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

**EX:No.9 DATE:12/04/25**

# AIM:

To Develop neural network-based time series forecasting model.

# ALGORITHM:

1. **Data Cleaning** – Loaded the dataset, parsed dates, fixed encoding issues, and selected only the PM2.5 column.
2. **Normalization** – Scaled PM2.5 values between 0 and 1 using MinMaxScaler to improve neural network performance.
3. **Sequence Creation** – Created supervised learning format by using the previous 10 timesteps to predict the next one.
4. **Train-Test Split** – Split the dataset into 80% training and 20% testing sets.
5. **Model Building** – Built an LSTM model with one LSTM layer (50 units) and one Dense output layer.
6. **Model Training** – Trained the model using training data for 20 epochs with a batch size of 32.
7. **Prediction & Inverse Scaling** – Predicted future values and converted them back to original scale.
8. **Visualization** – Plot values to evaluate model performance.

# CODE :

# 

# 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

# from sklearn.metrics import mean\_squared\_error

# df = pd.read\_csv('/content/Super\_Store\_data.csv', encoding='ISO-8859-1')

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

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

# monthly\_sales = df['Sales'].resample('M').sum()

# scaler = MinMaxScaler()

# sales\_scaled = scaler.fit\_transform(monthly\_sales.values.reshape(-1, 1))

# def create\_dataset(series, look\_back=6):

# X, y = [], []

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

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

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

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

# look\_back = 6

# X, y = create\_dataset(sales\_scaled, look\_back)

# 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:]

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

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

# model = Sequential()

# model.add(LSTM(64, activation='relu', input\_shape=(look\_back, 1)))

# model.add(Dense(1))

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

# model.fit(X\_train, y\_train, epochs=200, batch\_size=4, verbose=0)

# y\_pred = model.predict(X\_test)

# y\_pred\_inv = scaler.inverse\_transform(y\_pred.reshape(-1, 1))

# y\_test\_inv = scaler.inverse\_transform(y\_test.reshape(-1, 1))

# plt.figure(figsize=(10, 4))

# plt.plot(y\_test\_inv, label='Actual')

# plt.plot(y\_pred\_inv, label='Predicted')

# plt.title('LSTM - Monthly Sales Forecast')

# plt.xlabel('Months (Test Set)')

# plt.ylabel('Sales')

# plt.legend()

# plt.grid(True)

# plt.tight\_layout()

# plt.show()

# OUTPUT:

# 

**RESULT:**

Thus, the program using the time series data implementation has been done successfully.