EX:No.5

Implement programs for estimating & eliminating trend in time series data- aggregation, smoothing.

Aim:

Write a program to estimating & eliminating trend in time series data- aggregation, smoothing.

Algorithm:

Import Libraries

- Load required libraries: numpy, pandas, matplotlib, and statsmodels.

Load the Dataset

- Read the art market price dataset and set the 'Date' column as the index.

Select a Time Series Column

- Choose a column for analysis (e.g., price).

Apply Aggregation (Rolling Mean)

- Compute a 7-day rolling mean to smooth fluctuations in the data.

Apply Smoothing Techniques:

- Moving Average Smoothing: Compute a centered 7-day moving average.
- Exponential Smoothing: Apply Holt-Winters Exponential Smoothing to capture trends.

Extract Trend using Seasonal Decomposition

- Decompose the time series into trend, seasonal, and residual components using additive decomposition.

Visualize the Data

- Plot the original time series.
- Overlay the rolling mean, moving average, and exponential smoothing for comparison.
- Plot the trend component extracted from decomposition.

Output the Results

- Display the smoothed series and extracted trend.
- If needed, detrend the series by subtracting the estimated trend.

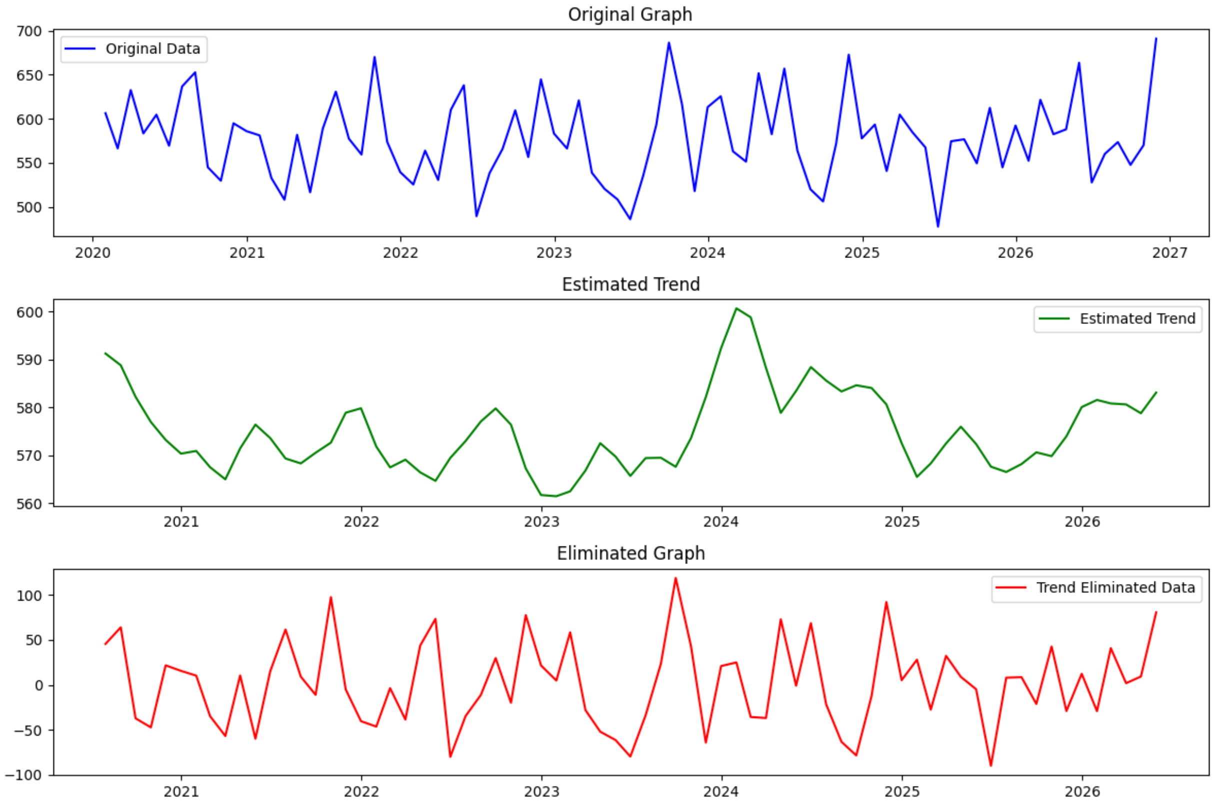
Code:

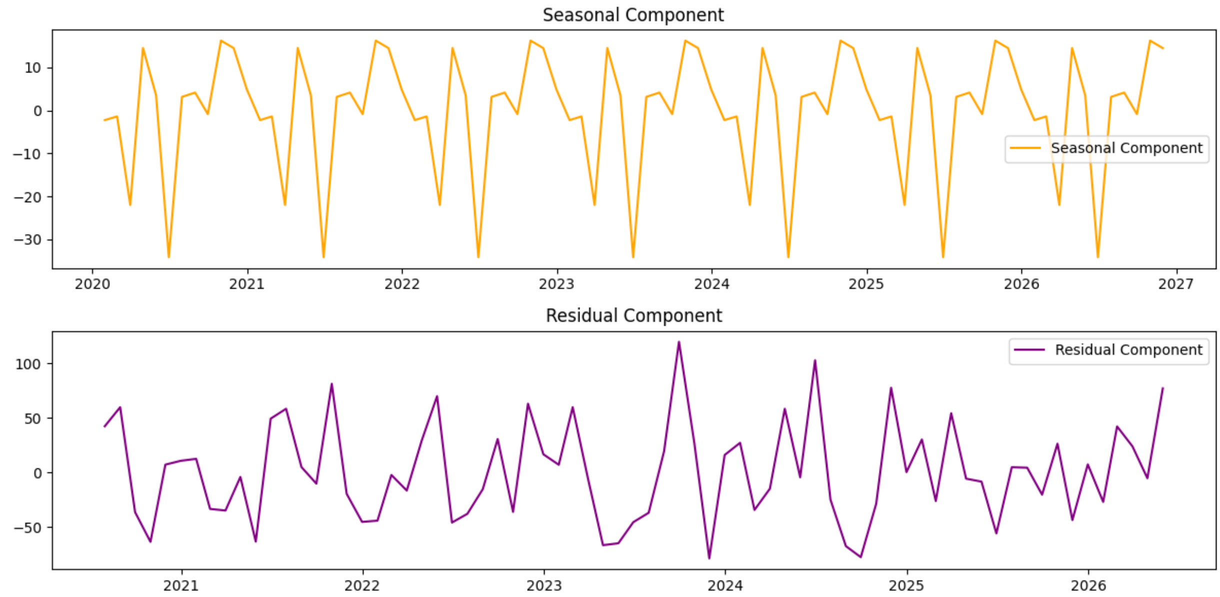
```
import pandas as pd

import matplotlib.pyplot as plt
```

```
from statsmodels.tsa.seasonal import seasonal_decompose
# Load the dataset (replace file name if needed)
file_path = "artmarket_with_dates.csv"
data = pd.read_csv(file_path)
# Convert 'Date' column to datetime and set as index
data['Date'] = pd.to_datetime(data['Date'])
data.set_index('Date', inplace=True)
# Convert 'Price ($)' to numeric and drop missing values
data['Price ($)'] = pd.to_numeric(data['Price ($)'], errors='coerce')
data = data.dropna(subset=['Price ($)'])
# Resample data to monthly frequency and calculate mean price
monthly_data = data['Price ($)'].resample('M').mean()
# Decompose the time series into trend, seasonal, and residual components
decomposition = seasonal_decompose(monthly_data, model='additive', period=12)
# Extract trend, seasonal, and residual components
trend = decomposition.trend
seasonal = decomposition.seasonal
residual = decomposition.resid
# Plot original data
plt.figure(figsize=(12, 8))
plt.subplot(3, 1, 1)
plt.plot(monthly_data, label='Original Data', color='blue')
plt.title('Original Graph')
plt.legend()
# Plot estimated trend separately
plt.subplot(3, 1, 2)
plt.plot(trend, label='Estimated Trend', color='green')
plt.title('Estimated Trend')
plt.legend()
# Plot detrended data (eliminated trend)
detrended_data = monthly_data - trend
plt.subplot(3, 1, 3)
plt.plot(detrended_data, label='Trend Eliminated Data', color='red')
plt.title('Eliminated Graph')
plt.legend()
plt.tight_layout()
plt.show()
# Plot seasonal and residual components separately
plt.figure(figsize=(12, 6))
#Plot seasonal component separately
plt.subplot(2, 1, 1)
plt.plot(seasonal, label='Seasonal Component', color='orange')
plt.title('Seasonal Component')
plt.legend()
# Plot residual component separately
plt.subplot(2, 1, 2)
plt.plot(residual, label='Residual Component', color='purple')
plt.title('Residual Component')
plt.legend()
plt.tight_layout()
plt.show()
```

Output:





Result:

Thus, the program to estimating & eliminating trend in time series data- aggregation, smoothing was done.