TIME SERIES FORECASTING

BUSINESS REPORT

BY

G S JAIGURURAM

TABLE OF CONTENTS

Problem Statement: 3
Context: 3
Objective: 3
Problem 1.1 Define the problem and perform Exploratory Data Analysis: 3
Problem 1.1.1 Read the data as an appropriate time series data: 3
Problem 1.1.2 Plot the Data: 3
Problem 1.1.3 Perform EDA: 4
Problem 1.1.4 Perform Decomposition: 4
Problem 1.2 Data Pre-processing: 5
Problem 1.2.1 Missing value treatment: 5
Problem 1.2.2 Visualize the processed data - Train-test split: 5
Problem 1.3 Model Building - Original Data: 5
Problem 1.3.1 Build forecasting models - Linear regression: 5
Problem 1.3.2 Build forecasting models – Simple Average: 6
Problem 1.3.3 Build forecasting models – Moving Average: 7
Problem 1.3.4 Build forecasting models – Single Exponential Smoothing: 8
Problem 1.3.5 Build forecasting models – Double Exponential Smoothing: 9
Problem 1.3.6 Build forecasting models – Triple Exponential Smoothing: 10
Problem 1.3.7 Check the performance of the models built: 11
Problem 1.3.8 Insights on models: 12
Problem 1.4 Check for Stationarity: 13
Problem 1.4.1 Stationarity Check: 13
Problem 1.4.2 Make the data stationary (if needed): 13
Problem 1.5 Model Building - Stationary Data: 13
Problem 1.5.1 Generate ACF & PACF Plot and find the AR, MA values: 13
Problem 1.5.2 Build different ARIMA models: 14
Problem 1.5.3 Build different SARIMA models: 15
Problem 1.5.4 Check the performance of the models built: 18
Problem 1.6 Compare the performance of the models: 18
Problem 1.6.1 Compare the performance of all the models built: 18
Problem 1.6.2 Choose the best model with proper rationale: 19
Problem 1.6.3 Rebuild the best model using the entire data: 19

Problem 1.6.4 Make a forecast for the next 12 months: 19
Problem 1.7 Actionable Insights & Recommendations: 20
Problem 1.7.1 Conclude with the key takeaways (actionable insights and recommendations) for the business: 20

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:

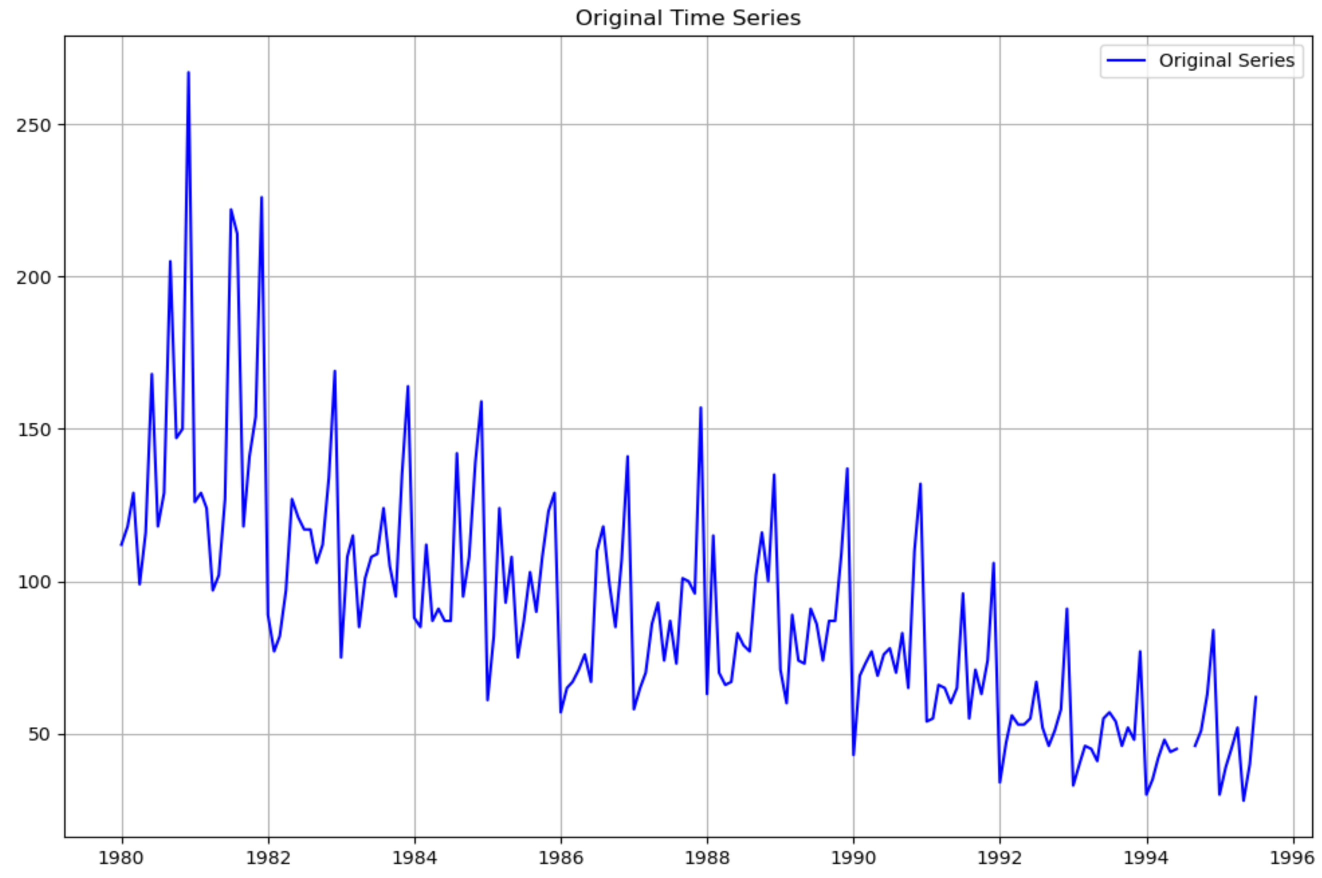
The primary objective of this project is to analyze and forecast wine sales trends for the 20th century based on historical data provided by ABC Estate Wines. We aim to equip ABC Estate Wines with the necessary insights and foresight to enhance sales performance, capitalize on emerging market opportunities, 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 Rose. The YearMonth column represents the time period in a "YYYY-MM" format, and the Rose 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: 90.

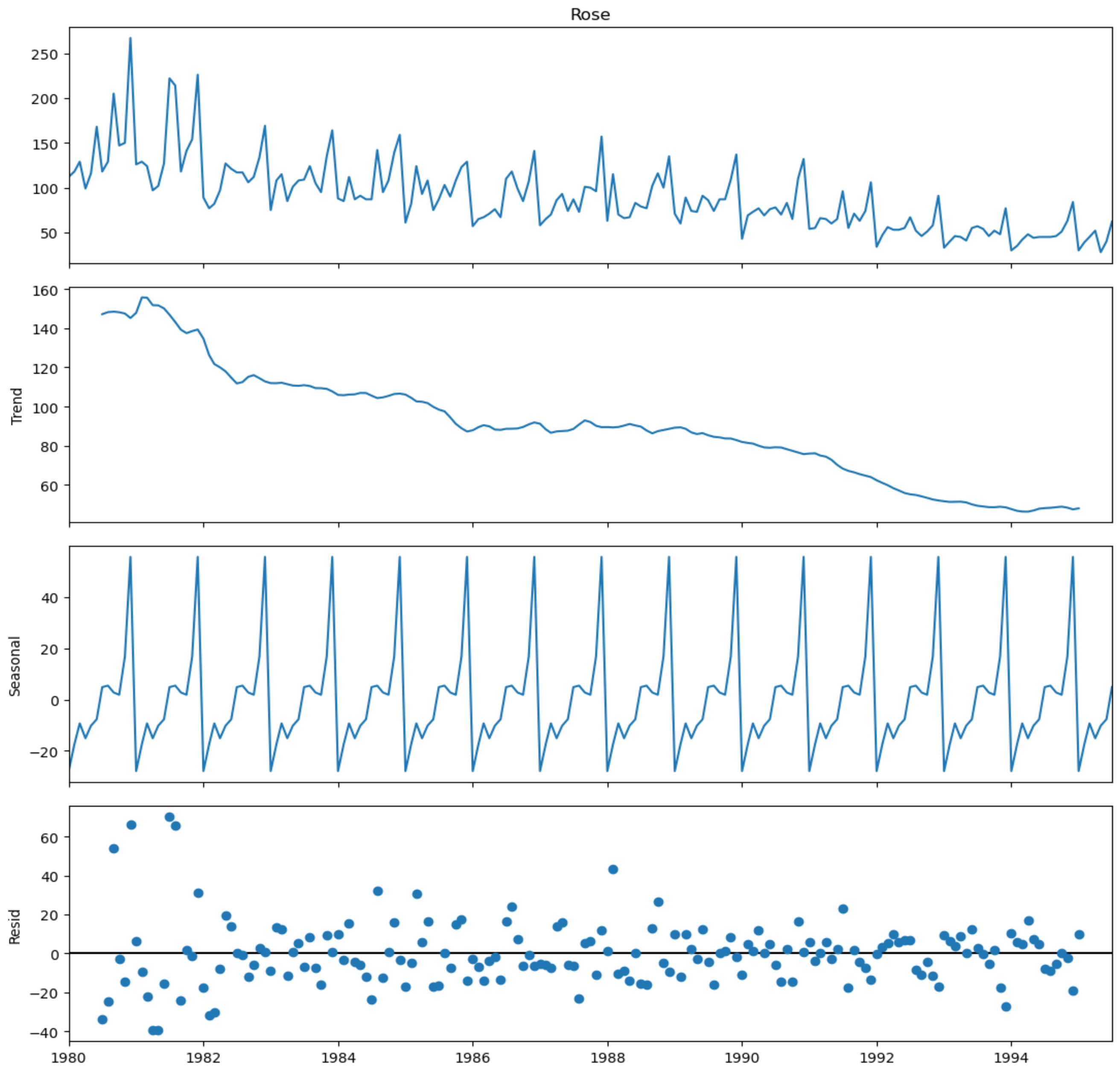
Minimum sales: 28 & Maximum sales: 267.

Standard deviation: 39, Indicating some variability in the sales data.

Problem 1.1.4 Perform Decomposition:

Decomposition: The plot displays the decomposed components:

- Trend: Shows a noticeable downward movement over time
- Seasonality: Exhibits periodic fluctuations, likely linked to seasonal variations in wine demand.
- Residual: Represents the remaining noise or irregular patterns after removing the trend and seasonality.



PROBLEM 1.2 DATA PRE-PROCESSING:

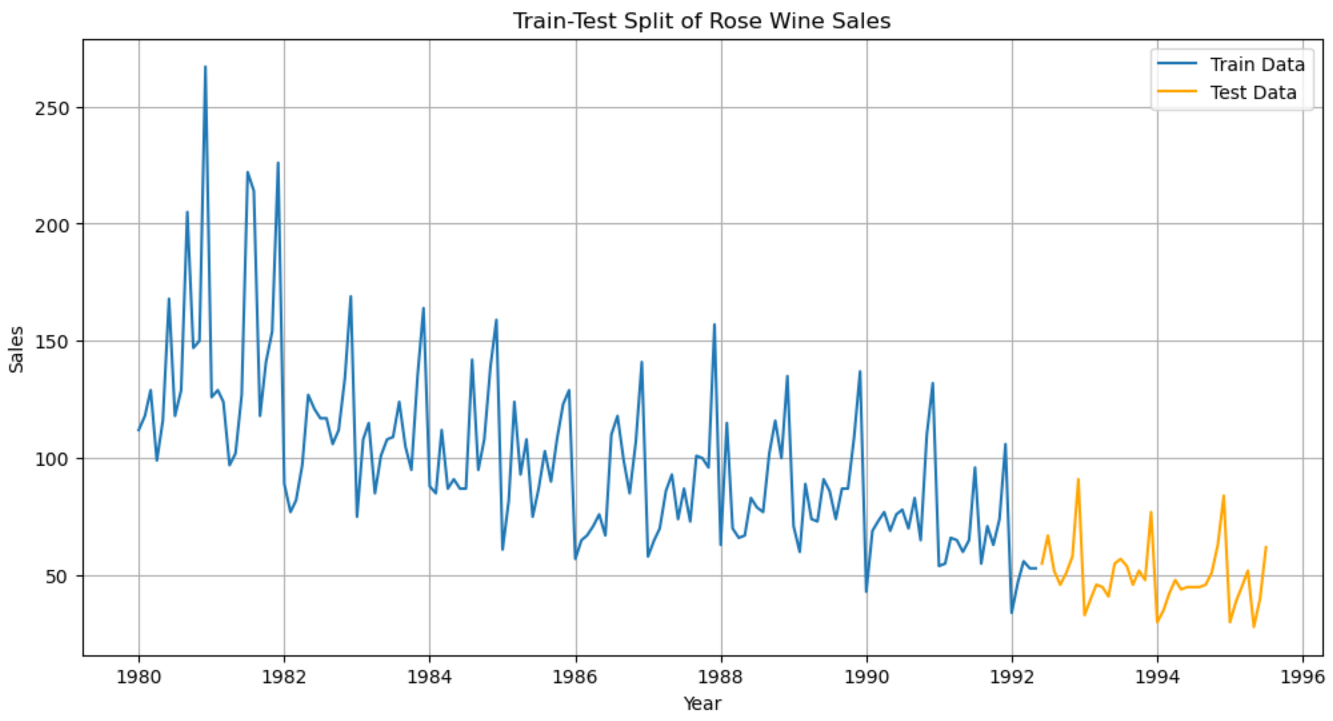
Problem 1.2.1 Missing value treatment:

We encounter missing value error while doing the decomposing the data and we treated it with the forward fill method.

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

Since the missing values are treated, 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:

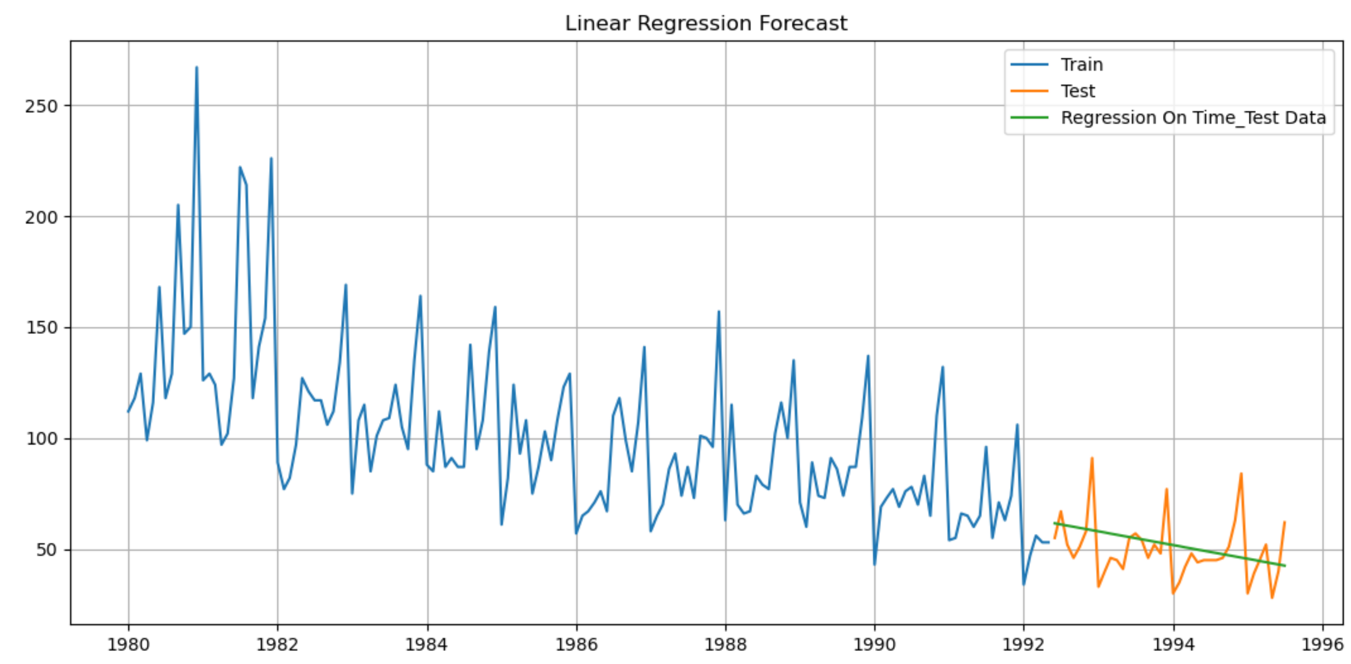
We will create a time instance to build a liner regression model. We take root mean square error value to evaluation of model performance.

Output:

```
For RegressionOnTime forecast on the Test Data,  RMSE is 13.74
```

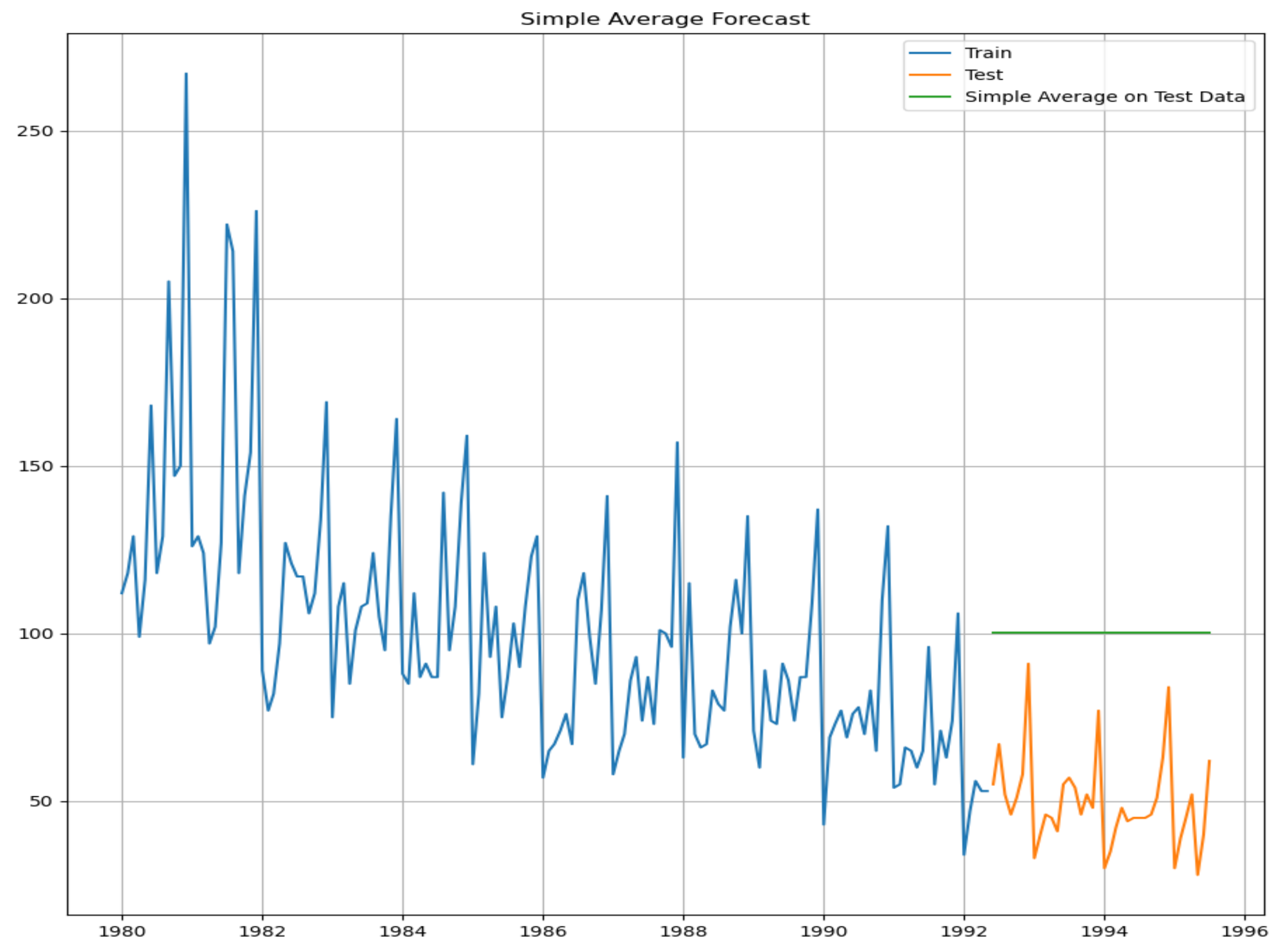
We store this RMSE Value in a new dataframe that will ease the comparison step.

```
          Test RMSE
RegressionOnTime      13.736625
```



Problem 1.3.2 Build forecasting models – Simple Average:

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



RMSE Value:

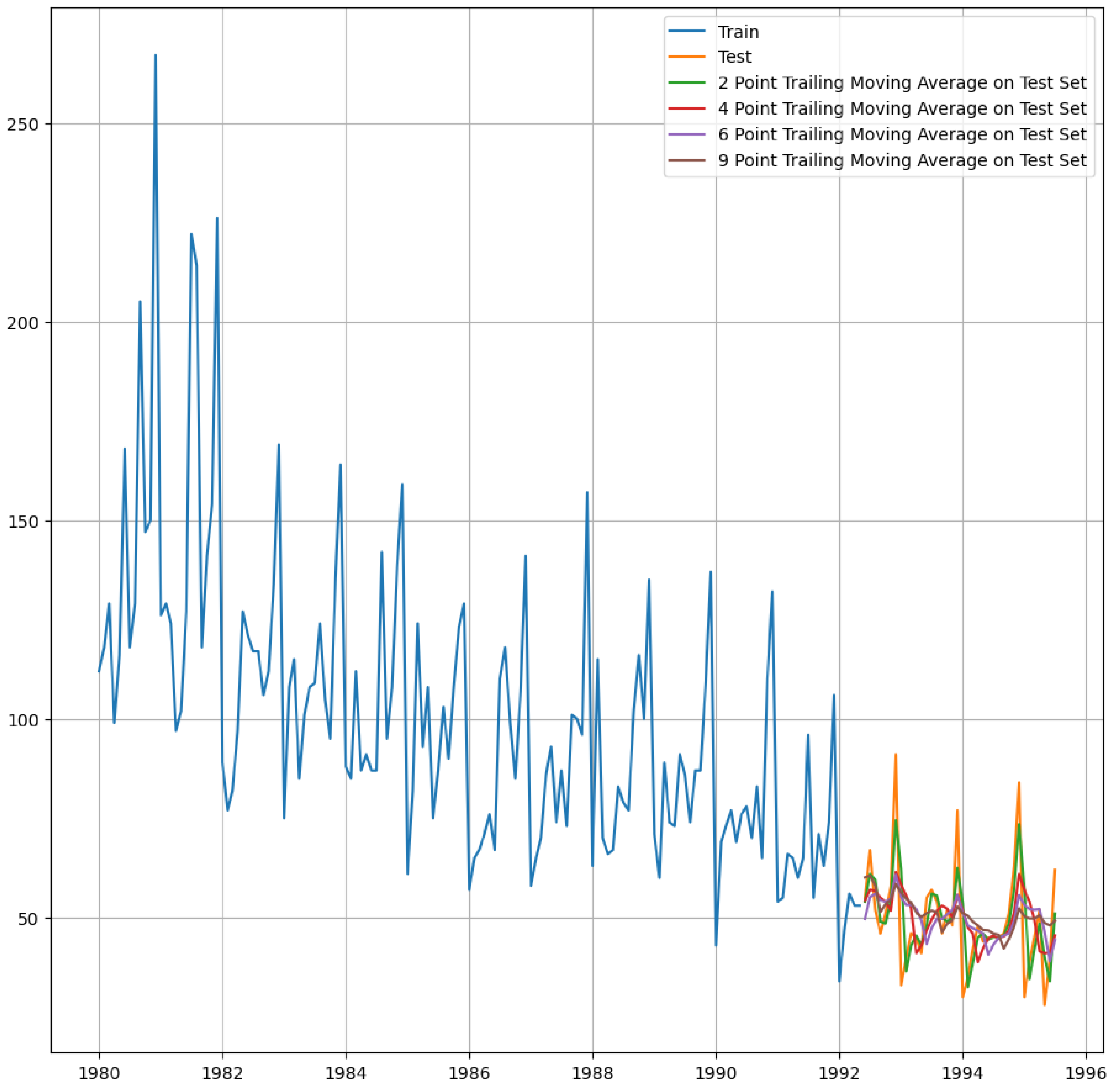
For Simple Average forecast on the Test Data, RMSE is 52.239

RMSE Dataframe:

	Test RMSE
RegressionOnTime	13.736625
SimpleAverageModel	52.239499

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:

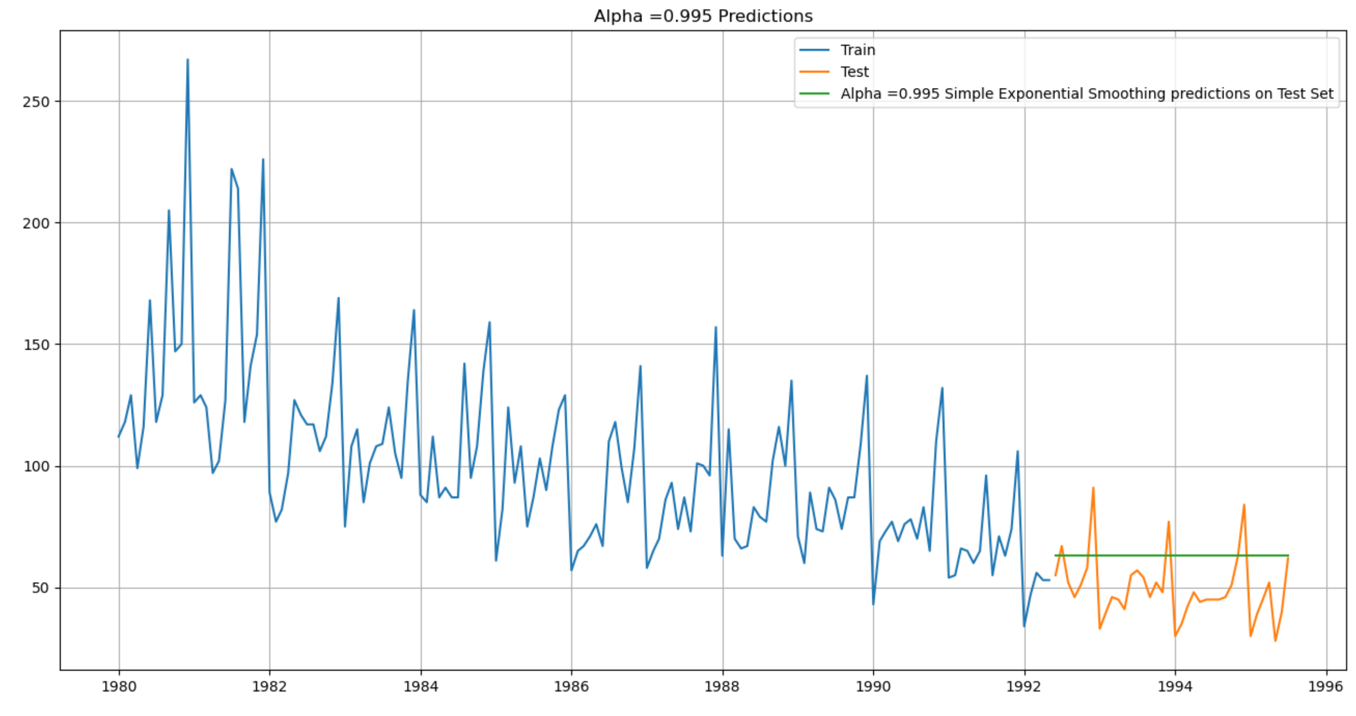
For 2 point Moving Average Model forecast on the Training Data, RMSE is 9.402
For 4 point Moving Average Model forecast on the Training Data, RMSE is 12.187
For 6 point Moving Average Model forecast on the Training Data, RMSE is 12.413
For 9 point Moving Average Model forecast on the Training Data, RMSE is 12.621

RMSE Dataframe:

	Test RMSE
RegressionOnTime	13.736625
SimpleAverageModel	52.239499
2pointTrailingMovingAverage	9.402197
4pointTrailingMovingAverage	12.186935
6pointTrailingMovingAverage	12.413090
9pointTrailingMovingAverage	12.620569

Problem 1.3.4 Build forecasting models – Single Exponential Smoothing:

Single Exponential Smoothing gives more weight to recent data.



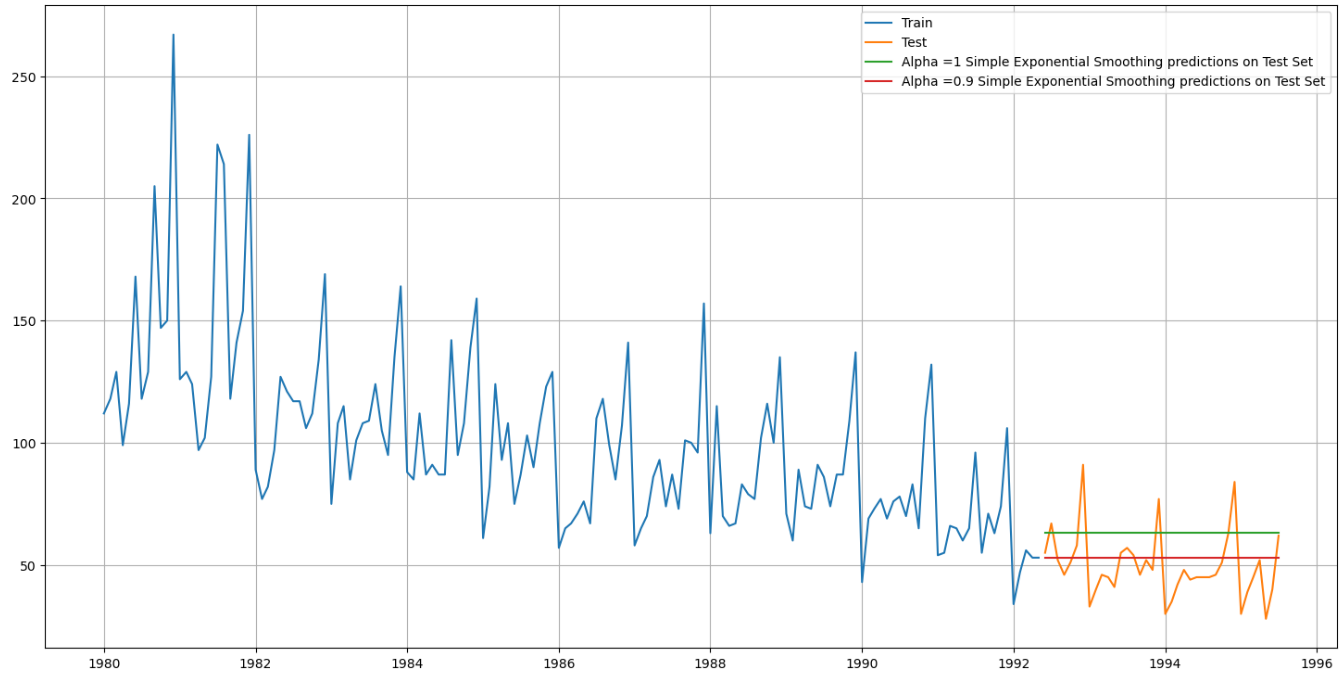
RMSE Value:

For Alpha =0.995 Single Exponential Smoothing Model forecast on the Test Data, RMSE is 19.021.

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

	Alpha Values	Train RMSE	Test RMSE
6	0.9	36.636990	13.838498
5	0.8	35.487361	13.845197
4	0.7	34.465075	13.850866
3	0.6	33.562525	13.873666
2	0.5	32.774372	13.962158
1	0.4	32.088907	14.220091
0	0.3	31.483475	14.878568

From the above we can see that alpha value 0.9 has the lowest test rmse values



RMSE Dataframe:

	Test RMSE
RegressionOnTime	13.736625
SimpleAverageModel	52.239499
2pointTrailingMovingAverage	9.402197
4pointTrailingMovingAverage	12.186935
6pointTrailingMovingAverage	12.413090
9pointTrailingMovingAverage	12.620569
Alpha=0.995,SimpleExponentialSmoothing	19.021186
Alpha=0.9,SimpleExponentialSmoothing	13.838498

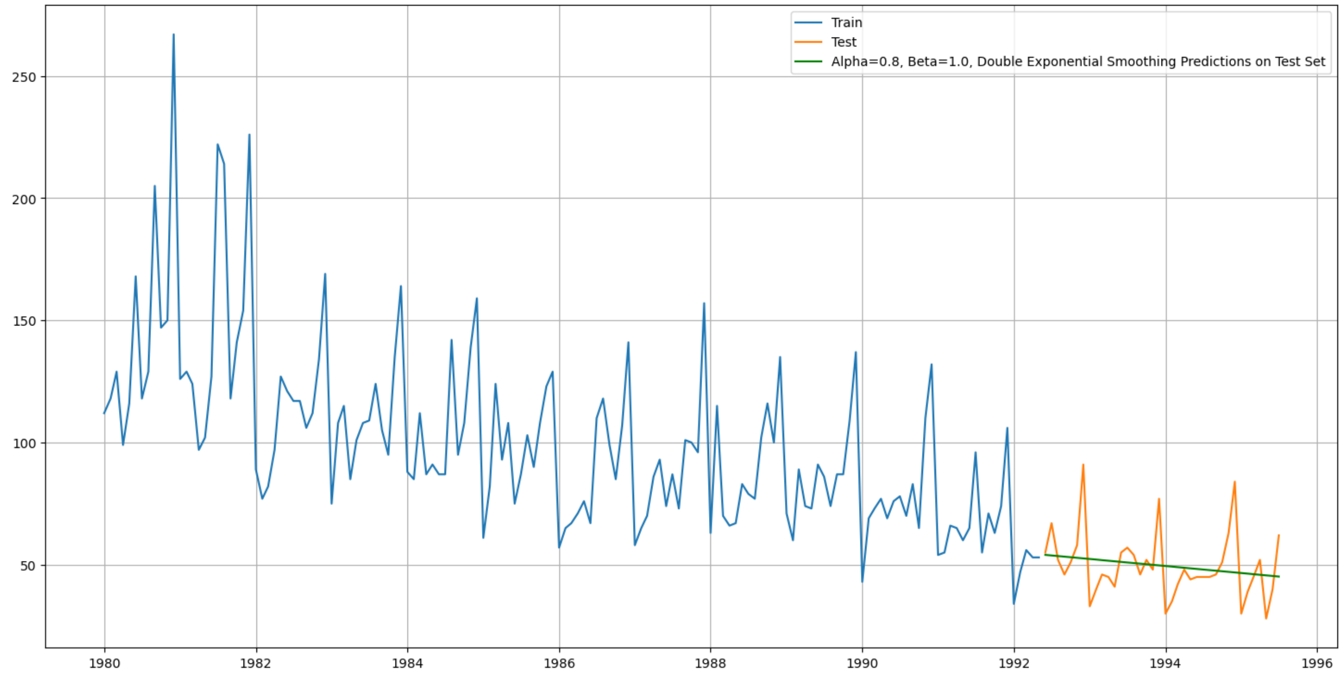
Problem 1.3.5 Build forecasting models – Double Exponential Smoothing:

Double Exponential Smoothing incorporates a trend component.

	Alpha Values	Beta Values	Train RMSE	Test RMSE
47	0.8	1.0	52.866144	13.180564
62	1.0	0.9	59.281740	13.195065
61	1.0	0.8	56.244829	13.195787
60	1.0	0.7	53.436790	13.202028

34	0.7	0.5	41.482288	13.232408

Above data is sort for the lowest test rmse value:



RSME Dataframe:

	Test RMSE
RegressionOnTime	13.736625
SimpleAverageModel	52.239499
2pointTrailingMovingAverage	9.402197
4pointTrailingMovingAverage	12.186935
6pointTrailingMovingAverage	12.413090
9pointTrailingMovingAverage	12.620569
Alpha=0.995,SimpleExponentialSmoothing	19.021186
Alpha=0.9,SimpleExponentialSmoothing	13.838498
Alpha=0.8,Beta=1.0,DoubleExponentialSmoothing	13.180564

Problem 1.3.6 Build forecasting models – Triple Exponential Smoothing:

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

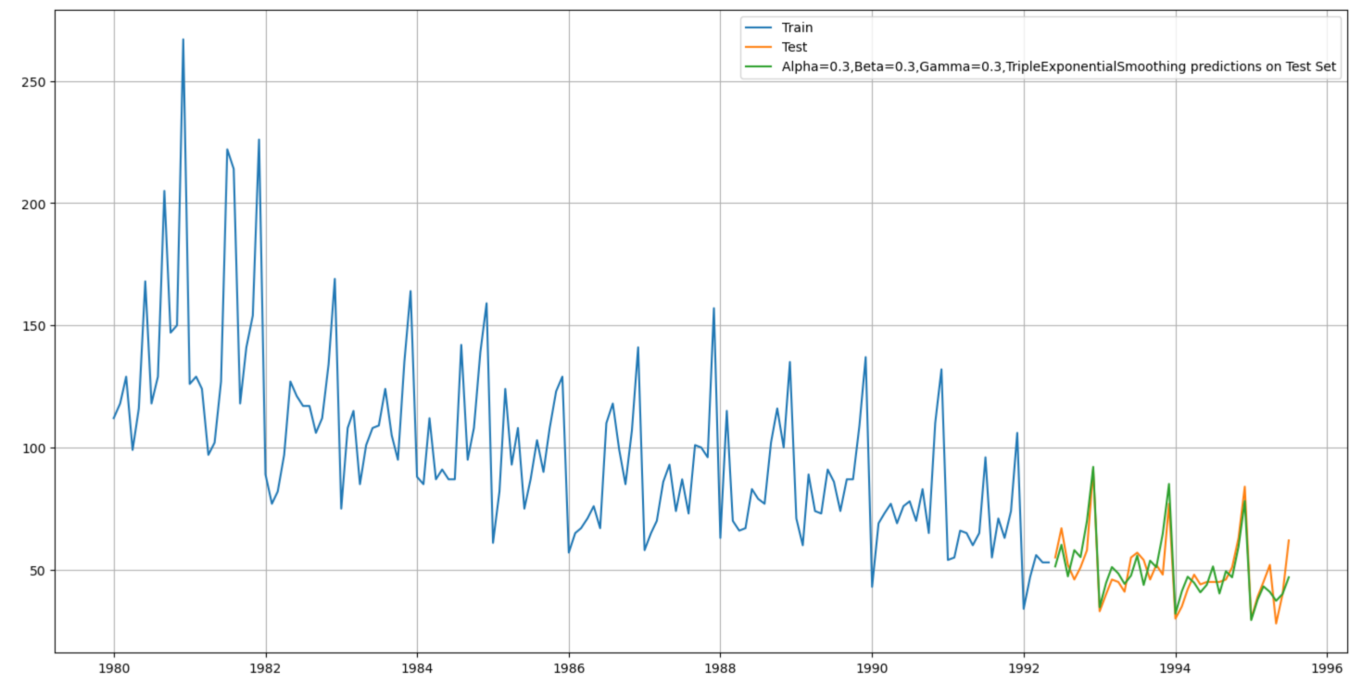
	Alpha Values	Beta Values	Gamma Values	Train RMSE	Test RMSE
0	0.3	0.3	0.3	22.302474	6.689740
1	0.3	0.3	0.4	23.489524	6.794501
2	0.3	0.3	0.5	25.066440	8.180679
74	0.4	0.4	0.5	27.573219	10.503173
195	0.6	0.3	0.6	32.788153	11.277663

Above is sorted as per the lowest test rmse value.

RMSE Dataframe:

	Test RMSE
RegressionOnTime	13.736625
SimpleAverageModel	52.239499
2pointTrailingMovingAverage	9.402197
4pointTrailingMovingAverage	12.186935

```
6pointTrailingMovingAverage      12.413090
9pointTrailingMovingAverage      12.620569
Alpha=0.995,SimpleExponentialSmoothing           19.021186
Alpha=0.9,SimpleExponentialSmoothing   13.838498
Alpha=0.8,Beta=1.0,DoubleExponentialSmoothing 13.180564
Alpha=0.1,Beta=0.0,Gamma=0.0,TripleExponentialSmoothing          8.406215
Alpha=0.3,Beta=0.3,Gamma=0.3,TripleExponentialSmoothing          6.689740
```



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=0.3,Beta=0.3,Gamma=0.3,TripleExponentialSmoothing	6.689740
Alpha=0.1,Beta=0.0,Gamma=0.0,TripleExponentialSmoothing	8.406215
2pointTrailingMovingAverage	9.402197
4pointTrailingMovingAverage	12.186935
6pointTrailingMovingAverage	12.413090
9pointTrailingMovingAverage	12.620569
Alpha=0.8,Beta=1.0,DoubleExponentialSmoothing	13.180564
RegressionOnTime	13.736625
Alpha=0.9,SimpleExponentialSmoothing	13.838498

```
Alpha=0.995,SimpleExponentialSmoothing   19.021186

SimpleAverageModel 52.239499
```

Problem 1.3.8 Insights on models:

Best Performing Model:

- The model with parameters Alpha=0.3, Beta=0.3, Gamma=0.3 using Triple Exponential Smoothing has the lowest RMSE of 6.69. This indicates that this model is the most accurate among those listed, effectively capturing trends and seasonality in the data.

Moderate Performance:

- The second-best model is another Triple Exponential Smoothing with parameters Alpha=0.1, Beta=0.0, Gamma=0.0, yielding an RMSE of 8.41. This model is less responsive to changes in level and trend but still performs well compared to simpler models.

Moving Average Models:

- The 2-point Trailing Moving Average has an RMSE of 9.40, indicating it performs reasonably well but is less accurate than the Triple Exponential Smoothing models. As the window size increases (4-point to 9-point), the RMSE values also increase slightly (from 12.19 to 12.62). This suggests that larger moving averages may introduce more lag and reduce responsiveness to recent changes.

Simple and Double Exponential Smoothing Models:

- The Double Exponential Smoothing model with parameters Alpha=0.8, Beta=1.0 has an RMSE of 13.18, indicating moderate accuracy but not as effective as the best Triple Exponential Smoothing model. The Simple Exponential Smoothing models show higher RMSE values (ranging from 13.84 to 19.02), suggesting they may not capture trends and seasonality as effectively as more complex models.

Regression on Time:

- The Regression on Time model has an RMSE of 13.74, which is comparable to the Simple Exponential Smoothing models but does not outperform them. This indicates 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 - 0.4573644979863173.`

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):

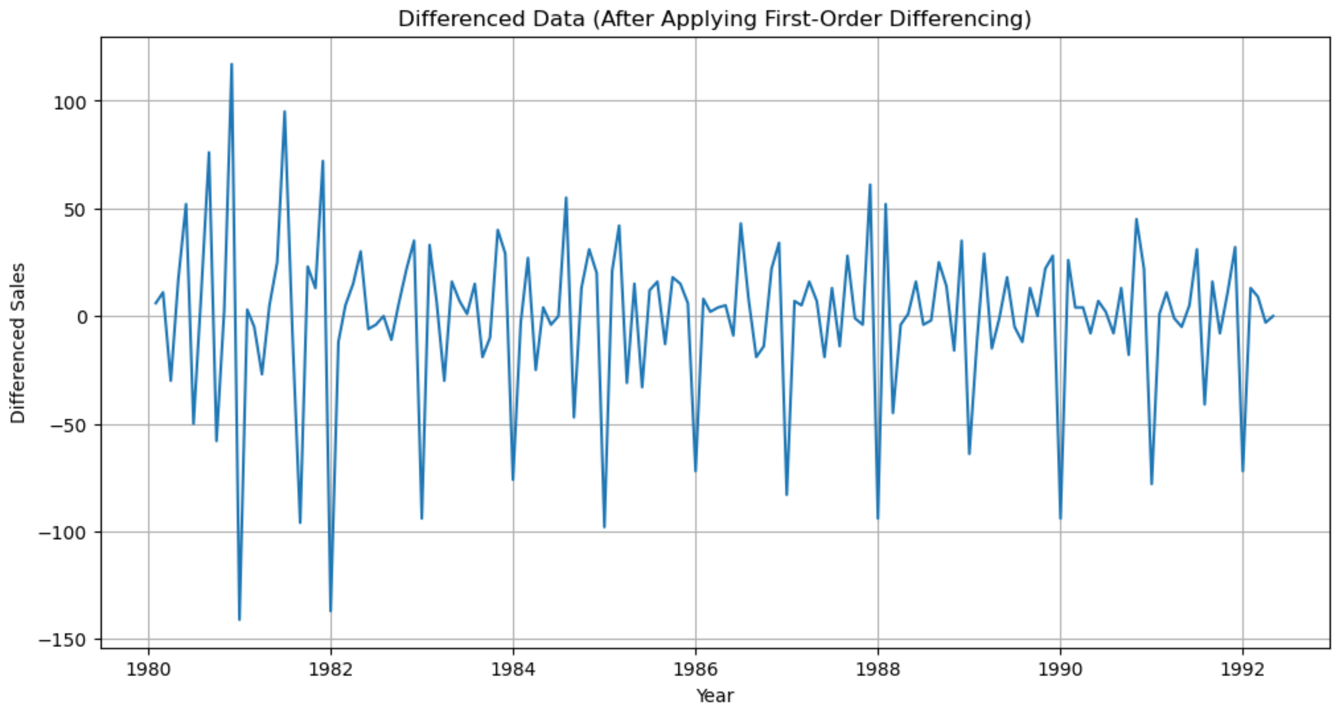
As our series was not stationary, we have to make it stationary using the Differencing method. Differencing is a common way to remove trends from a time series. We can apply first-order differencing to make the data stationary.

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

`p-value - 3.4961016848599786e-10.`

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

Timeseries plot for the difference data (stationary data).

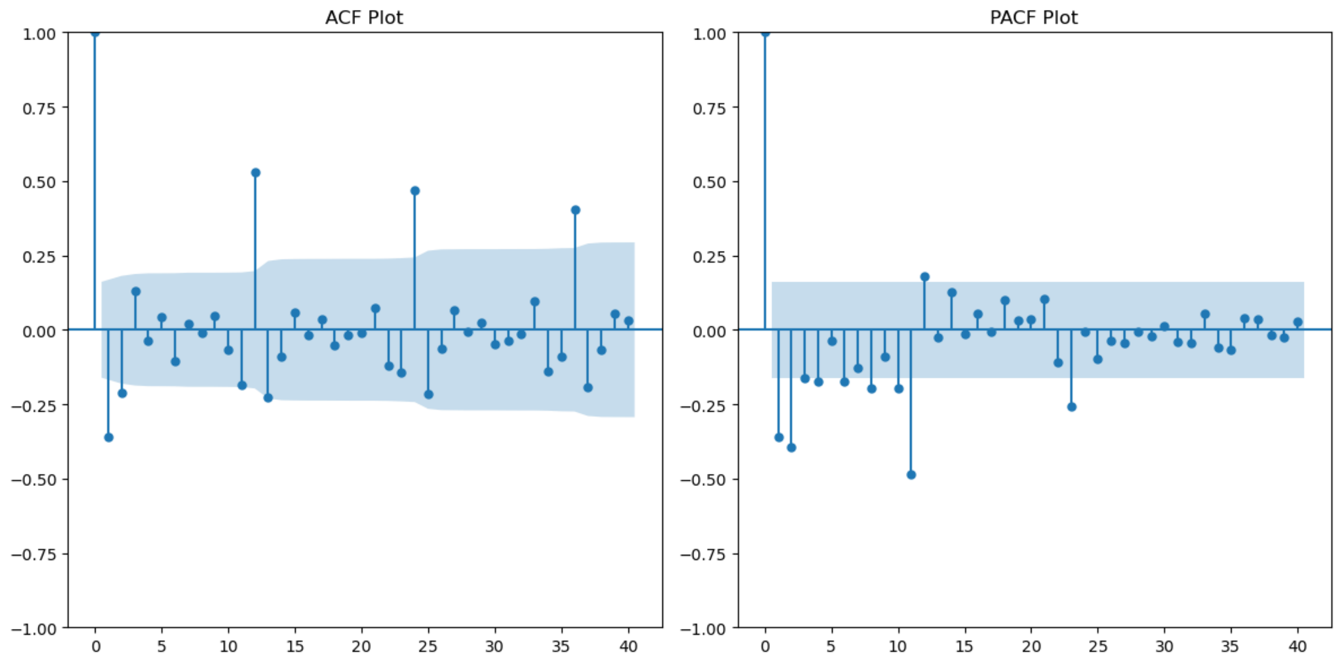


PROBLEM 1.5 MODEL BUILDING - STATIONARY DATA:

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 multiple significant spikes, suggesting an MA(1) or MA(2) components could be a good starting point.

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(2,1,2)(0,0,0)[0] intercept   : AIC=inf, Time=0.72 sec
 ARIMA(0,1,0)(0,0,0)[0] intercept   : AIC=1501.162, Time=0.05 sec
 ARIMA(1,1,0)(0,0,0)[0] intercept   : AIC=1482.523, Time=0.12 sec
 ARIMA(0,1,1)(0,0,0)[0] intercept   : AIC=inf, Time=0.24 sec
 ARIMA(0,1,0)(0,0,0)[0]             : AIC=1499.179, Time=0.05 sec
 ARIMA(2,1,0)(0,0,0)[0] intercept   : AIC=1459.779, Time=0.15 sec
 ARIMA(3,1,0)(0,0,0)[0] intercept   : AIC=1457.898, Time=0.19 sec
 ARIMA(4,1,0)(0,0,0)[0] intercept   : AIC=1455.517, Time=0.25 sec
 ARIMA(5,1,0)(0,0,0)[0] intercept   : AIC=1457.310, Time=0.37 sec
 ARIMA(4,1,1)(0,0,0)[0] intercept   : AIC=inf, Time=0.63 sec
 ARIMA(3,1,1)(0,0,0)[0] intercept   : AIC=inf, Time=0.41 sec
 ARIMA(5,1,1)(0,0,0)[0] intercept   : AIC=inf, Time=0.93 sec
 ARIMA(4,1,0)(0,0,0)[0]             : AIC=1453.703, Time=0.13 sec
 ARIMA(3,1,0)(0,0,0)[0]             : AIC=1456.025, Time=0.13 sec
 ARIMA(5,1,0)(0,0,0)[0]             : AIC=1455.511, Time=0.14 sec
 ARIMA(4,1,1)(0,0,0)[0]             : AIC=1439.786, Time=0.31 sec
 ARIMA(3,1,1)(0,0,0)[0]             : AIC=1438.569, Time=0.26 sec
 ARIMA(2,1,1)(0,0,0)[0]             : AIC=1437.705, Time=0.19 sec
 ARIMA(1,1,1)(0,0,0)[0]             : AIC=1437.156, Time=0.13 sec
 ARIMA(0,1,1)(0,0,0)[0]             : AIC=1438.644, Time=0.08 sec
 ARIMA(1,1,0)(0,0,0)[0]             : AIC=1480.559, Time=0.09 sec
 ARIMA(1,1,2)(0,0,0)[0]             : AIC=1435.657, Time=0.22 sec
 ARIMA(0,1,2)(0,0,0)[0]             : AIC=1436.199, Time=0.13 sec
 ARIMA(2,1,2)(0,0,0)[0]             : AIC=1437.638, Time=0.25 sec
 ARIMA(1,1,3)(0,0,0)[0]             : AIC=1437.640, Time=0.25 sec
 ARIMA(0,1,3)(0,0,0)[0]             : AIC=1436.548, Time=0.15 sec
 ARIMA(2,1,3)(0,0,0)[0]             : AIC=1432.027, Time=0.51 sec
 ARIMA(3,1,3)(0,0,0)[0]             : AIC=inf, Time=0.81 sec
 ARIMA(2,1,4)(0,0,0)[0]             : AIC=inf, Time=0.72 sec
 ARIMA(1,1,4)(0,0,0)[0]             : AIC=inf, Time=0.78 sec
 ARIMA(3,1,2)(0,0,0)[0]             : AIC=1439.521, Time=0.34 sec
 ARIMA(3,1,4)(0,0,0)[0]             : AIC=1435.254, Time=0.87 sec
 ARIMA(2,1,3)(0,0,0)[0] intercept   : AIC=inf, Time=0.68 sec

Best model:  ARIMA(2,1,3)(0,0,0)[0]
Total fit time: 11.298 seconds
Auto ARIMA RMSE: 19.727415342393787
```

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

```
Auto ARIMA RMSE: 19.727415342393787
```

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: 20.807042612472586
```

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(2,1,2)(1,0,1)[12] intercept   : AIC=inf, Time=1.74 sec
 ARIMA(0,1,0)(0,0,0)[12] intercept   : AIC=1501.162, Time=0.04 sec
 ARIMA(1,1,0)(1,0,0)[12] intercept   : AIC=1425.720, Time=0.33 sec
 ARIMA(0,1,1)(0,0,1)[12] intercept   : AIC=inf, Time=0.58 sec
 ARIMA(0,1,0)(0,0,0)[12]             : AIC=1499.179, Time=0.05 sec
 ARIMA(1,1,0)(0,0,0)[12] intercept   : AIC=1482.523, Time=0.13 sec
 ARIMA(1,1,0)(2,0,0)[12] intercept   : AIC=1404.734, Time=0.98 sec
 ARIMA(1,1,0)(2,0,1)[12] intercept   : AIC=1397.225, Time=1.70 sec
 ARIMA(1,1,0)(1,0,1)[12] intercept   : AIC=1395.476, Time=0.85 sec
 ARIMA(1,1,0)(0,0,1)[12] intercept   : AIC=1449.951, Time=0.36 sec
 ARIMA(1,1,0)(1,0,2)[12] intercept   : AIC=1397.295, Time=1.96 sec
 ARIMA(1,1,0)(0,0,2)[12] intercept   : AIC=1432.149, Time=1.18 sec
 ARIMA(1,1,0)(2,0,2)[12] intercept   : AIC=inf, Time=3.44 sec
 ARIMA(0,1,0)(1,0,1)[12] intercept   : AIC=1411.367, Time=0.46 sec
 ARIMA(2,1,0)(1,0,1)[12] intercept   : AIC=1369.134, Time=0.89 sec
 ARIMA(2,1,0)(0,0,1)[12] intercept   : AIC=1425.471, Time=0.46 sec
 ARIMA(2,1,0)(1,0,0)[12] intercept   : AIC=1398.773, Time=0.63 sec
 ARIMA(2,1,0)(2,0,1)[12] intercept   : AIC=1371.117, Time=1.79 sec
 ARIMA(2,1,0)(1,0,2)[12] intercept   : AIC=1371.121, Time=2.37 sec
 ARIMA(2,1,0)(0,0,0)[12] intercept   : AIC=1459.779, Time=0.18 sec
 ARIMA(2,1,0)(0,0,2)[12] intercept   : AIC=1404.172, Time=1.26 sec
 ARIMA(2,1,0)(2,0,0)[12] intercept   : AIC=1378.025, Time=1.36 sec
 ARIMA(2,1,0)(2,0,2)[12] intercept   : AIC=inf, Time=3.31 sec
 ARIMA(3,1,0)(1,0,1)[12] intercept   : AIC=1366.546, Time=1.14 sec
 ARIMA(3,1,0)(0,0,1)[12] intercept   : AIC=1422.693, Time=0.40 sec
 ARIMA(3,1,0)(1,0,0)[12] intercept   : AIC=1395.110, Time=0.61 sec
 ARIMA(3,1,0)(2,0,1)[12] intercept   : AIC=1368.546, Time=2.49 sec
 ARIMA(3,1,0)(1,0,2)[12] intercept   : AIC=1368.546, Time=2.37 sec
 ARIMA(3,1,0)(0,0,0)[12] intercept   : AIC=1457.898, Time=0.17 sec
 ARIMA(3,1,0)(0,0,2)[12] intercept   : AIC=1402.863, Time=1.32 sec
 ARIMA(3,1,0)(2,0,0)[12] intercept   : AIC=1375.868, Time=2.03 sec
 ARIMA(3,1,0)(2,0,2)[12] intercept   : AIC=inf, Time=3.72 sec
 ARIMA(4,1,0)(1,0,1)[12] intercept   : AIC=1358.866, Time=1.36 sec
 ARIMA(4,1,0)(0,0,1)[12] intercept   : AIC=1417.943, Time=0.51 sec
 ARIMA(4,1,0)(1,0,0)[12] intercept   : AIC=1387.843, Time=1.04 sec
 ARIMA(4,1,0)(2,0,1)[12] intercept   : AIC=1360.866, Time=3.71 sec
 ARIMA(4,1,0)(1,0,2)[12] intercept   : AIC=1360.866, Time=3.29 sec
 ARIMA(4,1,0)(0,0,0)[12] intercept   : AIC=1455.517, Time=0.23 sec
 ARIMA(4,1,0)(0,0,2)[12] intercept   : AIC=1398.559, Time=1.49 sec
 ARIMA(4,1,0)(2,0,0)[12] intercept   : AIC=1369.733, Time=2.03 sec
 ARIMA(4,1,0)(2,0,2)[12] intercept   : AIC=inf, Time=4.07 sec
 ARIMA(5,1,0)(1,0,1)[12] intercept   : AIC=1358.305, Time=1.95 sec
 ARIMA(5,1,0)(0,0,1)[12] intercept   : AIC=1418.964, Time=0.72 sec
 ARIMA(5,1,0)(1,0,0)[12] intercept   : AIC=1387.940, Time=0.99 sec
 ARIMA(5,1,0)(2,0,1)[12] intercept   : AIC=1360.304, Time=3.34 sec
 ARIMA(5,1,0)(1,0,2)[12] intercept   : AIC=1360.304, Time=2.87 sec
 ARIMA(5,1,0)(0,0,0)[12] intercept   : AIC=1457.310, Time=0.46 sec
 ARIMA(5,1,0)(0,0,2)[12] intercept   : AIC=1399.493, Time=2.39 sec
 ARIMA(5,1,0)(2,0,0)[12] intercept   : AIC=1369.672, Time=2.42 sec
 ARIMA(5,1,0)(2,0,2)[12] intercept   : AIC=inf, Time=5.02 sec
 ARIMA(5,1,1)(1,0,1)[12] intercept   : AIC=1362.725, Time=2.17 sec
 ARIMA(4,1,1)(1,0,1)[12] intercept   : AIC=1345.503, Time=2.69 sec
 ARIMA(4,1,1)(0,0,1)[12] intercept   : AIC=inf, Time=1.30 sec
 ARIMA(4,1,1)(1,0,0)[12] intercept   : AIC=inf, Time=1.57 sec
 ARIMA(4,1,1)(2,0,1)[12] intercept   : AIC=1348.419, Time=4.21 sec
 ARIMA(4,1,1)(1,0,2)[12] intercept   : AIC=1348.278, Time=4.78 sec
 ARIMA(4,1,1)(0,0,0)[12] intercept   : AIC=inf, Time=0.72 sec
 ARIMA(4,1,1)(0,0,2)[12] intercept   : AIC=inf, Time=3.37 sec
 ARIMA(4,1,1)(2,0,0)[12] intercept   : AIC=inf, Time=4.32 sec
 ARIMA(4,1,1)(2,0,2)[12] intercept   : AIC=inf, Time=4.74 sec
 ARIMA(3,1,1)(1,0,1)[12] intercept   : AIC=inf, Time=1.55 sec
 ARIMA(4,1,2)(1,0,1)[12] intercept   : AIC=1348.016, Time=2.40 sec
 ARIMA(3,1,2)(1,0,1)[12] intercept   : AIC=inf, Time=1.80 sec
 ARIMA(5,1,2)(1,0,1)[12] intercept   : AIC=1349.276, Time=2.40 sec
 ARIMA(4,1,1)(1,0,1)[12]             : AIC=1343.973, Time=1.76 sec
 ARIMA(4,1,1)(0,0,1)[12]             : AIC=1402.791, Time=0.68 sec
 ARIMA(4,1,1)(1,0,0)[12]             : AIC=1373.064, Time=0.87 sec
 ARIMA(4,1,1)(2,0,1)[12]             : AIC=1345.968, Time=2.90 sec
 ARIMA(4,1,1)(1,0,2)[12]             : AIC=1345.968, Time=2.74 sec
 ARIMA(4,1,1)(0,0,0)[12]             : AIC=1439.786, Time=0.29 sec
 ARIMA(4,1,1)(0,0,2)[12]             : AIC=1383.930, Time=2.34 sec
 ARIMA(4,1,1)(2,0,0)[12]             : AIC=1353.755, Time=2.02 sec
 ARIMA(4,1,1)(2,0,2)[12]             : AIC=inf, Time=15.68 sec
 ARIMA(3,1,1)(1,0,1)[12]             : AIC=inf, Time=3.30 sec
 ARIMA(4,1,0)(1,0,1)[12]             : AIC=1356.900, Time=1.05 sec
 ARIMA(5,1,1)(1,0,1)[12]             : AIC=1345.839, Time=2.10 sec
 ARIMA(4,1,2)(1,0,1)[12]             : AIC=1345.508, Time=1.81 sec
 ARIMA(3,1,0)(1,0,1)[12]             : AIC=1364.568, Time=1.00 sec
 ARIMA(3,1,2)(1,0,1)[12]             : AIC=1343.610, Time=1.08 sec
 ARIMA(3,1,2)(0,0,1)[12]             : AIC=1402.351, Time=0.63 sec
 ARIMA(3,1,2)(1,0,0)[12]             : AIC=1372.083, Time=0.65 sec
 ARIMA(3,1,2)(2,0,1)[12]             : AIC=1345.610, Time=2.77 sec
 ARIMA(3,1,2)(1,0,2)[12]             : AIC=1345.610, Time=3.21 sec
 ARIMA(3,1,2)(0,0,0)[12]             : AIC=1439.521, Time=0.33 sec
 ARIMA(3,1,2)(0,0,2)[12]             : AIC=1383.456, Time=2.02 sec
 ARIMA(3,1,2)(2,0,0)[12]             : AIC=1353.151, Time=2.03 sec
 ARIMA(3,1,2)(2,0,2)[12]             : AIC=inf, Time=3.68 sec
 ARIMA(2,1,2)(1,0,1)[12]             : AIC=1341.622, Time=1.03 sec
 ARIMA(2,1,2)(0,0,1)[12]             : AIC=1400.354, Time=0.68 sec
 ARIMA(2,1,2)(1,0,0)[12]             : AIC=1370.412, Time=0.60 sec
 ARIMA(2,1,2)(2,0,1)[12]             : AIC=1343.622, Time=2.61 sec
 ARIMA(2,1,2)(1,0,2)[12]             : AIC=1343.622, Time=2.77 sec
 ARIMA(2,1,2)(0,0,0)[12]             : AIC=1437.638, Time=0.25 sec
 ARIMA(2,1,2)(0,0,2)[12]             : AIC=1381.595, Time=1.70 sec
 ARIMA(2,1,2)(2,0,0)[12]             : AIC=1351.151, Time=1.72 sec
 ARIMA(2,1,2)(2,0,2)[12]             : AIC=inf, Time=3.51 sec
 ARIMA(1,1,2)(1,0,1)[12]             : AIC=1339.919, Time=0.79 sec
 ARIMA(1,1,2)(0,0,1)[12]             : AIC=1398.406, Time=0.44 sec
 ARIMA(1,1,2)(1,0,0)[12]             : AIC=1368.484, Time=0.42 sec
 ARIMA(1,1,2)(2,0,1)[12]             : AIC=1341.919, Time=2.15 sec

 Best model:  ARIMA(1,1,2)(1,0,1)[12]
 Total fit time: 189.911 seconds
 Auto SARIMA RMSE: 10.656532756396611
```

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

```
Auto SARIMA RMSE: 10.656532756396611
```

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: 11.266352946919064
```

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: 19.727415342393787
Manual ARIMA RMSE: 20.807042612472586
Auto SARIMA RMSE: 10.656532756396611
Manual SARIMA RMSE: 11.266352946919064
```

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:

RegressionOnTime	13.736625
SimpleAverageModel	52.239499
2pointTrailingMovingAverage	9.402197
4pointTrailingMovingAverage	12.186935
6pointTrailingMovingAverage	12.413090
9pointTrailingMovingAverage	12.620569
Alpha=0.995,SimpleExponentialSmoothing	19.021186
Alpha=0.9,SimpleExponentialSmoothing	13.838498
Alpha=0.8,Beta=1.0,DoubleExponentialSmoothing	13.180564
Alpha=0.1,Beta=0.0,Gamma=0.0,TripleExponentialSmoothing	8.406215
Alpha=0.3,Beta=0.3,Gamma=0.3,TripleExponentialSmoothing	6.689740
Auto ARIMA RMSE	19.727415
Manual ARIMA RMSE	20.807043
Auto SARIMA RMSE	10.656533

Manual SARIMA RMSE	11.266353

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: Alpha=0.3,Beta=0.3,Gamma=0.3,TripleExponentialSmoothing with RMSE: 6.689740159064305

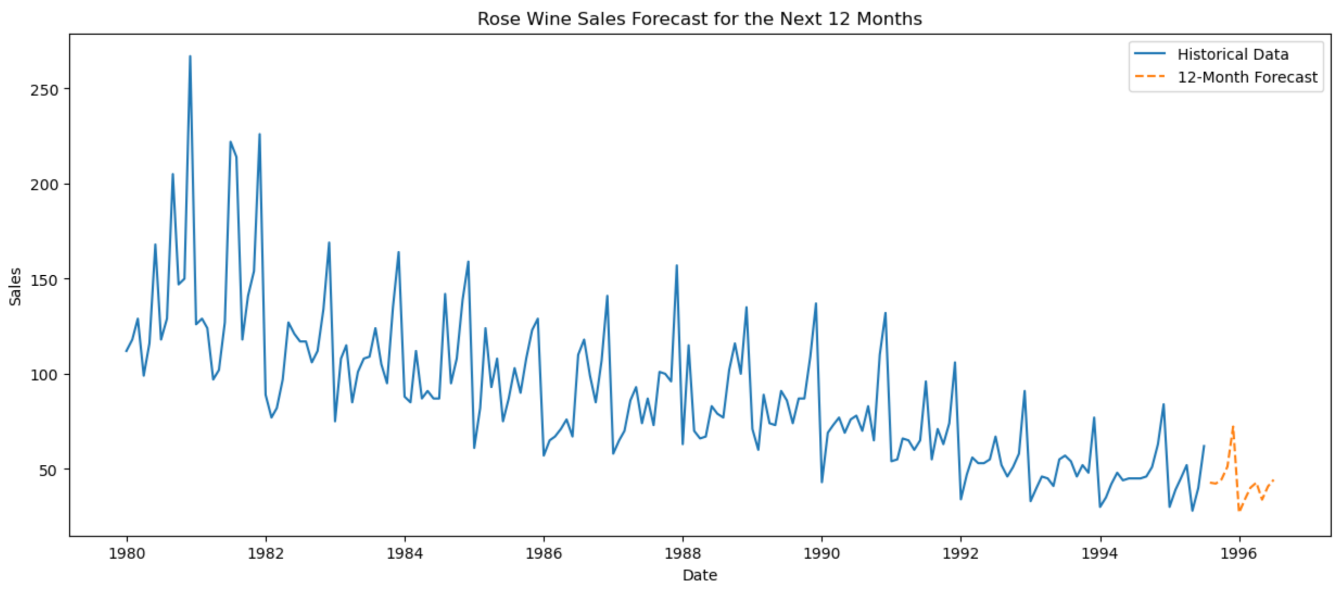
Problem 1.6.3 Rebuild the best model using the entire data:

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    42.778900
1995-09-01    42.282287
1995-10-01    44.492234
1995-11-01    51.203774
1995-12-01    72.248703
1996-01-01    26.949949
1996-02-01    34.092016
1996-03-01    40.118874
1996-04-01    42.787229
1996-05-01    33.844706
1996-06-01    40.811124
1996-07-01    44.304554
Freq: MS, 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 Trends and Patterns:

The historical data shows evident seasonal fluctuations in rose wine sales, which are captured effectively by the forecasting model. This understanding allows the business to plan inventory and marketing strategies around peak sales periods, ensuring product availability and targeted promotions.

Strategic Planning:

The forecast data enables the company to align its business strategies with expected market conditions, allowing for agile responses to changing consumer demands. This alignment ensures that the company remains competitive and can capitalize on emerging opportunities.

Forecast Accuracy:

The chosen model provides a reliable 12-month forecast, demonstrating its capability to predict future sales trends accurately. This prediction empowers the company to make informed decisions regarding production scaling, resource allocation, and financial planning.

Risk Management:

By anticipating potential dips in sales, the company can proactively manage risks associated with overproduction and stockouts. This foresight helps in maintaining optimal inventory levels, reducing storage costs, and minimizing wastage.

Model Choice:

Triple Exponential Smoothing 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 Recommendations:

Optimize Inventory Management:

Use the model's predictions to streamline inventory processes. This includes adjusting order quantities and delivery schedules to align with forecasted sales, thereby reducing excess inventory and associated costs.

Enhance Marketing Efforts:

Utilize forecast insights to design targeted marketing campaigns during high-demand periods. Special promotions and advertising should be synchronized with expected sales surges to maximize revenue.

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 forecasting model, the company can enhance its ability to anticipate market trends, optimize resource utilization, and ultimately drive growth in sales and profitability.