TIME SERIES FORECASTING

BUSINESS REPORT

BY

G S JAIGURURAM

TABLE OF CONTENTS

Problem Statement:

CONTEXT:

As an analyst at ABC Estate Wines, we are presented with historical data encompassing the sales of different types of wines throughout the 20th century. These datasets originate from the same company but represent sales figures for distinct wine varieties. Our objective is to delve into the data, analyze trends, patterns, and factors influencing wine sales over the course of the century. By leveraging data analytics and forecasting techniques, we aim to gain actionable insights that can inform strategic decision-making and optimize sales strategies for the future.

OBJECTIVE:

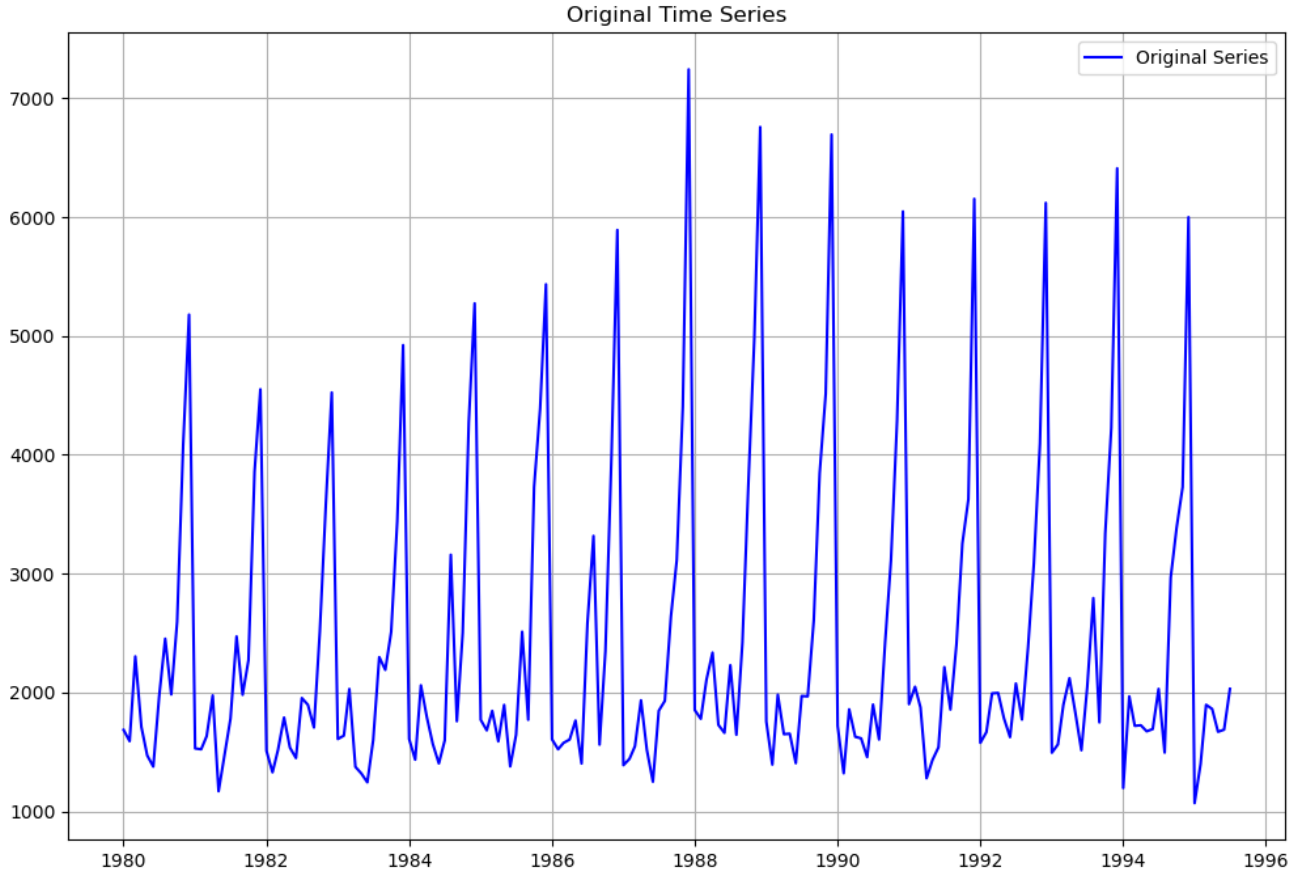
The primary objective of this project is to analyze and forecast wine sales trends for the 20th century based on historical data provided by ABC Estate Wines. We aim to equip ABC Estate Wines with the necessary insights and foresight to enhance sales performance, capitalize on emerging market opportunities, and maintain a competitive edge in the wine industry.

PROBLEM 1.1 DEFINE THE PROBLEM AND PERFORM EXPLORATORY DATA ANALYSIS:

Problem 1.1.1 Read the data as an appropriate time series data:

The data contains two columns: YearMonth and Sparkling. The YearMonth column represents the time period in a "YYYY-MM" format, and the Sparkling column contains the corresponding sales values. So, we will be converting the “YYYY-MM” to “YYYY-MM-DD” format.

Problem 1.1.2 Plot the Data:



Problem 1.1.3 Perform EDA:

Statistics Values: Mean sales: 2402.

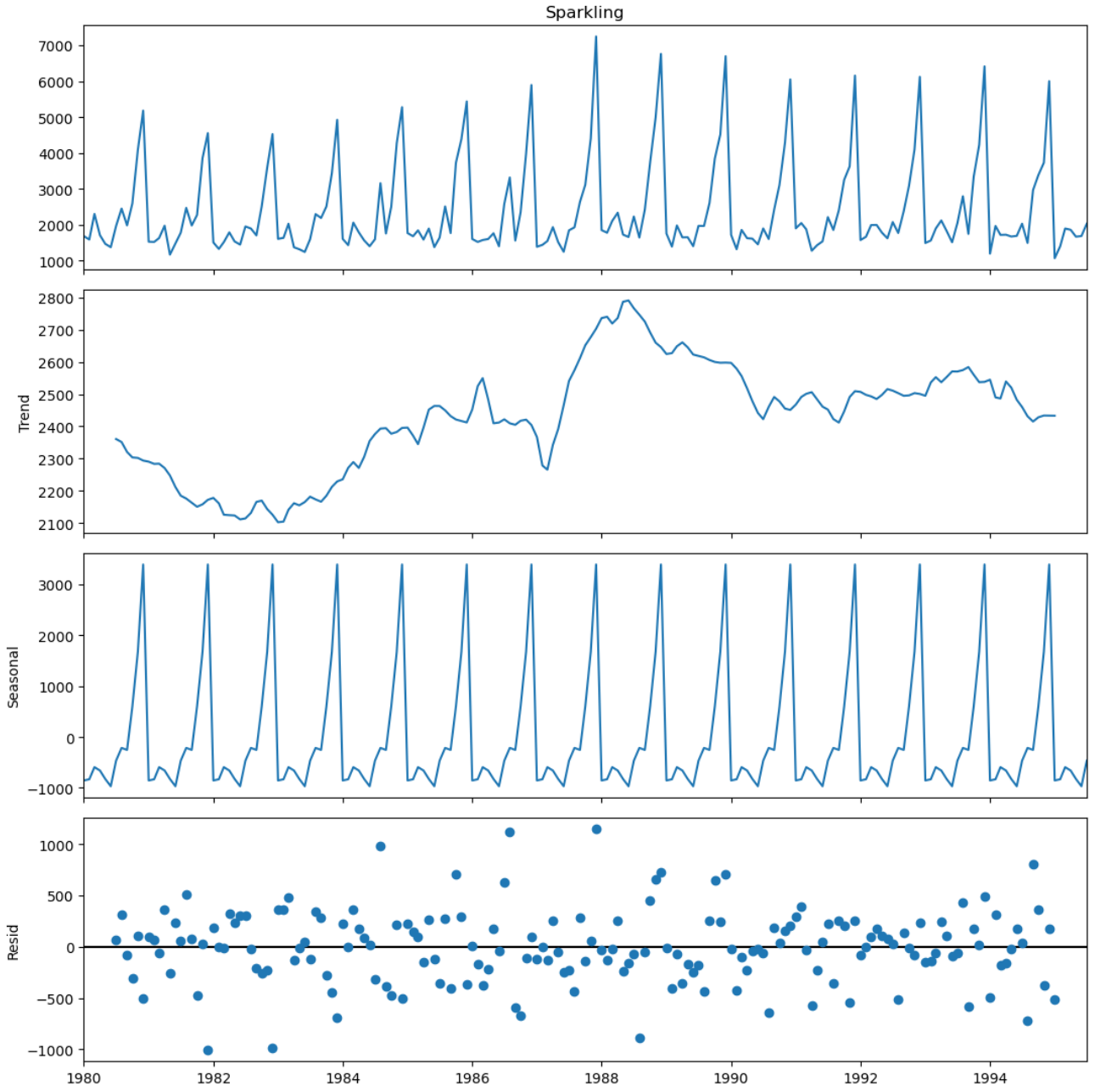
Minimum sales: 1070 & Maximum sales: 7242.

Standard deviation: 1295, Indicating some variability in the sales data.

Problem 1.1.4 Perform Decomposition:

Decomposition: The plot displays the decomposed components:

- Trend: Shows a general upward or downward movement over time.
- Seasonality: Displays repeating patterns at regular intervals.
- Residual: Represents the remaining noise after accounting for trend and seasonality.



PROBLEM 1.2 DATA PRE-PROCESSING:

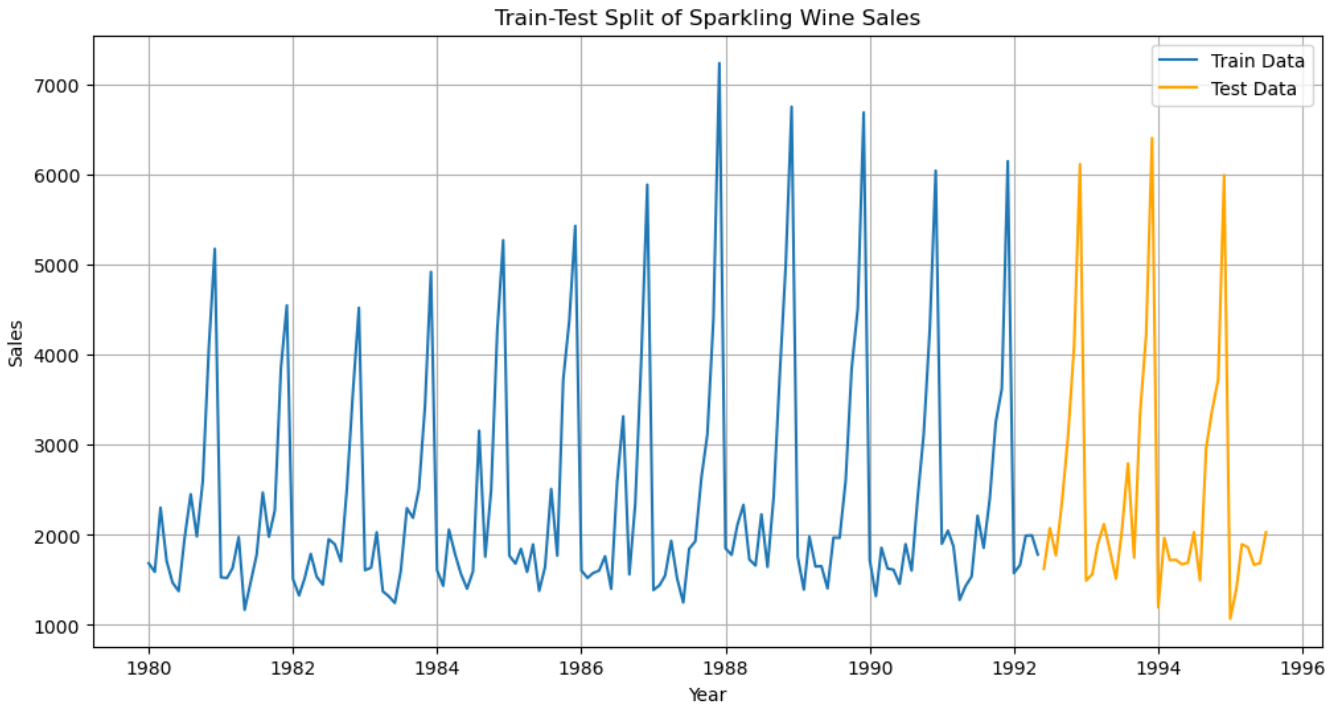
Problem 1.2.1 Missing value treatment:

We use isnull() function to find the missing values and result is that there are no missing values.

Problem 1.2.2 Visualize the processed data - Train-test split:

Since there are no missing values, proceed with train-test split. Splitting data into training (80%) and testing (20%).

Plotting line graph for training and testing data:



PROBLEM 1.3 MODEL BUILDING - ORIGINAL DATA:

Problem 1.3.1 Build forecasting models - Linear regression:

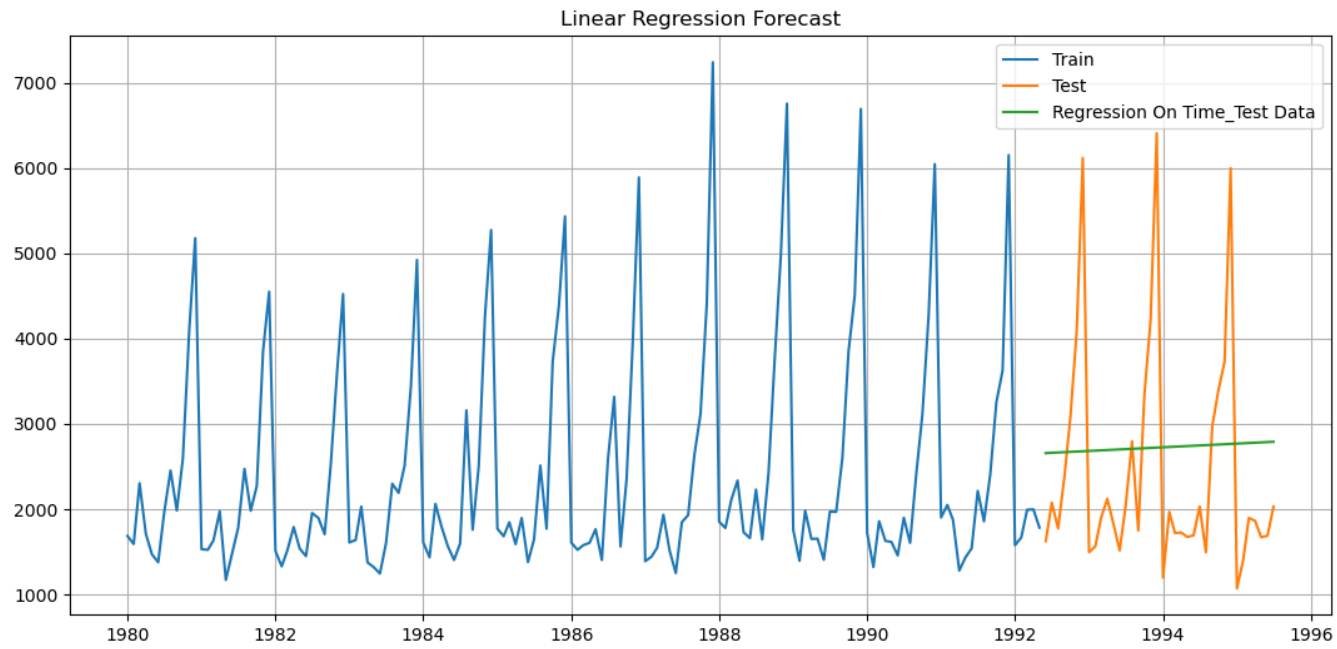
We will create a time instance to build a liner regression model. We take root mean square error value to evaluation of model performance.

Output:

For RegressionOnTime forecast on the Test Data, RMSE is 1359.71

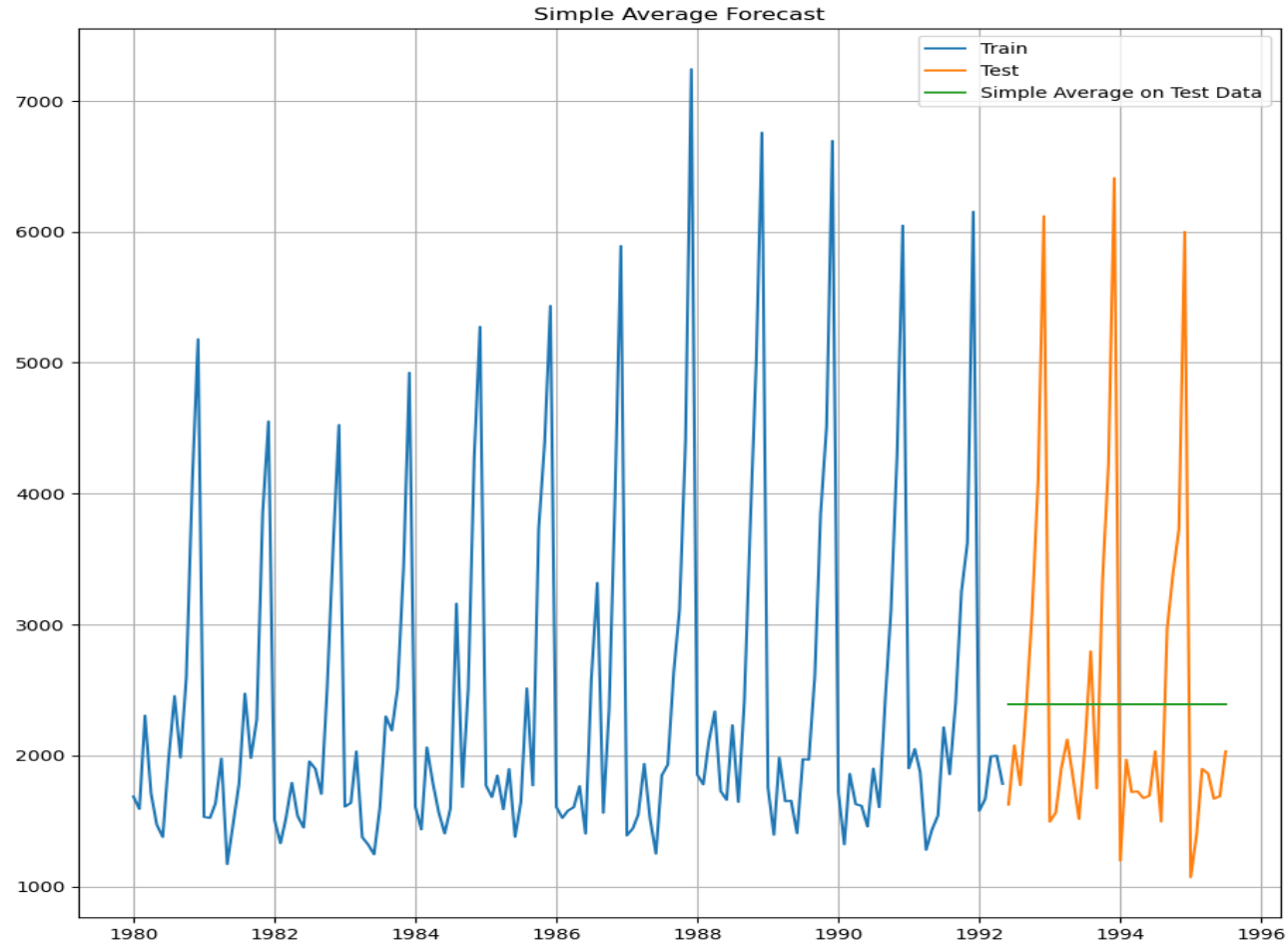
We store this RMSE Value in a new dataframe that will ease the comparison step.

	Test RMSE
RegressionOnTime	1359.708262



Problem 1.3.2 Build forecasting models – Simple Average:

This model forecasts future values based on the average of all past values.



RMSE Value:

For Simple Average forecast on the Test Data, RMSE is 1331.038

RMSE Dataframe:

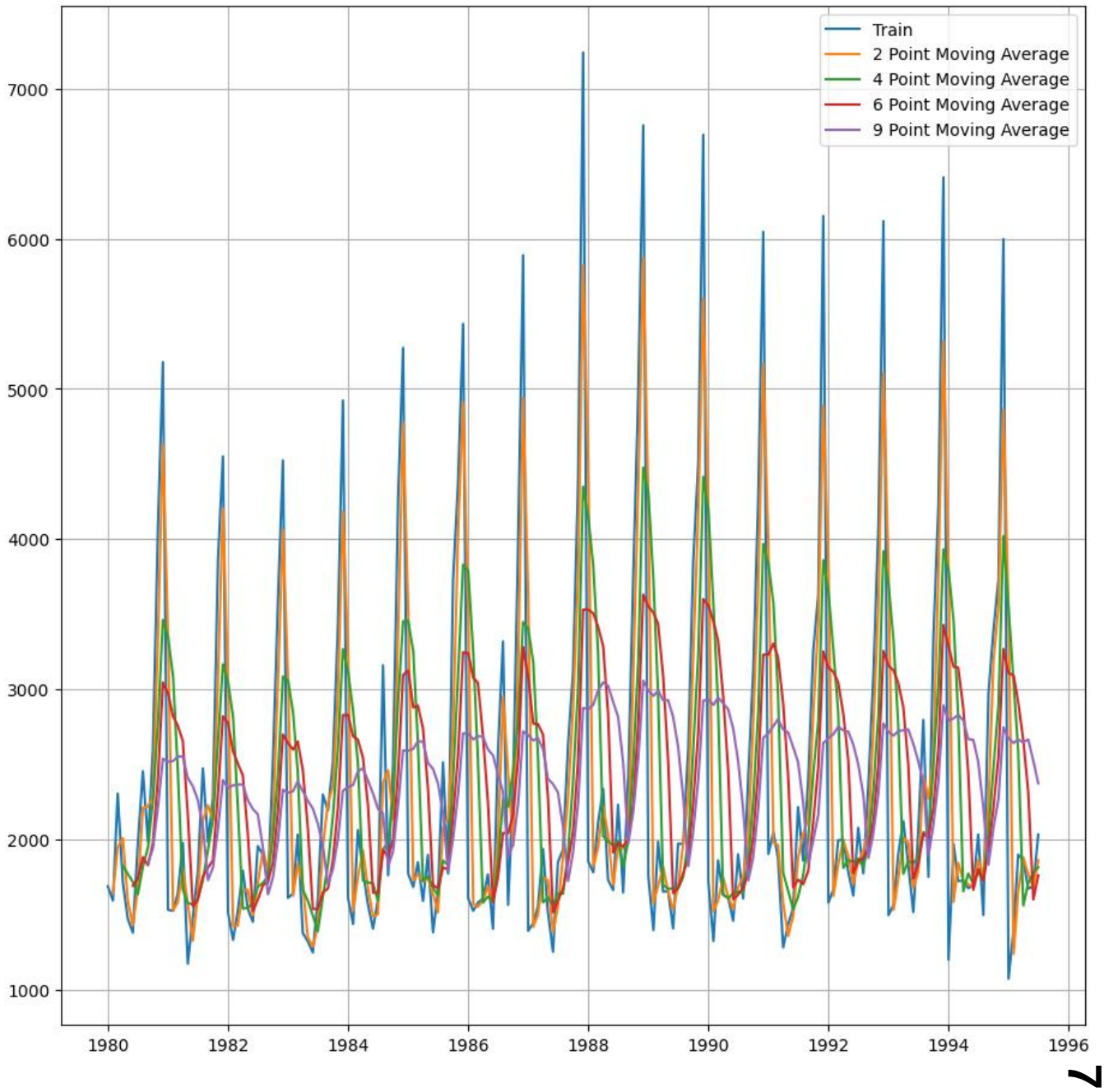
Test RMSE

RegressionOnTime 1359.708262

SimpleAverageModel 1331.037637

Problem 1.3.3 Build forecasting models – Moving Average:

A moving average model forecasts based on the average of a fixed window of past data points.



RMSE Value:

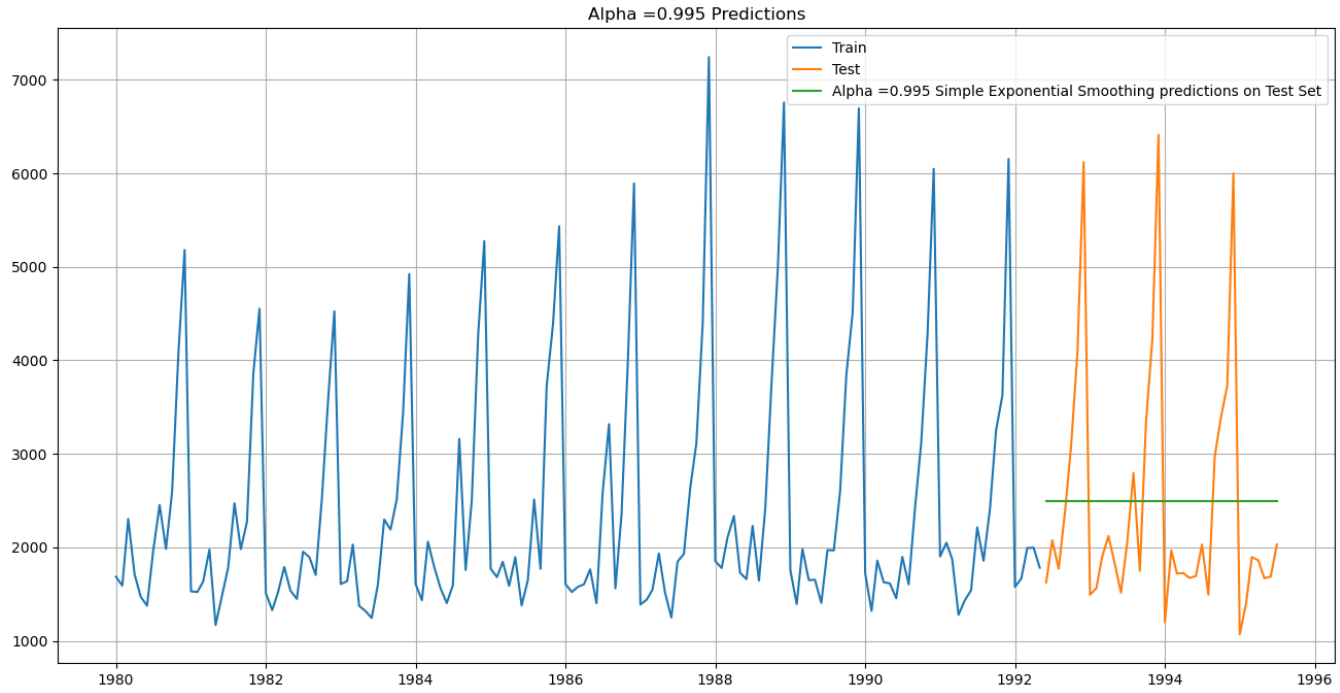
For 2 point Moving Average Model forecast on the Training Data, RMSE is 805.880
For 4 point Moving Average Model forecast on the Training Data, RMSE is 1161.283
For 6 point Moving Average Model forecast on the Training Data, RMSE is 1290.625
For 9 point Moving Average Model forecast on the Training Data, RMSE is 1375.582

RMSE Dataframe:

	Test RMSE
RegressionOnTime	1359.708262
SimpleAverageModel	1331.037637
2pointTrailingMovingAverage	805.879970
4pointTrailingMovingAverage	1161.282792
6pointTrailingMovingAverage	1290.624831
9pointTrailingMovingAverage	1375.582351

Problem 1.3.4 Build forecasting models – Single Exponential Smoothing:

Single Exponential Smoothing gives more weight to recent data.



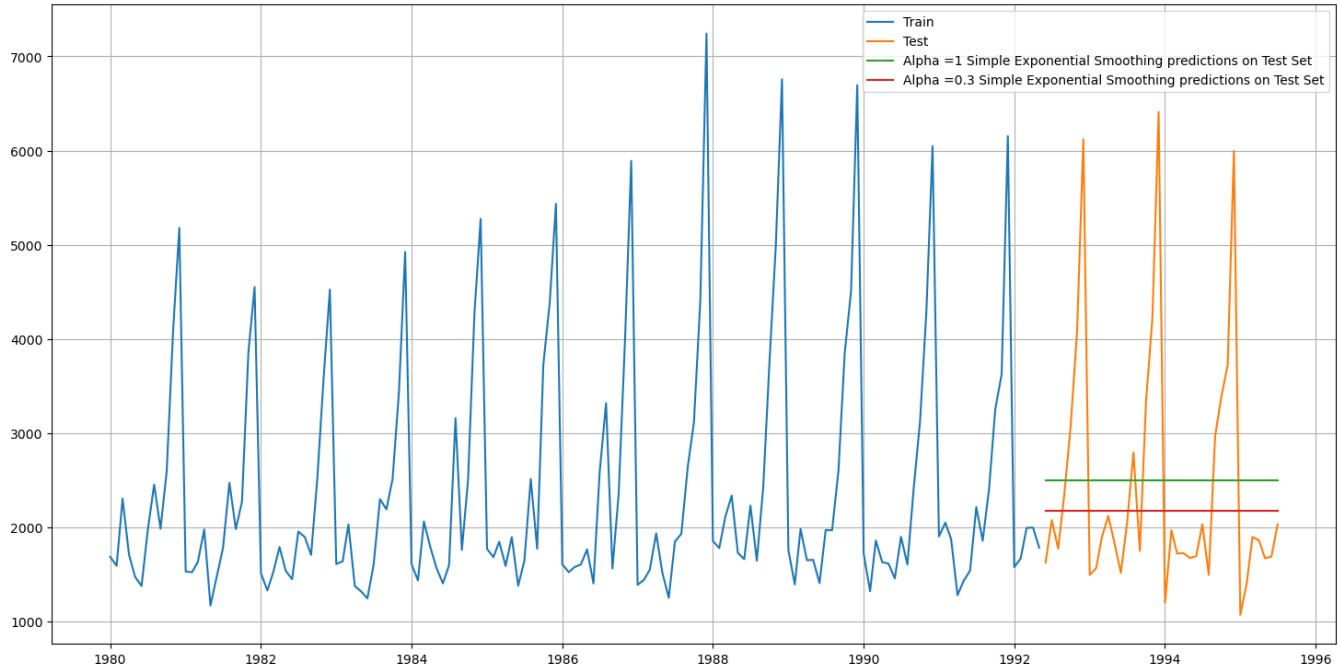
RMSE Value:

For Alpha =0.995 Single Exponential Smoothing Model forecast on the Test Data, RMSE is 1329.836.

We build the same model for different alpha value from 0.3 to 0.9.

	Alpha Values	Train RMSE	Test RMSE
0	0.3	1359.782485	1359.784870
1	0.4	1358.397528	1394.293942
2	0.5	1354.903094	1426.541499
3	0.6	1354.429363	1448.950515
4	0.7	1359.069181	1462.628689
5	0.8	1369.416613	1472.051592
6	0.9	1385.765178	1480.804187

From the above we can see that alpha value 0.3 has the lowest test rmse values



RMSE Dataframe:

	Test RMSE
RegressionOnTime	1359.708262
SimpleAverageModel	1331.037637
2pointTrailingMovingAverage	805.879970
4pointTrailingMovingAverage	1161.282792
6pointTrailingMovingAverage	1290.624831
9pointTrailingMovingAverage	1375.582351
Alpha=0.995,SimpleExponentialSmoothing	1329.835548
Alpha=0.3,SimpleExponentialSmoothing	1359.784870

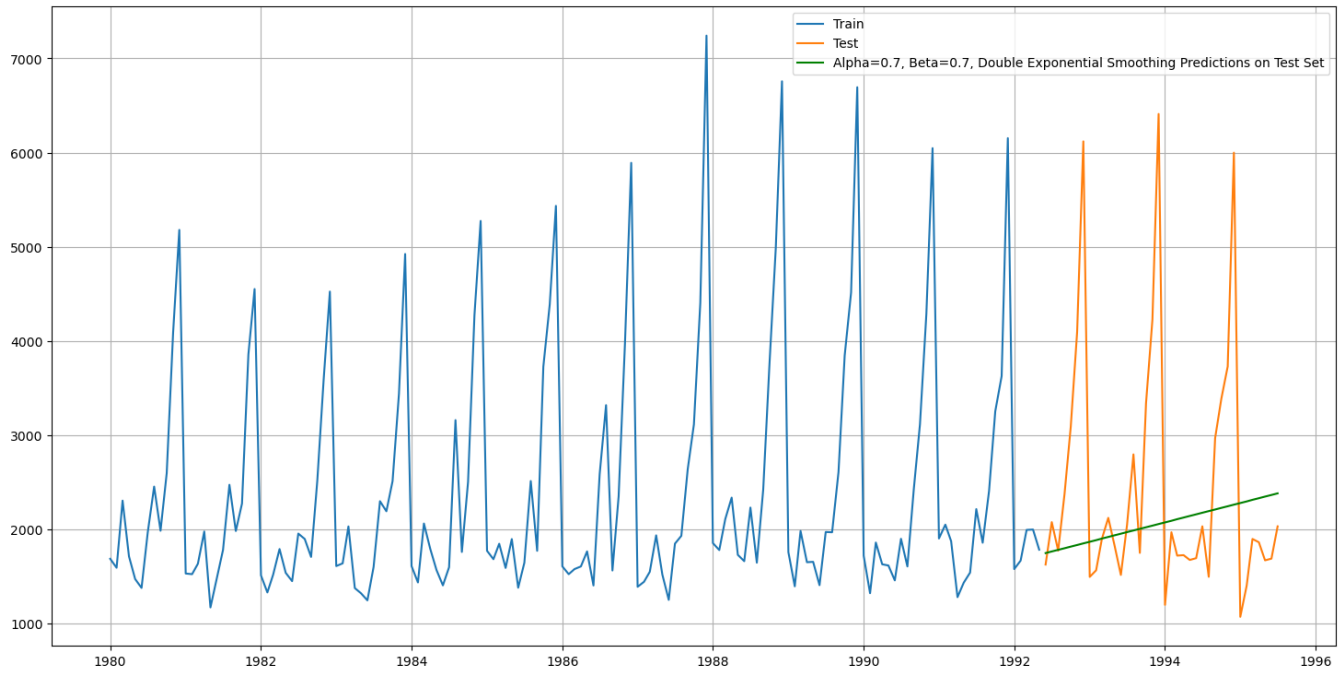
Problem 1.3.5 Build forecasting models – Double Exponential Smoothing:

Double Exponential Smoothing incorporates a trend component.

Alpha Values	Beta Values	Train RMSE	Test RMSE
36	0.7	0.7	1714.323131	1413.512841
45	0.8	0.8	1811.989180	1437.562918
46	0.8	0.9	1869.955439	1512.844361
22	0.5	0.9	1735.895456	1631.867102

44 0.8 0.7 1755.839715 1690.354869

Above data is sort for the lowest test rmse value:



RSME Dataframe:

	Test RMSE
RegressionOnTime	1359.708262
SimpleAverageModel	1331.037637
2pointTrailingMovingAverage	805.879970
4pointTrailingMovingAverage	1161.282792
6pointTrailingMovingAverage	1290.624831
9pointTrailingMovingAverage	1375.582351
Alpha=0.995,SimpleExponentialSmoothing	1329.835548
Alpha=0.3,SimpleExponentialSmoothing	1359.784870
Alpha=0.7,Beta=0.7,DoubleExponentialSmoothing	1413.512841

Problem 1.3.6 Build forecasting models – Triple Exponential Smoothing:

Triple Exponential Smoothing (Holt-Winters) adds a seasonality component.

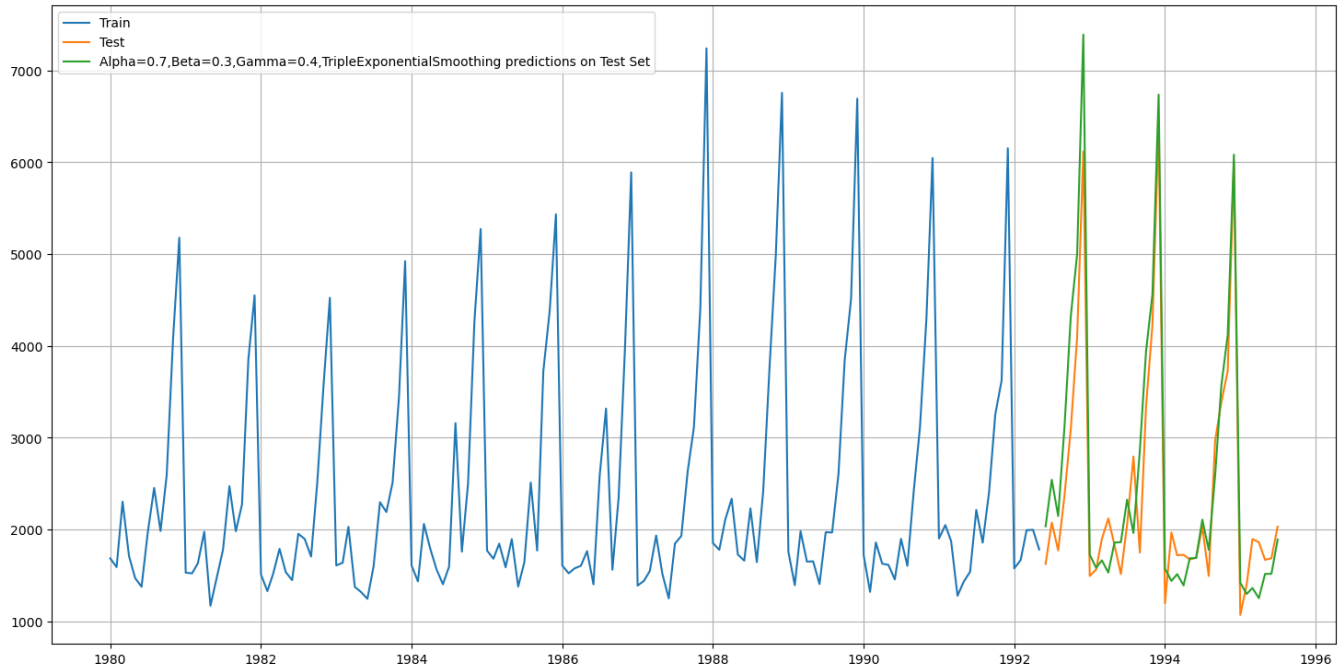
	Alpha Values	Beta Values	Gamma Values	Train RMSE	Test RMSE
257	0.7	0.3	0.4	528.822969	514.815688
200	0.6	0.4	0.3	492.947792	627.171672
96	0.4	0.7	0.3	500.669980	688.536684
144	0.5	0.5	0.3	488.750680	717.056843
256	0.7	0.3	0.3	492.626330	733.071728

Above is sorted as per the lowest test rmse value.

RMSE Dataframe:

	Test RMSE
RegressionOnTime	1359.708262
SimpleAverageModel	1331.037637
2pointTrailingMovingAverage	805.879970
4pointTrailingMovingAverage	1161.282792

6pointTrailingMovingAverage 1290.624831
9pointTrailingMovingAverage 1375.582351
Alpha=0.995,SimpleExponentialSmoothing 1329.835548
Alpha=0.3,SimpleExponentialSmoothing 1359.784870
Alpha=0.7,Beta=0.7,DoubleExponentialSmoothing 1413.512841
Alpha=0.676,Beta=0.088,Gamma=0.323,TripleExponentialSmoothing 318.532779
Alpha=1.0,Beta=0.7,Gamma=0.3,TripleExponentialSmoothing 514.815688



Problem 1.3.7 Check the performance of the models built:

We Compare the performance of model using the rmse value:

Sorted by RMSE values on the Test Data:

Test RMSE

Alpha=0.676,Beta=0.088,Gamma=0.323,TripleExponentialSmoothing 318.532779

Alpha=1.0,Beta=0.7,Gamma=0.3,TripleExponentialSmoothing 514.815688

2pointTrailingMovingAverage 805.879970

4pointTrailingMovingAverage 1161.282792

6pointTrailingMovingAverage 1290.624831

Alpha=0.995,SimpleExponentialSmoothing 1329.835548

SimpleAverageModel 1331.037637

RegressionOnTime 1359.708262

Alpha=0.3,SimpleExponentialSmoothing 1359.784870

9pointTrailingMovingAverage 1375.582351

Alpha=0.7,Beta=0.7,DoubleExponentialSmoothing 1413.512841

Problem 1.3.8 Insights on models:

Best Performing Model:

- Triple Exponential Smoothtng (Alpha=0.676, Beta=0.088, Gamma=0.323) has the lowest RMSE of 318.53. This indicates that this model provides the most accurate forecasts among those listed, effectively capturing trends and seasonality in the data.

Moderate Performance:

- The second-best model is also a Triple Exponential Smoothing model with parameters (Alpha=1.0, Beta=0.7, Gamma=0.3), which has an RMSE of 514.82. While it performs well, it is significantly less accurate than the best-performing model, suggesting that the specific parameter choices greatly influence its performance.

Moving Average Models:

- The 2-point Trailing Moving Average model has an RMSE of 805.88, indicating a moderate level of accuracy but still far from the top models. As the window size increases (4-point, 6-point, and 9-point), the RMSE values also increase significantly (1161.28, 1290.62, and 1375.58 respectively). This suggests that larger moving averages may introduce more lag and reduce responsiveness to recent changes in the data.

Simple Exponential Smoothing Models:

- The Simple Exponential Smoothing models show higher RMSE values, with parameters (Alpha=0.995) yielding 1329.84, and (Alpha=0.3) yielding 1359.78. This indicates that while Simple Exponential Smoothing can be effective in certain contexts, it may not capture trends and seasonality as effectively as more complex models like Triple Exponential Smoothing.

Regression on Time:

- The Regression on Time model has an RMSE of 1359.71, which is comparable to the Simple Exponential Smoothing models but does not outperform them. This suggests that while regression can be useful for trend analysis, it may not adequately capture seasonal patterns present in the data.

Overall Performance Trends:

- The trend indicates that more sophisticated models (like Triple Exponential Smoothing) generally outperform simpler models (like Simple Average or Simple Exponential Smoothing).
- Models that incorporate seasonality and trend components tend to yield lower RMSE values compared to those that do not.

PROBLEM 1.4 CHECK FOR STATIONARITY:

Problem 1.4.1 Stationarity Check:

To check for stationarity in the time series, we can use the Augmented Dickey-Fuller (ADF) test.

During the ADF test the p-value for the original dataset was:

`p-value - 0.628598166831955.`

As p-value is not less than 0.05 indicates that our series was not stationary.

Problem 1.4.2 Make the data stationary (if needed):

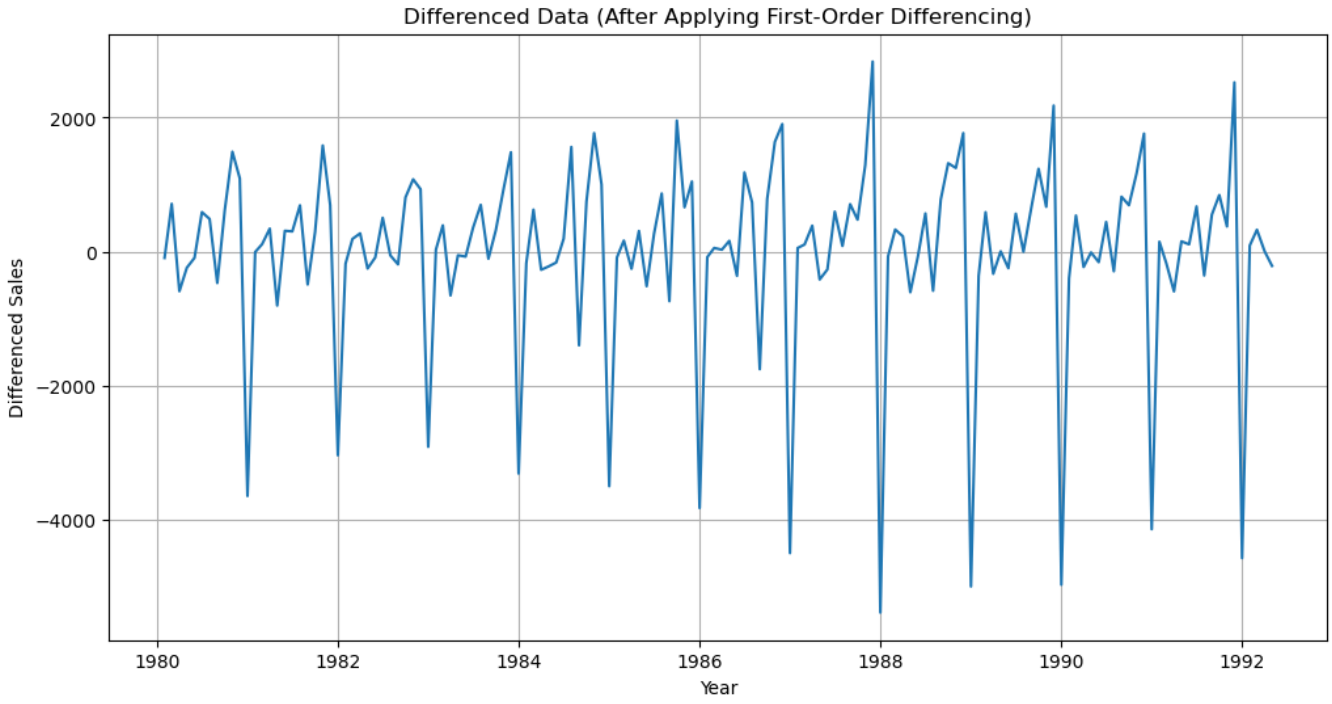
As our series was not stationary, we have to make it stationary using the Differencing method. Differencing is a common way to remove trends from a time series. We can apply first-order differencing to make the data stationary.

The p-value for the differenced dataset after apply differencing method was:

`p-value - 3.3653712299753526e-14.`

As p-value is less than 0.05 indicates that our series was stationary now.

Timeseries plot for the difference data (stationary data).

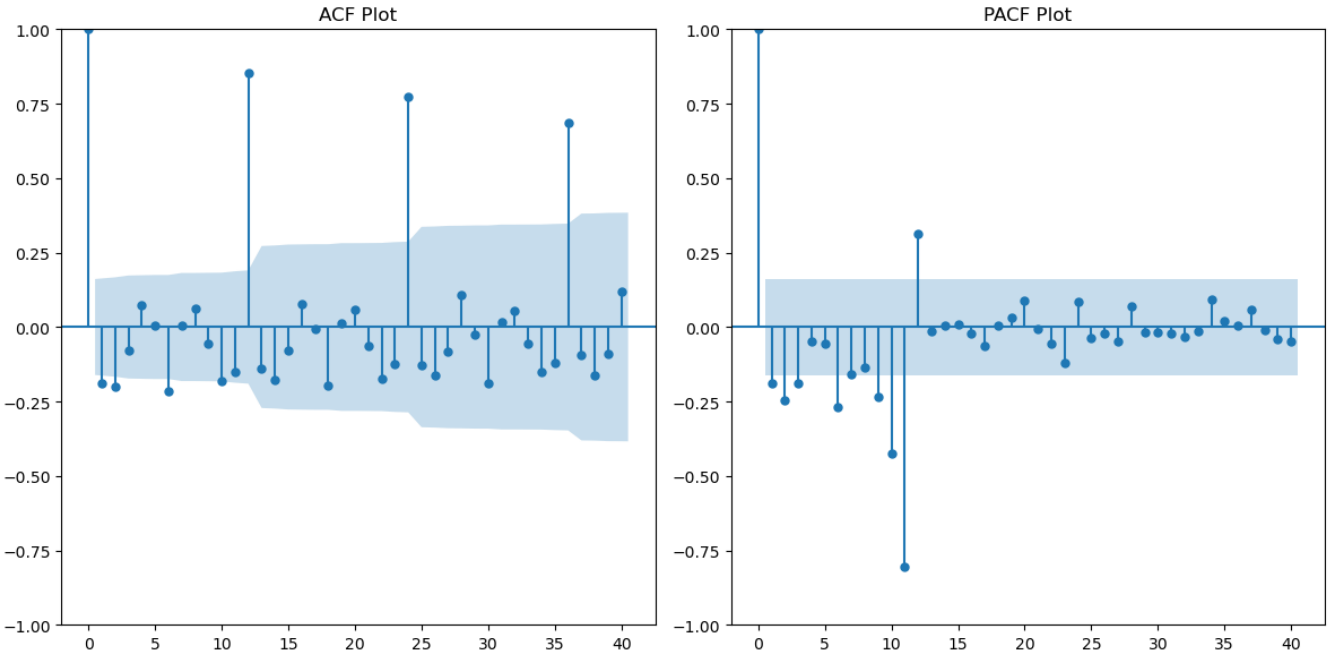


PROBLEM 1.5 MODEL BUILDING - STATIONARY DATA:

Problem 1.5.1 Generate ACF & PACF Plot and find the AR, MA values:

We'll first generate ACF (Autocorrelation Function) and PACF (Partial Autocorrelation Function) plots to identify the appropriate AR (Auto-Regressive) and MA (Moving Average) orders for the model.

To plot a ACF and PACF we are going to import the plot_acf & plot_pacf from the statsmodels.graphics.tsaplots library.



Insights:

Seasonality Detection

In the ACF plot, you can see significant spikes at regular intervals (e.g., every 12 lags), suggesting a seasonal pattern.

AR and MA Values:

AR (p) value: The PACF shows a significant spike at lag 1, suggesting an AR(1) component.

MA (q) value: The ACF shows a significant spike at lag 1, suggesting an MA(1) component.

Problem 1.5.2 Build different ARIMA models:

Problem 1.5.2.*1* Auto ARIMA:
Auto ARIMA automatically selects the best values for p, d, q (AR, differencing, and MA terms) by minimizing the AIC (Akaike Information Criterion).

We build the auto ARIMA model the does the stepwise search to minimize the AIC values and low AIC indicate towards the best model.

```
Performing stepwise search to minimize aic
 ARIMA(2,0,2)(0,0,0)[0]          : AIC=inf, Time=0.44 sec
 ARIMA(0,0,0)(0,0,0)[0]          : AIC=2780.518, Time=0.02 sec
 ARIMA(1,0,0)(0,0,0)[0]          : AIC=2578.818, Time=0.05 sec
 ARIMA(0,0,1)(0,0,0)[0]          : AIC=2677.322, Time=0.08 sec
 ARIMA(2,0,0)(0,0,0)[0]          : AIC=2578.299, Time=0.05 sec
 ARIMA(3,0,0)(0,0,0)[0]          : AIC=2574.041, Time=0.08 sec
```

```
ARIMA(4,0,0)(0,0,0)[0]          : AIC=2572.184, Time=0.08 sec
ARIMA(5,0,0)(0,0,0)[0]          : AIC=2574.112, Time=0.08 sec
ARIMA(4,0,1)(0,0,0)[0]          : AIC=2548.464, Time=0.38 sec
ARIMA(3,0,1)(0,0,0)[0]          : AIC=inf, Time=0.37 sec
ARIMA(5,0,1)(0,0,0)[0]          : AIC=2575.880, Time=0.20 sec
ARIMA(4,0,2)(0,0,0)[0]          : AIC=inf, Time=0.50 sec
ARIMA(3,0,2)(0,0,0)[0]          : AIC=inf, Time=0.33 sec
ARIMA(5,0,2)(0,0,0)[0]          : AIC=inf, Time=0.68 sec
ARIMA(4,0,1)(0,0,0)[0] intercept  : AIC=2537.225, Time=0.19 sec
ARIMA(3,0,1)(0,0,0)[0] intercept  : AIC=2537.499, Time=0.14 sec
ARIMA(4,0,0)(0,0,0)[0] intercept  : AIC=2536.326, Time=0.12 sec
ARIMA(3,0,0)(0,0,0)[0] intercept  : AIC=2535.510, Time=0.07 sec
ARIMA(2,0,0)(0,0,0)[0] intercept  : AIC=2534.263, Time=0.09 sec
ARIMA(1,0,0)(0,0,0)[0] intercept  : AIC=2536.204, Time=0.05 sec
ARIMA(2,0,1)(0,0,0)[0] intercept  : AIC=2536.508, Time=0.12 sec
ARIMA(1,0,1)(0,0,0)[0] intercept  : AIC=2535.730, Time=0.07 sec

Best model:  ARIMA(2,0,0)(0,0,0)[0] intercept
Total fit time: 4.242 seconds
Auto ARIMA RMSE: 1328.8422197840096
```

Evaluation of the Auto ARIMA model is also done using the RMSE values for the above model RMSE value was:

```
Auto ARIMA RMSE: 1328.8422197840096
```

Problem 1.5.2.*2* Manual ARIMA:

Based on the ACF and PACF plots, we can manually specify the ARIMA model parameters and fit the model. From the ACF and PACF we are going to set the p=2, q=2 and d=1.

Evaluation of the Manual ARIMA model is also done using the RMSE values for the above model RMSE value was:

```
Manual ARIMA RMSE: 1326.541693141659
```

Problem 1.5.3 Build different SARIMA models:

Problem 1.5.3.*1* Auto SARIMA:

We can build a SARIMA model that accounts for both seasonal and non-seasonal components.

For SARIMA, we account for seasonality by adding seasonal parameters (P, D, Q, s).

We build the auto SARIMA model the does the stepwise search to minimize the AIC values and low AIC indicate towards the best model.

```
Performing stepwise search to minimize aic
ARIMA(2,0,2)(1,1,1)[12] intercept  : AIC=2028.955, Time=2.36 sec
ARIMA(0,0,0)(0,1,0)[12] intercept  : AIC=2052.130, Time=0.04 sec
ARIMA(1,0,0)(1,1,0)[12] intercept  : AIC=2031.254, Time=0.41 sec
ARIMA(0,0,1)(0,1,1)[12] intercept  : AIC=2023.216, Time=0.46 sec
ARIMA(0,0,0)(0,1,0)[12]          : AIC=2050.241, Time=0.05 sec
ARIMA(0,0,1)(0,1,0)[12] intercept  : AIC=2046.569, Time=0.08 sec
ARIMA(0,0,1)(1,1,1)[12] intercept  : AIC=2024.299, Time=1.20 sec
```

ARIMA(0,0,1)(0,1,2)[12] intercept : AIC=2024.372, Time=1.12 sec
ARIMA(0,0,1)(1,1,0)[12] intercept : AIC=2030.648, Time=0.34 sec
ARIMA(0,0,1)(1,1,2)[12] intercept : AIC=2026.266, Time=2.36 sec
ARIMA(0,0,0)(0,1,1)[12] intercept : AIC=2027.137, Time=0.50 sec
ARIMA(1,0,1)(0,1,1)[12] intercept : AIC=2024.489, Time=1.22 sec
ARIMA(0,0,2)(0,1,1)[12] intercept : AIC=2025.101, Time=0.69 sec
ARIMA(1,0,0)(0,1,1)[12] intercept : AIC=2023.661, Time=0.70 sec
ARIMA(1,0,2)(0,1,1)[12] intercept : AIC=2027.132, Time=1.37 sec
ARIMA(0,0,1)(0,1,1)[12] : AIC=2023.251, Time=0.45 sec

Best model: ARIMA(0,0,1)(0,1,1)[12] intercept
Total fit time: 13.396 seconds
Auto SARIMA RMSE: 320.324912883244

Evaluation of the Auto SARIMA model is also done using the RMSE values for the above model RMSE value was:

Auto SARIMA RMSE: 320.324912883244

Problem 1.5.3.*2* Manual SARIMA:

In this we manually specify the SARIMA model's seasonal parameters, we can do so using the ExponentialSmoothing or ARIMA model from the statsmodels library.

Evaluation of the Manual SARIMA model is also done using the RMSE values for the above model RMSE value was:

Manual SARIMA RMSE: 312.76737287623274

Problem 1.5.4 Check the performance of the models built:

Now, we can compare the performance of the models based on the RMSE values calculated in the previous steps. The model with the lowest RMSE is considered the best fit for the data.

RMSE values of model:

Auto ARIMA RMSE: 1328.8422197840096
Manual ARIMA RMSE: 1326.541693141659
Auto SARIMA RMSE: 320.324912883244
Manual SARIMA RMSE: 312.76737287623274

PROBLEM 1.6 COMPARE THE PERFORMANCE OF THE MODELS:

Problem 1.6.1 Compare the performance of all the models built:

Now let compare the RMSE values of all the models:

Manual SARIMA RMSE	312.76737287623274
Alpha=0.676,Beta=0.088,Gamma=0.323,TripleExponentialSmoothing	318.532779
Auto SARIMA RMSE: 320.324912883244	320.324912883244
Alpha=1.0,Beta=0.7,Gamma=0.3,TripleExponentialSmoothing	514.815688
2pointTrailingMovingAverage	805.879970
4pointTrailingMovingAverage	1161.282792
6pointTrailingMovingAverage	1290.624831

Manual ARIMA RMSE	1326.541693141659
Auto ARIMA RMSE	1328.8422197840096
Alpha=0.995,SimpleExponentialSmoothing	1329.835548
SimpleAverageModel	1331.037637
RegressionOnTime	1359.708262
Alpha=0.3,SimpleExponentialSmoothing	1359.784870
9pointTrailingMovingAverage	1375.582351
Alpha=0.7,Beta=0.7,DoubleExponentialSmoothing	1413.512841

Problem 1.6.2 Choose the best model with proper rationale:

The model with the lowest RMSE is considered the best fit for the data. So,

Best Model: Manual SARIMA RMSE with RMSE: 312.76737287623274

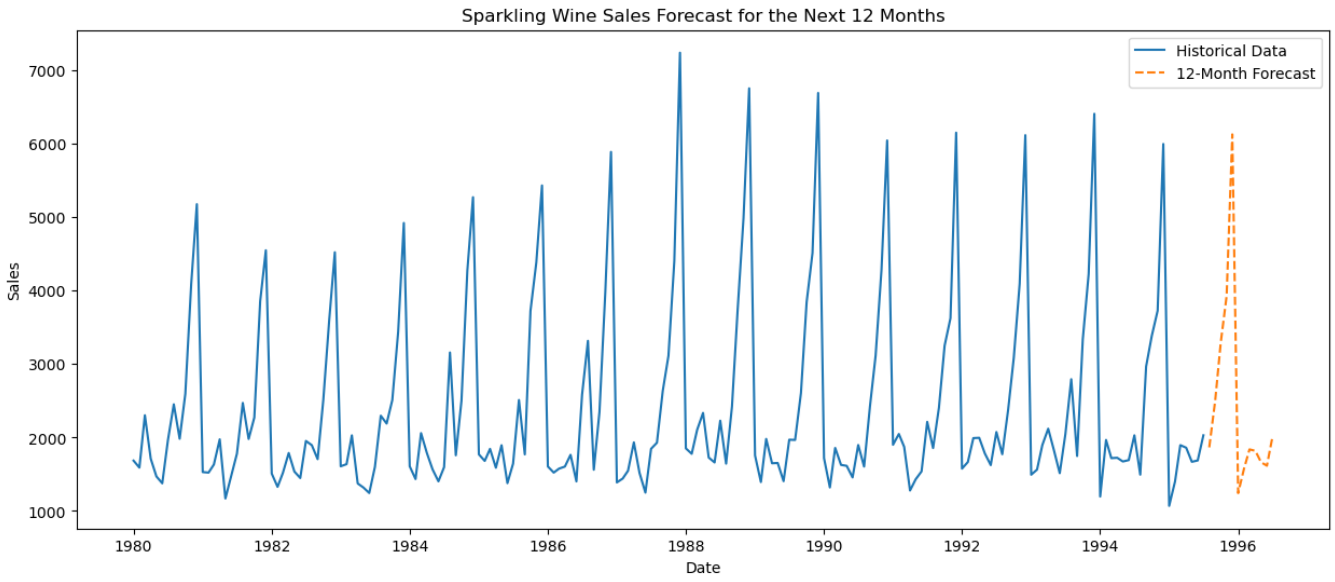
Problem 1.6.3 Rebuild the best model using the entire data:

We rebuild the best-performing model using the entire dataset.

Problem 1.6.4 Make a forecast for the next 12 months:

Forecast for the next 12 months.

Graphic forecast for next 12 months:



Numerical forecast for next 12 months:

Forecast for the next 12 months:

1995-08-01	1866.405603
1995-09-01	2492.673153
1995-10-01	3297.095457
1995-11-01	3929.714437
1995-12-01	6129.903208
1996-01-01	1243.835102
1996-02-01	1576.836427
1996-03-01	1839.227873
1996-04-01	1819.403970

1996-05-01 1664.272186
1996-06-01 1616.792081
1996-07-01 2017.439099
Freq: MS, Name: predicted_mean, dtype: float64

PROBLEM 1.7 ACTIONABLE INSIGHTS & RECOMMENDATIONS:

Problem 1.7.1 Conclude with the key takeaways (actionable insights and recommendations) for the business:

Seasonal Demand Patterns:

The sales data for sparkling wine shows clear seasonal peaks, particularly around certain periods each year. This suggests that demand is higher during specific months, likely due to holidays or celebrations.

Consistent Growth Trend:

Over the years, there is an upward trend in sales, indicating a growing market for sparkling wine. This suggests a positive long-term outlook for the product.

Predictive Accuracy:

The model used for forecasting captures both the seasonality and trend effectively, as demonstrated by the accurate 12-month forecast aligned with historical patterns.

Model Choice:

SARIMA Model:

This model was chosen because it effectively captures both the trend and seasonality present in the data. It uses seasonal differencing to handle the periodic peaks and troughs, making it well-suited for forecasting sales that fluctuate seasonally.

Business Benefits:

Inventory Management:

By understanding the seasonal demand, the company can optimize inventory levels, ensuring sufficient stock during peak periods while reducing excess inventory during off-peak times.

Marketing Strategies:

The forecast allows for targeted marketing campaigns during high-demand periods, potentially increasing sales and market share.

Resource Allocation:

With accurate sales predictions, resources such as labor and logistics can be better allocated to meet demand efficiently, reducing costs and improving service levels.

Strategic Planning:

The insights from the forecast can inform long-term strategic decisions, such as expanding production capacity or entering new markets during growth phases.

By adopting this model, the company can enhance its decision-making process, align operations with market demand, and capitalize on growth opportunities in the sparkling wine market