EX:No.9 221501043

12/04/25

**Program to develop neural network-based time series forecasting model.**

**Aim:**

Write a program to develop neural network-based time series forecasting model.

**Algorithm:**

1. **Load the Data**:
   * Read the CSV file containing the weather data.
   * Parse the date column as a datetime index.
2. **Clean the Data**:
   * Handle missing values by performing forward and backward filling.
   * Drop any remaining NaN values.
3. **Normalize the Data**:
   * Apply **Min-Max Scaling** to normalize each column's values between 0 and 1.
4. **Add Time-Based Features**:
   * Extract additional features from the datetime index: day, month and year
5. **Visualize the Data**:
   * Plot the time series for a specific column (e.g., temperature T) over time.
6. **Execute the Program**:
   * Sequentially call the functions to load, clean, normalize, add features, and visualize the data.

**Code:**

import pandas as pd

import numpy as np

import matplotlib.pyplot as plt

from tensorflow.keras.models import Sequential

from tensorflow.keras.layers import LSTM, Dense

from sklearn.preprocessing import MinMaxScaler

# Load and prepare the data

df = pd.read\_csv("daily-minimum-temperatures-in-me.csv")

df.columns = ['Date', 'Temperature']

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

df.set\_index('Date', inplace=True)

df = df[df['Temperature'].str.match(r'^-?\d+(\.\d+)?$', na=False)]

df['Temperature'] = df['Temperature'].astype(float)

# Normalize the data

scaler = MinMaxScaler()

scaled\_data = scaler.fit\_transform(df['Temperature'].values.reshape(-1, 1))

# Prepare the dataset for LSTM

def create\_dataset(data, look\_back=30):

X, y = [], []

for i in range(len(data) - look\_back):

X.append(data[i:i + look\_back])

y.append(data[i + look\_back])

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

look\_back = 30

X, y = create\_dataset(scaled\_data, look\_back)

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

# Split into train and test

train\_size = int(len(X) \* 0.8)

X\_train, X\_test = X[:train\_size], X[train\_size:]

y\_train, y\_test = y[:train\_size], y[train\_size:]

# Build the LSTM model

model = Sequential()

model.add(LSTM(50, return\_sequences=False, input\_shape=(look\_back, 1)))

model.add(Dense(1))

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

# Train the model

model.fit(X\_train, y\_train, epochs=10, batch\_size=16, verbose=1)

# Forecast the next 30 values

last\_30\_days = scaled\_data[-look\_back:]

forecast\_input = last\_30\_days.reshape(1, look\_back, 1)

forecast = []

for \_ in range(30):

pred = model.predict(forecast\_input)[0]

forecast.append(pred)

forecast\_input = np.append(forecast\_input[:, 1:, :], [[pred]], axis=1)

forecast = scaler.inverse\_transform(forecast)

# Plot

forecast\_dates = pd.date\_range(start=df.index[-1] + pd.Timedelta(days=1), periods=30)

plt.figure(figsize=(12, 6))

plt.plot(df.index[-100:], df['Temperature'][-100:], label='Historical Data')

plt.plot(forecast\_dates, forecast, label='LSTM Forecast (30 Days)', color='red')

plt.legend()

plt.title("Neural Network (LSTM) Forecast")

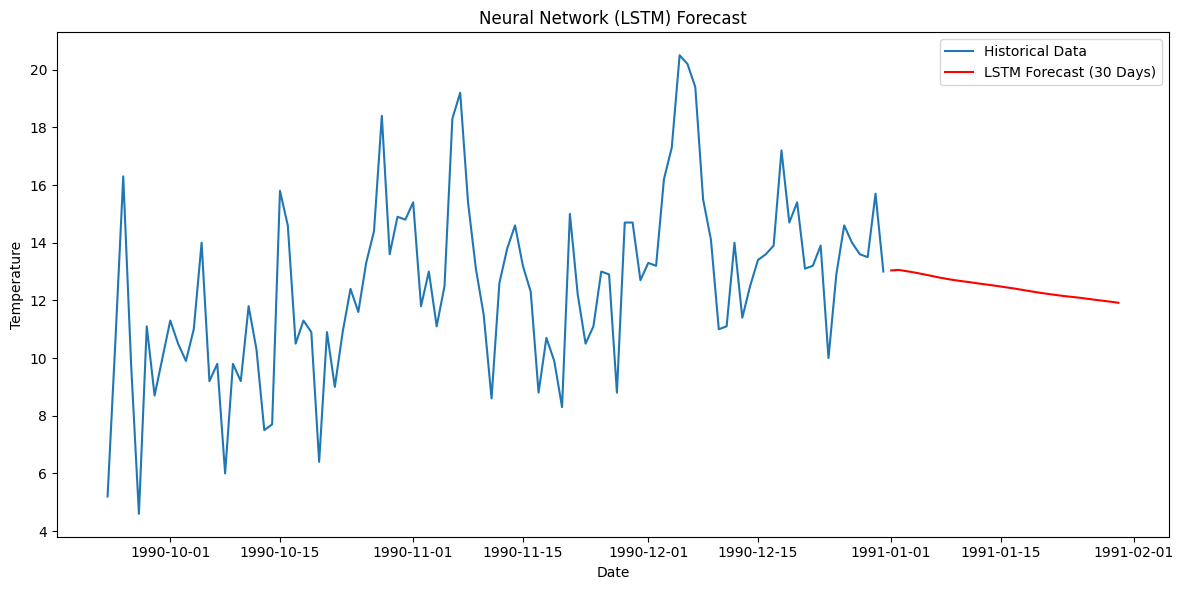
plt.xlabel("Date")

plt.ylabel("Temperature")

plt.tight\_layout()

plt.show()

**Output:**



**Result:**

Thus, the program to implement neural network using the time series data has been done successfully.