**TITLE NAME : Tesla Stock Price Prediction**

Project ON

Fuzzy Logic & Neural Network

Course Code: CSE-4107

**Submitted By :**

1. Mohammad Fahad Alam (ID- 1728071)

2. Joy Chowdhury (ID- 1728005)

**Supervised By :**

Dr. Fahim Irfan Alam

Associate Prof.

Chittagong University

Chittagong

**BGC TRUST UNIVERSITY BANGLADESH.**

DEPARTMENT OF COMPUTER SCIENCE & ENGINEERING

January 2021



**Table of Contents:**

Section 1: Introduction

Section 2: Dataset Description

Section 3: Data Analysis & Processing

Section 4: Model Presentation

Section 5: Result & Discussion

Section 6: Conclusion & Future Plan

**Section 1: Introduction-**

In this project we have tried to predict the ‘Tesla Stock Price’.

Nowadays **“TESLA,INC.”** have reached at the peak of it’s income.

Such a stellar performance has left many wondering what might come next for the stock. At the peak of the company's popularity, what are Tesla shares: a buy or sell? Should we invest in TESLA and ride the wave of hype, or stay sideways and wait for bulls to run out of breath?

This will help many of us to invest in **“TESLA,INC.”** or not?

**Section 2: Dataset Description-**

The dataset used in this project is Tesla stocks history (From November 2015 - November 2020).

I have downloaded this file from your given dataset, but we have csv file inside the project folder.

* Name of the file is **TESLA.csv**.

**Section 3: Data Analysis & Processing :**

**TESLA.csv this dataset will help us to predict**

**the stock of “TESLA,INC”.**

* **Supported Python version:**

         - Python version used in this project: 3.7

* **Libraries used:**
* [Pandas](http://pandas.pydata.org/)
* [Numpy](http://www.numpy.org/)
* [Matplotlib](https://matplotlib.org/)
* [Scikit-learn](http://scikit-learn.org/stable/)
* [TensorFlow](https://www.tensorflow.org/)

## Code Explanation:

This project has many different implementations.

#### (i)Import Libraries:

import numpy as np

import matplotlib.pyplot as plt

import pandas as pd

The first step, as always is to import the required libraries.

#### (ii)Import Dataset:

tesla\_training\_complete = pd.read\_csv(r'C:\Users\rjfah\OneDrive\Desktop\FN\TESLA.csv')

Execute the given script to import the data set.

tesla\_training\_processed = tesla\_training\_complete.iloc[:, 1:2].values

we will filter all the data from our training set and will retain only the values for the **Open** column.

we will filter all the data from our training set and will retain only the values for the **Open** column.

#### (iii)Data Normalization:

from sklearn.preprocessing import MinMaxScaler

scaler = MinMaxScaler(feature\_range = (0, 1))

tesla\_training\_scaled = scaler.fit\_transform(tesla\_training\_processed)

 We use MinMaxScaler class from the sklear.preprocessing library to scale our data between 0 and 1. The feature\_range parameter is used to specify the range of the scaled data.

**(iv)Convert Training Data to Right Shape:**

features\_set = []

labels = []

for i in range(50, 1250):

features\_set.append(tesla\_training\_scaled[i-50:i, 0])

labels.append(tesla\_training\_scaled[i, 0])

In the script above we create two lists: feature\_set and labels. There are 1260 records in the training data. We execute a loop that starts from 51st record and stores all the previous 50 records to the feature\_set list. The 51st record is stored in the labels list.

features\_set, labels = np.array(features\_set), np.array(labels)

We need to convert both the feature\_set and the labels list to the numpy array before we can use it for training.

features\_set = np.reshape(features\_set, (features\_set.shape[0], features\_set.shape[1], 1))

We need to convert our data into three-dimensional format. The first dimension is the number of records or rows in the dataset which is 1260 in our case. The second dimension is the number of time steps which is 50 while the last dimension is the number of indicators.

#### (v)Training The LSTM:

from keras.models import Sequential

from keras.layers import Dense

from keras.layers import LSTM

from keras.layers import Dropout

In the script above we imported the Sequential class from keras.models library and Dense, LSTM, and Dropout classes from keras.layers library.

model = Sequential()

we need to instantiate the Sequential class. This will be our model class and we will add LSTM, Dropout and Dense layers to this model.

##### Creating LSTM and Dropout Layers **:**

model.add(LSTM(units=50, return\_sequences=True, input\_shape=(features\_set.shape[1], 1)))

Add LSTM layer to the model that we just creat

model.add(Dropout(0.2))

 Now add a dropout layer to our model. Dropout layer is added to avoid over-fitting.

model.add(LSTM(units=50, return\_sequences=True))

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))

Add three more LSTM and dropout layers to our model.

##### Creating Dense Layer:

model.add(Dense(units = 1))

To make our model more robust, we add a dense layer at the end of the model. The number of neurons in the dense layer will be set to 1 since we want to predict a single value in the output.

##### Model Compilation**:**

model.compile(optimizer = 'adam', loss = 'mean\_squared\_error')

Finally, we need to compile our LSTM before we can train it on the training data.

##### Algorithm Training**:**

model.fit(features\_set, labels, epochs = 100, batch\_size = 32)

Now is the time to train the model that we defined in the previous few steps.

#### (vi)Testing our LSTM:

tesla\_testing\_complete = pd.read\_csv(r'C:\Users\rjfah\OneDrive\Desktop\FN\TESLA.csv')

tesla\_testing\_processed = tesla\_testing\_complete.iloc[:, 1:2].values

First import our test data.

##### Converting Test Data to Right Format**:**

tesla\_total = pd.concat((tesla\_training\_complete['Open'], tesla\_testing\_complete['Open']), axis=0)

we need to concatenate our training data and test data before preprocessing.

test\_inputs = tesla\_total[len(tesla\_total) - len(tesla\_testing\_complete) - 50:].values

Prepared our test inputs.

test\_inputs = test\_inputs.reshape(-1,1)

test\_inputs = scaler.transform(test\_inputs)

As we did for the training set, we need to scale our test data.

test\_features = []

for i in range(50, 70):

test\_features.append(test\_inputs[i-50:i, 0])

We scaled our data, now let's prepare our final test input set that will contain previous 50 stock prices for the month of January.

test\_features = np.array(test\_features)

test\_features = np.reshape(test\_features, (test\_features.shape[0], test\_features.shape[1], 1))

Finally, we need to convert our data into the three-dimensional format which can be used as input to the LSTM.

##### Making Predictions**:**

predictions = model.predict(test\_features)

we simply need to call the predict method on the model that we trained.

predictions = scaler.inverse\_transform(predictions)

we can use the ìnverse\_transform method of the scaler object we created during training.

plt.figure(figsize=(10,6))

plt.plot(tesla\_testing\_processed, color='blue', label='Actual Tesla Stock Price')

plt.plot(predictions , color='red', label='Predicted Tesla Stock Price')

plt.title('Tesla Stock Price Prediction')

plt.xlabel('Date')

plt.ylabel('Tesla Stock Price')

plt.legend()

plt.show()

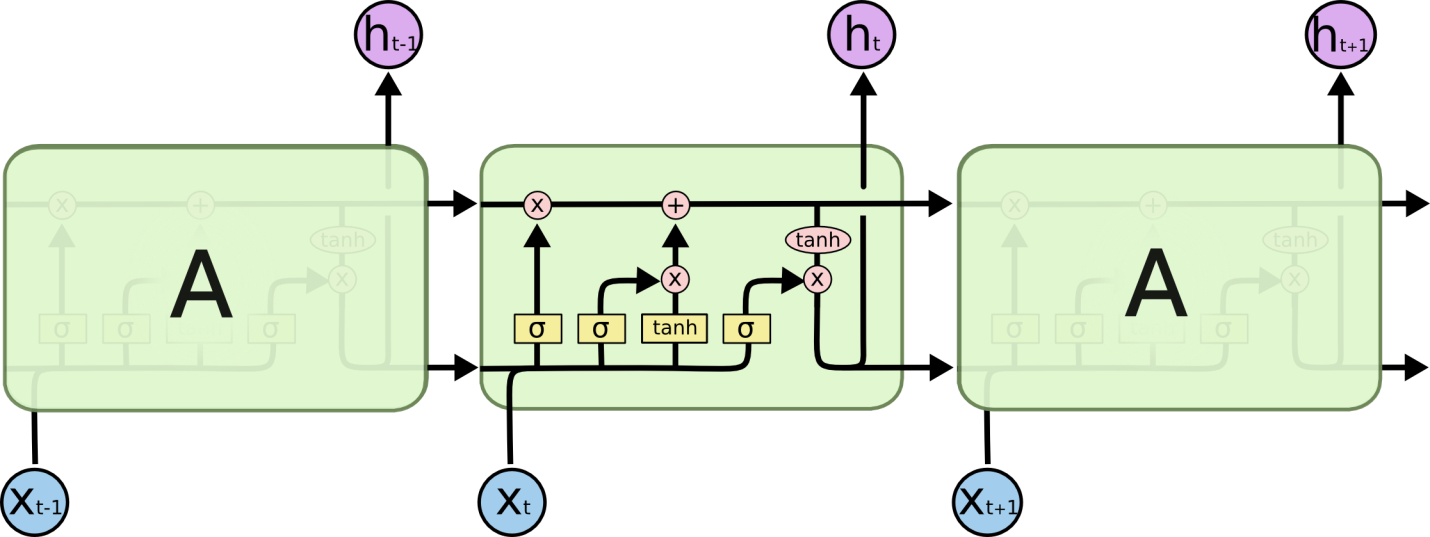
Finally, let's see how well did our algorithm predicted the future stock prices.

**Section 4: Model Presentation-**

Here we have given a RNN model that use to predict the stock price .

The picture took from **Colah’s Blog**.

We have used LSTM as a type of RNN.



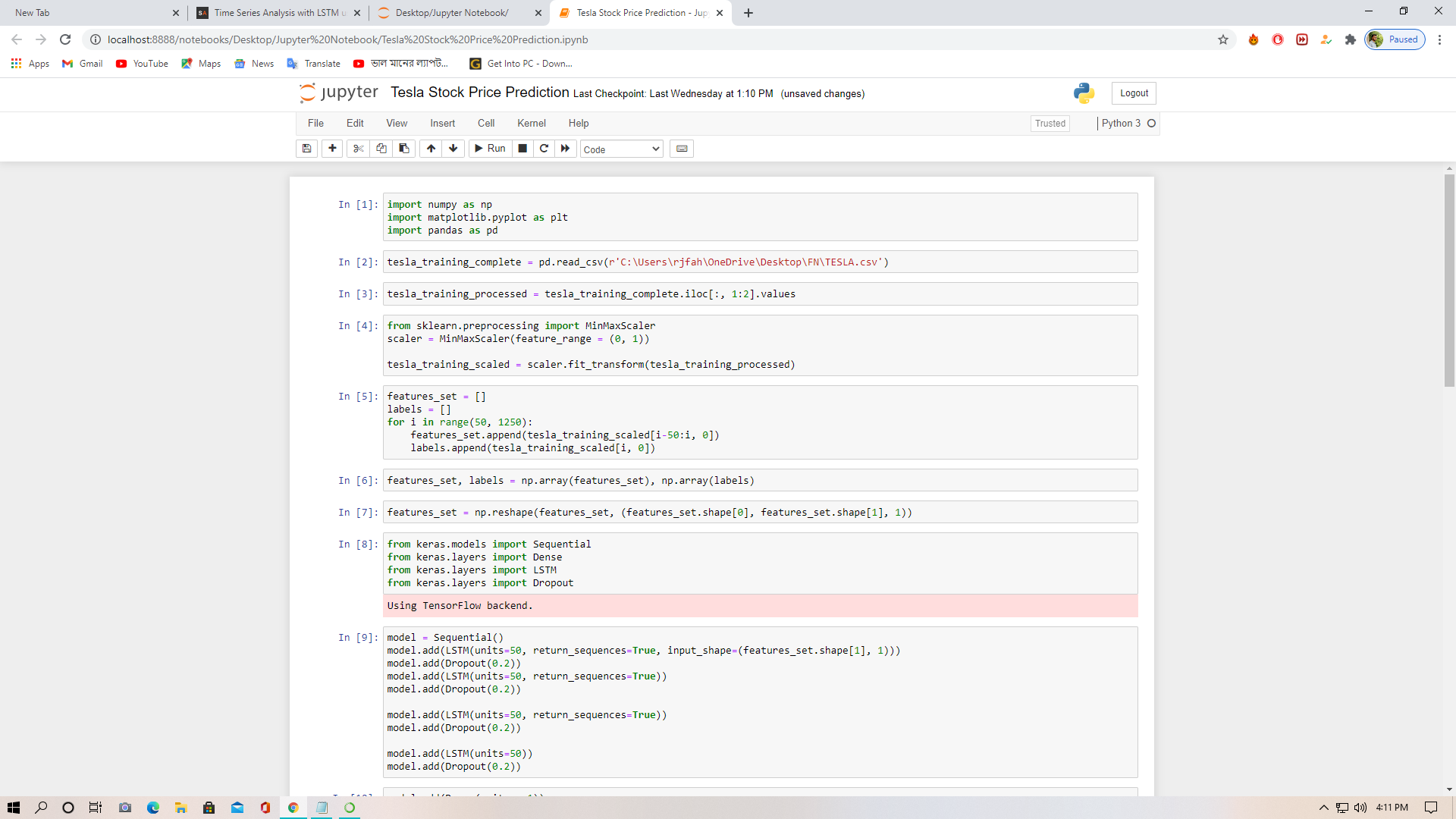
**Figure**: RNN model

**Section 5: Result & Discussion-**

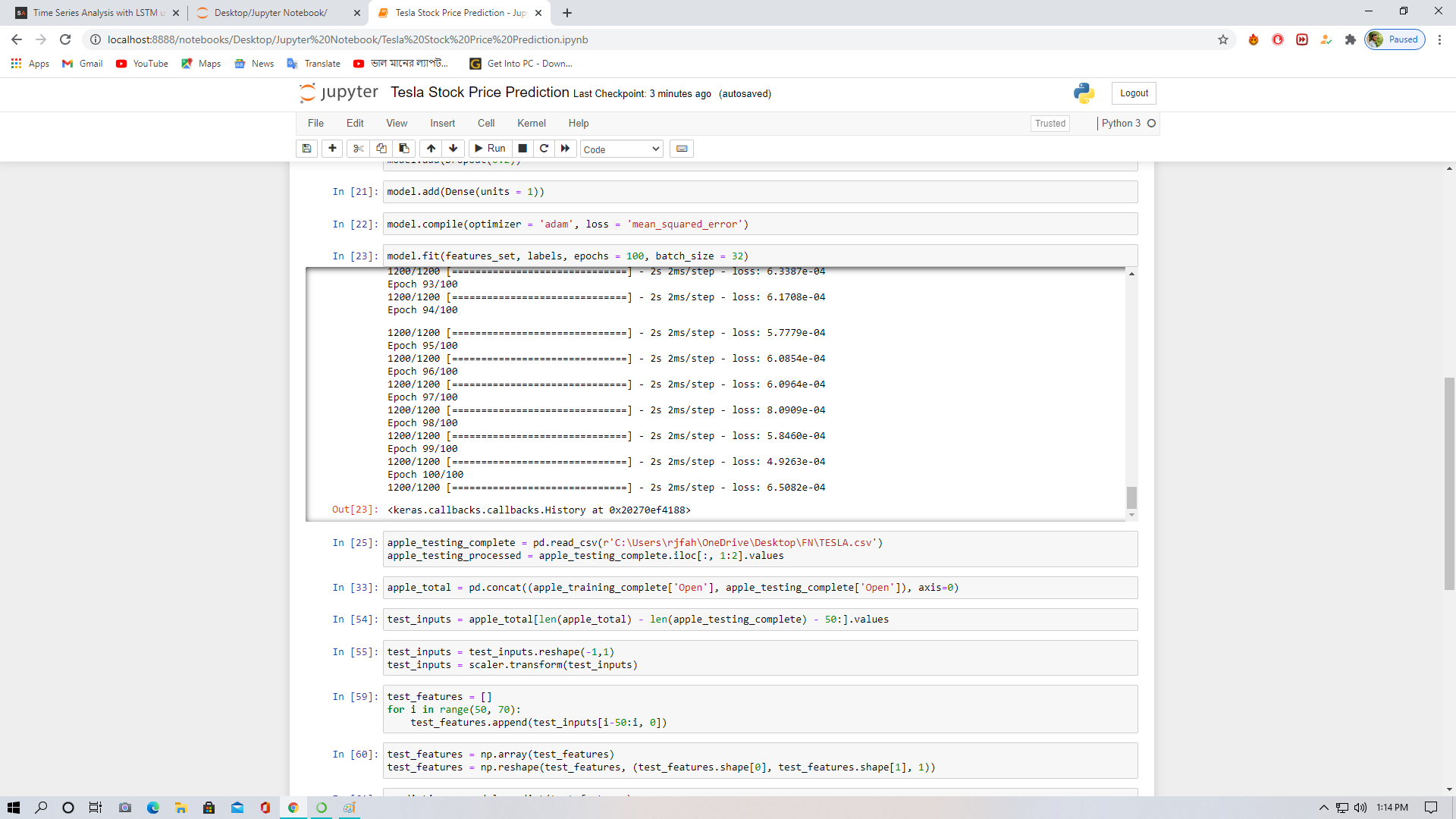
* Result Analysis:

Here the same Snapshot of the project.

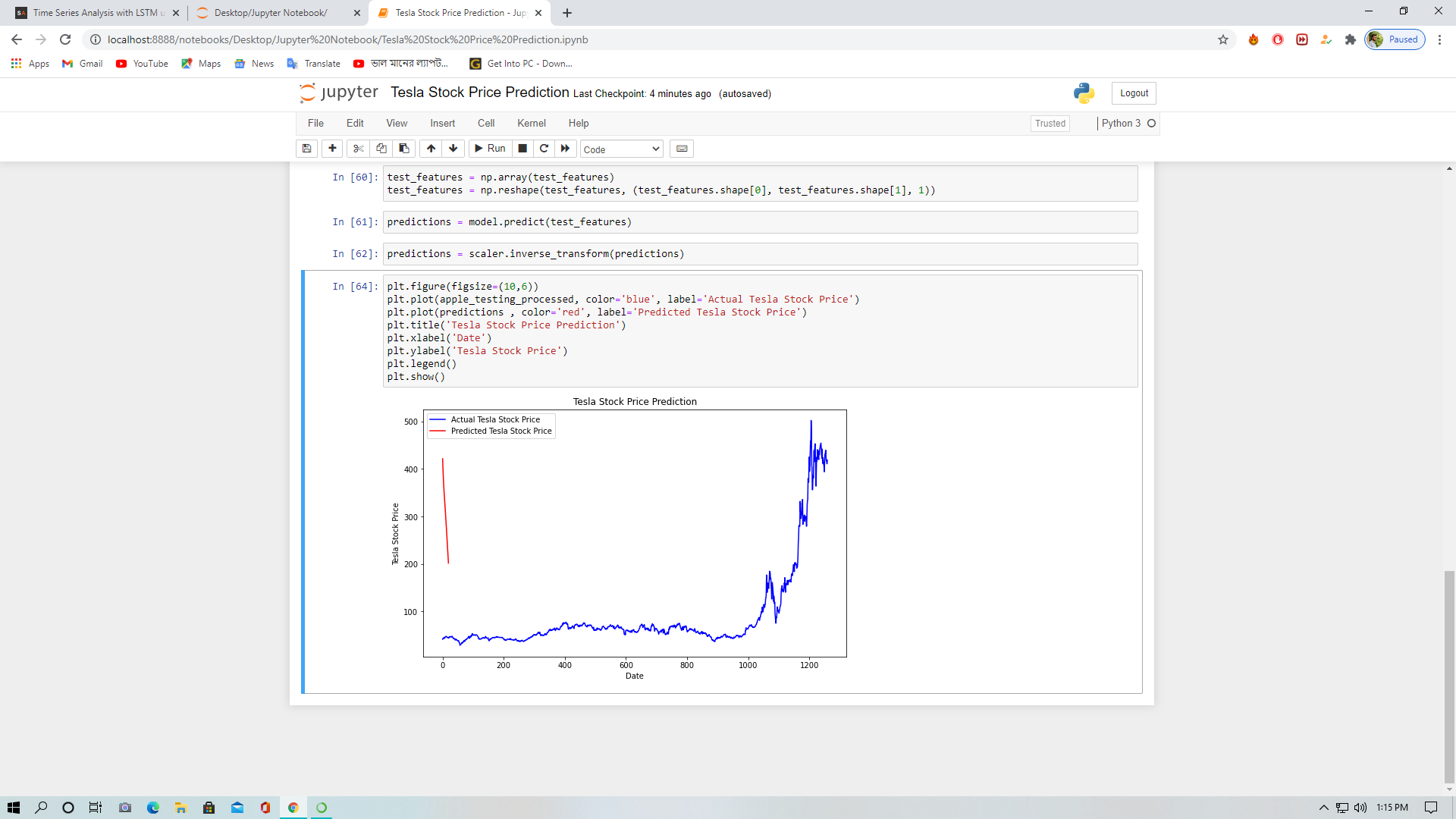
Snapshot-1



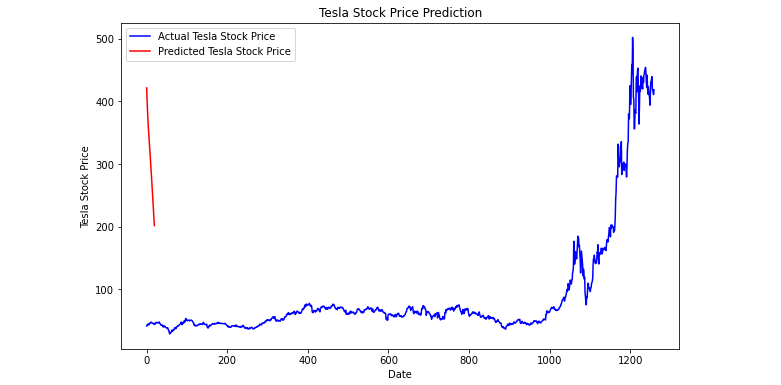
Snapshot-2



Snapshot-3



Snapshot of the result:



* Discussion:

We have faced many problem to do this predicting project.But, at last this project is ready to roar and predict the price of **“TESLA,INC”.**

A long short-term memory network (LSTM) is one of the most commonly used neural networks for time series analysis. The ability of LSTM to remember previous information makes it ideal for such tasks. In this article, we saw how we can use LSTM for the Tesla stock price prediction.

**Section 6: Conclusion & Future Plan-**

This project will help us to predict to the stock.

And further, we can invest by watching the prediction result.

In future this predicting must be updated or will be useable for the regular customer/user/everyone.

Our plan is to make a SOFTWARE on the basis of this predicting project.

……………………………………………………………………………………………

**Thank YOU**