Experiment 4 : Smoothing and Trend Removal in Time Series Data

1. Importing Necessary Libraries

In this section, we import essential Python libraries that facilitate data handling, visualization, and time series decomposition. These libraries help in analyzing and processing time series data.

```python
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
from statsmodels.tsa.seasonal import seasonal_decompose
from statsmodels.tsa.holtwinters import ExponentialSmoothing
```

2. Loading the Dataset

The dataset containing stock prices is loaded from a CSV file. The 'Date' column is set as the index to facilitate time series operations, ensuring chronological arrangement.

```python
file_path = '/mnt/data/MSFT.csv'
df = pd.read_csv(file_path, parse_dates=['Date'],
index_col='Date')
```

3. Plotting the Original Time Series

The original time series data is plotted to observe patterns, trends, and fluctuations. This provides insights into the nature of the stock price movements over time.

Python
```python
plt.figure(figsize=(10, 5))
plt.plot(df['Close'], label='Original Time Series', color='blue')
plt.title('Original Time Series')
plt.legend()
plt.show()
```

---

4. Moving Average Smoothing

Moving Average is used to smooth short-term fluctuations and highlight longer-term trends. A rolling window of 10 days is used to compute the average price.

Python
```python
window_size = 10
rolling_mean = df['Close'].rolling(window=window_size).mean()
```

---

5. Exponential Smoothing

Exponential Smoothing applies weighted averages where recent data points have higher influence. It helps in trend identification while keeping responsiveness to changes.

Python
```python
exp_smooth = ExponentialSmoothing(df['Close'], trend='add',
seasonal=None, damped_trend=True).fit()
df['Smoothed'] = exp_smooth.fittedvalues
```

---

6. Trend Extraction using Seasonal Decomposition

Seasonal decomposition splits the time series into trend, seasonal, and residual components. This enables us to analyze and remove the trend component.

```python
decomposition = seasonal_decompose(df['Close'], model='additive', 
period=30)
trend = decomposition.trend
detrended_series = df['Close'] - trend
```

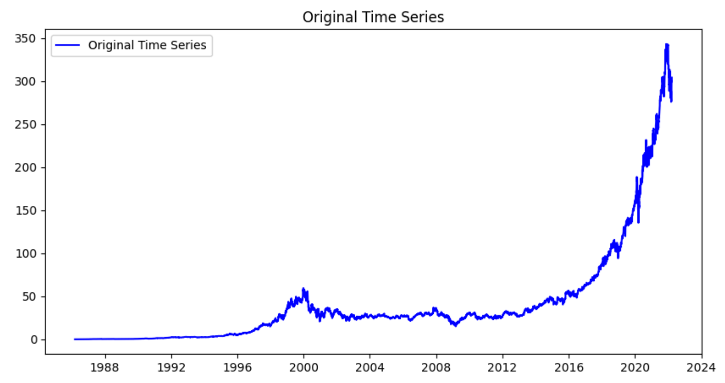
---

7. Visualization: Before and After Trend Removal

A side-by-side visualization compares the original time series with the detrended series, highlighting the effectiveness of the trend removal process.

```python
plt.figure(figsize=(12,6))
plt.subplot(2,1,1)
plt.plot(df['Close'], label='Original', color='blue')
plt.plot(trend, label='Trend', color='red')
plt.title('Before Removing Trend')
plt.legend()

plt.subplot(2,1,2)
plt.plot(detrended_series, label='Detrended Series', 
color='purple')
plt.title('After Removing Trend')
plt.legend()
plt.show()
```



8. Result

The implementation of smoothing techniques and trend removal has been successfully completed. By applying moving average, exponential smoothing, and decomposition, we effectively analyzed and eliminated trends in the time series data.