EXP:9 17/04/2025	**NEURAL NETWORK MODEL FOR FORECASTING IN TIME SERIES DATA.**

AIM:

To Implement a program for Developing neural network-based time series forecasting model.

PROCEDURE:

```
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

# Load the dataset
df = pd.read_csv("monthly-beer.csv")
df['Month'] = pd.to_datetime(df['Month'])
df.set_index('Month', inplace=True)
df.rename(columns={'Monthly beer production': 'Production'}, inplace=True)

# Normalize the data
scaler = MinMaxScaler()
scaled_data = scaler.fit_transform(df[['Production']])

# 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)

sequence_length = 12
X, y = create_sequences(scaled_data, sequence_length)

# Split into train/test
split = int(len(X) * 0.8)
X_train, X_test = X[:split], X[split:]
```

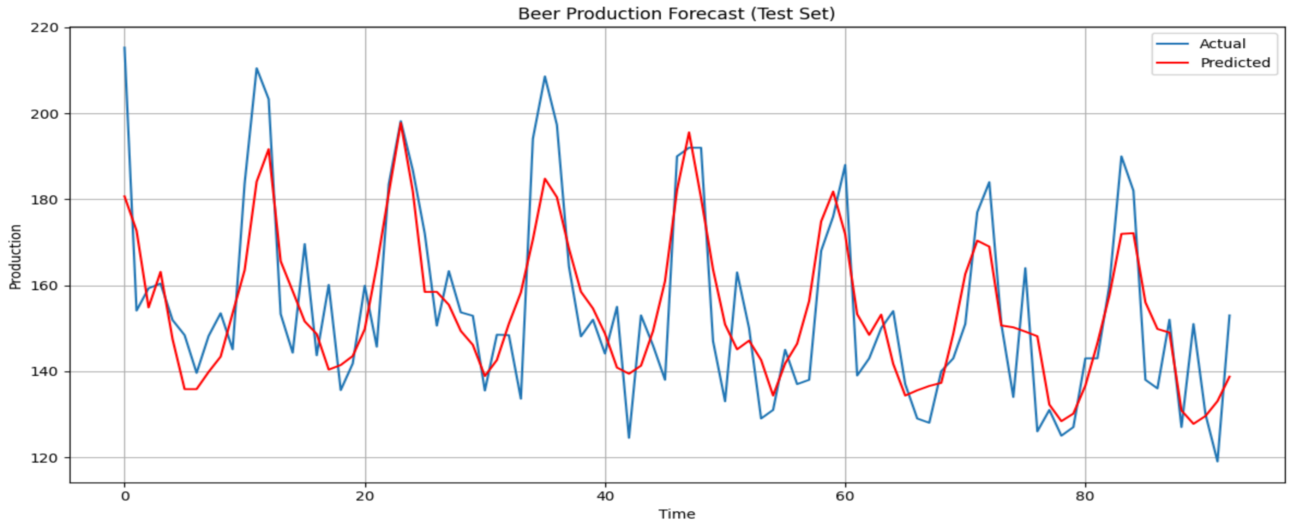
```
y_train, y_test = y[:split], y[split:]

# Build LSTM model
model = Sequential([
    LSTM(50, activation='relu', input_shape=(X.shape[1], 1)),
    Dense(1)
])
model.compile(optimizer='adam', loss='mse')
model.fit(X_train, y_train, epochs=50, verbose=1)

# Predict on test set
predictions = model.predict(X_test)
predicted_values = scaler.inverse_transform(predictions)
actual_values = scaler.inverse_transform(y_test)

# Plot predictions
plt.figure(figsize=(12, 6))
plt.plot(actual_values, label='Actual')
plt.plot(predicted_values, label='Predicted', color='red')
plt.title('Beer Production Forecast (Test Set)')
plt.xlabel('Time')
plt.ylabel('Production')
plt.legend()
plt.grid(True)
plt.tight_layout()
plt.show()
```

OUTPUT:



RESULT :

Thus the Program has been Implemented and executed successfully.