EX:No.9 **DATE:12/04/25**	**Develop neural network-based time series forecasting model.**

AIM:

To develop a neural network-based time series forecasting model using LSTM (Long Short-Term Memory) to predict future values of the 'MA' variable from the given time-series dataset.

ALGORITHM:

Step 1: Import necessary Libraries Load Pandas, NumPy, Matplotlib, Keras, and Scikit-learn.

Step 2: Load Data the dataset and parse the date column.

Step 3: Resample the data to a monthly frequency and handle missing values using interpolation.

Step 4: Create input/output sequences for LSTM.

Step 5: Fit model on training data.

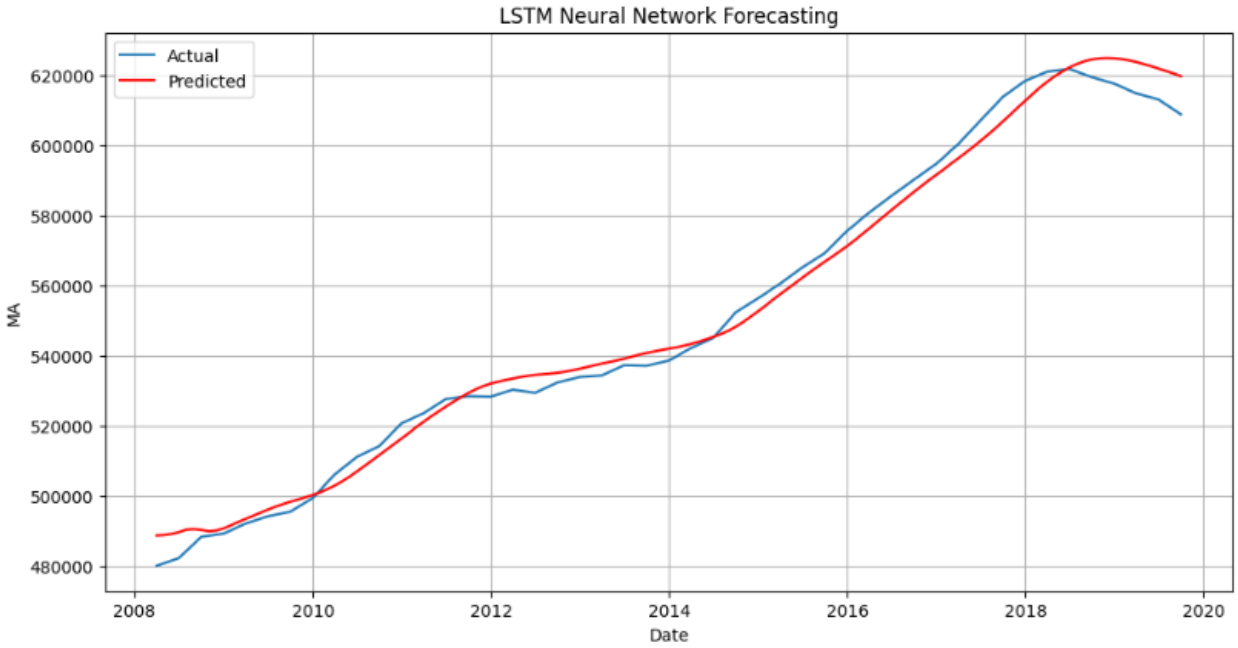
Step 6: Predict & Plot future values and plot actual vs predicted.

CODE AND DESCRIPTION:

```
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
from sklearn.preprocessing import MinMaxScaler
from keras.models import Sequential
from keras.layers import Dense, LSTM

# Load the dataset
file_path = "/content/ma_lga_12345.csv"
df = pd.read_csv(file_path)

# Convert 'saledate' to datetime and set as index
df['saledate'] = pd.to_datetime(df['saledate'], format="%d/%m/%Y")
df.set_index('saledate', inplace=True)
# Use 'MA' column and resample to monthly frequency
monthly_df = df['MA'].resample('ME').mean().interpolate()
```

```
# Normalize the data
scaler = MinMaxScaler()
scaled_data = scaler.fit_transform(monthly_df.values.reshape(-1, 1))

# Prepare data for LSTM
X, y = [], []
window_size = 12  # 12 months look-back
for i in range(window_size, len(scaled_data)):
    X.append(scaled_data[i - window_size:i, 0])
    y.append(scaled_data[i, 0])

X, y = np.array(X), np.array(y)
X = X.reshape((X.shape[0], X.shape[1], 1))  # 3D input for LSTM

# Build the LSTM model
model = Sequential()
model.add(LSTM(units=50, input_shape=(X.shape[1], 1)))
model.add(Dense(1))
model.compile(optimizer='adam', loss='mean_squared_error')

# Train the model
model.fit(X, y, epochs=50, batch_size=16, verbose=1)

# Predict
predicted = model.predict(X)
predicted = scaler.inverse_transform(predicted)
actual = scaler.inverse_transform(y.reshape(-1, 1))

# Plot actual vs predicted
plt.figure(figsize=(12, 6))
plt.plot(monthly_df.index[window_size:], actual, label='Actual')
plt.plot(monthly_df.index[window_size:], predicted, label='Predicted', color='red')
plt.title('LSTM Neural Network Forecasting')
plt.xlabel('Date')
plt.ylabel('MA')
plt.legend()
plt.grid(True)
plt.show()
```

OUTPUT



RESULT:

Thus, the program has been completed and verified successfully.