Open Ended Problem

# AIM: Implement Stock Exchange Prediction Small Project with Charts in Python.

import math

import pandas\_datareader as web

import numpy as np

import pandas as pd

from sklearn.preprocessing import MinMaxScaler

from tensorflow.python.keras.layers import Dense,LSTM

from tensorflow.python.keras import Sequential

import matplotlib.pyplot as plt

plt.style.use('fivethirtyeight')

#get the stock quote

df=web.DataReader('TSLA',data\_source='yahoo',start='2010-01-01',end='2019-12-17')

df

#visualize the closing price hystpl

plt.figure(figsize=(16,8))

plt.title('Close price History')

plt.plot(df['Close'])

plt.xlabel('Date',fontsize=18)

plt.ylabel('Close Price USD ($) ',fontsize=18)

plt.show()

#get the numbers of rows and columns

df.shape

#create a new dataframe with only the 'Close column

data=df.filter(['Close'])

#convert the dataframe to a numpy array

dataset=data.values

print(len(dataset))

#get the number of rowa to train the model on [HAS ISSUE WITH LENGTH]

training\_data\_len=math.ceil(len(dataset) \* .8)

training\_data\_len

#Scale the data

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

scaled\_data=scaler.fit\_transform(dataset)

print(len(scaled\_data))

scaled\_data

#create the training dataset

#create the scaed training data set

train\_data=scaled\_data[0:training\_data\_len,:] #0-1908 [issue here]

#split the data into x\_train and y\_train data sets

x\_train=[]

y\_train=[]

for i in range(60,len(train\_data)):

x\_train.append(train\_data[i-60:i,0])

y\_train.append(train\_data[i,0])

print(len(x\_train))

print()

print(len(y\_train))

#Convert the x\_train and y\_train to numpy arrays

x\_train,y\_train = np.array(x\_train),np.array(y\_train)

#Reshape the data

x\_train=np.reshape(x\_train,(x\_train.shape[0],x\_train.shape[1],1))

x\_train.shape

#Build the LSTM model

model=Sequential()

model.add(LSTM(50,return\_sequences=True,input\_shape=(x\_train.shape[1],1)))

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

model.add(Dense(25))

model.add(Dense(1))

#Compile model

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

#train the model

model.fit(x\_train,y\_train,batch\_size=1,epochs=1)

#create the testing dataset

#create a new array containing scaled value

test\_data=scaled\_data[training\_data\_len-60:,:]

#Create the data set x\_test and y\_test

x\_test=[]

y\_test=dataset[training\_data\_len:,:]

for i in range(60,len(test\_data)):

x\_test.append(test\_data[i-60:i,0])

#Convert the data into numpy dataset

x\_test=np.array(x\_test)

#Reshape the data

x\_test=np.reshape(x\_test,(x\_test.shape[0],x\_test.shape[1],1))

#Get the model predicted price values

predictions=model.predict(x\_test)

predictions=scaler.inverse\_transform(predictions)

#Get the root mean squared error (RMSE)

rmse=np.sqrt(np.mean(predictions-y\_test)\*\*2)

rmse

#Plot the data

train=data[:training\_data\_len]

valid = data[training\_data\_len:]

valid['Predictions']=predictions

#visual

plt.figure(figsize=(16,8))

plt.title('Model')

plt.xlabel('Date',fontsize=18)

plt.ylabel('Close Price USD($)',fontsize=18)

plt.plot(train['Close'])

plt.plot(valid[['Close','Predictions']])

plt.legend(['Train','Val','Predictions'],loc='lower right')

plt.show

#Show the valid and predicted prices valid

#Get the quote

apple\_quote = web.DataReader('TSLA',data\_source='yahoo', start='2010-01-01' ,end='2020-2-5')

#create a new datafreame new

df =apple\_quote.filter(['Close'])

#get the last 60 days values

last\_60\_days=new\_df[-60:].values

#scale the data to be values btn 0-1

last\_60\_days\_scaled = scaler.transform(last\_60\_days)

X\_test= []

X\_test.append(last\_60\_days\_scaled)

X\_test = np.array(X\_test)

#Resahpe

X\_test = np.reshape(X\_test,(X\_test.shape[0],X\_test.shape[1],1))

pred\_price=model.predict(X\_test)

pred\_price=scaler.inverse\_transform(pred\_price)

print("Predicted price of given date is : ")

print(pred\_price)

## OUTPUT:

### Data-of-tesla company:



### Training –of –model-LSTM:



### Graph-of-prediction-and-training:



### Predicted-price for date 2020-04-04:

