EX:No.6 221501060

05/04/25

**Implement program to apply moving average smoothing for data preparation and time series forecasting.**

**Aim:**

To Implement program to apply moving average smoothing for data preparation and time series forecasting**.**

**Algorithm:**

1. **Load the Data**:
   * Read the CSV file containing the weather data.
   * Parse the date column as a datetime index.
2. **Clean the Data**:
   * Handle missing values by performing forward and backward filling.
   * Drop any remaining NaN values.
3. **Normalize the Data**:
   * Apply **Min-Max Scaling** to normalize each column's values between 0 and 1.
4. **Add Time-Based Features**:
   * Extract additional features from the datetime index: day, month and year
5. **Visualize the Data**:
   * Plot the time series for a specific column (e.g., temperature T) over time.
6. **Execute the Program**:
   * Sequentially call the functions to load, clean, normalize, add features, and visualize the data.

**Code:**

import pandas as pd

import numpy as np

import matplotlib.pyplot as plt

date\_range = pd.date\_range(start='2025-01-01', periods=100, freq='H')

energy\_consumption = np.random.normal(loc=150, scale=10, size=len(date\_range

df = pd.DataFrame({

'Timestamp': date\_range,

'Energy\_Consumption': energy\_consumption

})

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

plt.figure(figsize=(12, 4))

plt.plot(df['Energy\_Consumption'], label='Original Energy Consumption', alpha=0.5)

plt.title("Original Energy Consumption Data")

plt.xlabel("Time")

plt.ylabel("Energy (kWh)")

plt.legend()

plt.show()

window\_size = 5 # you can change this depending on your data (e.g., 24 for daily average on hourly data)

df['Smoothed'] = df['Energy\_Consumption'].rolling(window=window\_size).mean()

plt.figure(figsize=(12, 4))

plt.plot(df['Energy\_Consumption'], label='Original', alpha=0.4)

plt.plot(df['Smoothed'], label=f'{window\_size}-Point Moving Average', color='red')

plt.title("Smoothed Energy Consumption Data")

plt.xlabel("Time")

plt.ylabel("Energy (kWh)")

plt.legend()

plt.show()

forecast\_steps = 10

last\_known = df['Smoothed'].dropna().iloc[-1]

forecast\_index = pd.date\_range(start=df.index[-1] + pd.Timedelta(hours=1), periods=forecast\_steps, freq='H')

forecast\_values = [last\_known] \* forecast\_steps

forecast\_df = pd.DataFrame({'Forecast': forecast\_values}, index=forecast\_index)

plt.figure(figsize=(12, 4))

plt.plot(df['Smoothed'], label='Smoothed Data')

plt.plot(forecast\_df, label='Naive Forecast', linestyle='--', color='green')

plt.title("Naive Forecasting from Smoothed Energy Data")

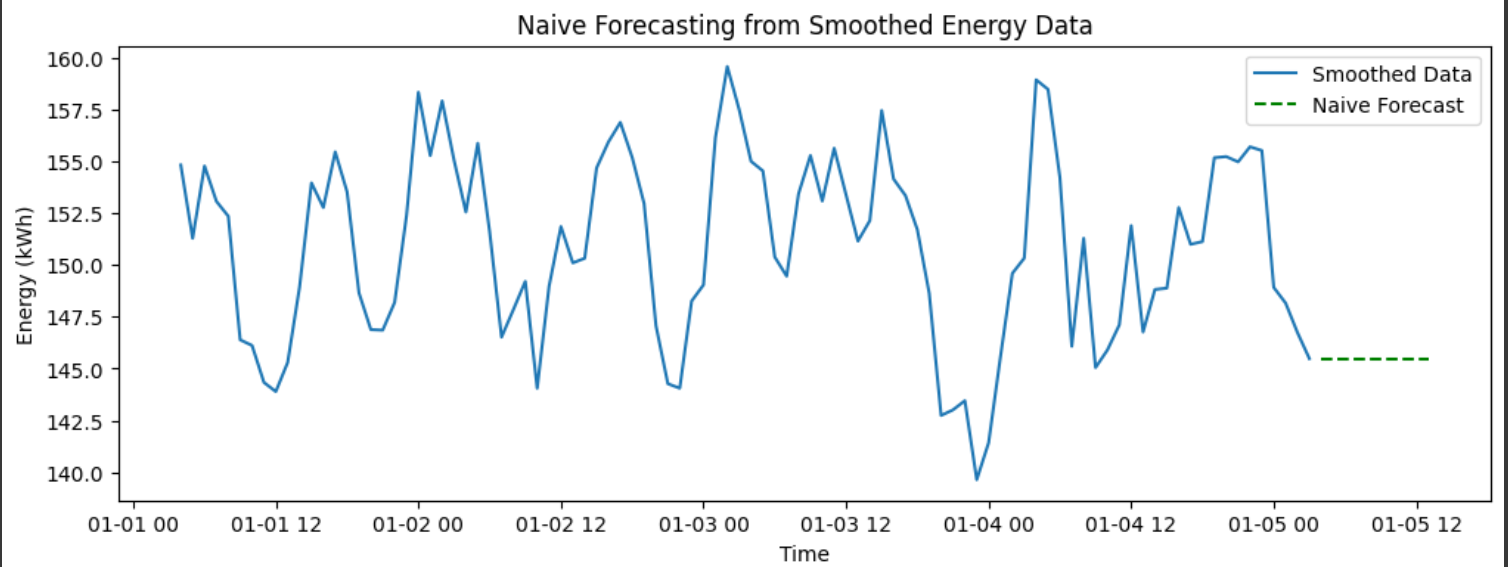
plt.xlabel("Time")

plt.ylabel("Energy (kWh)")

plt.legend()

plt.show()

**Output:**



**Result:**

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