EX:No.6 221501062

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

**Aim:**

Write a program to apply moving average smoothing for data preparation and time series forecasting.

**Algorithm:**

1.**Import Required Libraries**

* Load pandas, numpy, matplotlib.pyplot, and sklearn.metrics.

2.**Load Dataset**

* Read the CSV file and parse 'Date' as a datetime index.

3.**Select Target Column**

* Choose a time series column for smoothing (e.g., Temperature).

4.**Apply Moving Average Smoothing**

* Calculate the **7-day rolling average** of the target column.

5.**Forecast Future Values**

* Use the **last value** of the moving average as the forecast for the next 10 days (simple static forecast).
* Create future dates and build a forecast DataFrame.

6.**Visualization**

* Plot:
  + Original time series.
  + Smoothed series.
  + Forecast for the next 10 days.
* Add labels, title, and legend for clarity.

7.**Evaluate the Smoothing (Optional)**

* Compute **RMSE** between original and smoothed values (excluding initial NaN values).

**Code:**

import numpy as np

import pandas as pd

import matplotlib.pyplot as plt

from sklearn.metrics import mean\_squared\_error

# Load the dataset

df = pd.read\_csv(r"C:\Users\kishore\Downloads\pune\_precipitation data.csv", parse\_dates=['date'], index\_col='date')

# Select the target time series column

time\_series = df['T']

# Apply Moving Average Smoothing with a window size of 7

df['MA\_7'] = time\_series.rolling(window=7).mean()

# Forecasting: Extend the moving average trend into the future (simple approach)

forecast\_steps = 10

last\_ma = df['MA\_7'].dropna().iloc[-1]

forecast = [last\_ma] \* forecast\_steps

forecast\_dates = pd.date\_range(start=df.index[-1] + pd.Timedelta(days=1), periods=forecast\_steps, freq='D')

# Create a forecast DataFrame

forecast\_df = pd.DataFrame({'Date': forecast\_dates, 'Forecast': forecast})

forecast\_df.set\_index('Date', inplace=True)

# Plot original time series and smoothed series

plt.figure(figsize=(14, 6))

plt.plot(time\_series, label='Original Time Series', color='lightblue')

plt.plot(df['MA\_7'], label='7-Day Moving Average', color='orange')

plt.plot(forecast\_df['Forecast'], label='Forecast (next 10 days)', color='red', linestyle='--')

plt.title('Moving Average Smoothing and Forecasting')

plt.xlabel('Date')

plt.ylabel('Temperature')

plt.legend()

plt.grid(True)

plt.tight\_layout()

plt.show()

rmse = np.sqrt(mean\_squared\_error(time\_series[6:], df['MA\_7'][6:]))

print(f'RMSE between Original and Smoothed series: {rmse:.2f}')

**Output:**

**A graph showing a wave of orange lines

AI-generated content may be incorrect.**

**Result:**

Thus, the program to apply moving average smoothing for data preparation and time series forecasting was done.