EX:No.6 221501039

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:

Step 1: Import Libraries

- Import the necessary libraries: `pandas`, `numpy`, and `matplotlib.pyplot`.

Step 2: Create or Load Time Series Data

- You can either generate synthetic time series data or load from a CSV.
- Ensure the index is a `DateTimeIndex`.

Step 3: Apply Moving Average Smoothing

- Use the **rolling mean** technique from `pandas.Series.rolling(window).mean()`:
    - `window` defines how many data points the average should be taken over.
- The moving average reduces the noise and highlights the trend component.

Step 4: Forecast Future Points (Simple Forecasting)

- Use the last smoothed value as a naive forecast (or average of last few).

Step 5: Plot Original vs Smoothed Series

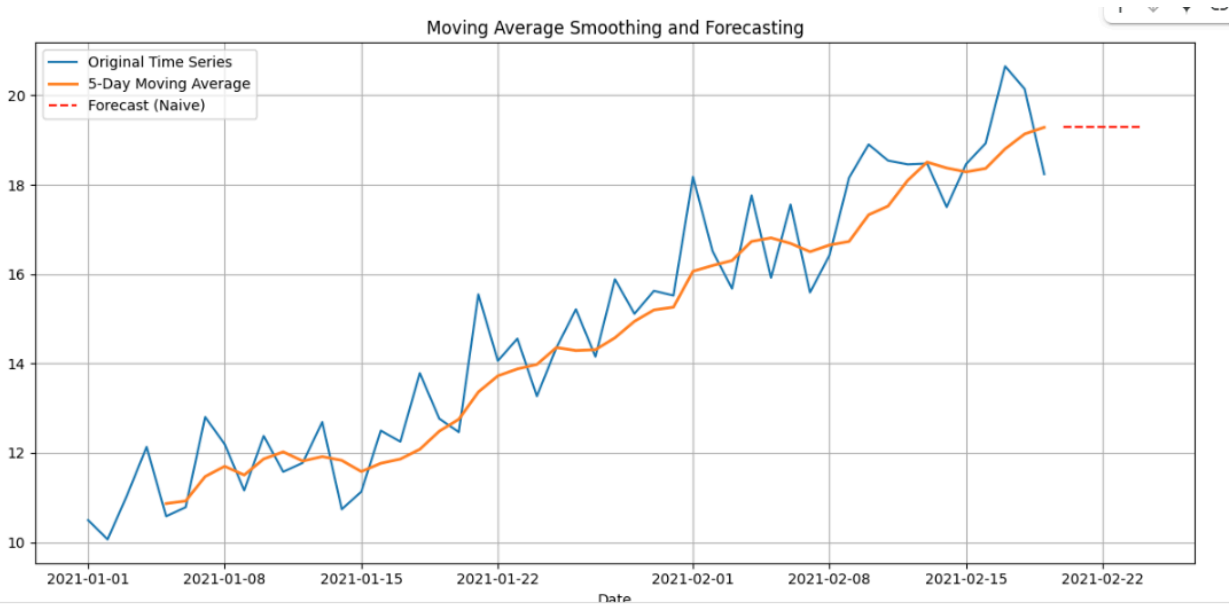
- Plot both the original and smoothed time series to visualize the smoothing effect.

Code:

```
import pandas as pd

import numpy as np

import matplotlib.pyplot as plt


# Step 1: Generate synthetic time series data

np.random.seed(42)

date_range = pd.date_range(start='2021-01-01', periods=50, freq='D')  #

Daily data
```

```
trend = np.linspace(10, 20, 50)
noise = np.random.normal(0, 1, 50)
data = trend + noise

# Step 2: Create time series as a pandas Series
ts = pd.Series(data, index=date_range)

# Step 3: Apply moving average smoothing (window size = 5)
window_size = 5
smoothed_ts = ts.rolling(window=window_size).mean()

# Step 4: Forecasting (Naive: use last smoothed value as future
prediction)
last_value = smoothed_ts.dropna().iloc[-1]
forecast_days = 5
future_dates = pd.date_range(start=ts.index[-1] + pd.Timedelta(days=1),
periods=forecast_days)
forecast_values = pd.Series([last_value] * forecast_days,
index=future_dates)

# Step 5: Plotting
plt.figure(figsize=(12, 6))
plt.plot(ts, label='Original Time Series')
plt.plot(smoothed_ts, label=f'{window_size}-Day Moving Average',
linewidth=2)
plt.plot(forecast_values, label='Forecast (Naive)', linestyle='--',
color='red')
plt.title('Moving Average Smoothing and Forecasting')
plt.xlabel('Date')
plt.ylabel('Value')
plt.legend()
plt.grid(True)
plt.tight_layout()
plt.show()
```

Output:



Result:

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