Work this Code on GoogleColab

import numpy as np # linear algebra

import pandas as pd # data processing, CSV file I/O (e.g. pd.read\_csv)

import matplotlib.pyplot as plt

from keras.models import Sequential

import matplotlib.patches as mpatches

from keras.layers import Dense

from keras.layers import Dropout

from keras.layers import LSTM

from sklearn.preprocessing import MinMaxScaler

from sklearn.metrics import mean\_squared\_error

from keras.layers import Dense,RepeatVector

from keras.layers import Flatten

from keras.layers import TimeDistributed

from keras.layers.convolutional import Conv1D

from keras.layers.convolutional import Conv2D

from keras.layers.convolutional import MaxPooling1D

from keras.layers.convolutional import MaxPooling2D

from sklearn.model\_selection import train\_test\_split

from datetime import timedelta

cities = pd.read\_excel(r'Karachi.xlsx')

khi = cities.loc[cities['City'] == 'Karachi', ['Date','Maximum temperature']]  # CHANGE ACCORDINGLY

print(khi.head())

khi['Date'] = pd.to\_datetime(khi['Date'])

khi.set\_index('Date', inplace= True)

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

sc = scaler.fit\_transform(khi)

timestep = 365

X= []

Y=[]

for i in range(len(sc)- (timestep)):

    X.append(sc[i:i+timestep])

    Y.append(sc[i+timestep])

X=np.asanyarray(X)

Y=np.asanyarray(Y)

k = 3000

Xtrain = X[:k,:,:]

Xtest = X[k:,:,:]

Ytrain = Y[:k]

Ytest= Y[k:]

print(Xtrain.shape)

print(Xtest.shape)

model = Sequential()

model.add(Conv1D(filters=256, kernel\_size=2, activation='relu', input\_shape=(365,1)))

model.add(Conv1D(filters=128, kernel\_size=2, activation='relu'))

model.add(MaxPooling1D(pool\_size=2))

model.add(Flatten())

model.add(RepeatVector(30))

model.add(LSTM(128, activation='relu'))

model.add(Dense(100, activation='relu'))

model.add(Dense(1))

model.compile(loss='mse', optimizer='adam',metrics=['mse', 'mae'])

history= model.fit(Xtrain,Ytrain,epochs=1,verbose=1 )

print(history.history['mse'])

print(history.history['mae'])

plt.figure(figsize=(8,5))

plt.plot(history.history['loss'])

plt.xlabel("Epochs")

plt.ylabel("Loss")

plt.title("Model Accuracy")

preds= model.predict(Xtest)

preds = scaler.inverse\_transform(preds)

Ytest=np.asanyarray(Ytest)

Ytest=Ytest.reshape(-1,1)

Ytest = scaler.inverse\_transform(Ytest)

Ytrain=np.asanyarray(Ytrain)

Ytrain=Ytrain.reshape(-1,1)

Ytrain = scaler.inverse\_transform(Ytrain)

print(mean\_squared\_error(Ytest,preds))

plt.figure(figsize=(20,9))

plt.plot(Ytest , 'blue', linewidth=5)

plt.plot(preds,'r' , linewidth=4)

plt.legend(('Test','Predicted'))

plt.title(" Prediction") # CHANGE ACCORDINGLY

plt.show()

test = pd.DataFrame(Ytest,columns=['Actual'])

pred = pd.DataFrame(preds,columns=['Predict'])

results = pd.concat([test,pred],axis=1)

print(results.head())

resutls=pd.DataFrame(results)

def insert\_end(Xin,new\_input):

    for i in range(timestep-1):

        Xin[:,i,:] = Xin[:,i+1,:]

    Xin[:,timestep-1,:] = new\_input

    return Xin

print(khi)

future=1825

forcast = []

Xin = Xtest[-1:,:,:]

time=[]

for i in range(future):

    out = model.predict(Xin, batch\_size=1)

    forcast.append(out[0,0])

    Xin = insert\_end(Xin,out[0,0])

    time.append(pd.to\_datetime(khi.index[-1])+timedelta(days=i+1))

forcasted\_output=np.asanyarray(forcast)

forcasted\_output=forcasted\_output.reshape(-1,1)

forcasted\_output = scaler.inverse\_transform(forcasted\_output)

forcasted\_output = pd.DataFrame(forcasted\_output)

date = pd.DataFrame(time)

df\_result = pd.concat([date,forcasted\_output],axis=1)

df\_result.columns = "Date","Forecasted"

print(df\_result.columns)

print(df\_result)

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

plt.title(' Prediction') # CHANGE ACCORDINGLY

plt.xlabel('Date', fontsize=18)

plt.ylabel('Temperature' ,fontsize=18)

plt.plot(khi['Maximum temperature'])

plt.plot(df\_result.set\_index('Date')[['Forecasted']],"r--")

df\_result=pd.DataFrame(df\_result)

writer=pd.ExcelWriter(r"Karachi\_maxtemp.xlsx") # CHANGE ACCORDINGLY

df\_result.to\_excel(writer,'Sheet1')

writer.save()