import numpy as np

import pandas as pd

import matplotlib.pyplot as plt

import yfinance as yf

from sklearn.preprocessing import MinMaxScaler

from tensorflow.keras.models import Sequential

from tensorflow.keras.layers import Dense, LSTM

# Step 1: Load Stock Data

def load\_stock\_data(ticker='AAPL', start='2015-01-01', end='2024-01-01'):

df = yf.download(ticker, start=start, end=end)

return df[['Close']]

# Step 2: Preprocess Data

def preprocess\_data(data, time\_step=60):

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

data\_scaled = scaler.fit\_transform(data)

X, y = [], []

for i in range(time\_step, len(data\_scaled)):

X.append(data\_scaled[i-time\_step:i, 0])

y.append(data\_scaled[i, 0])

X, y = np.array(X), np.array(y)

X = np.reshape(X, (X.shape[0], X.shape[1], 1)) # LSTM input shape

return X, y, scaler

# Step 3: Build LSTM Model

def build\_model(input\_shape):

model = Sequential()

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

model.add(LSTM(units=50))

model.add(Dense(1))

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

return model

# Step 4: Predict and Visualize

def plot\_predictions(actual, predicted):

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

plt.plot(actual, label='Actual Price')

plt.plot(predicted, label='Predicted Price')

plt.title('Stock Price Prediction')

plt.xlabel('Time')

plt.ylabel('Price')

plt.legend()

plt.show()

# Main Execution

if \_\_name\_\_ == "\_\_main\_\_":

df = load\_stock\_data('AAPL')

X, y, scaler = preprocess\_data(df.values)

model = build\_model((X.shape[1], 1))

model.fit(X, y, epochs=10, batch\_size=32, verbose=1)

# Predict on the same data (for simplicity)

predicted = model.predict(X)

predicted\_prices = scaler.inverse\_transform(predicted.reshape(-1, 1))

actual\_prices = scaler.inverse\_transform(y.reshape(-1, 1))

plot\_predictions(actual\_prices, predicted\_prices)