematics. It shows the trend of the market. To show the graph we used the library such as matplotlib and functions such as plot(), xlabel(), ylabel(), title(), etc.

Let's have a quick review of **RELIANCE** shares from **January 1, 2010 to April 1, 2022.**

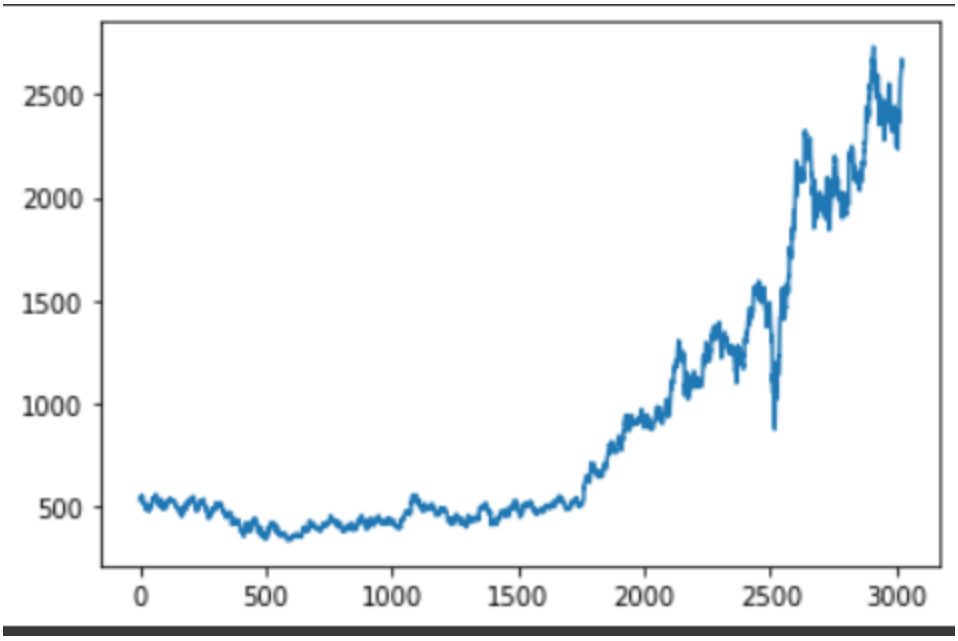


Fig: RELIANCE shares

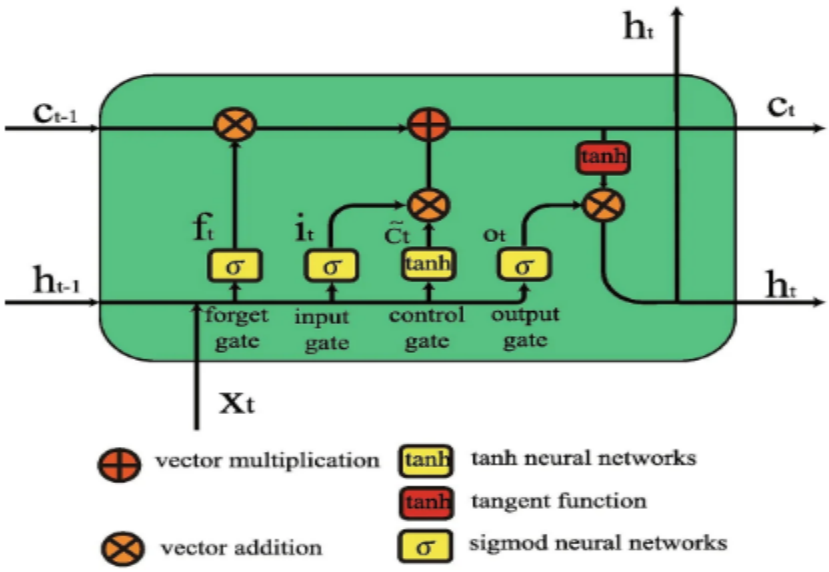
X-axis shows the ***Time/Trading Days***.

Y-axis shows the ***Closing price of the Stock in Rupees***.

Algorithms and Techniques

Stock price prediction using machine learning and deep learning is the process of predicting the future value of stock prices. With a daily challenge of market trends changing as well as everyday news, political situations the prediction of the stock market is tough. High accuracy is needed in stock price prediction because without high accuracy anyone can lose a huge amount of money.We should treat the stock price data as time-series. Using **Recurrent Neural networks(RNNs)** or **LSTMs** we can work on predicting the values more accurately. The machine learning Models assign weights to each market feature. This determines and reads the history of that company to analyze and predict the future value.
LSTM is a Recurrent Neural Network that works on data sequences, learning to retain only relevant information from a time window. New information the network learns is added to a "memory" that gets updated with each timestep based on how significant the new sample seems to the model. Over the years, LSTM has revolutionized speech and handwriting recognition, language understanding, forecasting, and several other applications that have become the new normal today.

A standard LSTM cell comprises three gates: the input, output, and forget gate. These gates learn their weights and determine how much of the current data sample should be remembered and how much of the past learned content should be forgotten. This simple structure is an improvement over the previous and similar RNN model.



Methodology

Implementation

1. Importing Libraries and Modules:

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

from sklearn.preprocessing import MinMaxScaler
from keras.models import Sequential
from keras.layers import Dense, Dropout, LSTM
```

For the project we will use:

- numpy: To perform a wide variety of mathematical operations.
- pandas: To work with relational or labeled data both easily and intuitively
- pandas_datareader: It allows to create a pandas DataFrame object by using various data sources from the internet
- sklearn: High-level API, user-friendly, used for prediction algorithms
- keras: supports multiple backend neural network computation.

2. Preparation of Data:

```
company = input("Enter the Company you want to predict price: ")

start = dt.datetime(2020,1,1)
end = dt.datetime(2022,1,1)

data = web.DataReader(company, 'yahoo', start, end)
```

company here will accept the value from the user of the stock price he wants to know.

start and end with datetime() function will hold the value of starting and ending dates of the dataset the user wants to know.

data will carry all the dataset sent by the pandas_datareader from the YAHOO! FINANCE. web here is used as pandas_datareader as it was declared in the import section.

pandas_datareader allows us to read the dataset using the following functions:

- head():

```
df.head()
```

Date	High	Low	Open	Close	Volume	Adj Close
2010-01-04	540.427307	506.127411	540.427307	532.700500	35372156.0	484.265961
2010-01-05	569.551208	527.697937	569.551208	530.323059	9872785.0	482.104767
2010-01-06	542.111328	530.298279	534.879883	538.891846	10933743.0	489.894470
2010-01-07	552.265076	533.938782	538.891846	547.832092	12090336.0	498.021820
2010-01-08	551.670715	542.854309	548.797974	546.395691	6973331.0	496.716034

- tail()

```
df.tail()
```

Date	High	Low	Open	Close	Volume	Adj Close
2022-03-28	2629.750000	2586.500000	2610.000000	2621.949951	4564891.0	2621.949951
2022-03-29	2638.000000	2607.399902	2638.000000	2622.550049	4007695.0	2622.550049
2022-03-30	2688.000000	2617.100098	2639.899902	2672.949951	7297028.0	2672.949951
2022-03-31	2669.699951	2628.600098	2664.949951	2634.750000	6102744.0	2634.750000
2022-04-01	2665.149902	2622.000000	2636.000000	2655.850098	3656408.0	2655.850098

3. Understanding the Dataset:

```
scaler = MinMaxScaler(feature_range=(0,1))
scaled_data = scaler.fit_transform(data['Close'].values.reshape(-1,1))
```

Here we will be sorting the dataset so that it will be easy to understand.

We can further display the dataset using head() or tail().

4. Mean of the dataset:

```
ma50 = df.Close.rolling(50).mean()

ma100 = df.Close.rolling(100).mean()
ma200 = df.Close.rolling(200).mean()
```

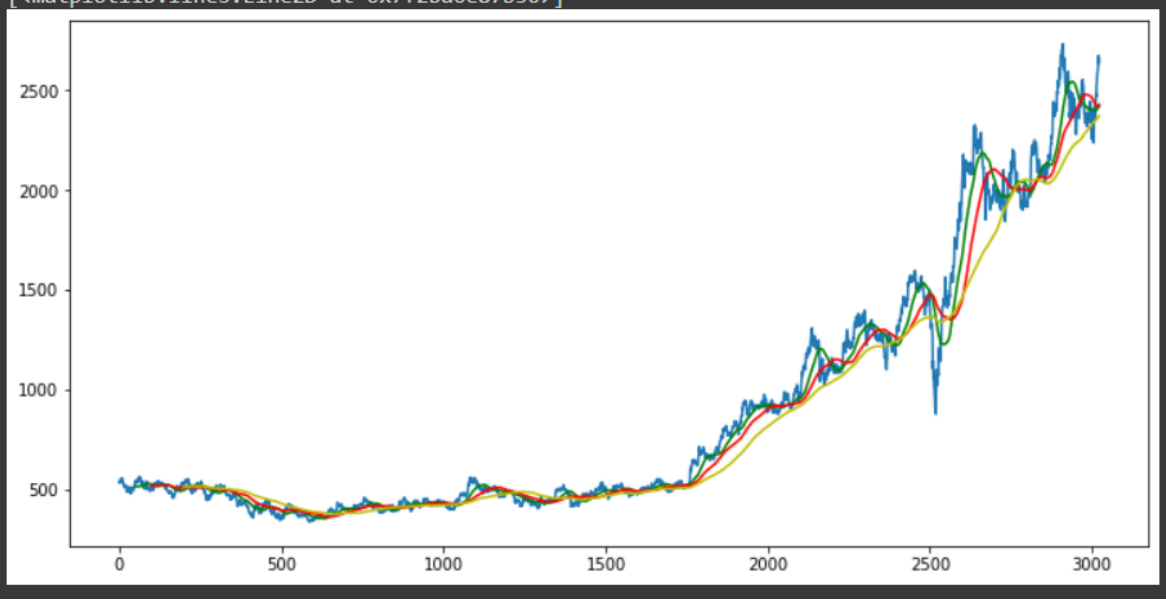
Mean can be calculated by the following functions. This gives us the idea how the market is moving and also provides trend.

5. Visualizing the Stock Data:

To visualize the data we need to use matplotlib. Here we can label the x-axis, y-axis and many more.

```
[ ] plt.figure(figsize=(12,6))
    plt.plot(df.Close)
    plt.plot(ma50, 'g')
    plt.plot(ma100, 'r')
    plt.plot(ma200, 'y')
```

```
[<matplotlib.lines.Line2D at 0x7f2ba0e87b50>]
```



6. Building LSTM Model and Compiling:

```
prediction_days = 100

x_train = []
y_train = []

for x in range(prediction_days, len(scaled_data)):
    x_train.append(scaled_data[x-prediction_days:x, 0])
    y_train.append(scaled_data[x, 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))

model = Sequential()

model.add(LSTM(units=50, return_sequences = True, input_shape=(x_train.shape[1], 1)))
model.add(Dropout(0.2))
model.add(LSTM(units=50, return_sequences = True))
model.add(Dropout(0.2))
model.add(LSTM(units=50))
model.add(Dropout(0.2))
model.add(Dense(units=1))

model.compile(optimizer='adam', loss ='mean_squared_error')
model.fit(x_train, y_train, epochs=25, batch_size=32)
```

LSTM will have three layers with 50 neurons each, one dense layer and three layers of dropout. The LSTM model is compiled using mean squared error loss function and adam optimizer.

```
Epoch 1/2
13/13 [==============================] - 5s 72ms/step - loss: 0.0506
Epoch 2/2
13/13 [==============================] - 1s 71ms/step - loss: 0.0131
```

7. Testing the model on testing data:

```
'''Test the model accuaracy on Existing data'''

test_start = dt.datetime(2020,1,1)
test_end = dt.datetime.now()

test_data = web.DataReader(company, 'yahoo', test_start, test_end)
actual_prices = test_data['Close'].values

total_dataset = pd.concat((data['Close'], test_data["Close"]), axis=0)

model_inputs = total_dataset[len(total_dataset)-len(test_data)-prediction_days:].values
model_inputs = model_inputs.reshape(-1,1)
model_inputs = scaler.transform(model_inputs)

x_test = []

for x in range(prediction_days, len(model_inputs)):
    x_test.append(model_inputs[x-prediction_days:x, 0])

x_test = np.array(x_test)
x_test = np.reshape(x_test, (x_test.shape[0], x_test.shape[1],1))

predicted_prices = model.predict(x_test)
predicted_prices = scaler.inverse_transform(predicted_prices)
```

The above code will test the LSTM model.

8. Making Prediction and Plotting graph:

```
plt.plot(actual_prices, color="black", label=f"Actaul {company} Price")
plt.plot(predicted_prices, color="green", label=f"Predicted {company} Prices")
plt.title(f"{company} Share Price")
plt.xlabel('Time')
plt.ylabel(f'{company} Share Price')
plt.legend()
plt.show()

real_data = [model_inputs[len(model_inputs) + 1 - prediction_days:len(model_inputs+1), 0]]
real_data = np.array(real_data)
real_data = np.reshape(real_data, (real_data.shape[0], real_data.shape[1],1))

prediction = model.predict(real_data)
prediction = scaler.inverse_transform(prediction)
print(f"Prediction:{prediction}")
```

The final step is to plot and visualize the data. To visualize the data we use these basic functions like title, label, plot as per how we want our graph to look like.

Outputs

Results

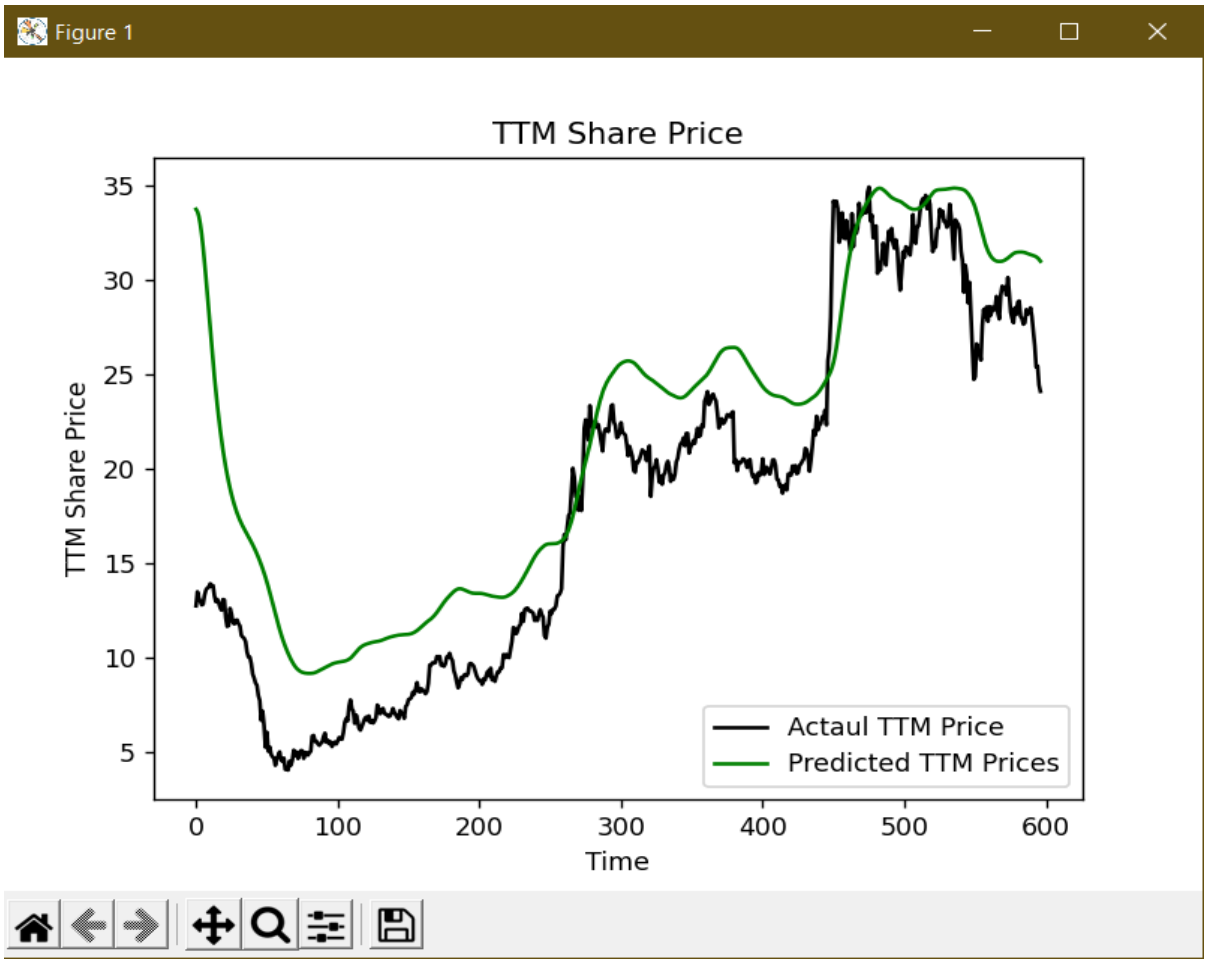


Fig: Shows the Graph of Tata-Motors

Similarly we can get the graph of our companies:

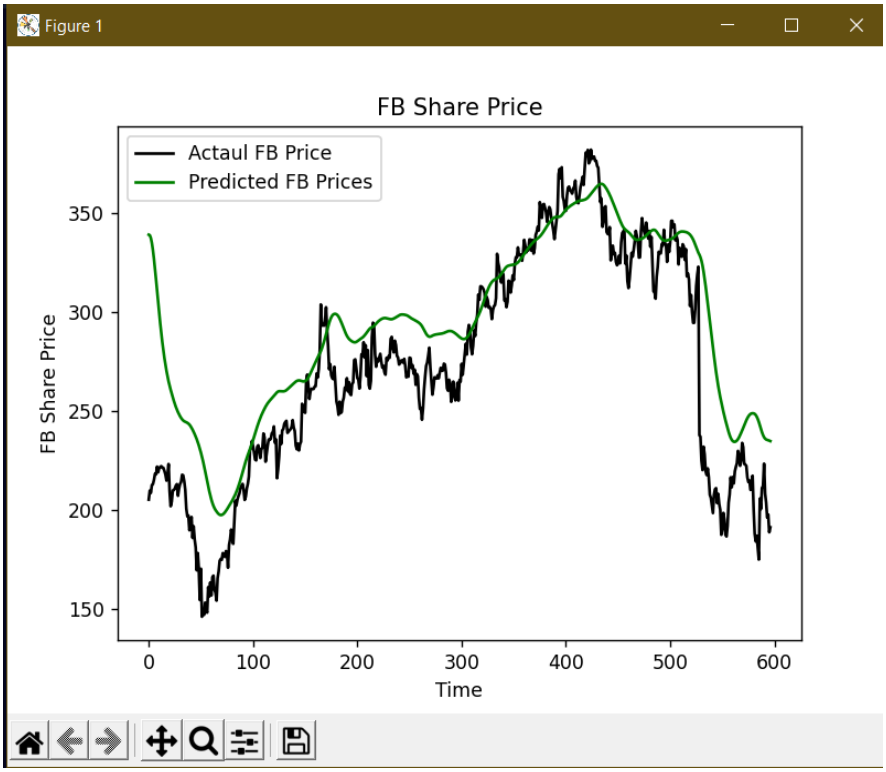


Fig: Meta share graph

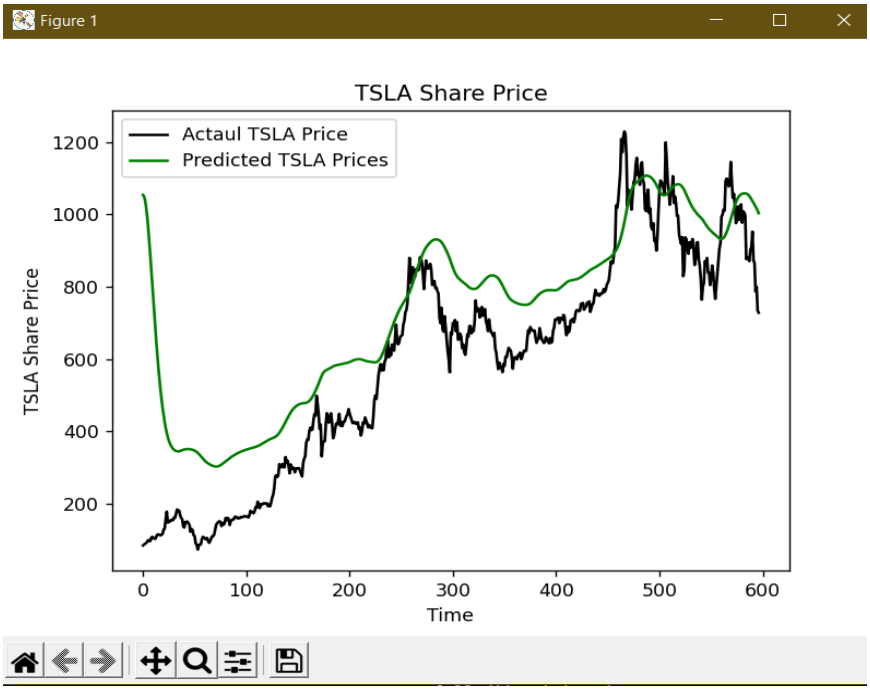


Fig: Tesla share graph

Conclusion

We have learnt a lot from this project. The knowledge we gained on how the LSTM works, the use of numpy, the mathematical shortcuts it gave. The use of pandas_datareader which helped in picking the live API from YAHOO! FINANCE. Also pandas where it helped to easily access the 2D-dataset. As we know the project is not yet complete because the stock market is a real time project. There are everyday incidents that cause fatal or much gain in the stock market.

We started the project with the view of learning new algorithms, in which we were successful. Long short term memory helped a lot in predicting the value. The keras and sklearn the high-level API helped a lot. The mathematical libraries like numpy and matplotlib helped to access the dataset and to plot on the graph.

Improvement

This was our first project on machine learning. We had no prior experience in python projects. We got the results as we desired but are still working on this project to make further improvements. The improvement we want are as follows:

- A UI can be provided where users can check the value for future dates.
- There can be more listed companies we have used only YAHOO! FINANCE.

Reference

- https://en.wikipedia.org/wiki/Long_short-term_memory
- https://finance.yahoo.com/
- https://www.askpython.com/python/examples/stock-price-prediction-python