Exp No: 4 **Implement programs for estimating & eliminating trend in time series data- aggregation, smoothing**

Date: 8/3/25

Objective:

The objective of this experiment is to estimate and eliminate trends in time series data using aggregation and smoothing techniques. This is essential for improving the accuracy of predictive models and ensuring reliable time series forecasting. The experiment focuses on handling missing values, applying moving average smoothing, and utilizing exponential smoothing techniques to preprocess time series data.

Background and Scope:

Time series data consists of observations indexed in time order, which often exhibit trends, seasonal patterns, and random noise. Estimating and eliminating trends is a critical preprocessing step to enhance forecasting models. This experiment covers:

- **Aggregation**: Using a moving average to smooth fluctuations and highlight long-term trends.
- **Smoothing**: Applying exponential smoothing to reduce noise while preserving important patterns.
- **Seasonal Decomposition**: Breaking down the time series into trend, seasonal, and residual components.
- **Holt-Winters Method**: A forecasting technique that considers both trend and seasonality for improved predictions.

By implementing these techniques, we can enhance the data quality for modeling and achieve more accurate forecasting results.

Steps for Time Series Trend Estimation and Elimination

Step 1: Load the Dataset

- The dataset is loaded from a CSV file into a Pandas DataFrame.
- The 'DATE' column is converted to datetime format and set as the index.
- The dataset is converted to a fixed frequency time series to maintain consistency.

```
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

# Load the dataset
```

```
file_path = "AirPassengers.csv"  # Update with correct path

df = pd.read_csv(file_path)

# Convert DATE column to datetime format and set as index

df['Month'] = pd.to_datetime(df['Month'], dayfirst=True)

df.set_index('Month', inplace=True)

# Ensure time series has a fixed frequency (Month Start)

df = df.asfreq('MS')
```

Step 2: Handle Missing Values

- Since missing values interfere with analysis, they are filled using linear interpolation.

```
# Fill missing values

df['#Passengers'] = df['#Passengers'].interpolate(method='linear')
```

Step 3: Apply Moving Average for Aggregation

- A 12-month moving average is applied to smooth out short-term fluctuations and highlight trends.

```
# Moving Average Smoothing

df['Moving_Avg_12'] = df['#Passengers'].rolling(window=12, min_periods=1).mean()
```

Step 4: Apply Exponential Smoothing

- Exponential Smoothing with alpha = 0.3 is used to give more weight to recent observations while reducing noise.

```
# Exponential Smoothing

df['Exp_Smooth_Alpha0.3'] = df['#Passengers'].ewm(alpha=0.3).mean()
```

Step 5: Seasonal Decomposition

- The time series is decomposed into trend, seasonal, and residual components.
- Missing values are handled to ensure smooth decomposition.

```
# Seasonal Decomposition

decomposed = seasonal_decompose(df['#Passengers'], model='additive', period=12)

# Extract components

df['Trend'] = decomposed.trend

df['Seasonal'] = decomposed.seasonal

df['Residual'] = decomposed.resid

# Drop NaN values after decomposition
```

df.dropna(inplace=True)

Step 6: Apply Holt-Winters Exponential Smoothing

- This method accounts for both trend and seasonality to enhance forecasting accuracy.

Holt-Winters Exponential Smoothing

hw_model = ExponentialSmoothing(df['#Passengers'], trend='add', seasonal='add', seasonal_periods=12)

hw_fit = hw_model.fit()

df['HoltWinters'] = hw_fit.fittedvalues

Step 7: Plot Aggregation and Smoothing Results

Plot Original vs Aggregation & Smoothing

plt.figure(figsize=(12, 8))

plt.subplot(3, 1, 1)

plt.plot(df['#Passengers'], label="Original Series", color='blue')

plt.plot(df['Moving_Avg_12'], label="12-Month Moving Avg", color='red', linestyle='dashed')

plt.title("Aggregation: Moving Average")

plt.legend()

plt.subplot(3, 1, 2)

plt.plot(df['#Passengers'], label="Original Series", color='blue')

plt.plot(df['Exp_Smooth_Alpha0.3'], label="Exponential Smoothing (α=0.3)", color='green', linestyle='dashed')

plt.title("Smoothing: Exponential Smoothing")

plt.legend()

plt.subplot(3, 1, 3)

plt.plot(df['#Passengers'], label="Original Series", color='blue')

plt.plot(df['HoltWinters'], label="Holt-Winters Smoothing", color='purple', linestyle='dashed')

plt.title("Trend Removal: Holt-Winters Method")

plt.legend()

plt.tight_layout()

plt.show()

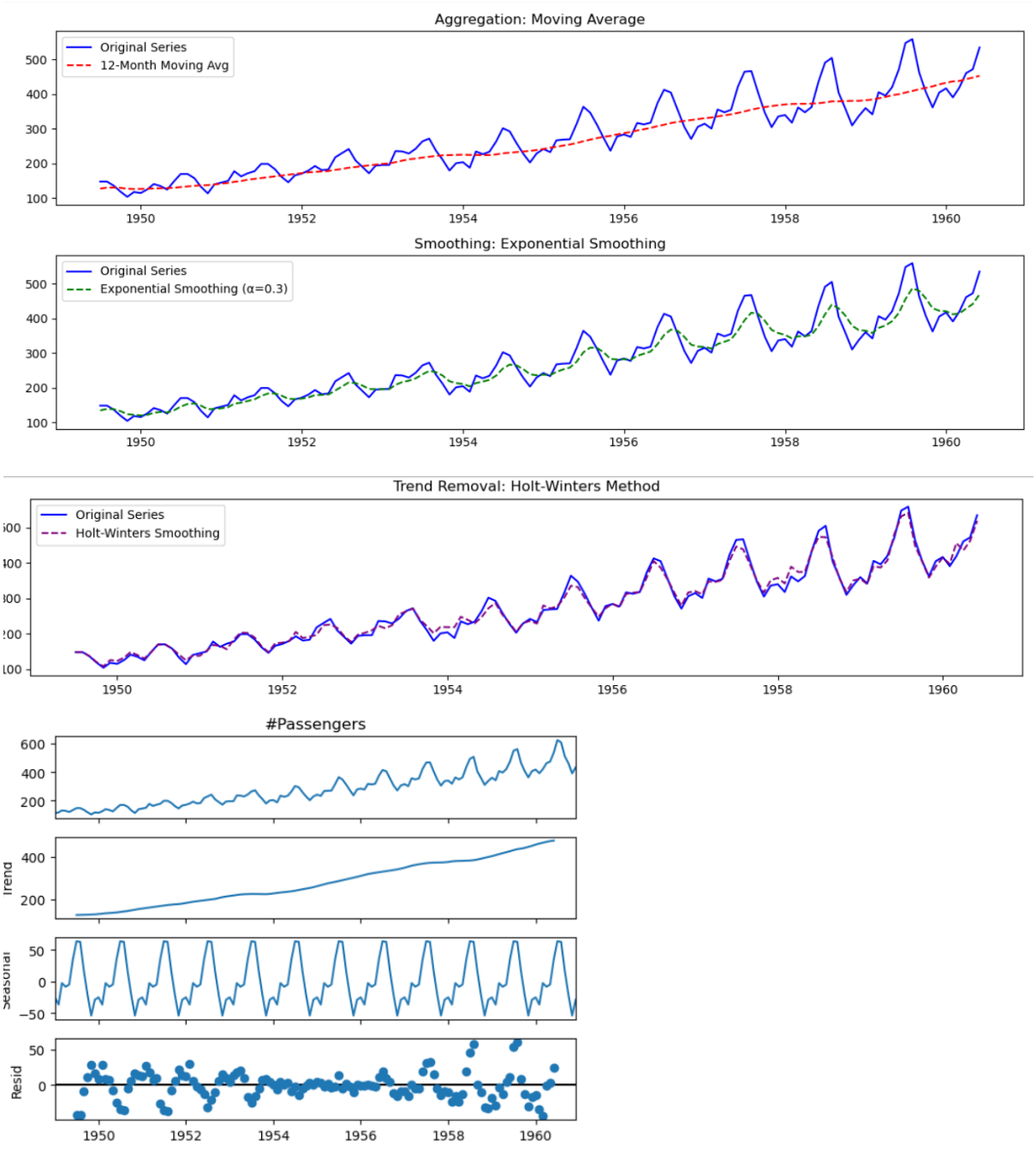
Step 8: Display Seasonal Decomposition Components

Display Seasonal Decomposition Components

decomposed.plot()

plt.show()

Output :



Conclusion:

This experiment successfully demonstrated trend estimation and elimination in time series data using: Moving Average Aggregation to smooth fluctuations. Exponential Smoothing to reduce noise. Seasonal Decomposition to extract key components. Holt-Winters Exponential Smoothing for improved forecasting.