**EX:No.3 221501034**

**07/01/25**

**IMPLEMENTING LINEAR REGRESSION MODEL USING TIME SERIES DATASET**

**AIM**:

To implement linear regression using time series data

**IMPLEMENTATION**:

import numpy as np

import pandas as pd

import matplotlib.pyplot as plt

from sklearn.model\_selection import train\_test\_split

from sklearn.linear\_model import LinearRegression

from sklearn.metrics import mean\_squared\_error

# Step 1: Generate Synthetic Time Series Data

np.random.seed(42)

n = 100 # Number of data points

time = np.arange(n) # Time steps

trend = 0.5 \* time # Linear trend

noise = np.random.normal(0, 5, n) # Gaussian noise

y = trend + noise # Synthetic target variable

# Create a DataFrame

df = pd.DataFrame({'Time': time, 'Value': y})

# Step 2: Split Data into Training and Testing Sets

X = df[['Time']]

y = df['Value']

X\_train, X\_test, y\_train, y\_test = train\_test\_split(X, y, test\_size=0.2, shuffle=False)

# Step 3: Train a Linear Regression Model

model = LinearRegression()

model.fit(X\_train, y\_train)

# Step 4: Predict and Evaluate

y\_pred = model.predict(X)

mse = mean\_squared\_error(y\_test, model.predict(X\_test))

rmse = np.sqrt(mse)

# Print Model Performance

print(f'RMSE: {rmse:.2f}')

# Plot Results

plt.figure(figsize=(10, 5))

plt.scatter(X.values, y, label='Actual Data', color='blue', alpha=0.5)

plt.plot(X.values, y\_pred, label='Linear Regression Prediction', color='red', linewidth=2)

plt.xlabel('Time')

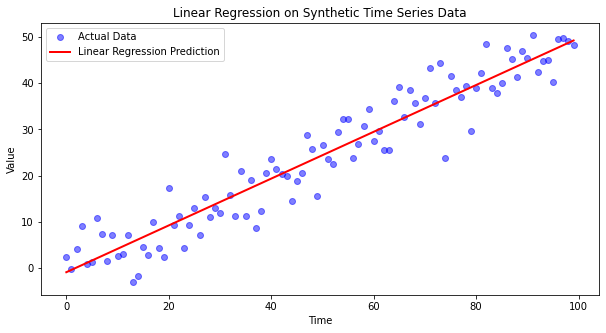
plt.ylabel('Value')

plt.legend()

plt.title('Linear Regression on Synthetic Time Series Data')

plt.show()

**OUTPUT** :



**RESULT**:

The program is created and executed successfully