**9.Develop neural network-based time series forecasting model**

**AIM:**

To develop neural network-based time series forecasting model using the given dataset.

**PROCEDURE:**

1.Import the necessary libraries:

import pandas as pd

import numpy as np

import matplotlib.pyplot as plt

from sklearn.preprocessing import MinMaxScaler

from tensorflow.keras.models import Sequential

from tensorflow.keras.layers import LSTM, Dense

2.Load dataset:

df = pd.read\_csv('/content/PRICE\_AND\_DEMAND\_201801\_NSW1.csv')

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

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

3. Select only numeric columns for resampling:

numeric\_columns = ['TOTALDEMAND']

df = df[numeric\_columns]

4. Resample to hourly intervals and drop NaNs:

df = df.resample('H').mean()

df.dropna(inplace=True)

5. Normalize the data:

scaler = MinMaxScaler()

data\_scaled = scaler.fit\_transform(df[['TOTALDEMAND']])

6. Create sequences:

def create\_sequences(data, seq\_length):

    X, y = [], []

    for i in range(len(data) - seq\_length):

        X.append(data[i:i+seq\_length])

        y.append(data[i+seq\_length])

    return np.array(X), np.array(y)

SEQ\_LENGTH = 24

X, y = create\_sequences(data\_scaled, SEQ\_LENGTH)

7. Train-test split:

split = int(0.8 \* len(X))

X\_train, X\_test = X[:split], X[split:]

y\_train, y\_test = y[:split], y[split:]

model = Sequential([

    LSTM(64, activation='relu', input\_shape=(SEQ\_LENGTH, 1)),

    Dense(1)

])

model.compile(optimizer='adam', loss='mse')

model.fit(X\_train, y\_train, epochs=10, batch\_size=32, validation\_split=0.1)

8. Evaluate and plot results:

predictions = model.predict(X\_test)

predicted\_demand = scaler.inverse\_transform(predictions)

actual\_demand = scaler.inverse\_transform(y\_test)

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

plt.plot(actual\_demand, label='Actual Demand')

plt.plot(predicted\_demand, label='Predicted Demand')

plt.legend()

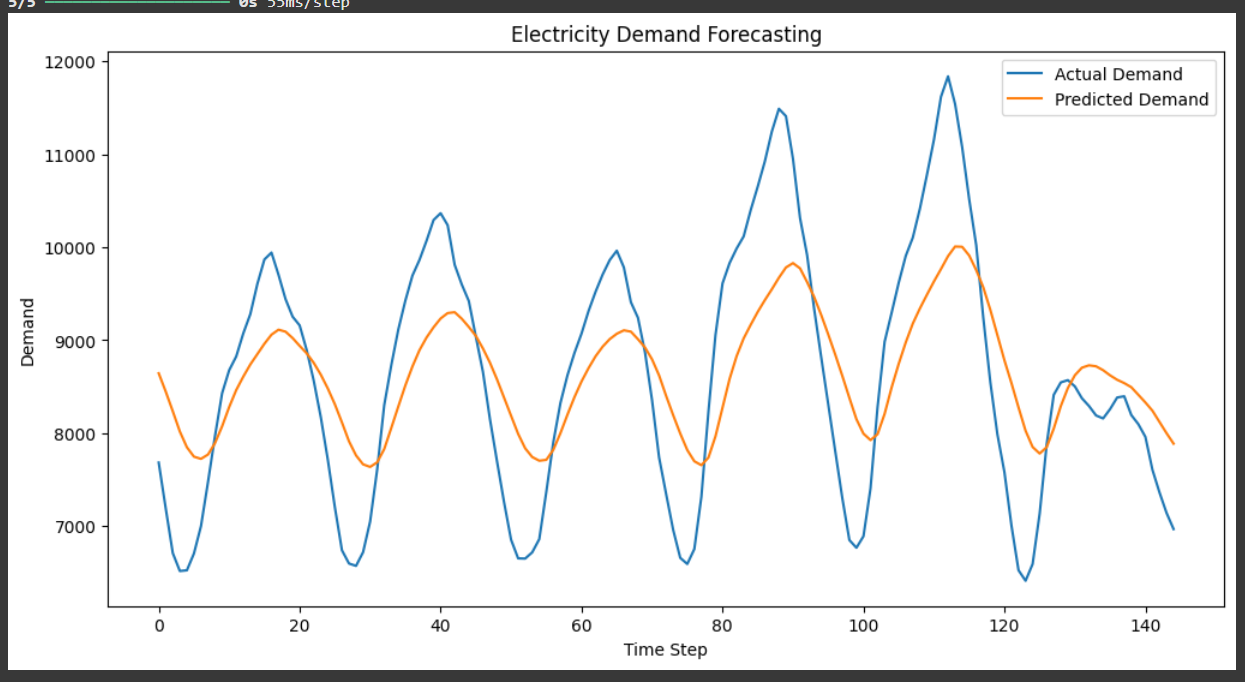
plt.title("Electricity Demand Forecasting")

plt.xlabel("Time Step")

plt.ylabel("Demand")

plt.show()

**OUTPUT:**



**RESULT:**

Thus the program has been implemented and executed successfully.