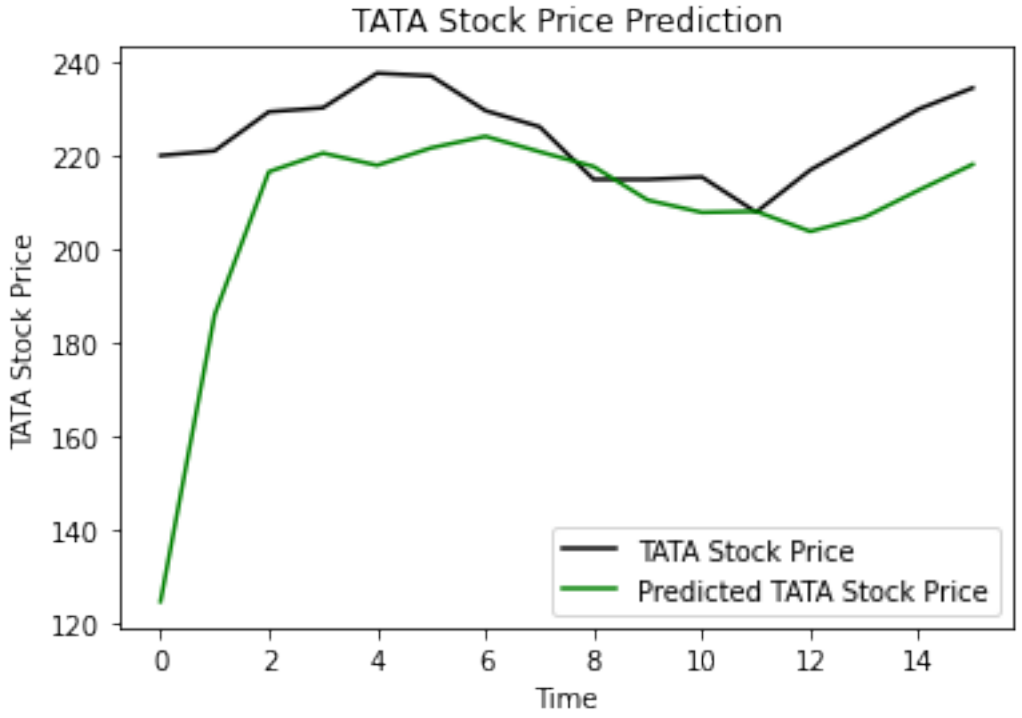
StockMarketPrediction-ML

December 16, 2021

[14]:
```
# Author : Saeid Rezaei , STU:205812010
# Date Dec , 16 2021
# Final Project :CP640 - Machine Learing
# Desciprtion :
# In this project , I will be analysyng the Stock market for few stock such as␣
 ↪google, TATA
# I have used Long sorth-term memory (LSTM) model to predict the future loss␣
 ↪based on history of price changes
# This is usfull ML method that acts better that Time Series for Stock Market
# Data has been cleaned prior to this scrip using Perl to fill the missing␣
 ↪oinformation
# Logic for missing informations:
# If closing price is missing - fill the value based on previous day price
# If date is missing , record is eliminated
# if price is not in the threshold - (different previous and currect is more␣
 ↪than 1000) flag and remove

import numpy as np
import matplotlib.pyplot as plt
import pandas as pd
# Import NSE index for TATA stock for several years
# Like I said above data is cleaned based on Perl scrip (Please see Cleanup.pl␣
 ↪under /scr)
dataset_train = pd.read_csv('NSE-TATAGLOBAL.csv')

# Create traning sets and test based on 1 by 2. means based on volumn of data␣
 ↪training is two times more than test
training_set = dataset_train.iloc[:, 1:2].values
# review the data in DataFram
dataset_train.head()

# Use skearn lib to calculate the min and max scale of data and price
from sklearn.preprocessing import MinMaxScaler
sc = MinMaxScaler(feature_range = (0, 1))
training_set_scaled = sc.fit_transform(training_set)

# Create 2D matrix , in this case X is an input for next Y tranig sets.
```

```
# I have used 60 step that shows better changes , it could be change to other␣
 ↪value based on price change and market trend


X_train = []
y_train = []
for i in range(60, 2035):
    X_train.append(training_set_scaled[i-60:i, 0])
    y_train.append(training_set_scaled[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))


# Create Model based on traing data sets
# Using keras lib
# Create LSTM Model
from keras.models import Sequential
from keras.layers import Dense
from keras.layers import LSTM
from keras.layers import Dropout


# Load test data sets for TATA stock
# again data is cleaned and test data / train ratio is 1:2
dataset_test = pd.read_csv('tatatest.csv')
real_stock_price = dataset_test.iloc[:, 1:2].values

# Predict the model based on test data
dataset_total = pd.concat((dataset_train['Open'], dataset_test['Open']), axis =␣
 ↪0)
inputs = dataset_total[len(dataset_total) - len(dataset_test) - 60:].values
inputs = inputs.reshape(-1,1)
inputs = sc.transform(inputs)
X_test = []
for i in range(60, 76):
    X_test.append(inputs[i-60:i, 0])
X_test = np.array(X_test)
X_test = np.reshape(X_test, (X_test.shape[0], X_test.shape[1], 1))
predicted_stock_price = regressor.predict(X_test)
predicted_stock_price = sc.inverse_transform(predicted_stock_price)




# plot the prediction
```

```
plt.plot(real_stock_price, color = 'black', label = 'TATA Stock Price')
plt.plot(predicted_stock_price, color = 'green', label = 'Predicted TATA Stock␣
 ↪Price')
plt.title('TATA Stock Price Prediction')
plt.xlabel('Time')
plt.ylabel('TATA Stock Price')
plt.legend()
plt.show()
```



[ ]:
```
!dpkg --configure -a
!wget -nc https://raw.githubusercontent.com/brpy/colab-pdf/master/colab_pdf.py
from colab_pdf import colab_pdf
colab_pdf('StockMarketPrediction-ML.ipynb')
```

```
File colab_pdf.py already there; not retrieving.

Mounted at /content/drive/

WARNING: apt does not have a stable CLI interface. Use with caution in scripts.


WARNING: apt does not have a stable CLI interface. Use with caution in scripts.
```

```
Extracting templates from packages: 100%
```

[3]:
```
dataset_train = pd.read_csv('NSE-TATAGLOBAL.csv')
training_set = dataset_train.iloc[:, 1:2].values
```

[4]:
```
dataset_train.head()
```

[4]:
```
         Date    Open    High  ...   Close  Total Trade Quantity  Turnover
(Lacs)
0  2018-09-28  234.05  235.95  ...  233.75               3069914
7162.35
1  2018-09-27  234.55  236.80  ...  233.25               5082859
11859.95
2  2018-09-26  240.00  240.00  ...  234.25               2240909
5248.60
3  2018-09-25  233.30  236.75  ...  236.10               2349368
5503.90
4  2018-09-24  233.55  239.20  ...  233.30               3423509
7999.55

[5 rows x 8 columns]
```

[5]:
```
from sklearn.preprocessing import MinMaxScaler
sc = MinMaxScaler(feature_range = (0, 1))
training_set_scaled = sc.fit_transform(training_set)
```

[6]:
```
X_train = []
y_train = []
for i in range(60, 2035):
    X_train.append(training_set_scaled[i-60:i, 0])
    y_train.append(training_set_scaled[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))
```

[7]:
```
from keras.models import Sequential
from keras.layers import Dense
from keras.layers import LSTM
from keras.layers import Dropout
```

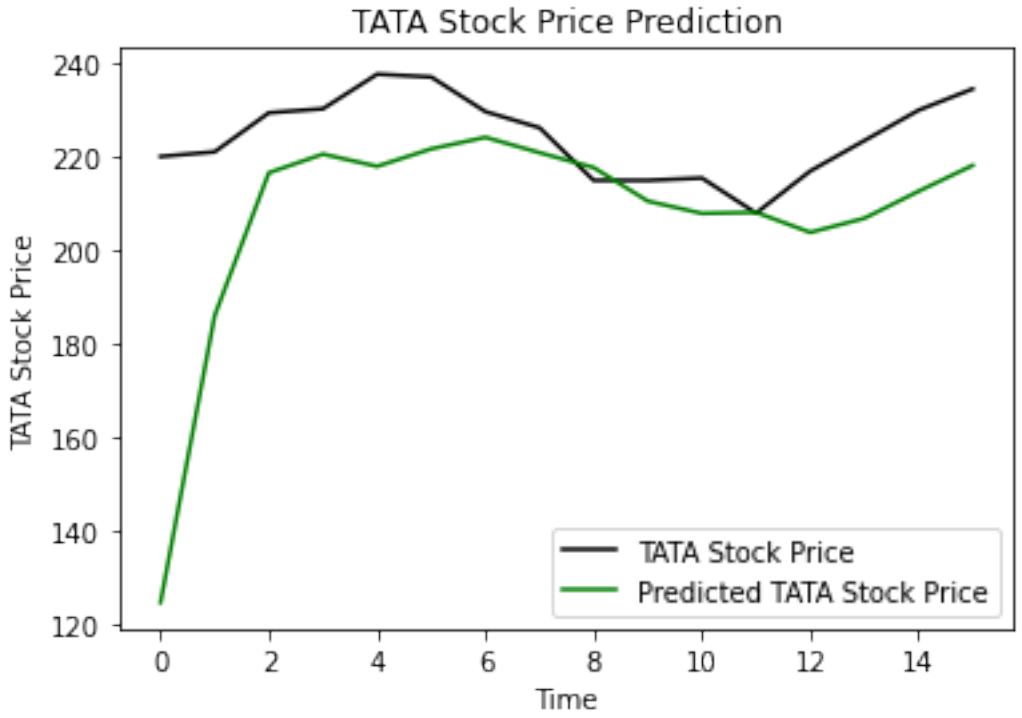
[ ]:
```

```

[9]:
```
dataset_test = pd.read_csv('tatatest.csv')
real_stock_price = dataset_test.iloc[:, 1:2].values
```

[10]:
```
dataset_total = pd.concat((dataset_train['Open'], dataset_test['Open']), axis =␣
↪0)
inputs = dataset_total[len(dataset_total) - len(dataset_test) - 60:].values
inputs = inputs.reshape(-1,1)
inputs = sc.transform(inputs)
```

```
X_test = []
for i in range(60, 76):
    X_test.append(inputs[i-60:i, 0])
X_test = np.array(X_test)
X_test = np.reshape(X_test, (X_test.shape[0], X_test.shape[1], 1))
predicted_stock_price = regressor.predict(X_test)
predicted_stock_price = sc.inverse_transform(predicted_stock_price)
```

[11]:
```
plt.plot(real_stock_price, color = 'black', label = 'TATA Stock Price')
plt.plot(predicted_stock_price, color = 'green', label = 'Predicted TATA Stock␣
 ↪Price')
plt.title('TATA Stock Price Prediction')
plt.xlabel('Time')
plt.ylabel('TATA Stock Price')
plt.legend()
plt.show()
```



[8]:
```
regressor = Sequential()

regressor.add(LSTM(units = 50, return_sequences = True, input_shape = (X_train.
 ↪shape[1], 1)))
regressor.add(Dropout(0.2))

regressor.add(LSTM(units = 50, return_sequences = True))
```

```
regressor.add(Dropout(0.2))

regressor.add(LSTM(units = 50, return_sequences = True))
regressor.add(Dropout(0.2))

regressor.add(LSTM(units = 50))
regressor.add(Dropout(0.2))

regressor.add(Dense(units = 1))

regressor.compile(optimizer = 'adam', loss = 'mean_squared_error')

regressor.fit(X_train, y_train, epochs = 100, batch_size = 32)
```

```
Epoch 1/100
62/62 [==============================] - 17s 142ms/step - loss: 0.0100
Epoch 2/100
62/62 [==============================] - 9s 144ms/step - loss: 0.0030
Epoch 3/100
62/62 [==============================] - 9s 142ms/step - loss: 0.0027
Epoch 4/100
62/62 [==============================] - 9s 139ms/step - loss: 0.0023
Epoch 5/100
62/62 [==============================] - 9s 139ms/step - loss: 0.0025
Epoch 6/100
62/62 [==============================] - 9s 140ms/step - loss: 0.0021
Epoch 7/100
62/62 [==============================] - 9s 142ms/step - loss: 0.0020
Epoch 8/100
62/62 [==============================] - 9s 140ms/step - loss: 0.0024
Epoch 9/100
62/62 [==============================] - 9s 142ms/step - loss: 0.0018
Epoch 10/100
62/62 [==============================] - 10s 170ms/step - loss: 0.0018
Epoch 11/100
62/62 [==============================] - 9s 142ms/step - loss: 0.0020
Epoch 12/100
62/62 [==============================] - 9s 139ms/step - loss: 0.0018
Epoch 13/100
62/62 [==============================] - 9s 137ms/step - loss: 0.0017
Epoch 14/100
62/62 [==============================] - 9s 140ms/step - loss: 0.0016
Epoch 15/100
62/62 [==============================] - 9s 140ms/step - loss: 0.0017
Epoch 16/100
62/62 [==============================] - 9s 143ms/step - loss: 0.0015
Epoch 17/100
```

```
62/62 [==============================] - 9s 143ms/step - loss: 0.0015
Epoch 18/100
62/62 [==============================] - 9s 139ms/step - loss: 0.0014
Epoch 19/100
62/62 [==============================] - 9s 144ms/step - loss: 0.0017
Epoch 20/100
62/62 [==============================] - 9s 141ms/step - loss: 0.0014
Epoch 21/100
62/62 [==============================] - 9s 142ms/step - loss: 0.0015
Epoch 22/100
62/62 [==============================] - 9s 141ms/step - loss: 0.0013
Epoch 23/100
62/62 [==============================] - 9s 142ms/step - loss: 0.0013
Epoch 24/100
62/62 [==============================] - 9s 145ms/step - loss: 0.0014
Epoch 25/100
62/62 [==============================] - 9s 142ms/step - loss: 0.0014
Epoch 26/100
62/62 [==============================] - 9s 141ms/step - loss: 0.0012
Epoch 27/100
62/62 [==============================] - 9s 142ms/step - loss: 0.0012
Epoch 28/100
62/62 [==============================] - 9s 141ms/step - loss: 0.0011
Epoch 29/100
62/62 [==============================] - 9s 143ms/step - loss: 0.0011
Epoch 30/100
62/62 [==============================] - 9s 141ms/step - loss: 0.0013
Epoch 31/100
62/62 [==============================] - 9s 140ms/step - loss: 0.0011
Epoch 32/100
62/62 [==============================] - 9s 141ms/step - loss: 0.0010
Epoch 33/100
62/62 [==============================] - 9s 139ms/step - loss: 0.0012
Epoch 34/100
62/62 [==============================] - 9s 138ms/step - loss: 0.0010
Epoch 35/100
62/62 [==============================] - 9s 142ms/step - loss: 0.0011
Epoch 36/100
62/62 [==============================] - 9s 141ms/step - loss: 0.0012
Epoch 37/100
62/62 [==============================] - 9s 140ms/step - loss: 0.0010
Epoch 38/100
62/62 [==============================] - 9s 139ms/step - loss: 8.5557e-04
Epoch 39/100
62/62 [==============================] - 9s 140ms/step - loss: 9.6375e-04
Epoch 40/100
62/62 [==============================] - 9s 142ms/step - loss: 0.0010
Epoch 41/100
```

```
62/62 [==============================] - 9s 140ms/step - loss: 9.9628e-04
Epoch 42/100
62/62 [==============================] - 9s 140ms/step - loss: 9.8303e-04
Epoch 43/100
62/62 [==============================] - 9s 140ms/step - loss: 8.4373e-04
Epoch 44/100
62/62 [==============================] - 9s 140ms/step - loss: 8.9220e-04
Epoch 45/100
62/62 [==============================] - 9s 138ms/step - loss: 9.5627e-04
Epoch 46/100
62/62 [==============================] - 9s 140ms/step - loss: 8.9705e-04
Epoch 47/100
62/62 [==============================] - 9s 141ms/step - loss: 8.5974e-04
Epoch 48/100
62/62 [==============================] - 9s 142ms/step - loss: 8.3529e-04
Epoch 49/100
62/62 [==============================] - 9s 139ms/step - loss: 9.1079e-04
Epoch 50/100
62/62 [==============================] - 9s 141ms/step - loss: 0.0010
Epoch 51/100
62/62 [==============================] - 9s 141ms/step - loss: 8.5367e-04
Epoch 52/100
62/62 [==============================] - 9s 139ms/step - loss: 8.5003e-04
Epoch 53/100
62/62 [==============================] - 9s 141ms/step - loss: 7.8702e-04
Epoch 54/100
62/62 [==============================] - 9s 142ms/step - loss: 7.6615e-04
Epoch 55/100
62/62 [==============================] - 9s 140ms/step - loss: 6.9579e-04
Epoch 56/100
62/62 [==============================] - 9s 138ms/step - loss: 7.4900e-04
Epoch 57/100
62/62 [==============================] - 9s 139ms/step - loss: 8.5222e-04
Epoch 58/100
62/62 [==============================] - 9s 140ms/step - loss: 8.1124e-04
Epoch 59/100
62/62 [==============================] - 9s 139ms/step - loss: 8.8609e-04
Epoch 60/100
62/62 [==============================] - 9s 138ms/step - loss: 7.7964e-04
Epoch 61/100
62/62 [==============================] - 9s 138ms/step - loss: 7.3759e-04
Epoch 62/100
62/62 [==============================] - 9s 140ms/step - loss: 8.3362e-04
Epoch 63/100
62/62 [==============================] - 9s 146ms/step - loss: 7.4163e-04
Epoch 64/100
62/62 [==============================] - 10s 167ms/step - loss: 8.4333e-04
Epoch 65/100
```

```
62/62 [==============================] - 9s 139ms/step - loss: 7.5498e-04
Epoch 66/100
62/62 [==============================] - 9s 139ms/step - loss: 6.9727e-04
Epoch 67/100
62/62 [==============================] - 9s 138ms/step - loss: 8.3132e-04
Epoch 68/100
62/62 [==============================] - 9s 138ms/step - loss: 8.3326e-04
Epoch 69/100
62/62 [==============================] - 9s 140ms/step - loss: 8.2899e-04
Epoch 70/100
62/62 [==============================] - 9s 141ms/step - loss: 7.2405e-04
Epoch 71/100
62/62 [==============================] - 9s 139ms/step - loss: 7.2507e-04
Epoch 72/100
62/62 [==============================] - 9s 140ms/step - loss: 7.7836e-04
Epoch 73/100
62/62 [==============================] - 9s 141ms/step - loss: 7.4451e-04
Epoch 74/100
62/62 [==============================] - 9s 140ms/step - loss: 8.2306e-04
Epoch 75/100
62/62 [==============================] - 9s 142ms/step - loss: 8.3614e-04
Epoch 76/100
62/62 [==============================] - 9s 141ms/step - loss: 7.2929e-04
Epoch 77/100
62/62 [==============================] - 9s 141ms/step - loss: 6.9544e-04
Epoch 78/100
62/62 [==============================] - 9s 140ms/step - loss: 7.2137e-04
Epoch 79/100
62/62 [==============================] - 9s 138ms/step - loss: 6.6986e-04
Epoch 80/100
62/62 [==============================] - 9s 140ms/step - loss: 6.4556e-04
Epoch 81/100
62/62 [==============================] - 9s 140ms/step - loss: 6.7964e-04
Epoch 82/100
62/62 [==============================] - 9s 140ms/step - loss: 6.4693e-04
Epoch 83/100
62/62 [==============================] - 9s 141ms/step - loss: 5.8525e-04
Epoch 84/100
62/62 [==============================] - 9s 142ms/step - loss: 7.1220e-04
Epoch 85/100
62/62 [==============================] - 9s 143ms/step - loss: 6.4041e-04
Epoch 86/100
62/62 [==============================] - 9s 142ms/step - loss: 7.8026e-04
Epoch 87/100
62/62 [==============================] - 9s 141ms/step - loss: 6.7657e-04
Epoch 88/100
62/62 [==============================] - 9s 140ms/step - loss: 6.5434e-04
Epoch 89/100
```

```
62/62 [==============================] - 9s 142ms/step - loss: 6.0078e-04
Epoch 90/100
62/62 [==============================] - 9s 141ms/step - loss: 6.6962e-04
Epoch 91/100
62/62 [==============================] - 9s 140ms/step - loss: 6.7122e-04
Epoch 92/100
62/62 [==============================] - 9s 140ms/step - loss: 6.5553e-04
Epoch 93/100
62/62 [==============================] - 9s 139ms/step - loss: 6.4642e-04
Epoch 94/100
62/62 [==============================] - 9s 139ms/step - loss: 6.5292e-04
Epoch 95/100
62/62 [==============================] - 9s 138ms/step - loss: 7.6738e-04
Epoch 96/100
62/62 [==============================] - 9s 140ms/step - loss: 6.5305e-04
Epoch 97/100
62/62 [==============================] - 9s 138ms/step - loss: 7.3798e-04
Epoch 98/100
62/62 [==============================] - 9s 139ms/step - loss: 7.1912e-04
Epoch 99/100
62/62 [==============================] - 9s 139ms/step - loss: 6.1914e-04
Epoch 100/100
62/62 [==============================] - 9s 139ms/step - loss: 6.6658e-04
```

[8]: `<keras.callbacks.History at 0x7fad2427b850>`