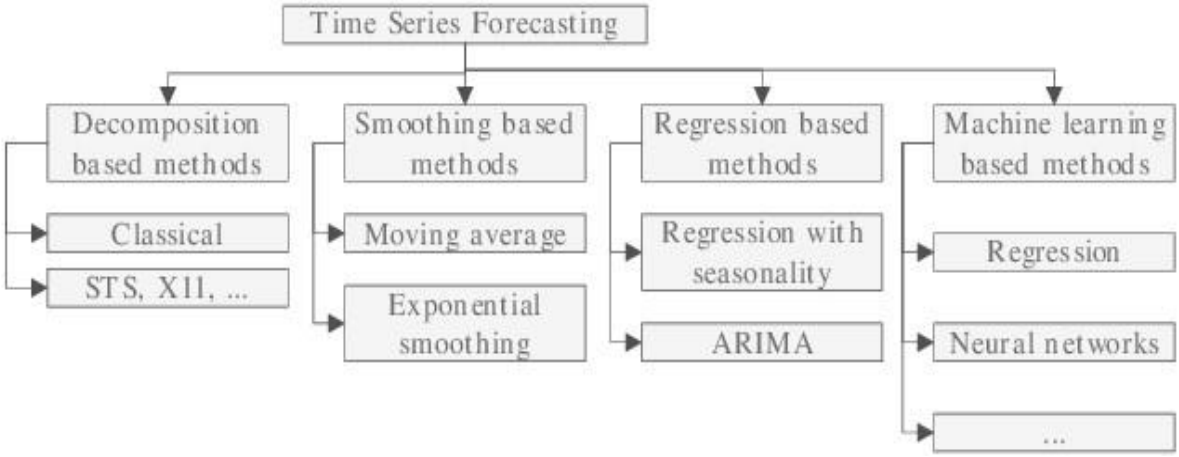


TIME SETIES FORECASTING

BUSINESS REPORT

This Business report will provide the detailed explanation of how we performed analysis according to the problem statement given in the assignment. It will also provide the relative resolution and explanation with regards to the problem statement.

MANISHA PARAKANDLA
PGP DSBA GREATLEARNING | ONLINE

Contents

TIME SERIES ANALYSIS ON SOFTDRINK DATA

PROBLEM STATEMENT:

You are an analyst in the RST soft drink company and you are expected to forecast the sales of the production of the soft drink for the upcoming 12 months from where the data ends. The data for the production of soft drink has been given to you from January 1980 to July 1995.

Data Set: **SoftDrink.csv**

First, we import all the necessary libraries such as seaborn, pandas, sklearn etc to perform our analysis.
Next, we import the dataset **SoftDrink.csv.**

1.Read the data as an appropriate Time Series data and plot the data.

When we read the data using pandas read_csv function, the sample of the data looks like:

	YearMonth	SoftDrinkProduction
0	1980-01	1954
1	1980-02	2302
2	1980-03	3054
3	1980-04	2414
4	1980-05	2226

Here, we can see that the YearMonth is showing as 'object' variable when we use info () function. we need to convert this data into time series data by parsing the YearMonth column and making it as index.

The Time series is having monthly series.

	SoftDrinkProduction
YearMonth	
1980-01-01	1954
1980-02-01	2302
1980-03-01	3054
1980-04-01	2414
1980-05-01	2226

TIME SERIES PLOT FOR SOFTDRINK PRODUCTION DATA:

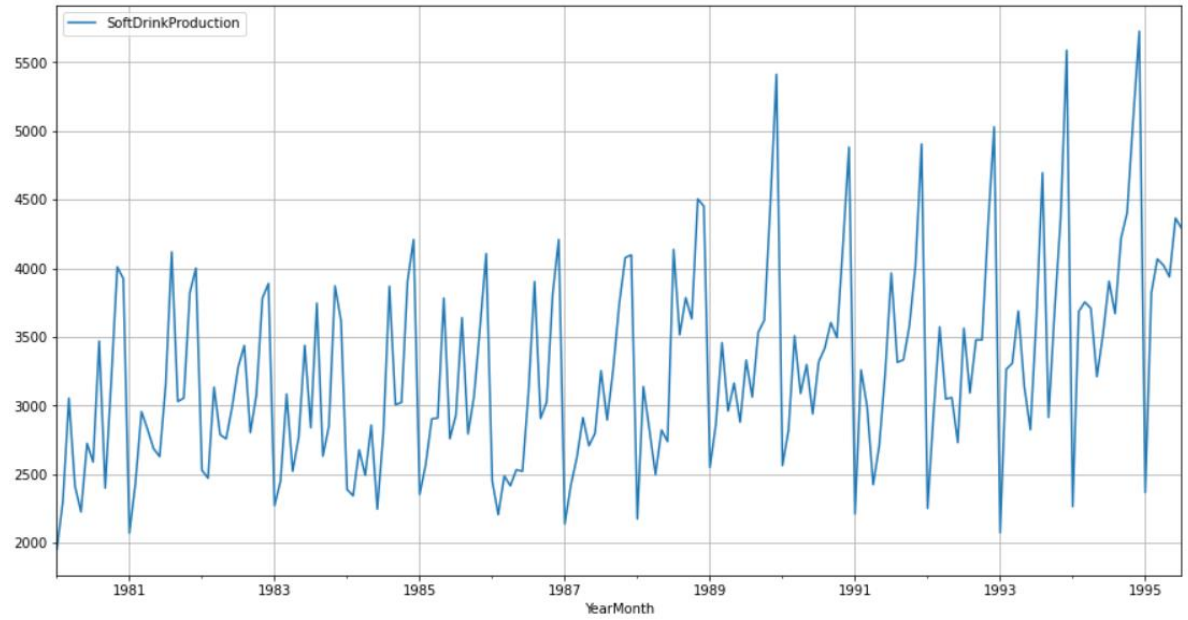


Figure 1 Time series plot for soft drink sales

Time series plots can help us spot time-dependent patterns in the data values, such as increasing or decreasing trends, changes in the mean or variability over time.

Time series plots help us look for repeating patterns such as cycles or seasonal fluctuations.

Time series data has four aspects of behaviour: Trend, Seasonality, Cycles & Unexplained Variation.

Trend is the overall long time direction of the series.

Seasonality occurs when there is repeated behaviour in the data which occurs at regular intervals. It is related to seasonal natural or human behaviour.

From the above time series plot we can see that both the trend and seasonality.

The plot clearly shows that the series is of multiplicative model.

2.Perform appropriate Exploratory Data Analysis to understand the data and also perform decomposition.

The Shape of the dataset is (187,1).

1.There are 187 observations which represent the monthly sales of respective wines from the year 1980 to July 1995.

2.The data has two variables the YearMonth of sales and the sales for the respective month of the year.

3.There are no null values present in the data.

4. Checking the info of the data:

```
<class 'pandas.core.frame.DataFrame'>
DatetimeIndex: 187 entries, 1980-01-01 to 1995-07-01
Data columns (total 1 columns):
 #   Column               Non-Null Count  Dtype
---  ------               --------------  -----
 0   SoftDrinkProduction  187 non-null    int64
dtypes: int64(1)
memory usage: 2.9 KB
```

we can see the index as DatetimeIndex and variable SoftDrinkproduction of int64 datatype.

5.Description of the data:

	count	mean	std	min	25%	50%	75%	max
SoftDrinkProduction	187.0	3262.609626	728.357367	1954.0	2748.0	3134.0	3741.0	5725.0

Boxplot for yearly/Monthly sales for soft drink:

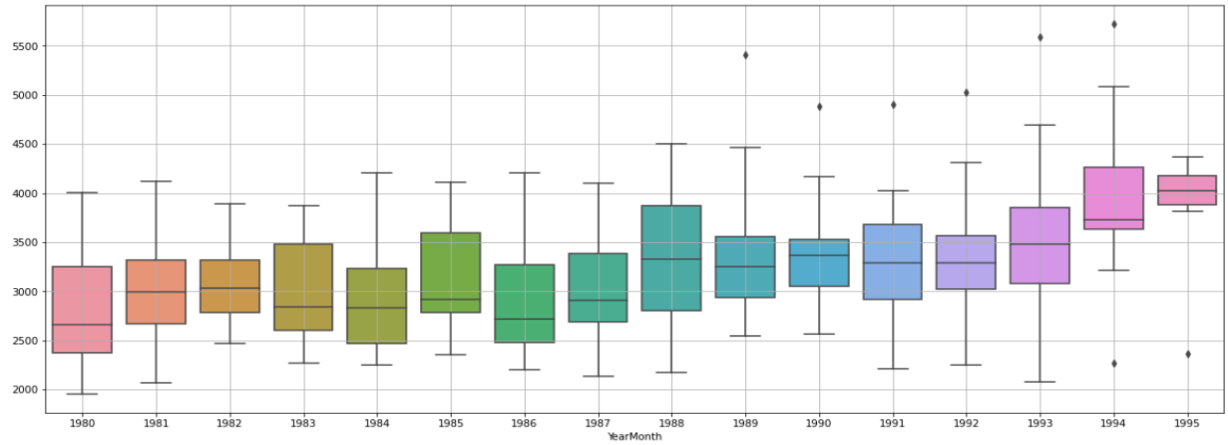


Figure 2 Yearly Boxplot

From the Yearly boxplot, we can see the trend, Seasonality and the outliers present in the data from 1989 to 1995.

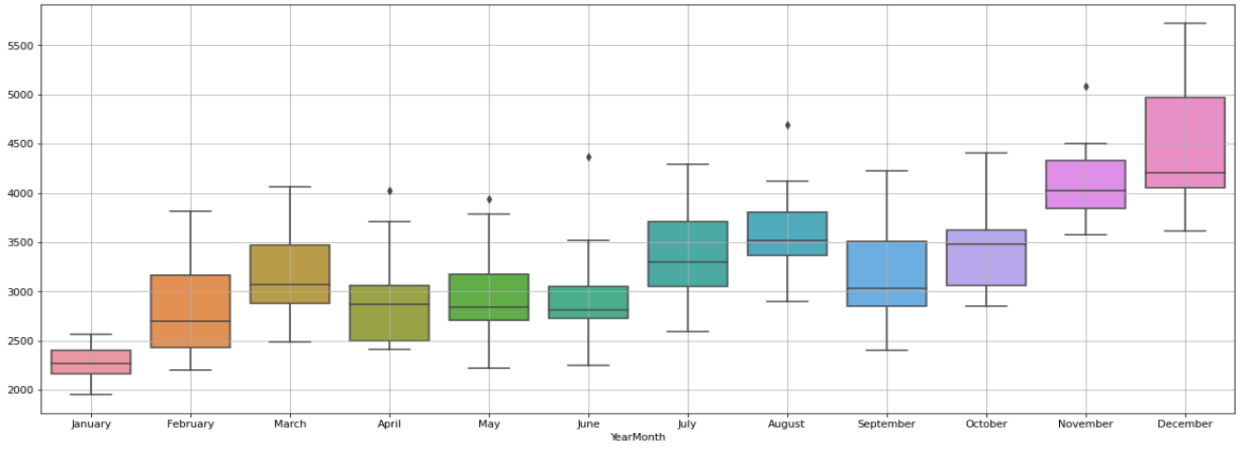


Figure 3 Monthly Boxplot

We can observe the slight spike & drop of trend in the monthly boxplot and there are less number of outliers present in monthly series data.

- Outliers are present only in April, May, June, August, November months.

- We can observe the spike of sales from September to December.

- The December month is having the highest spike of sales every year.

Plot a time series month plot to understand the spread of Sales across different years and within different months across years

YearMonth	1	2	3	4	5	6	7	8	9	10	11	12
YearMonth												
1980	1954.0	2302.0	3054.0	2414.0	2226.0	2725.0	2589.0	3470.0	2400.0	3180.0	4009.0	3924.0
1981	2072.0	2434.0	2956.0	2828.0	2687.0	2629.0	3150.0	4119.0	3030.0	3055.0	3821.0	4001.0
1982	2529.0	2472.0	3134.0	2789.0	2758.0	2993.0	3282.0	3437.0	2804.0	3076.0	3782.0	3889.0
1983	2271.0	2452.0	3084.0	2522.0	2769.0	3438.0	2839.0	3746.0	2632.0	2851.0	3871.0	3618.0
1984	2389.0	2344.0	2678.0	2492.0	2858.0	2246.0	2800.0	3869.0	3007.0	3023.0	3907.0	4209.0
1985	2353.0	2570.0	2903.0	2910.0	3782.0	2759.0	2931.0	3641.0	2794.0	3070.0	3576.0	4106.0
1986	2452.0	2206.0	2488.0	2416.0	2534.0	2521.0	3093.0	3903.0	2907.0	3025.0	3812.0	4209.0
1987	2138.0	2419.0	2622.0	2912.0	2708.0	2798.0	3254.0	2895.0	3263.0	3736.0	4077.0	4097.0
1988	2175.0	3138.0	2823.0	2498.0	2822.0	2738.0	4137.0	3515.0	3785.0	3632.0	4504.0	4451.0
1989	2550.0	2867.0	3458.0	2961.0	3163.0	2880.0	3331.0	3062.0	3534.0	3622.0	4464.0	5411.0
1990	2564.0	2820.0	3508.0	3088.0	3299.0	2939.0	3320.0	3418.0	3604.0	3495.0	4163.0	4882.0
1991	2211.0	3260.0	2992.0	2425.0	2707.0	3244.0	3965.0	3315.0	3333.0	3583.0	4021.0	4904.0
1992	2252.0	2952.0	3573.0	3048.0	3059.0	2731.0	3563.0	3092.0	3478.0	3478.0	4308.0	5029.0
1993	2075.0	3264.0	3308.0	3688.0	3136.0	2824.0	3644.0	4694.0	2914.0	3686.0	4358.0	5587.0
1994	2265.0	3685.0	3754.0	3708.0	3210.0	3517.0	3905.0	3670.0	4221.0	4404.0	5086.0	5725.0
1995	2367.0	3819.0	4067.0	4022.0	3937.0	4365.0	4290.0	NaN	NaN	NaN	NaN	NaN

Cumulative % and Month on Month % sales plots of soft drink sales:

Time series month plot helps us to understand the spread of Sales across different years and within different months across years.

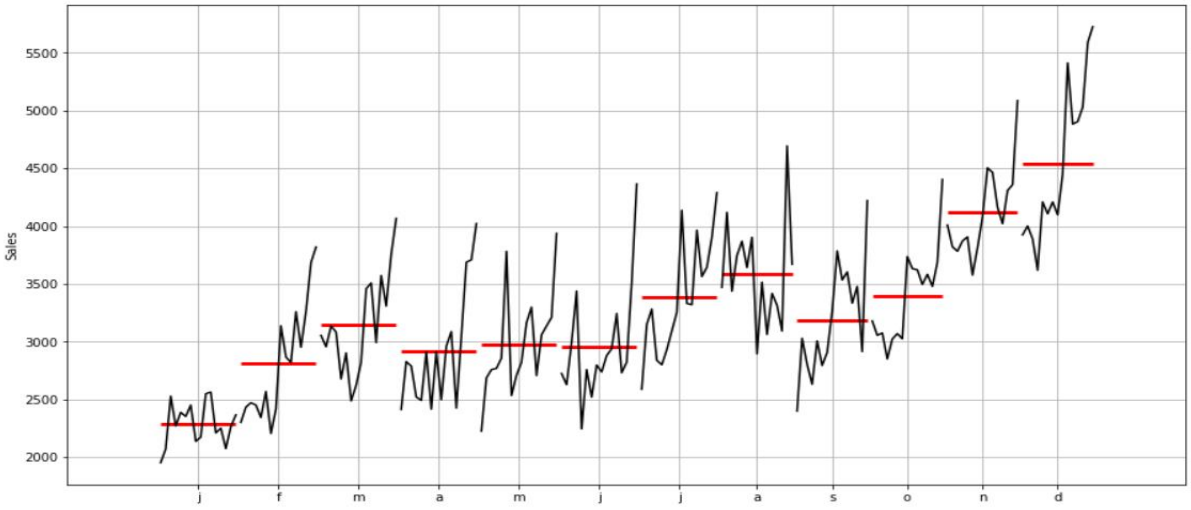


Figure 4 Timeseries month plot for spread of sales

→From the above discrete time series plot, we can understand the behaviour of the time series data through the median which is represented as *red line*.

→The median is varying for each month.

The ECD curve tells us what percentage of data points refer to what number of sales.

The ECD curve is an estimator of the cumulative distribution function.

It essentially allows you a feature of your data in order from least to greatest and see the whole feature as it is distributed across the dataset

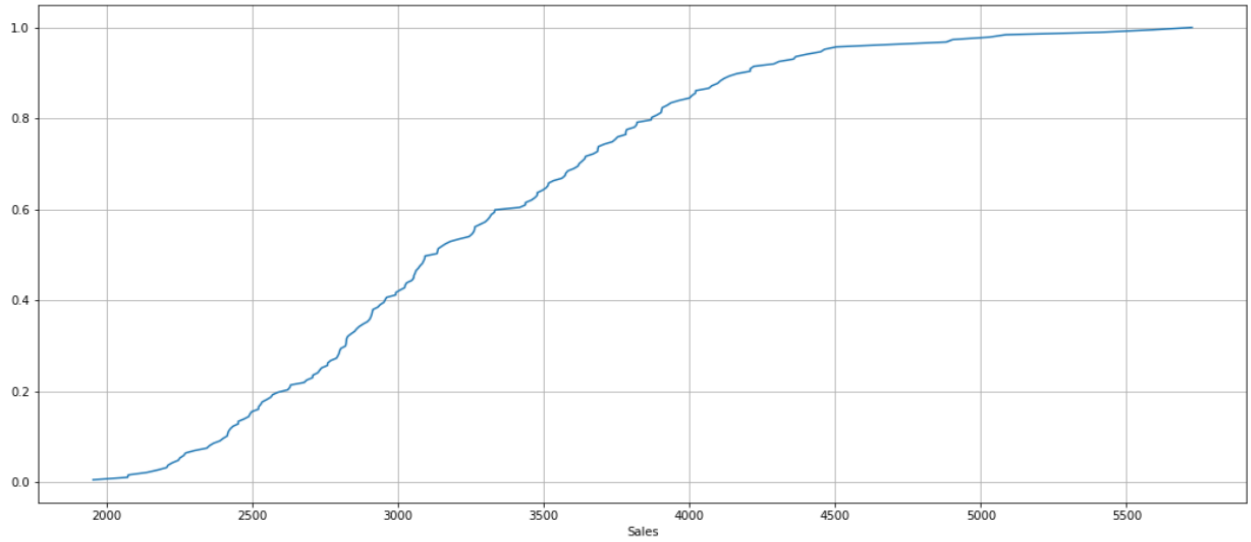


Figure 5 Empirical cumulative distribution curve

Line plot for monthly sales for soft drink sales:

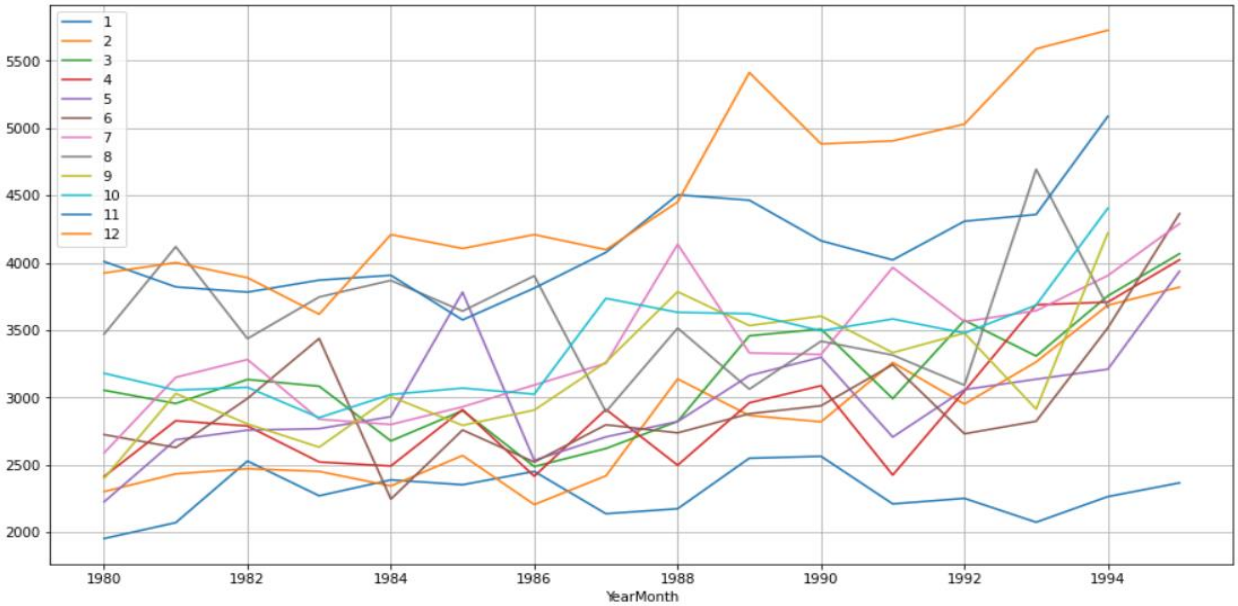


Figure 6 Line plot for soft drink sales

The above line plot shoes the monthly sales performance of SoftDrinkProduction across every year. Generally, November and December months are having the highest sales compared to others. The SoftDrinkProduction are highest and almost in peak stage in the year 1994. December month is having the huge spike of sales in 1989 & 1994. Also, for November month we can see the highest peak sales in 1988 and 1994.

Read the monthly data into a quarterly and yearly format. Compare the time series plot and draw inferences.

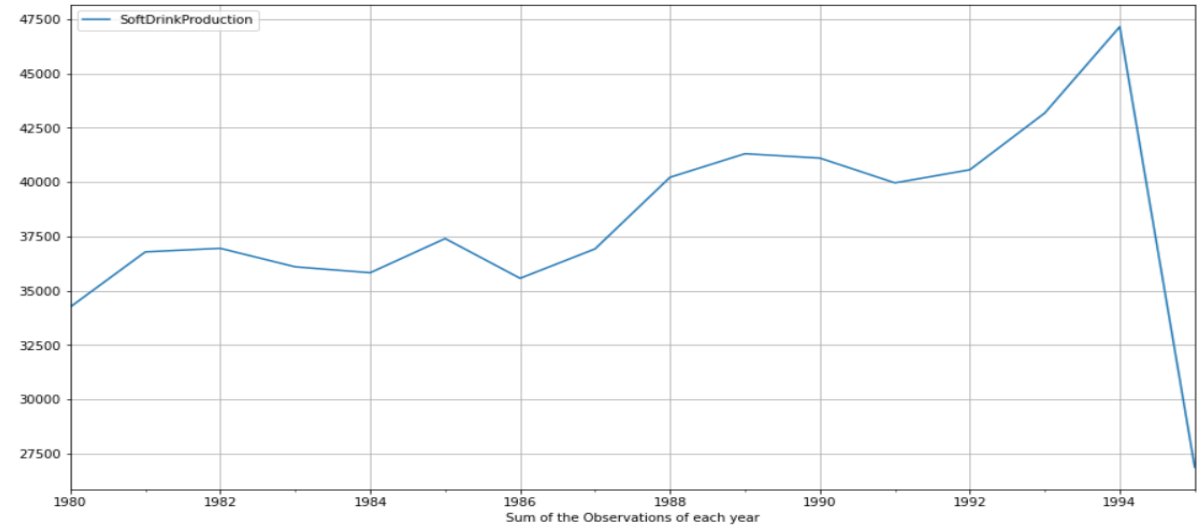


Figure 7 yearly plot – sum

From the above plot, we can observe there is seasonality and trend. The sales across years for all the months are displayed in the graph. we can observe a clear highest spike from 1991 to 1994. Also, the year 1994 has the highest peak point in sales. After that we can see the sudden fall in the sales.

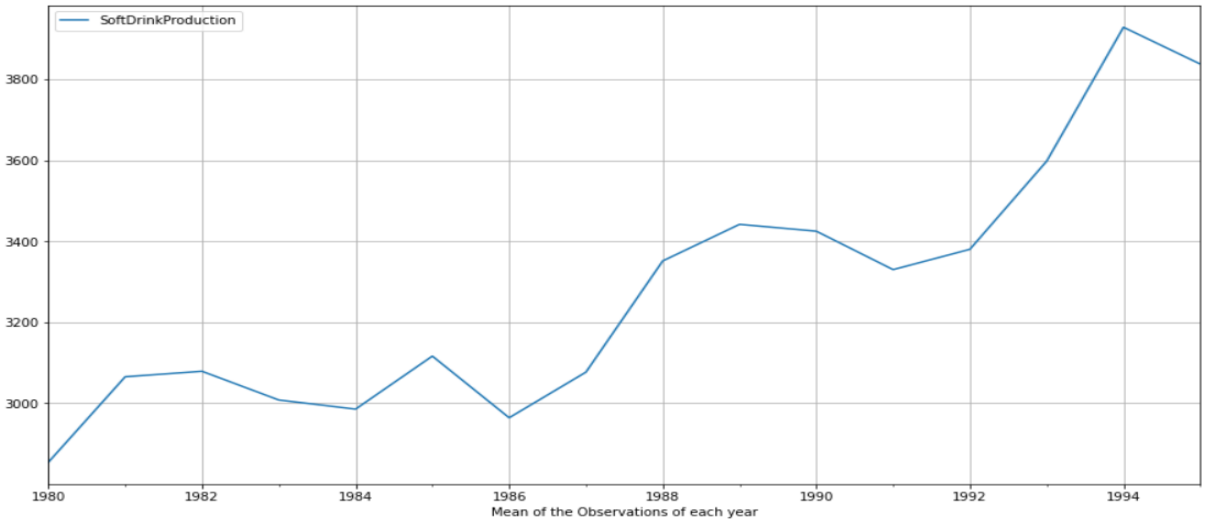
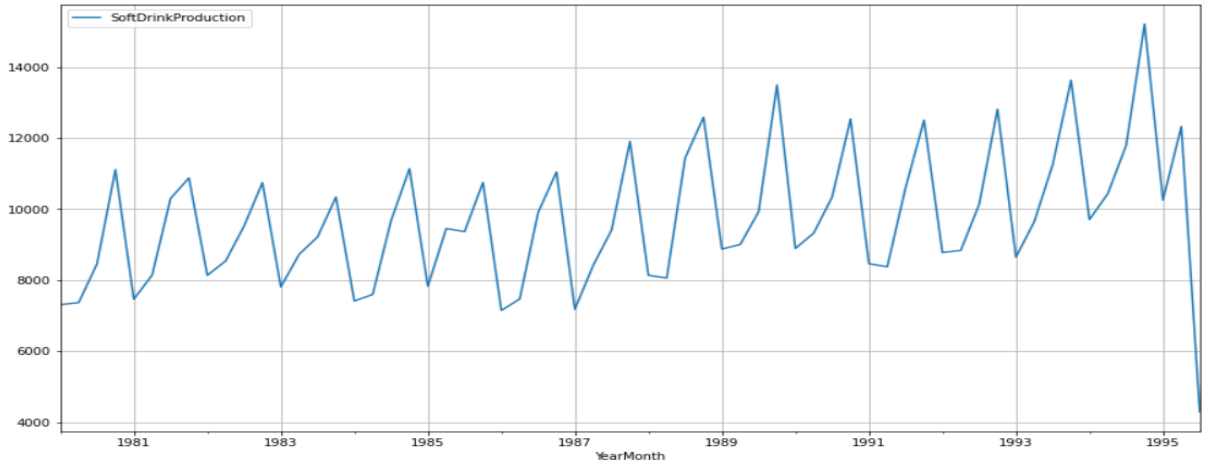
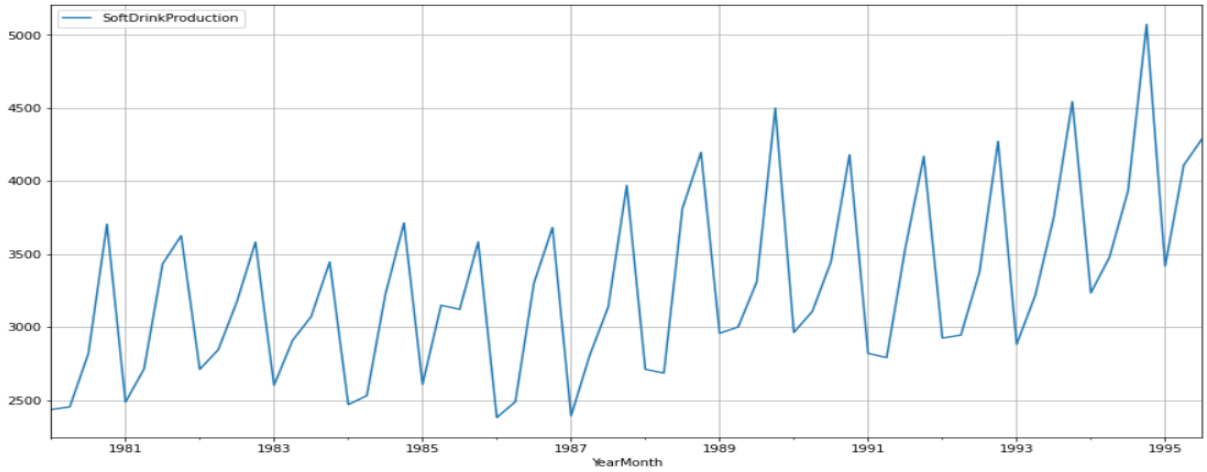


Figure 8 Yearly plot – Mean

From the above plot, we can observe the trend, seasonality and the average sales across each year.

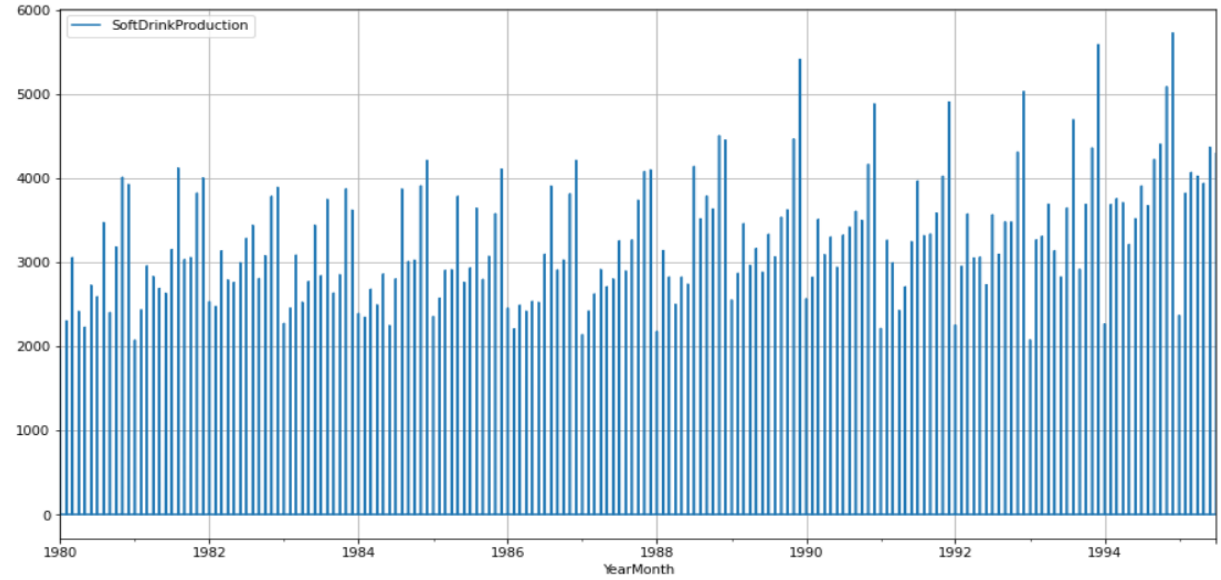
Quarterly plots:





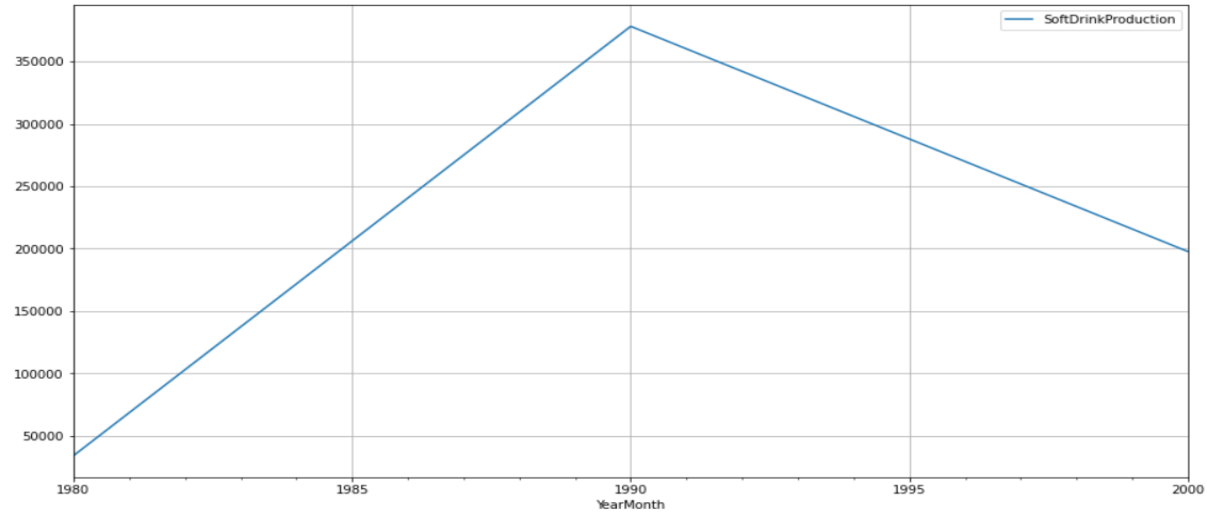
From the above Quarterly plots, we can see the slight smoothening of the curve. The trend and seasonality are also clearly visible. Both the sum and average sales plots are almost same.

Daily plot:



Resampling the data to daily interval where the number of observations 0, which is not a good practise because it does not give us clear understanding of the behaviour of the time series.

Decade Plot:



from the Decade Plot, we observe the smoothened seasonality. But the trend is clearly visible.

Average sales & Percentage change of Sales with respect to time:

The average Retail Sales per month and the month-on-month percentage change of Retail sales, group by date and get average Sales, and precent change

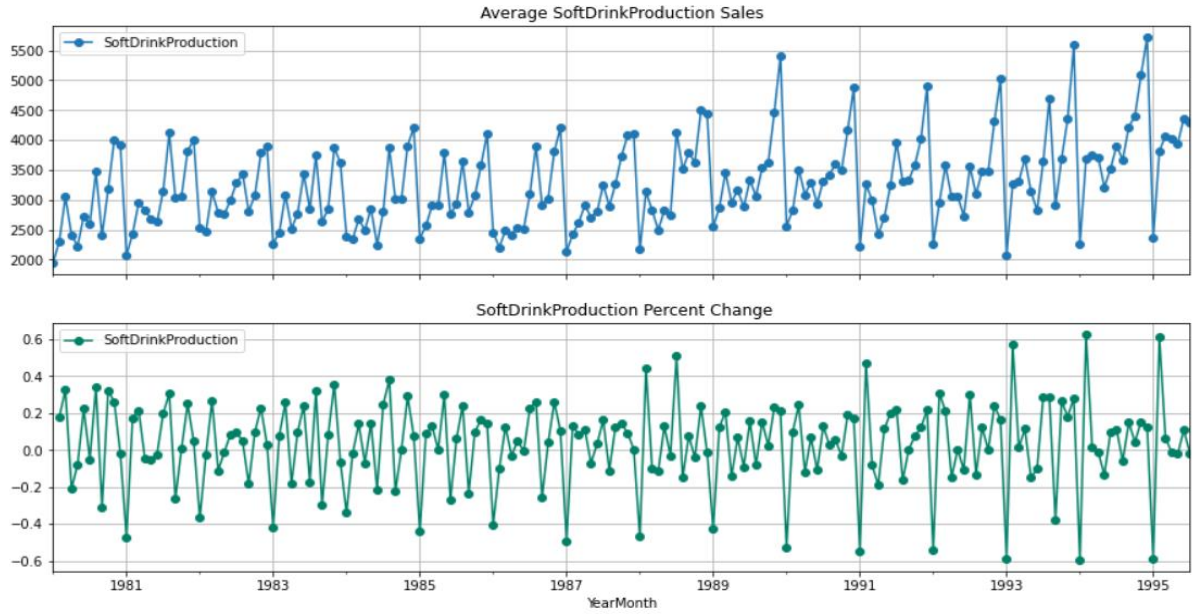


Figure 9 Average sales & Percentage change of Sales

From the above plot,

we can see the average sales of the soft drink production for every month in a year.

The dots represent the average sales of the month.

From the percent change plot of the sales,

we can see that at what percent the sales are increasing or decreasing from month to month.

Decomposition:

Decomposing the time series data plays an important role in forecasting and improving forecast accuracy.

Seasonal_decompose uses the classical decomposition method. There are two types of decomposition in this.

1. Additive Decomposition: It is a Linear model. The components are added together.

i.e., **Y = T + S + R.**

2. Multiplicative Decomposition: It's a non-linear model. The components are multiplied together.

i.e., **Y = T * S * R.**

Let us now look at the time series plot before performing the decomposition:

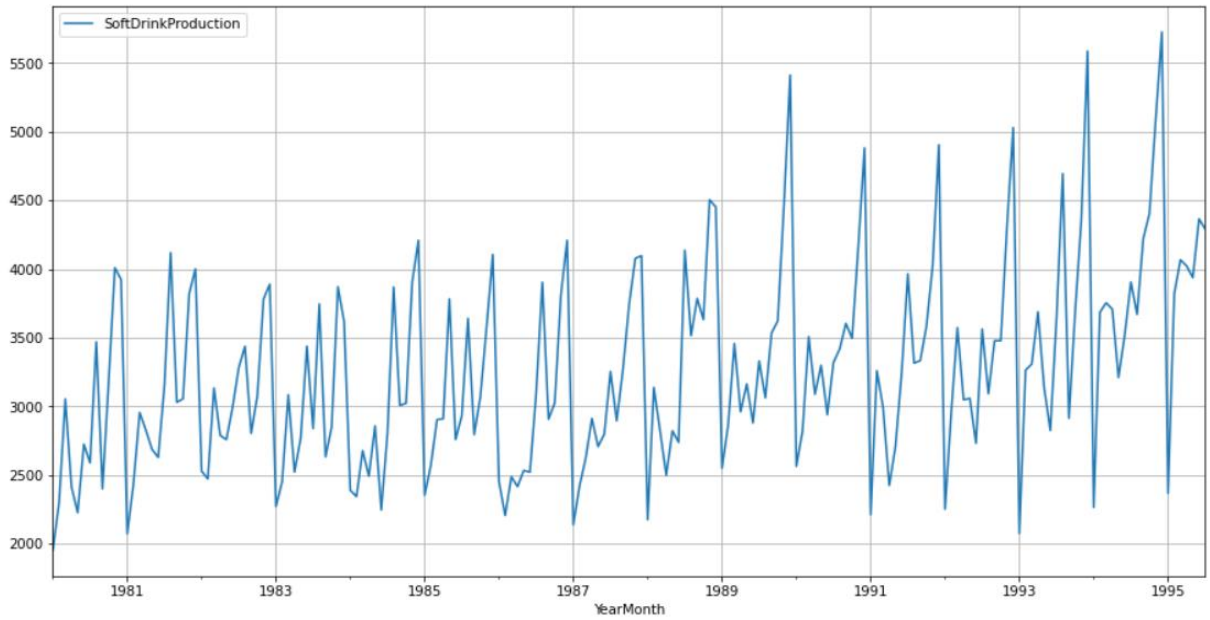


Figure 10 Time series plot -2

The time series data shows both trend & seasonality.

The plot clearly shows that the series is of multiplicative model.

Here, I am doing both the Additive and Multiplicative models to understand the difference between then clearly.

Additive Decomposition:

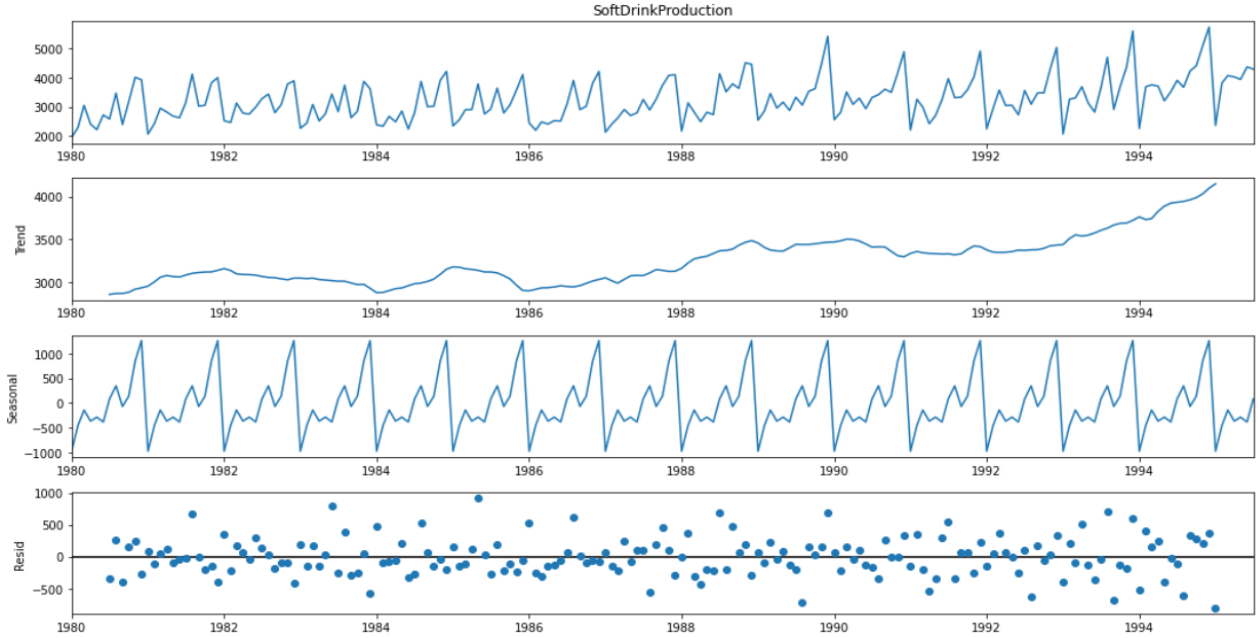


Figure 11 Additive Decomposition

From the above decomposition plot, we can see that the time series data is having the trend, seasonality in increasing fashion. Here, we can see very high residual from -500 to +500 on Y- axis.

Hence, It is very clear that the time series is not additive.

Multiplicative Decomposition:

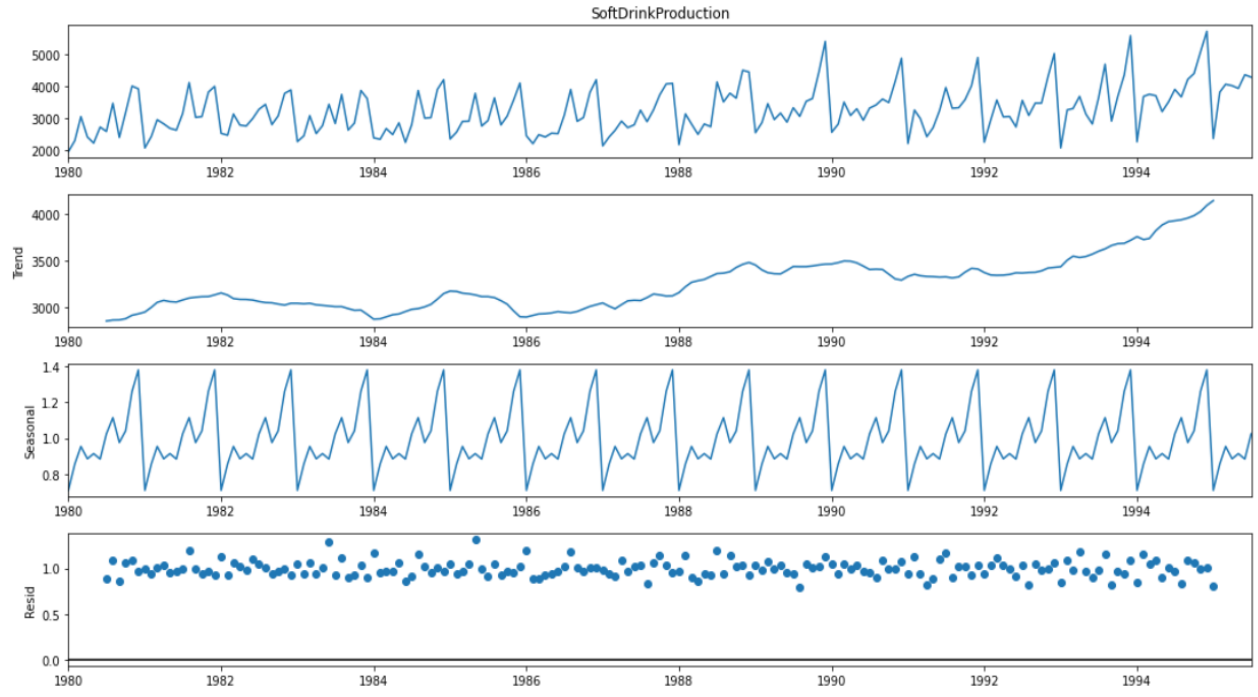


Figure 12 Multiplicative Decomposition

From the above decomposition plot, we can see that the time series data is having the trend, seasonality. The residual is lying from 0.5 to 1.0 on Y- axis which is very low when compared to the additive model.

Also, the magnitude of the seasonal component changes with time.

Hence, It is very clear that the time series is Multiplicative.

3. Split the data into training and test. The test data should start in 1991.

Split the data into training and test. The test data should start in 1991.

The train data of shoe sales has been split up to the year 1990 and has 132 data points.

The test data has been split from the year 1991. and has 55 data points.

From train-test split we will be predicting the future sales in comparison with past years' sale.

Shape of train data: (132,1)

Shape of test data: (55,1)

First/last few rows of training and testing data:

First few rows of Training Data

YearMonth	SoftDrinkProduction
1980-01-01	1954
1980-02-01	2302
1980-03-01	3054
1980-04-01	2414
1980-05-01	2226

Last few rows of Training Data

YearMonth	SoftDrinkProduction
1990-08-01	3418
1990-09-01	3604
1990-10-01	3495
1990-11-01	4163
1990-12-01	4882

First few rows of Test Data

YearMonth	SoftDrinkProduction
1991-01-01	2211

1991-02-01	3260
1991-03-01	2992
1991-04-01	2425
1991-05-01	2707

Last few rows of Test Data

YearMonth	SoftDrinkProduction
1995-03-01	4067
1995-04-01	4022
1995-05-01	3937
1995-06-01	4365
1995-07-01	4290

We can see that now the train & test data are split.

The test data starts from 1991 where the train data starts before 1981 & ends 1990.

From train test split we can predict the similar performance compared to past years.

Plot the Train test split of Time series data:

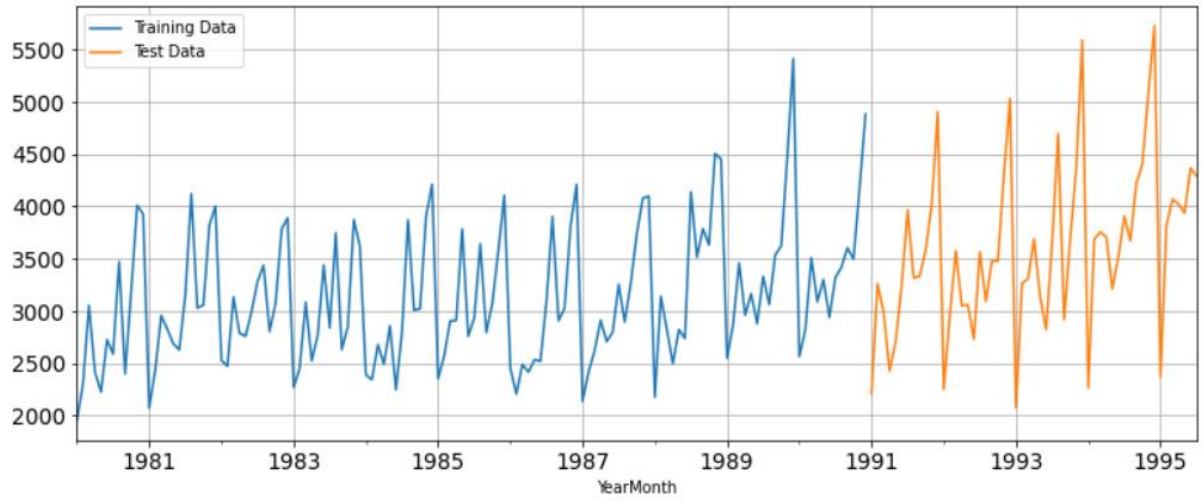


Figure 13 Train - test Split

4. Build all the exponential smoothing models on the training data and evaluate the model using RMSE on the test data. Other models such as regression, naïve forecast models and simple average models. should also be built on the training data and check the performance on the test data using RMSE.

RMSE — Root Mean Squared Error

RMSE tells you how many units your model is wrong on average.

Model1: Linear Regression

For linear regression, the equation will be y=a+b (time)

For this particular linear regression, we are going to regress the 'SoftDrinkProduction' variable against the order of the occurrence. For this we need to modify our training data before fitting it into a linear regression.

```
Training Time instance
 [1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13, 14, 15, 16, 17, 18, 19, 20, 21, 22, 23, 24, 25, 26, 27, 28, 29, 30, 31, 32, 33, 3
4, 35, 36, 37, 38, 39, 40, 41, 42, 43, 44, 45, 46, 47, 48, 49, 50, 51, 52, 53, 54, 55, 56, 57, 58, 59, 60, 61, 62, 63, 64, 65,
66, 67, 68, 69, 70, 71, 72, 73, 74, 75, 76, 77, 78, 79, 80, 81, 82, 83, 84, 85, 86, 87, 88, 89, 90, 91, 92, 93, 94, 95, 96, 97,
98, 99, 100, 101, 102, 103, 104, 105, 106, 107, 108, 109, 110, 111, 112, 113, 114, 115, 116, 117, 118, 119, 120, 121, 122, 123,
124, 125, 126, 127, 128, 129, 130, 131, 132]
Test Time instance
 [43, 44, 45, 46, 47, 48, 49, 50, 51, 52, 53, 54, 55, 56, 57, 58, 59, 60, 61, 62, 63, 64, 65, 66, 67, 68, 69, 70, 71, 72, 73, 7
4, 75, 76, 77, 78, 79, 80, 81, 82, 83, 84, 85, 86, 87, 88, 89, 90, 91, 92, 93, 94, 95, 96, 97]
```

we have generated the numerical time instance order for both the training and test data.

Now we will add these values in the training and test data.

First few rows of Training Data

YearMonth	SoftDrinkProduction	Time
1980-01-01	1954	1
1980-02-01	2302	2
1980-03-01	3054	3
1980-04-01	2414	4
1980-05-01	2226	5

Last few rows of Training Data

YearMonth	SoftDrinkProduction	Time
1990-08-01	3418	128
1990-09-01	3604	129
1990-10-01	3495	130
1990-11-01	4163	131
1990-12-01	4882	132

First few rows of Test Data

YearMonth	SoftDrinkProduction	Time
1991-01-01	2211	43
1991-02-01	3260	44
1991-03-01	2992	45
1991-04-01	2425	46
1991-05-01	2707	47

Last few rows of Test Data

YearMonth	SoftDrinkProduction	Time
1995-03-01	4067	93
1995-04-01	4022	94
1995-05-01	3937	95
1995-06-01	4365	96
1995-07-01	4290	97

Predictions on Train Data:

```
[array([2787.26304397, 2792.40661073, 2797.55017749, 2802.69374425,
       2807.83731101, 2812.98087777, 2818.12444453, 2823.26801128,
       2828.41157804, 2833.5551448 , 2838.69871156, 2843.84227832,
       2848.98584508, 2854.12941184, 2859.27297859, 2864.41654535,
       2869.56011211, 2874.70367887, 2879.84724563, 2884.99081239,
       2890.13437915, 2895.27794591, 2900.42151266, 2905.56507942,
       2910.70864618, 2915.85221294, 2920.9957797 , 2926.13934646,
       2931.28291322, 2936.42647998, 2941.57004673, 2946.71361349,
       2951.85718025, 2957.00074701, 2962.14431377, 2967.28788053,
       2972.43144729, 2977.57501404, 2982.7185808 , 2987.86214756,
       2993.00571432, 2998.14928108, 3003.29284784, 3008.4364146 ,
       3013.57998136, 3018.72354811, 3023.86711487, 3029.01068163,
       3034.15424839, 3039.29781515, 3044.44138191, 3049.58494867,
       3054.72851542, 3059.87208218, 3065.01564894, 3070.1592157 ,
       3075.30278246, 3080.44634922, 3085.58991598, 3090.73348274,
       3095.87704949, 3101.02061625, 3106.16418301, 3111.30774977,
       3116.45131653, 3121.59488329, 3126.73845005, 3131.8820168 ,
       3137.02558356, 3142.16915032, 3147.31271708, 3152.45628384,
       3157.5998506 , 3162.74341736, 3167.88698412, 3173.03055087,
       3178.17411763, 3183.31768439, 3188.46125115, 3193.60481791,
       3198.74838467, 3203.89195143, 3209.03551818, 3214.17908494,
       3219.3226517 , 3224.46621846, 3229.60978522, 3234.75335198,
       3239.89691874, 3245.0404855 , 3250.18405225, 3255.32761901,
       3260.47118577, 3265.61475253, 3270.75831929, 3275.90188605,
       3281.04545281, 3286.18901956, 3291.33258632, 3296.47615308,
       3301.61971984, 3306.7632866 , 3311.90685336, 3317.05042012,
       3322.19398688, 3327.33755363, 3332.48112039, 3337.62468715,
       3342.76825391, 3347.91182067, 3353.05538743, 3358.19895419,
       3363.34252094, 3368.4860877 , 3373.62965446, 3378.77322122,
       3383.91678798, 3389.06035474, 3394.2039215 , 3399.34748826,
       3404.49105501, 3409.63462177, 3414.77818853, 3419.92175529,
       3425.06532205, 3430.20888881, 3435.35245557, 3440.49602233,
       3445.63958908, 3450.78315584, 3455.9267226 , 3461.07028936])]
```

Predictions on Test Data:

```
[array([3003.29284784, 3008.4364146 , 3013.57998136, 3018.72354811,
       3023.86711487, 3029.01068163, 3034.15424839, 3039.29781515,
       3044.44138191, 3049.58494867, 3054.72851542, 3059.87208218,
       3065.01564894, 3070.1592157 , 3075.30278246, 3080.44634922,
       3085.58991598, 3090.73348274, 3095.87704949, 3101.02061625,
       3106.16418301, 3111.30774977, 3116.45131653, 3121.59488329,
       3126.73845005, 3131.8820168 , 3137.02558356, 3142.16915032,
       3147.31271708, 3152.45628384, 3157.5998506 , 3162.74341736,
       3167.88698412, 3173.03055087, 3178.17411763, 3183.31768439,
       3188.46125115, 3193.60481791, 3198.74838467, 3203.89195143,
       3209.03551818, 3214.17908494, 3219.3226517 , 3224.46621846,
       3229.60978522, 3234.75335198, 3239.89691874, 3245.0404855 ,
       3250.18405225, 3255.32761901, 3260.47118577, 3265.61475253,
       3270.75831929, 3275.90188605, 3281.04545281])]
```

Linear Regression train test plot:

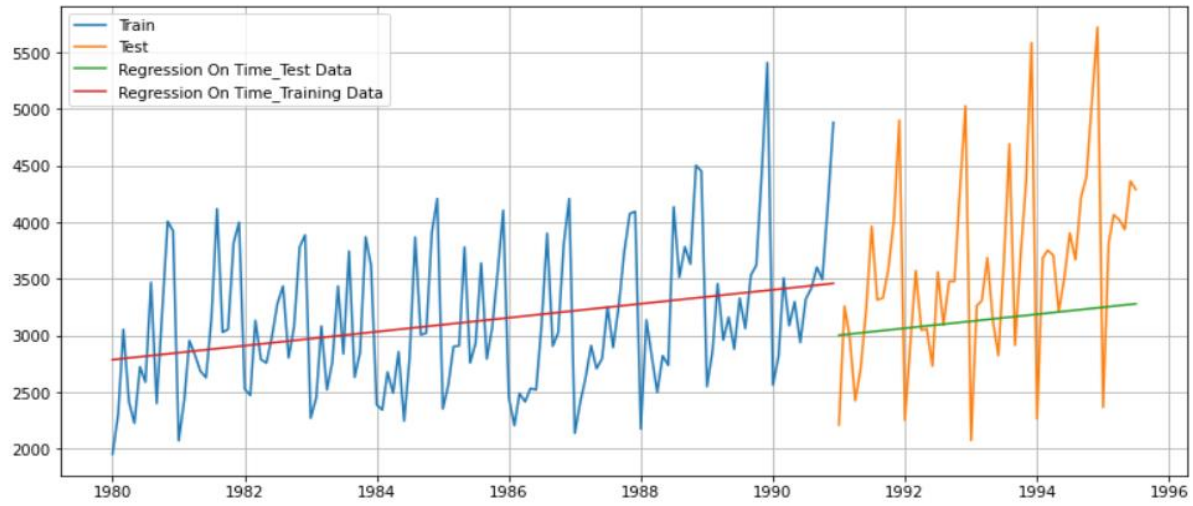


Figure 14 Linear Regression Train - Test Plot

From the plot we are analysing the Test data: The Orange coloured area represents the test data and the green line represents the linear regression values that we have calculated and which cuts the curves almost mean / average of the data.

The average trend is shown between 1992 and 1994. We can observe that, the gap from reg ression line to upward peak and downward peak which we predicted is high. From this, we c an understand that the error is high. Hence, we can use the Linear Regression model to pred ict the trend in the timeseries data. Regression On Time forecast on the Test Data:
RMSE is 898.173

Model2: Naive Forecast Model

Naive forecast involves using the previous observation directly as the forecast without any change. It is often called the persistence forecast as the prior observation is persisted.

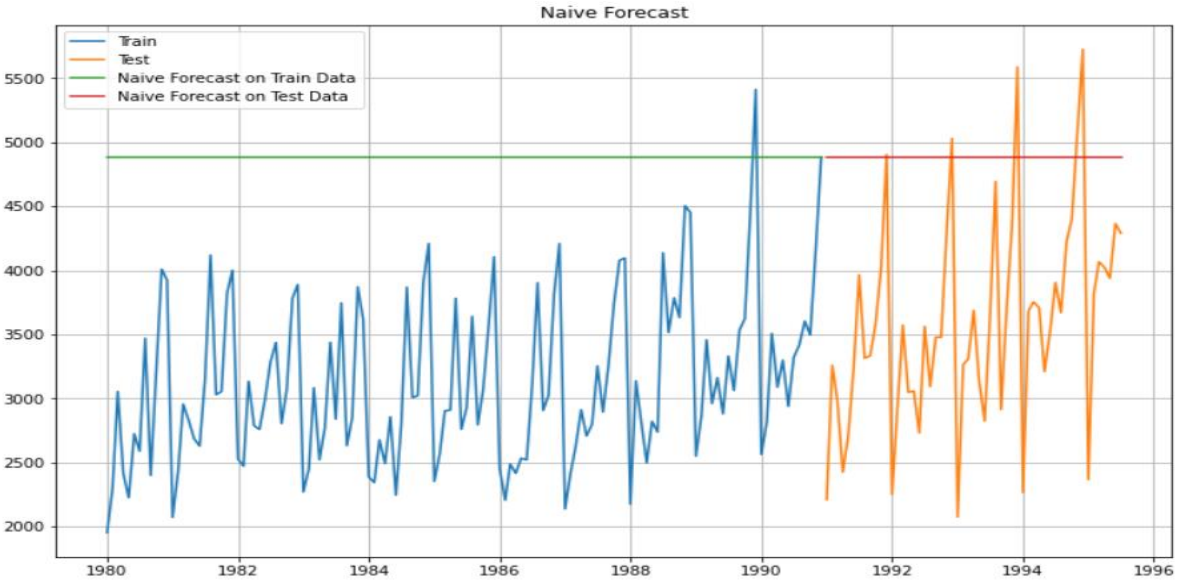


Figure 15 Naive- train- test plot

From the plot we are analysing the Test data:

The Orange coloured area represents the test data and the red line represents the naive forecast.

The error is very high for predicted values in the test data.

The error in naive model is very high compared to the linear regression model.

We cannot predict the trend in the naive forecast model.

Naive Model forecast on the Test Data, **RMSE is 1519.259**

we can observe that the RMSE for Naive model is higher than the Regression model.

Model3: Simple Average Model

This model averages the data by months or quarters or years and then calculate the average for the period. Then find out, what percentage it is to the grand average.

For this particular simple average method, we will forecast by using the average of the training values.

YearMonth	SoftDrinkProduction	mean_forecast
1980-01-01	1954	3124.166667
1980-02-01	2302	3124.166667
1980-03-01	3054	3124.166667
1980-04-01	2414	3124.166667
1980-05-01	2226	3124.166667

YearMonth	SoftDrinkProduction	mean_forecast
1991-01-01	2211	3124.166667
1991-02-01	3260	3124.166667
1991-03-01	2992	3124.166667
1991-04-01	2425	3124.166667
1991-05-01	2707	3124.166667

Simple Average Train – Test plot:

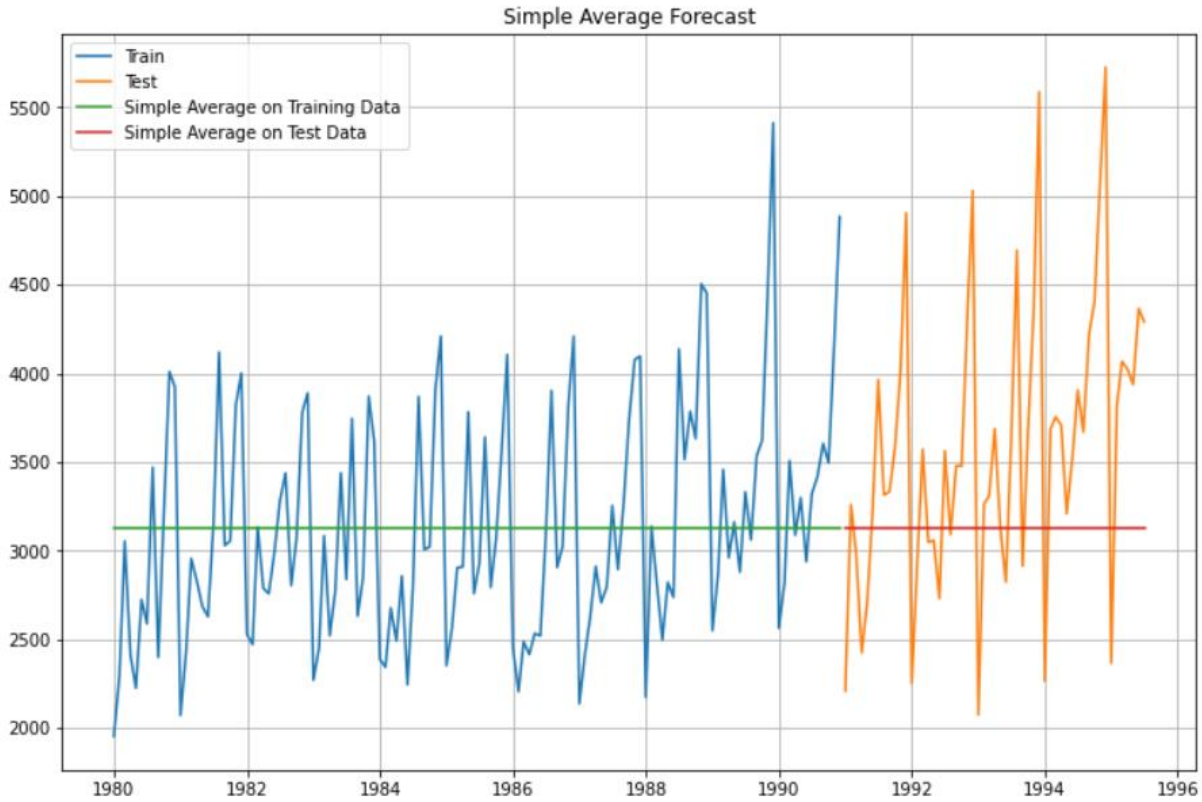


Figure 16 Simple Average Train – Test

From the plot we are analysing the Test data:

The Orange coloured area represents the test data.

The red line represents the predicted simple average values which takes the average of the train data and cuts test curve almost near to the mean of the data.

The error is very high for predicted values in the test data.

We cannot predict the trend in Simple Average model also.

Simple Average forecast on the Test Data:

RMSE is 934.353

Model4: Moving Average Model

The moving average is a statistical method used for forecasting the long-term trends.

We take an average of a set of numbers in a given range while moving the range.

The moving average method is used with time-series data to smooth out short-term fluctuations and long-term trends.

We are rolling means or moving averages for different intervals.

The best interval can be determined by the maximum accuracy or the minimum error.

YearMonth	SoftDrinkProduction	Trailing_2	Trailing_4	Trailing_6	Trailing_9
1980-01-01	1954	NaN	NaN	NaN	NaN
1980-02-01	2302	2128.0	NaN	NaN	NaN
1980-03-01	3054	2678.0	NaN	NaN	NaN
1980-04-01	2414	2734.0	2431.0	NaN	NaN
1980-05-01	2226	2320.0	2499.0	NaN	NaN

While performing rolling windows we will get NaN values, so we have to be careful.

The window of the moving average is need to be carefully selected. Because too big a window will result in not having any test set as the whole series might get averaged over.

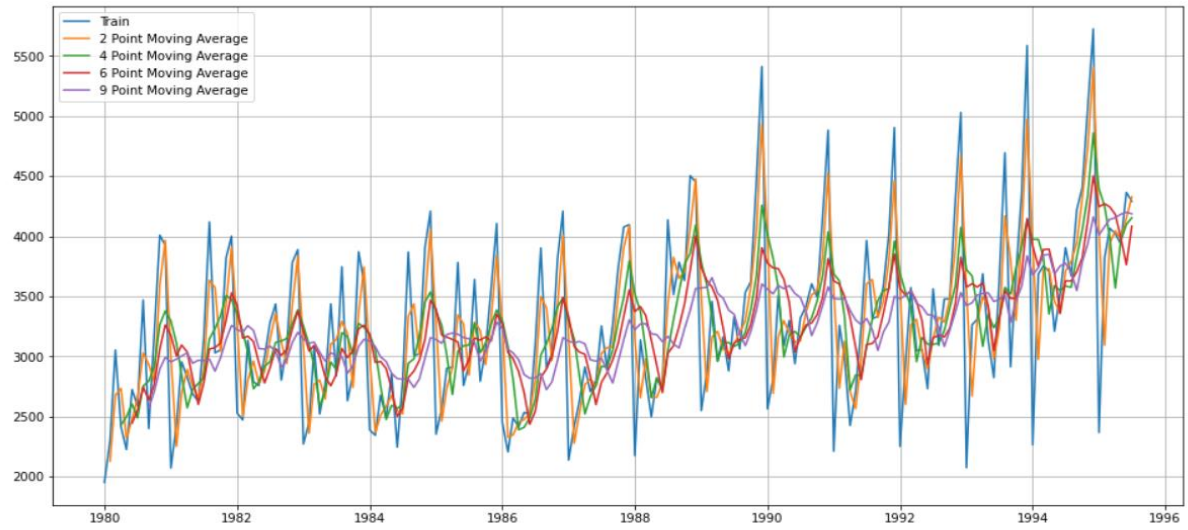


Figure 17 Simple Average model plot

From the above Moving Average plot, 2 point moving average is almost tracing the original train data. Hence, it'll have less error. when we are increasing the rolling window value like 4 point moving average it got smoothened a bit compared to the 2point moving average. when we observe 9 point moving average plot the fluctuations are smoothed much compared to 2point, 4point, 6 point moving averages. so, as per the increasing year the smoothing would be higher.

YearMonth	SoftDrinkProduction	Trailing_2	Trailing_4	Trailing_6	Trailing_9
1980-01-01	1954	NaN	NaN	NaN	NaN
1980-02-01	2302	2128.0	NaN	NaN	NaN
1980-03-01	3054	2678.0	NaN	NaN	NaN
1980-04-01	2414	2734.0	2431.0	NaN	NaN
1980-05-01	2226	2320.0	2499.0	NaN	NaN

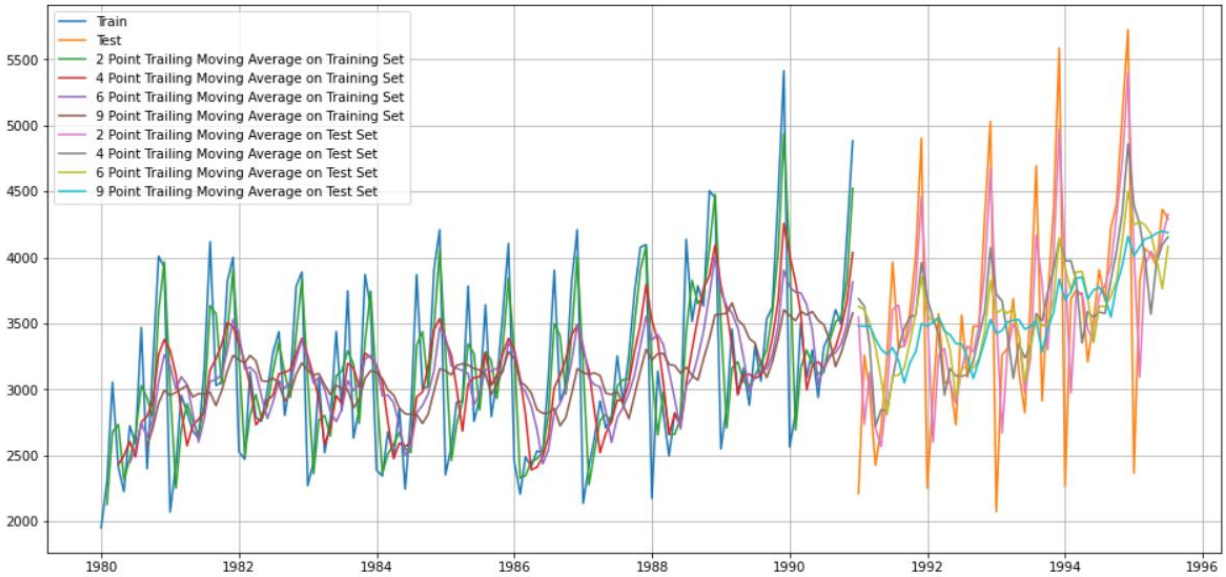


Figure 18 Simple Average Model Train- Test Plot

we can observe that the 2point moving average is almost tracing the original data and having minimum error. Hence, Lower the rolling window value less the error, higher the rolling window value higher the error.

2 point Moving Average Model forecast on the Testing Data:
RMSE is 556.725
4 point Moving Average Model forecast on the Testing Data:
RMSE is 687.182
6 point Moving Average Model forecast on the Testing Data:
RMSE is 710.514
9 point Moving Average Model forecast on the Testing Data:
RMSE is 735.890
Before we go on to build the various Exponential Smoothing models, let us look at all the models and compare the Time Series plots:

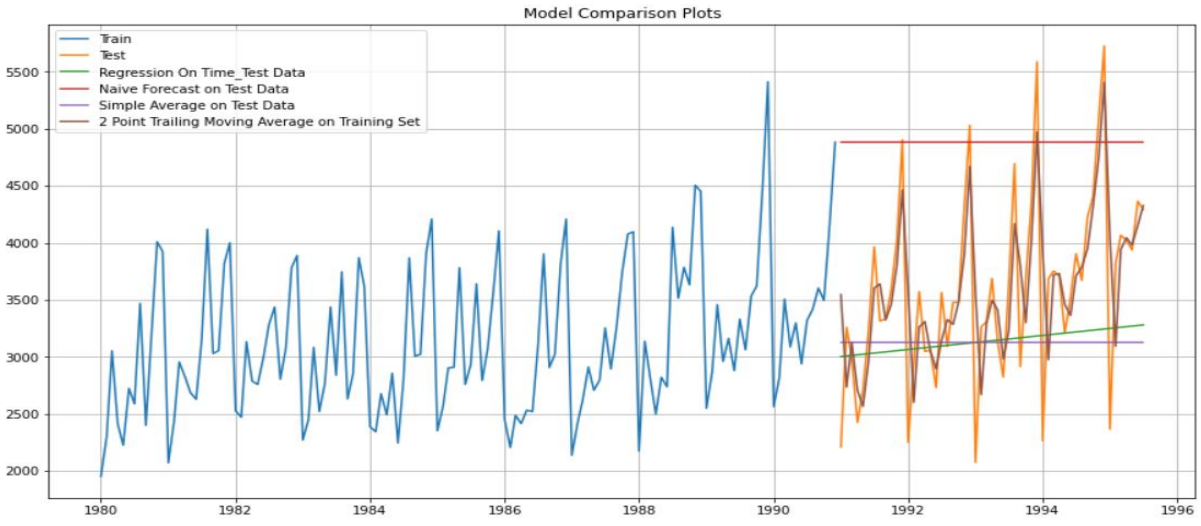


Figure 19 Model Comparison plot

From the plot, we can observe that the 2point moving average model having the least error.

Model5: Simple Exponential Smoothing (automated)

Simple Exponential Smoothing is a time series forecasting method used for univariate data without a trend or seasonality.

It requires a single parameter, called alpha (a), which is also called the smoothing factor or smoothing coefficient.

We are now fitting the model to the train and test data.

SimpleExpSmoothing Model Results

Dep. Variable:	SoftDrinkProduction	No. Observations:	132
Model:	SimpleExpSmoothing	SSE	54364308.436
Optimized:	True	AIC	1710.551
Trend:	None	BIC	1716.317
Seasonal:	None	AICC	1710.866
Seasonal Periods:	None	Date:	Sun, 03 Apr 2022
Box-Cox:	False	Time:	01:43:16
Box-Cox Coeff.:	None		

	coeff	code	optimized
smoothing_level	0.2162886	alpha	True
initial_level	2297.4229	l.0	True

```
{'smoothing_level': 0.2162885602609007,
 'smoothing_trend': nan,
 'smoothing_seasonal': nan,
 'damping_trend': nan,
 'initial_level': 2297.422897653051,
 'initial_trend': nan,
 'initial_seasons': array([], dtype=float64),
 'use_boxcox': False,
 'lamda': None,
 'remove_bias': False}
```

we get the optimised parameters by using the autofit. smoothing level is alpha, smoothing trend is beta and smoothing seasonal is gamma.

we don’t consider the trend and seasonal it is showing as nan because we are now using the simple exponential smoothing.

It requires a single parameter, called alpha (a), which is also called the smoothing factor or smoothing coefficient.

For prediction, we need to use autofit.forecast for the steps of length of the test data. So that we can produce as many forecasted values as the length of the test data.

Prediction on train-test:

YearMonth	SoftDrinkProduction	predict
1980-01-01	1954	2297.422898
1980-02-01	2302	2223.144454
1980-03-01	3054	2240.200006
1980-04-01	2414	2416.215635
1980-05-01	2226	2415.736419

YearMonth	SoftDrinkProduction	predict
1991-01-01	2211	3853.781071
1991-02-01	3260	3853.781071
1991-03-01	2992	3853.781071
1991-04-01	2425	3853.781071
1991-05-01	2707	3853.781071

For prediction, we need to use autofit.forecast for the steps of length of the test data so that we can produce as many forecasted values as the length of the test data.

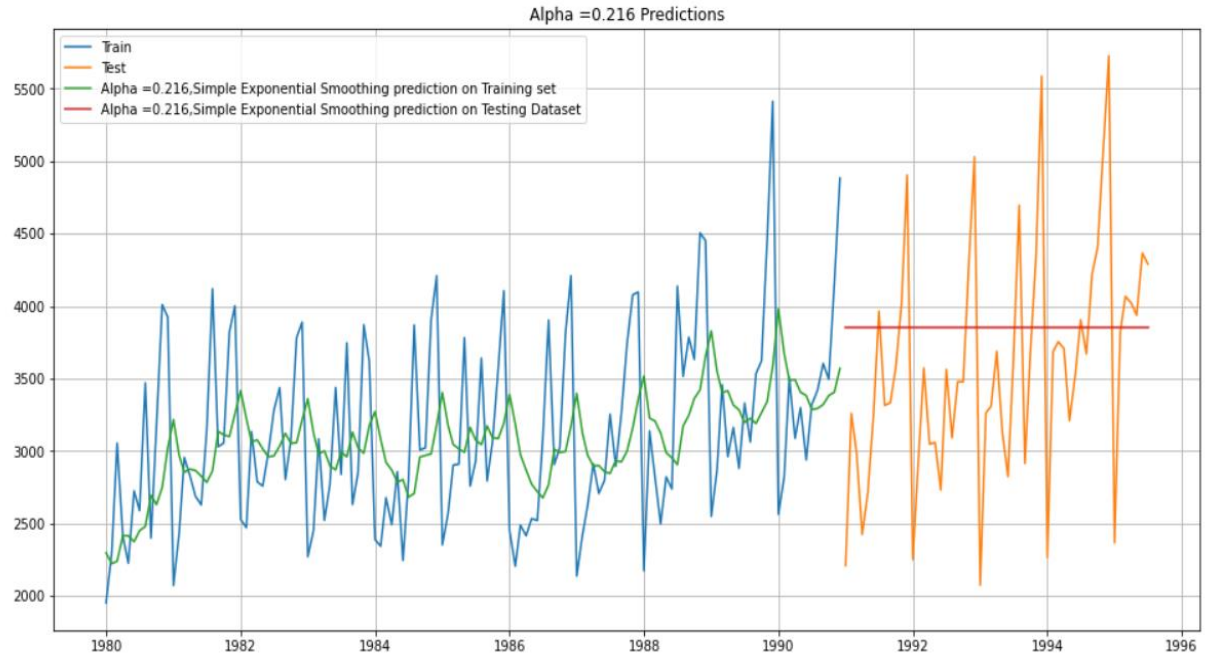


Figure 20 Simple Exponential Smoothing Train - Test Plot

Alpha =0.216 Simple Exponential Smoothing Model forecast on the Training Data,

RMSE is 847.635

Setting different alpha values:

Note: The higher the alpha value more weightage is given to the more recent observation. That means, what happened recently will happen again.

Model6: Simple Exponential Model with alpha in range of 0.1 to 0.1

we are manually giving alpha values in a range and while fitting the model range is given for smoothing level.

Hence, we are not asking model to give us optimized value. we keep it as false and we us brute force method.

while performing metrics we keep squared as false since by default it takes as true and wil give us MSE value instead of RMSE as we are using mean squared error.

	Alpha Values	Test RMSE	Train RMSE
0	0.1	807.346881	629.183163
1	0.2	838.357158	639.434755
2	0.3	910.187416	648.029705
3	0.4	1005.179377	655.598855
4	0.5	1105.985227	664.145354
5	0.6	1203.565956	674.666863
6	0.7	1294.680933	687.232192
7	0.8	1378.198740	701.529689
8	0.9	1453.359494	717.278420

After sorting the values with respect to test RMSE, I'm choosing alpha with 0.1 since it is having low **RMSE 807.346881.**

SimpleExpSmoothing Model Results

Dep. Variable:	SoftDrinkProduction	No. Observations:	132
Model:	SimpleExpSmoothing	SSE	52255031.765
Optimized:	True	AIC	1705.328
Trend:	None	BIC	1711.093
Seasonal:	None	AICC	1705.642
Seasonal Periods:	None	Date:	Sun, 03 Apr 2022
Box-Cox:	False	Time:	01:45:41
Box-Cox Coeff.:	None		

	coeff	code	optimized
smoothing_level	0.1000000	alpha	False
initial_level	2746.0946	l.0	True

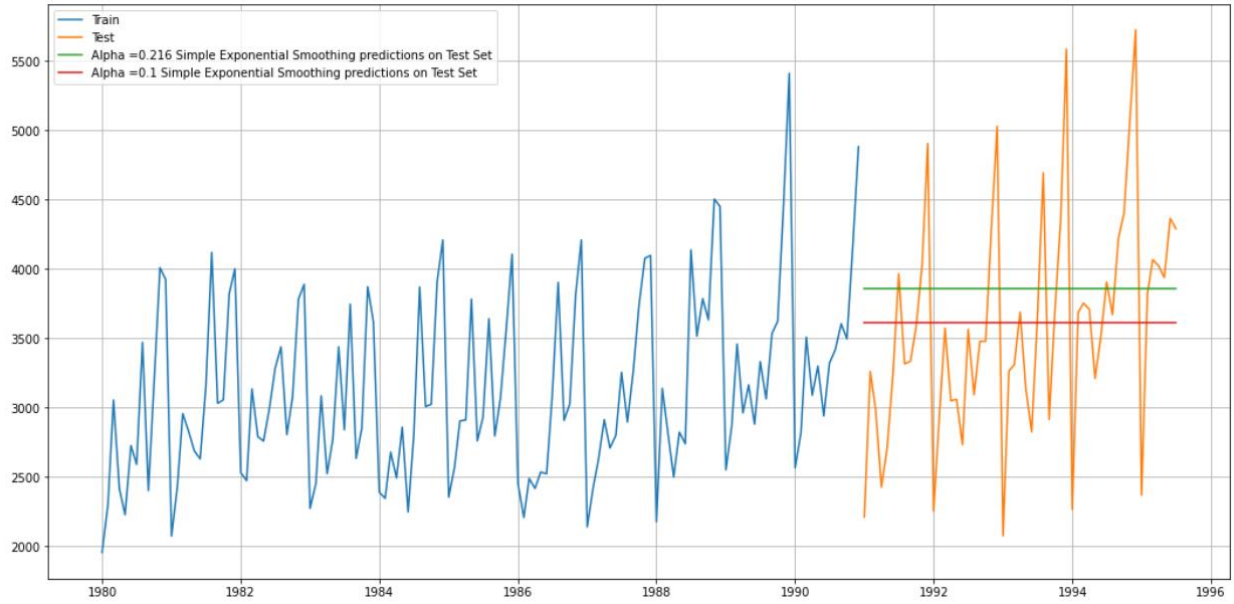


Figure 21 SES train-test plot

alpha = 0.1 having less error but it is not having any trend or seasonality

Model7: Double Exponential Smoothing

Double Exponential Smoothing is an extension to Exponential Smoothing that explicitly adds support for trends in the univariate time series.

Including the alpha parameter for controlling smoothing factor for the level, an additional smoothing factor is added to control the decay of the influence of the change in trend called beta.

Two parameters alpha and beta are estimated in this model. Level and Trend are accounted for in this model.

we are using Holt method where we consider smoothing level and trend but not seasonality that is alpha and beta respectively.

Holt Model Results

Dep. Variable:	SoftDrinkProduction	No. Observations:	132
Model:	Holt	SSE	59570554.781
Optimized:	True	AIC	1726.623
Trend:	Additive	BIC	1738.154
Seasonal:	None	AICC	1727.295
Seasonal Periods:	None	Date:	Sun, 03 Apr 2022
Box-Cox:	False	Time:	01:46:14
Box-Cox Coeff.:	None		

	coeff	code	optimized
smoothing_level	0.3940653	alpha	True
smoothing_trend	0.0677053	beta	True
initial_level	2225.8990	l.0	True
initial_trend	63.127865	b.0	True

	name	param	optimized
smoothing_level	alpha	0.394065	True
smoothing_trend	beta	0.067705	True
initial_level	l.0	2225.899026	True
initial_trend	b.0	63.127865	True

Predict on train-test:

YearMonth	SoftDrinkProduction	(predict, 0.39, 0.0)
1980-01-01	1954	2289.026891
1980-02-01	2302	2211.193673
1980-03-01	3054	2303.589281
1980-04-01	2414	2675.933259
1980-05-01	2226	2642.359169

YearMonth	SoftDrinkProduction	(predict, 0.39, 0.0)
1991-01-01	2211	4252.678190
1991-02-01	3260	4306.875038
1991-03-01	2992	4361.071886
1991-04-01	2425	4415.268733
1991-05-01	2707	4469.465581

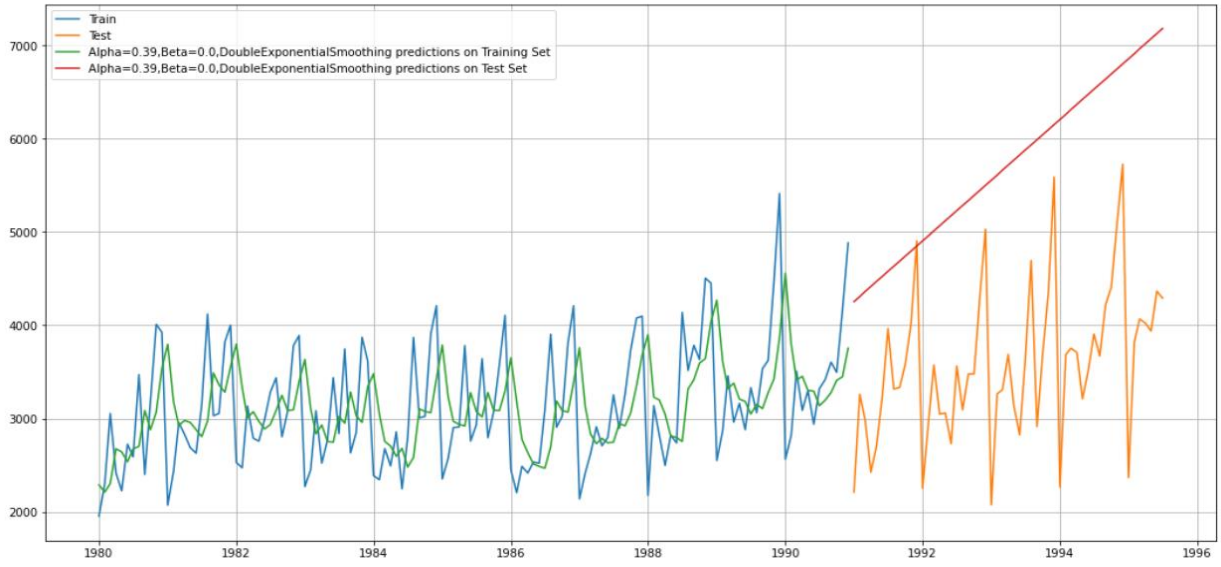


Figure 22 Double Exponential Smoothing Train- Test plot

Alpha=0.39 and Beta=0 Double Exponential Smoothing Model forecast on the Testing Data, **RMSE is 2301.543**

Model8: Simple Exponential Model with alpha in range of 0.1 to 0.1

Here we create an empty data frame to store the alpha, beta values for train and test data.

After that we use a loop with a specific range to get the alpha beta values on both train and test data.

After we get the data into a data frame, we sort the data in the ascending order of RMSE values.

	Alpha Values	Beta Values	Train RMSE	Test RMSE
0	0.1	0.1	636.982831	985.423307
1	0.1	0.2	649.470017	1084.073803
8	0.2	0.1	657.308598	1524.527761
2	0.1	0.3	660.310399	1715.626059
16	0.3	0.1	671.320234	2306.003980
...	...	...	...	...
62	0.8	0.7	922.464802	20984.112822
55	0.7	0.8	912.492050	21620.121482
70	0.9	0.7	964.699606	21623.751079
63	0.8	0.8	959.389040	22747.464839
71	0.9	0.8	1005.454300	23239.543208

72 rows × 4 columns

After sorting values I'm choosing alpha = 0.1 and beta = 0.1 since it is having low test RMSE.

```
                              Holt Model Results
================================================================================
Dep. Variable:     SoftDrinkProduction   No. Observations:                  132
Model:                            Holt   SSE                       53558620.762
Optimized:                        True   AIC                           1712.580
Trend:                       Additive    BIC                           1724.111
Seasonal:                        None    AICC                          1713.252
Seasonal Periods:                None    Date:                Sun, 03 Apr 2022
Box-Cox:                        False    Time:                        01:47:10
Box-Cox Coeff.:                  None
================================================================================
                         coeff                 code              optimized
--------------------------------------------------------------------------------
smoothing_level            0.1000000             alpha                 False
smoothing_trend            0.1000000              beta                 False
initial_level              2657.2260               l.0                  True
initial_trend              31.203326               b.0                  True
--------------------------------------------------------------------------------
```

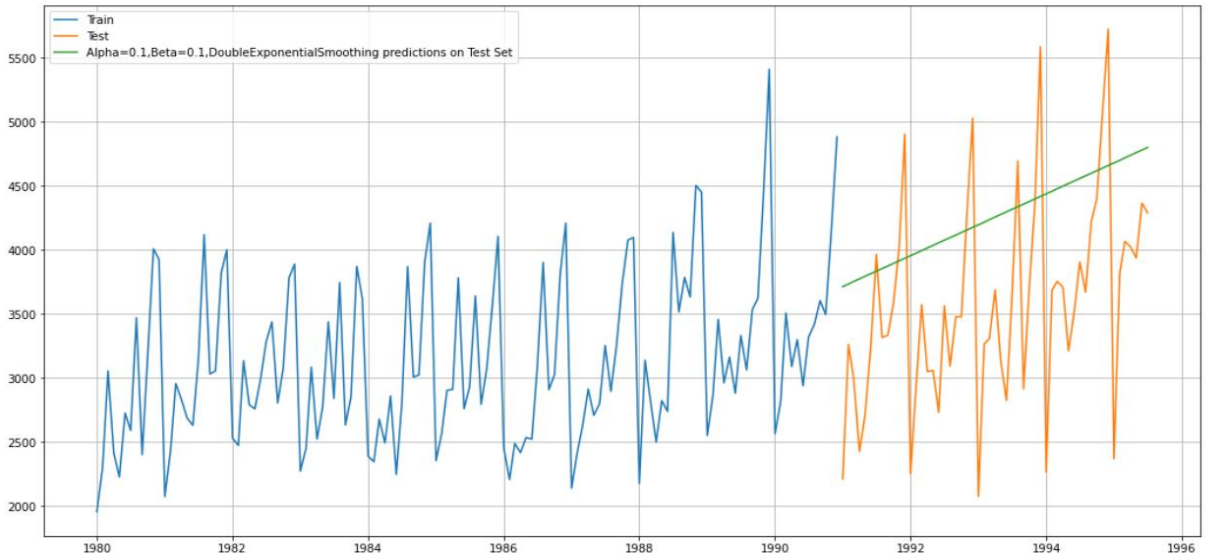


Figure 23 DES train-test plot

Model:9 Triple Exponential Smoothing

Triple Exponential Smoothing is an extension of Exponential Smoothing that explicitly adds support for seasonality to the univariate time series.

This method is also called Holt-Winters Exponential Smoothing.

In addition to the alpha and beta smoothing factors, a new parameter is added called gamma (g) that controls the influence on the seasonal component.

Parameters & Prediction on train test:

```
{'smoothing_level': 0.11102761730983592,
 'smoothing_trend': 0.04932588178949664,
 'smoothing_seasonal': 0.3949408071529521,
 'damping_trend': nan,
 'initial_level': 2312.469315120127,
 'initial_trend': 5.940363987039598,
 'initial_seasons': array([1.07863773, 1.13580187, 1.38746591, 1.24816588, 1.28452544,
       1.26879511, 1.36688932, 1.66203911, 1.29410569, 1.38551337,
       1.76694885, 1.80085756]),
 'use_boxcox': False,
 'lamda': None,
 'remove_bias': False}
```

ExponentialSmoothing Model Results

Dep. Variable:	SoftDrinkProduction	No. Observations:	132
Model:	ExponentialSmoothing	SSE	13997124.553
Optimized:	True	AIC	1559.446
Trend:	Additive	BIC	1605.571
Seasonal:	Multiplicative	AICC	1565.499
Seasonal Periods:	12	Date:	Sun, 03 Apr 2022
Box-Cox:	False	Time:	01:48:29
Box-Cox Coeff.:	None		

	coeff	code	optimized
smoothing_level	0.1110276	alpha	True
smoothing_trend	0.0493259	beta	True
smoothing_seasonal	0.3949408	gamma	True
initial_level	2312.4693	l.0	True
initial_trend	5.9403640	b.0	True
initial_seasons.0	1.0786377	s.0	True
initial_seasons.1	1.1358019	s.1	True
initial_seasons.2	1.3874659	s.2	True
initial_seasons.3	1.2481659	s.3	True
initial_seasons.4	1.2845254	s.4	True
initial_seasons.5	1.2687951	s.5	True
initial_seasons.6	1.3668893	s.6	True
initial_seasons.7	1.6620391	s.7	True
initial_seasons.8	1.2941057	s.8	True
initial_seasons.9	1.3855134	s.9	True
initial_seasons.10	1.7669488	s.10	True
initial_seasons.11	1.8008576	s.11	True

YearMonth	SoftDrinkProduction	auto_predict
1980-01-01	1954	2500.724156
1980-02-01	2302	2572.929845
1980-03-01	3054	3108.856243
1980-04-01	2414	2793.300477
1980-05-01	2226	2831.301752

YearMonth	SoftDrinkProduction	auto_predict
1991-01-01	2211	2540.621164
1991-02-01	3260	2896.115301
1991-03-01	2992	3329.155911
1991-04-01	2425	2978.160902
1991-05-01	2707	3180.844354

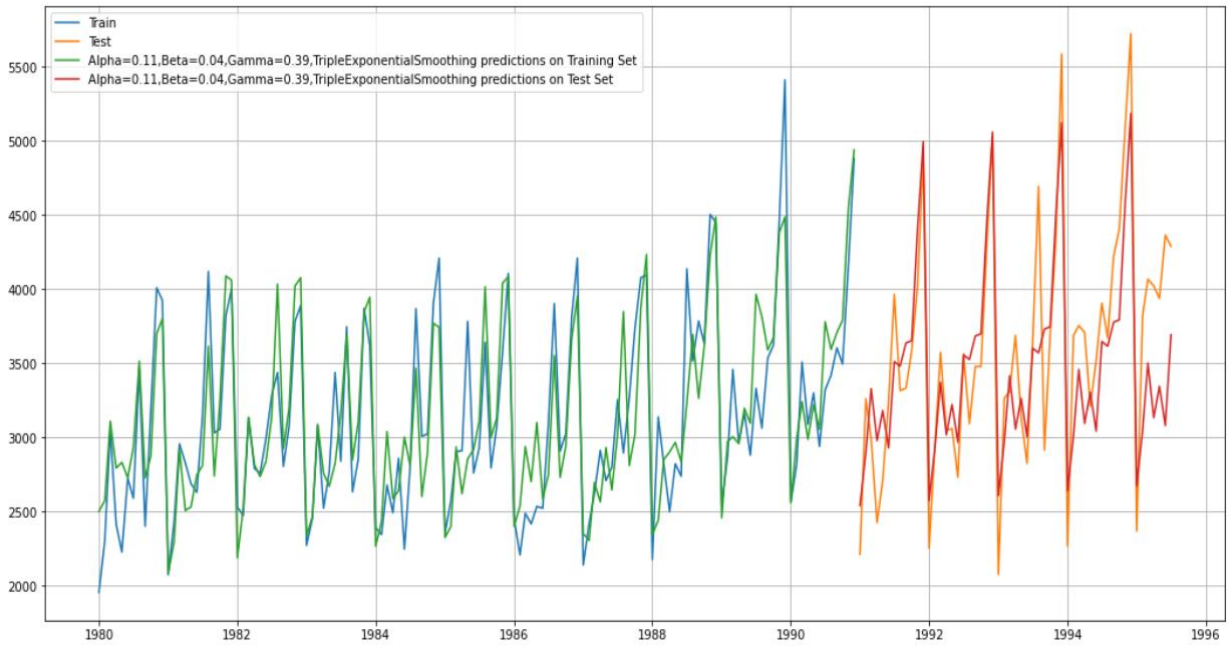


Figure 24 Triple Exponential Smoothing Train – Test

Alpha=0.11, Beta=0.04, Gamma=0.39, Triple Exponential Smoothing Model forecast on the T est Data,

RMSE is 460.438

Model10: Triple Exponential Smoothing in range 0.1 to 0.1

Here we create an empty data frame to store the alpha, beta, gamma values for train and test data.

After that we use a loop with a specific range to get the alpha beta values on both train and test data.

After we get the data into a data frame, we sort the data in the ascending order of RMSE values.

	Alpha Values	Beta Values	Gamma Values	Train RMSE	Test RMSE
211	0.4	0.4	0.4	4.131097e+02	4.273311e+02
259	0.5	0.2	0.3	3.581696e+02	4.364422e+02
315	0.6	0.1	0.3	3.526730e+02	4.411676e+02
161	0.3	0.6	0.3	3.917324e+02	4.420957e+02
379	0.7	0.1	0.4	3.698664e+02	4.645893e+02
...	...	...	...	...	...
545	0.9	0.6	0.9	4.923867e+05	1.464006e+07
550	0.9	0.7	0.7	1.058364e+07	1.591301e+07
222	0.4	0.5	0.8	2.556663e+03	1.892297e+07
425	0.7	0.7	0.8	1.679066e+03	1.986653e+07
249	0.4	0.9	0.7	4.378220e+03	2.031246e+07

567 rows × 5 columns

After sorting the values with respect to low RMSE value I'm choosing alpha = 0.4, beta = 0.4, gamma = 0.4.

```
                       ExponentialSmoothing Model Results
==============================================================================
Dep. Variable:      SoftDrinkProduction   No. Observations:                  132
Model:             ExponentialSmoothing   SSE                     22527071.319
Optimized:                         True   AIC                         1622.260
Trend:                         Additive   BIC                         1668.385
Seasonal:                Multiplicative   AICC                        1628.313
Seasonal Periods:                    12   Date:                Sun, 03 Apr 2022
Box-Cox:                          False   Time:                        01:49:44
Box-Cox Coeff.:                    None
==============================================================================
                          coeff                 code              optimized
------------------------------------------------------------------------------
smoothing_level            0.4000000                alpha                False
smoothing_trend            0.4000000                 beta                False
smoothing_seasonal         0.4000000                gamma                False
initial_level              2313.3629                  l.0                 True
initial_trend              17.604242                  b.0                 True
initial_seasons.0          1.3792667                  s.0                 True
initial_seasons.1          1.4536834                  s.1                 True
initial_seasons.2          1.7598303                  s.2                 True
initial_seasons.3          1.6682886                  s.3                 True
initial_seasons.4          1.8112700                  s.4                 True
initial_seasons.5          1.5450927                  s.5                 True
initial_seasons.6          1.5058065                  s.6                 True
initial_seasons.7          2.1812803                  s.7                 True
initial_seasons.8          1.5656242                  s.8                 True
initial_seasons.9          1.8238774                  s.9                 True
initial_seasons.10         2.3624641                 s.10                 True
initial_seasons.11         2.4076096                 s.11                 True
------------------------------------------------------------------------------
```

Plotting on both the Training and Test data using brute force alpha, beta and gamma determination.

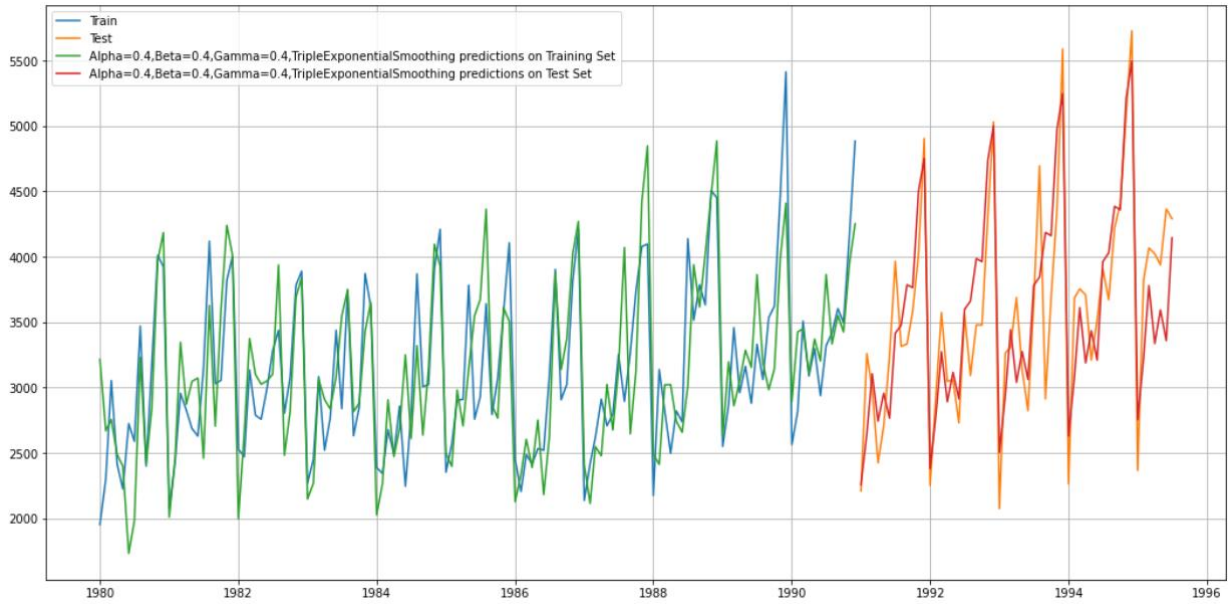


Figure 25 TES train-test plot

With the brute force method for triple exponential smoothing, we got less error and tracing same as original test data compared to triple exponential smoothing with optimized values.

```
Sorted by RMSE values on the Test Data:
```

	Test RMSE
Alpha=0.4,Beta=0.4,Gamma=0.4,TripleExponentialSmoothing	427.331067
Alpha=0.11,Beta=0.04,Gamma=0.39,TripleExponentialSmoothing	460.437728
2pointTrailingMovingAverage	556.725418
4pointTrailingMovingAverage	687.181726
6pointTrailingMovingAverage	710.513877
9pointTrailingMovingAverage	735.889827
Alpha=0.1,SimpleExponentialSmoothing	807.346881
Alpha=0.216,SimpleExponentialSmoothing	847.635259
RegressionOnTime	898.172528
SimpleAverageModel	934.353358
Alpha=0.1,Beta=0.1,DoubleExponentialSmoothing	985.423307
NaiveModel	1519.259233
Alpha=0.39 and Beta=0,DoubleExponentialSmoothing	2301.542724

We can now see that the best optimized model is the Triple Exponential Smoothing with multiplicative seasonality with the parameters. alpha = 0.4, beta = 0.4 and gamma = 0.4.

The Triple exponential model is having both the trend and seasonality, It can work better compared to the other models.

For the purpose of better understanding we are performing all the models and comparing them with respect to the RMSE values.

Before moving to the ARIMA models, let us plot all the models and compare the Time Series plots.

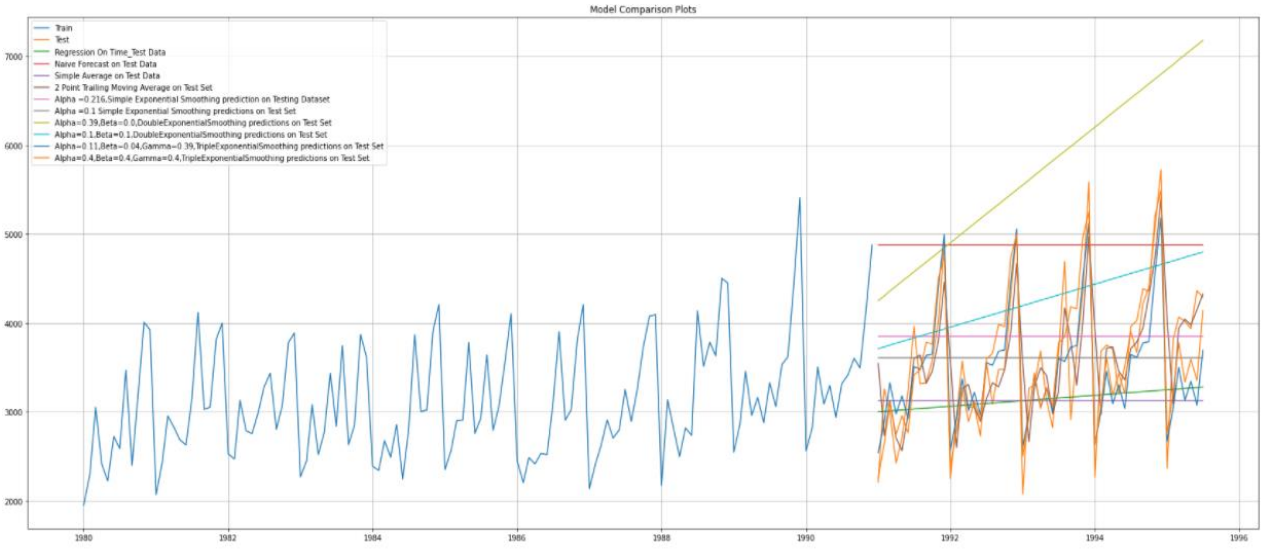


Figure 26 Model comparison plot-2

From the above plot, we can observe that Triple exponential model is best optimised as per the analysis till now, as the curve is almost tracing the original data.

5.Check for the stationarity of the data on which the model is being built on using appropriate statistical tests and also mention the hypothesis for the statistical test. If the data is found to be non-stationary, take appropriate steps to make it stationary. Check the new data for stationarity and comment.

Note: Stationarity should be checked at alpha = 0.05.

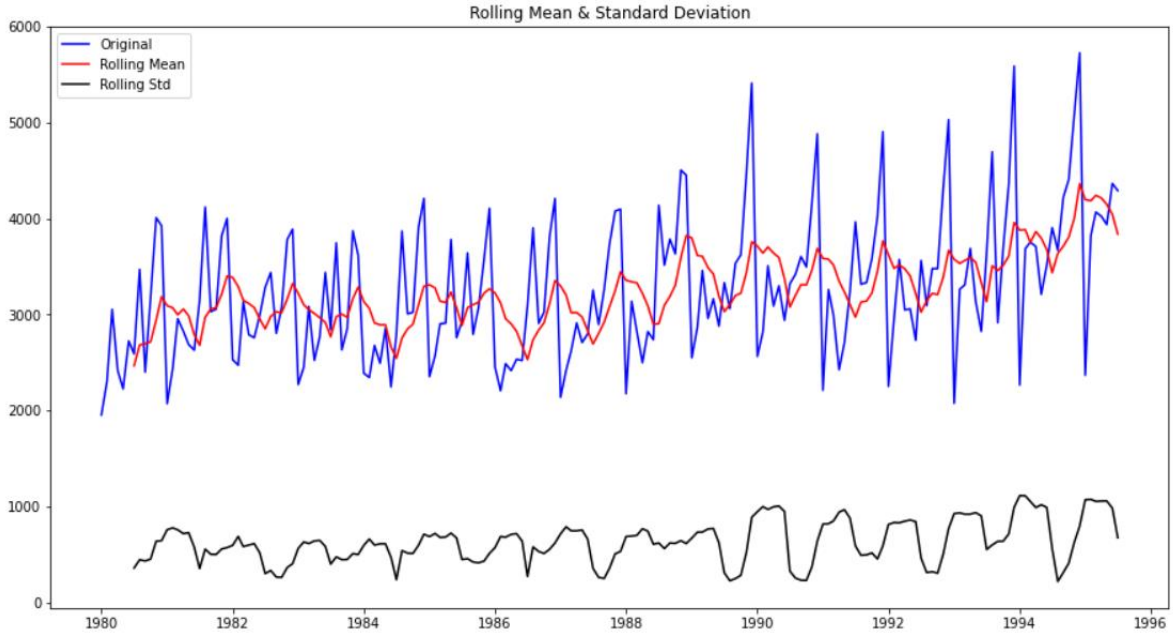
From the below plot, we can observe that p-value is greater than 0.05 which is 5% significant level.

At a 5% significant level, we can see that the time series is non stationary.

So, Let's take a difference of order 1 and check whether the Time Series is stationary or not.

The null hypothesis for ADF test (H0) is that the time series is non-stationary

The alternate hypothesis for ADF test (H1) is that time series is stationary.

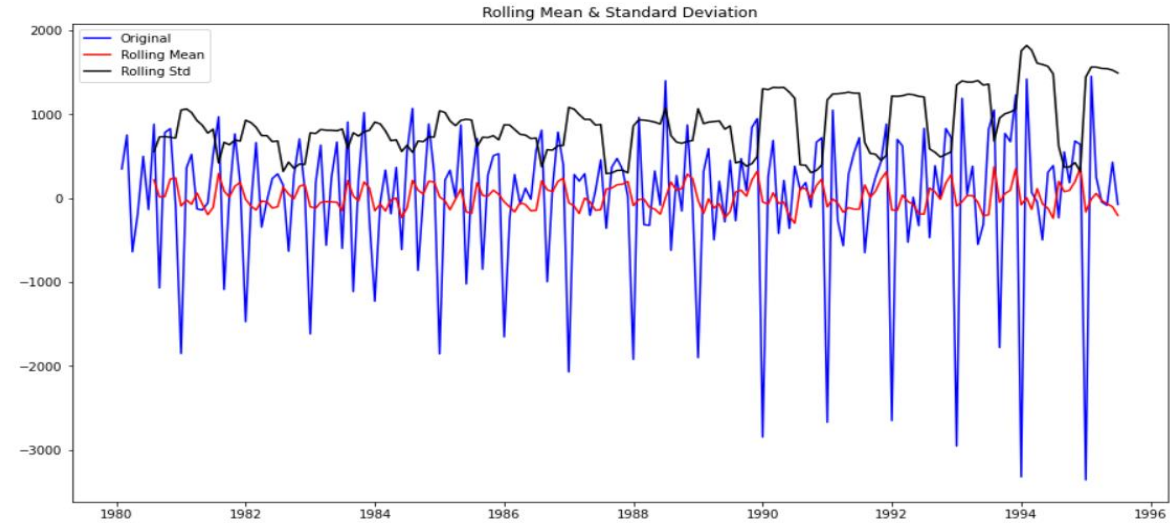



```
Results of Dickey-Fuller Test:
Test Statistic                   1.098734
p-value                          0.995206
#Lags Used                      12.000000
Number of Observations Used    174.000000
Critical Value (1%)             -3.468502
Critical Value (5%)             -2.878298
Critical Value (10%)            -2.575704
dtype: float64
```

The null hypothesis for ADF test (H0) is that the time series is non-stationary

The alternate hypothesis for ADF test (H1) is that time series is stationary

Since, the p-value of the ADF test is greater than the critical value at 5%, we cannot reject the null hypothesis. Thus, the given time given series is non stationary



```
Results of Dickey-Fuller Test:
Test Statistic                -9.313527e+00
p-value                        1.033701e-15
#Lags Used                     1.100000e+01
Number of Observations Used    1.740000e+02
Critical Value (1%)           -3.468502e+00
Critical Value (5%)           -2.878298e+00
Critical Value (10%)          -2.575704e+00
dtype: float64
```

Now, we can see that after taking a difference of order 1 and dropping null values, because when we do difference first and last we get null values.

Now, the p-value is less than 0.05, The time series is stationary. Here, we have done first order difference, the d value in ARIMA model becomes 1.

Plot the Autocorrelation and the Partial Autocorrelation function plots on the whole data.

From the ACF plot below, we can observe the auto correlation which is the shaded region represents the confidence range of that particular correlation.

The values which are going beyond this confidence range represents the significant auto correlation. we can observe the evidence of seasonality in the original series. But for now, I won't take original series since it is a non-stationary, Instead, will choose 1st order difference series. From the difference plot, we can see that, lags 0,1,6... are having the significant auto correlation we can take these values for q (AR).

Generally, we take small value of lag so that our model would be simple as we go further with big values the model becomes complex. Hence, we will find the optimum model at lower values of lags.

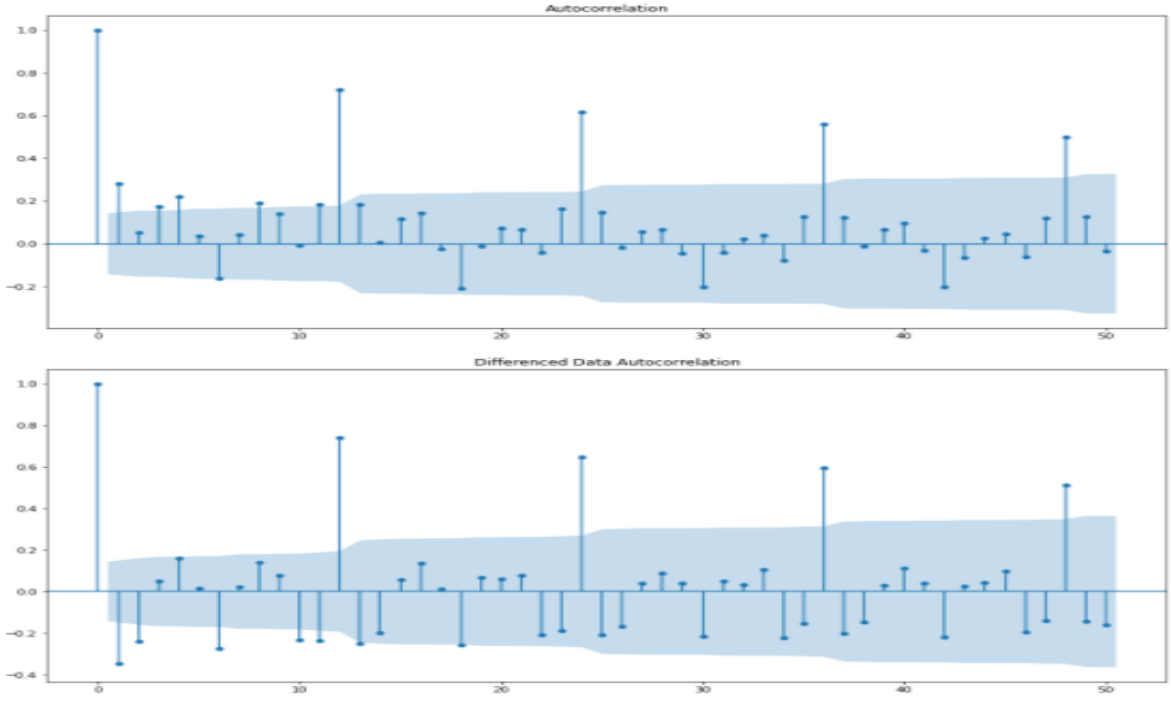


Figure 27 ACF plots

From the above PACF Plot,

The First order difference PACF (partial auto correlation) gives p (MA) value in p,d,q.

If we check for the lags which are having the significant auto correlation are 0,1,2,3,6.. we need our model to be optimum and simple. so, we need to choose the lags with low value.

For present scenario, we can choose p = 0.

The values of p,d,q becomes (0,1,0). since, it's a first order difference at d=1.

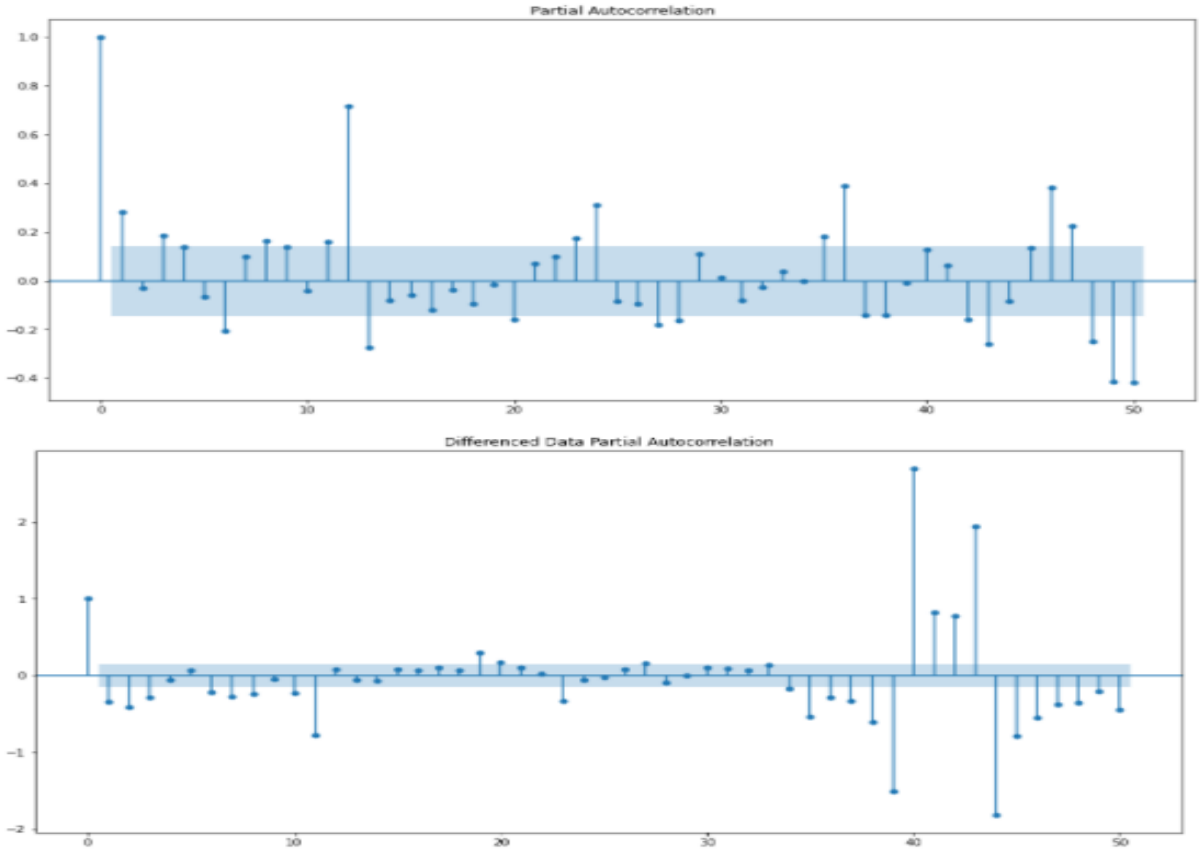
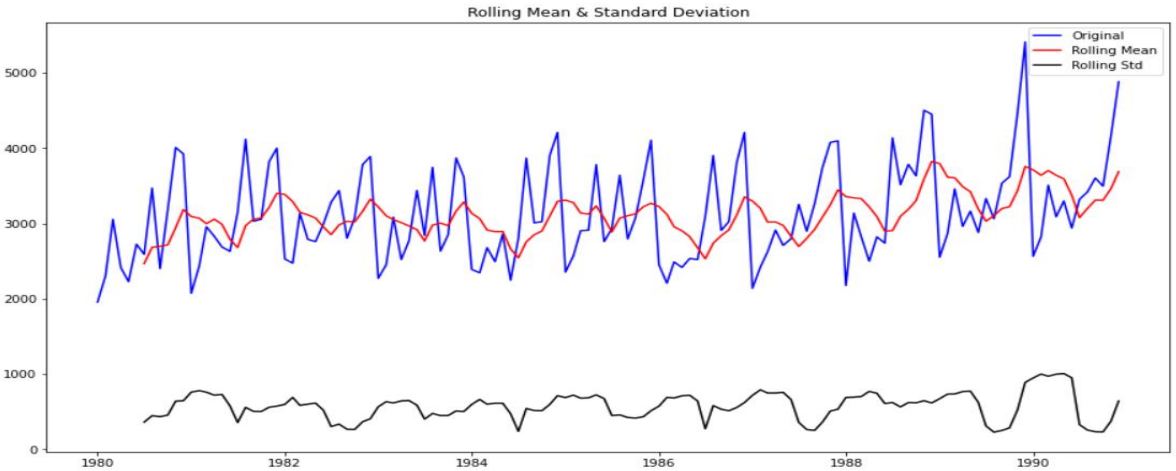


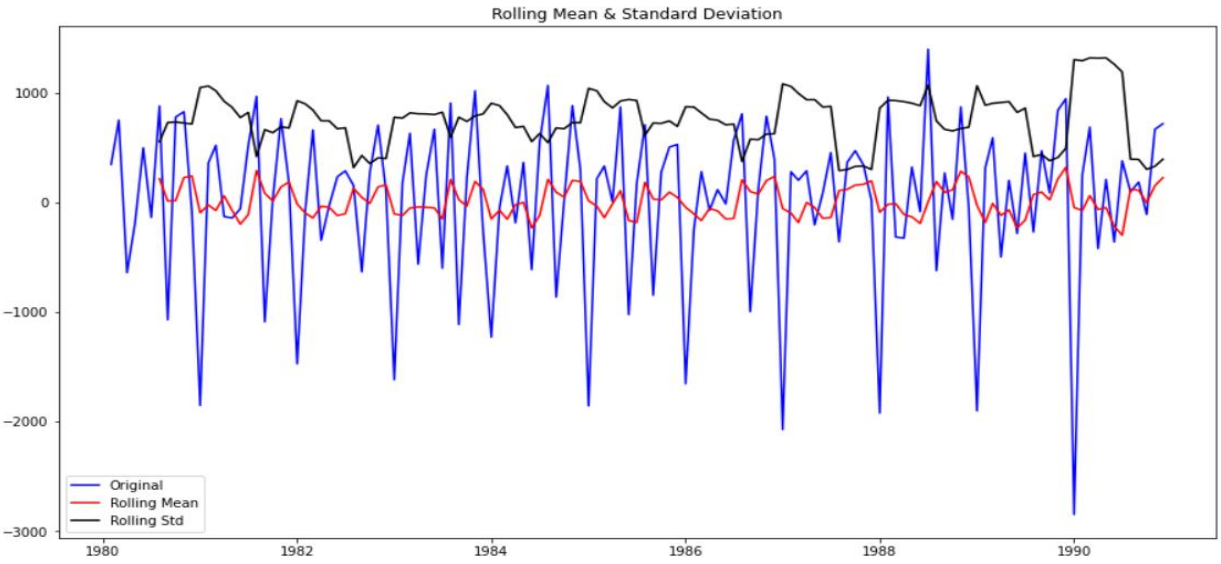
Figure 28 PACF plots

Check for stationarity of the Training Data Time Series:



```
Results of Dickey-Fuller Test:
Test Statistic                  -0.990112
p-value                          0.756854
#Lags Used                      12.000000
Number of Observations Used    119.000000
Critical Value (1%)             -3.486535
Critical Value (5%)             -2.886151
Critical Value (10%)            -2.579896
dtype: float64
```

we can observe that the series is not stationary at α = 0.05.




```
Results of Dickey-Fuller Test:
Test Statistic                -7.299886e+00
p-value                        1.347278e-10
#Lags Used                     1.100000e+01
Number of Observations Used    1.190000e+02
Critical Value (1%)           -3.486535e+00
Critical Value (5%)           -2.886151e+00
Critical Value (10%)          -2.579896e+00
dtype: float64
```

We see that after taking a difference of order 1 the series have become stationary at α = 0.05.

If the series is non-stationary, make it stationary by taking a difference of the Time Series. Then, we can use this particular differenced series to train the ARIMA models. We do not need to worry about stationarity for the Test Data because we are not building any models on the Test Data, we are evaluating our models over there. You can look at other kinds of transformations as part of making the time series stationary like taking logarithms.

6. Build an automated version of the ARIMA/SARIMA model in which the parameters are selected using the lowest Akaike Information Criteria (AIC) on the training data and evaluate this model on the test data using RMSE.

The data is having seasonality. Ideally, we should build the SARIMA model. But for demonstration purposes we are building an ARIMA model both by looking at the minimum AIC criterion and by looking at the ACF and the PACF plots.

MODEL11: AUTOMATED ARIMA BASED ON AIC CRITERIA

The parameters of the ARIMA model are defined as follows:

- **p**: The number of lag observations included in the model, also called the lag order.
- **d**: The number of times that the raw observations are differenced, also called the degree of differencing.
- **q**: The size of the moving average window, also called the order of moving average.

```
Some parameter combinations for the Model...
Model: (0, 1, 1)
Model: (0, 1, 2)
Model: (1, 1, 0)
Model: (1, 1, 1)
Model: (1, 1, 2)
Model: (2, 1, 0)
Model: (2, 1, 1)
Model: (2, 1, 2)
```

Here, we got the combination of different parameters of p and q in the range of 0 and 2. We can take the value of d as 1, because we need to take a difference of the series to make it stationary.

we are passing the values to model with the values of train data of soft drink sales and order as param in which it takes the p,d,q combinations we got from the itertools and calculate the AIC.

```
ARIMA(0, 1, 0) - AIC:2103.7338336875
ARIMA(0, 1, 1) - AIC:2069.5996302115227
ARIMA(0, 1, 2) - AIC:2056.4892632434817
ARIMA(1, 1, 0) - AIC:2097.8721216490417
ARIMA(1, 1, 1) - AIC:2061.523083927102
ARIMA(1, 1, 2) - AIC:2056.7156820689142
ARIMA(2, 1, 0) - AIC:2073.234860536023
ARIMA(2, 1, 1) - AIC:2059.1006718140716
ARIMA(2, 1, 2) - AIC:2058.712702099906
```

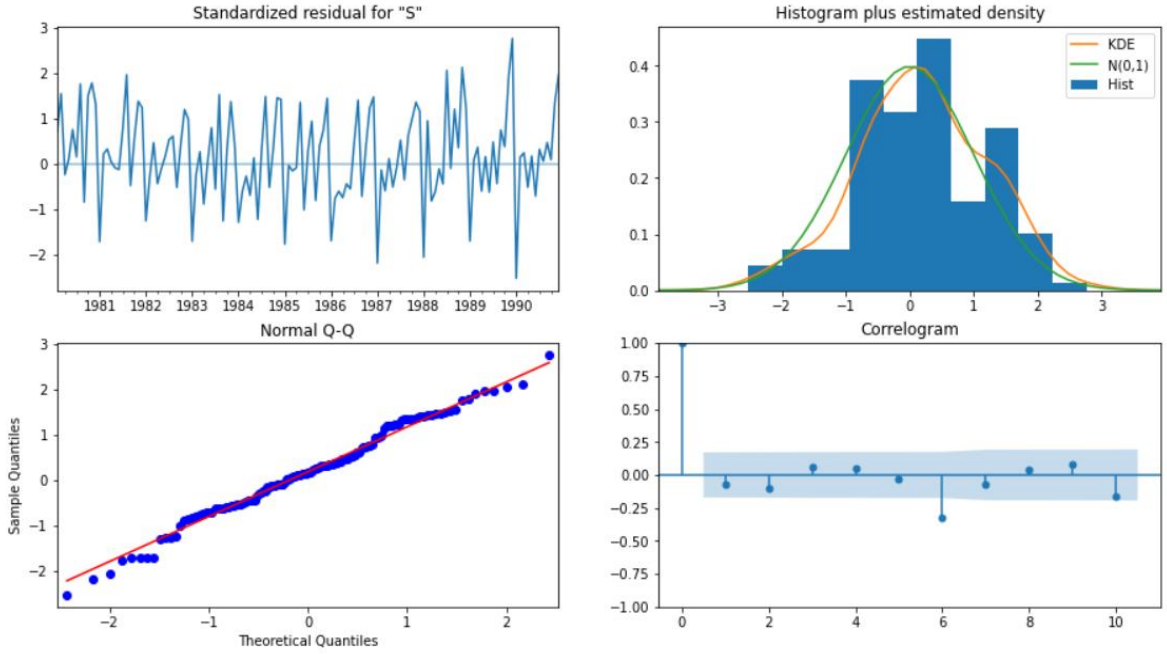
Sort the above AIC values in the ascending order to get the parameters for the minimum AIC value

	param	AIC
2	(0, 1, 2)	2056.489263
5	(1, 1, 2)	2056.715682
8	(2, 1, 2)	2058.712702
7	(2, 1, 1)	2059.100672
4	(1, 1, 1)	2061.523084
1	(0, 1, 1)	2069.599630
6	(2, 1, 0)	2073.234861
3	(1, 1, 0)	2097.872122
0	(0, 1, 0)	2103.733834

After sorting the values I'm choosing (0,1,2) with AIC = 2056.489263. we can now pass these parameters and see the statistical summary.

```
                               SARIMAX Results
==============================================================================
Dep. Variable:     SoftDrinkProduction   No. Observations:                  132
Model:                  ARIMA(0, 1, 2)   Log Likelihood               -1025.245
Date:                 Sun, 03 Apr 2022   AIC                           2056.489
Time:                         17:28:24   BIC                           2065.115
Sample:                     01-01-1980   HQIC                          2059.994
                          - 12-01-1990
Covariance Type:                   opg
==============================================================================
                 coef    std err          z      P>|z|      [0.025      0.975]
------------------------------------------------------------------------------
ma.L1         -0.5407      0.085     -6.392      0.000      -0.707      -0.375
ma.L2         -0.3913      0.113     -3.475      0.001      -0.612      -0.171
sigma2      3.572e+05   4.62e+04      7.725      0.000    2.67e+05    4.48e+05
===================================================================================
Ljung-Box (L1) (Q):                   0.61   Jarque-Bera (JB):                 0.39
Prob(Q):                              0.44   Prob(JB):                         0.82
Heteroskedasticity (H):               1.31   Skew:                            -0.13
Prob(H) (two-sided):                  0.37   Kurtosis:                         2.91
===================================================================================
```

From the summary, AIC = 2056.489 & P-value of ar and ms < 0.05.



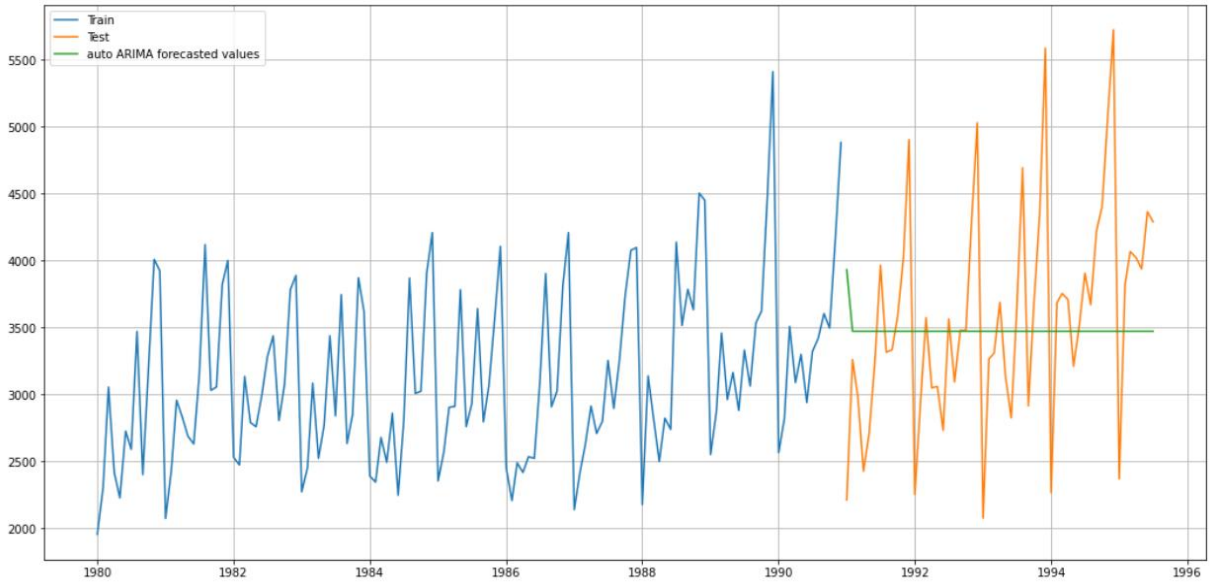


Figure 29 Autofit ARIMA plot

RMSE for the autofit ARIMA model: 831.6158500856835

Model:12 Automated SARIMA model with seasonality 6 & 12

Configuring a SARIMA requires selecting hyperparameters for both the trend and seasonal elements of the series.

→There are three trend elements that require configuration.

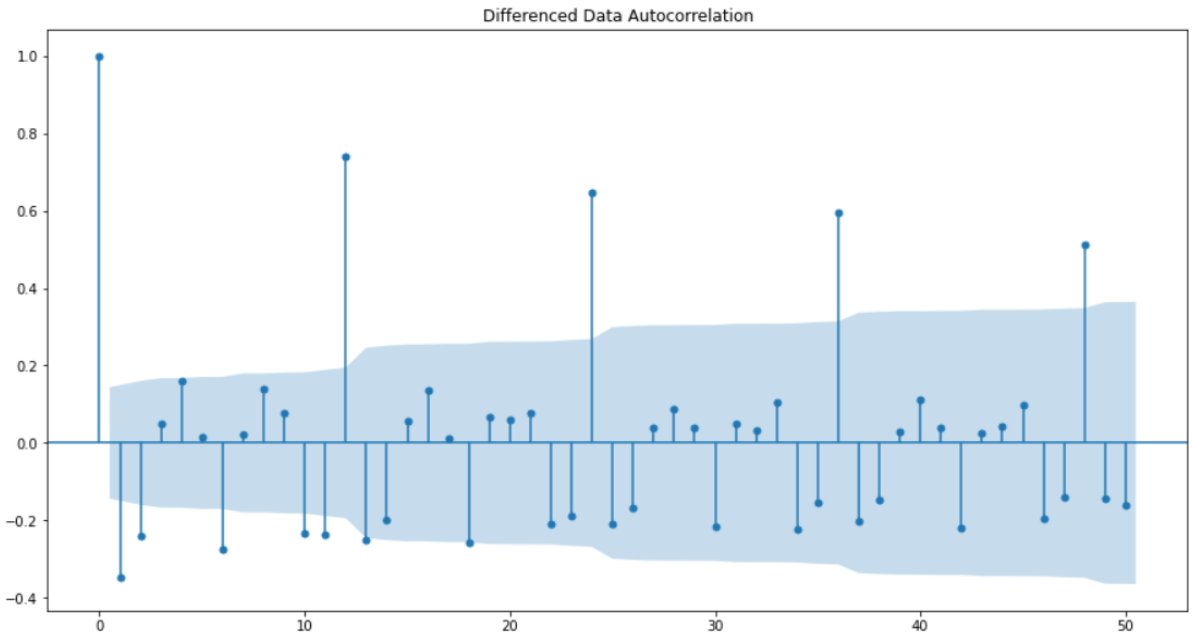
They are the same as the ARIMA model;

- **p**: Trend autoregression order.
- **d**: Trend difference order.
- **q**: Trend moving average order.

→ There are four seasonal elements that are not part of ARIMA that must be configured; they are:

- **P**: Seasonal autoregressive order.
- **D**: Seasonal difference order.
- **Q**: Seasonal moving average order.
- **m**: The number of time steps for a single seasonal period.

Let us look at the ACF plot once more to understand the seasonal parameter for the SARIMA model.



We see that there can be a seasonality of 6 as well as 12. We will run our auto SARIMA models by setting seasonality both as 6 and 12.

Setting the seasonality as 6 for the first iteration of the auto SARIMA model.

```
Examples of some parameter combinations for Model...
Model: (0, 1, 1)(0, 0, 1, 6)
Model: (0, 1, 2)(0, 0, 2, 6)
Model: (1, 1, 0)(1, 0, 0, 6)
Model: (1, 1, 1)(1, 0, 1, 6)
Model: (1, 1, 2)(1, 0, 2, 6)
Model: (2, 1, 0)(2, 0, 0, 6)
Model: (2, 1, 1)(2, 0, 1, 6)
Model: (2, 1, 2)(2, 0, 2, 6)
```

We are passing the pdq values for the param and Train values of Shoe sales to the SARIMAX and fitting the model and calculating the AIC values.

So, it is difficult to search for lowest AIC value, we sort the values with respect to AIC.

	param	seasonal	AIC
26	(0, 1, 2)	(2, 0, 2, 6)	1686.172022
53	(1, 1, 2)	(2, 0, 2, 6)	1688.105594
80	(2, 1, 2)	(2, 0, 2, 6)	1689.372224
17	(0, 1, 1)	(2, 0, 2, 6)	1698.846967
44	(1, 1, 1)	(2, 0, 2, 6)	1700.331864

After sorting, the combination (0, 1, 2) (2, 0, 2, 6) have less AIC = 1686.172022.

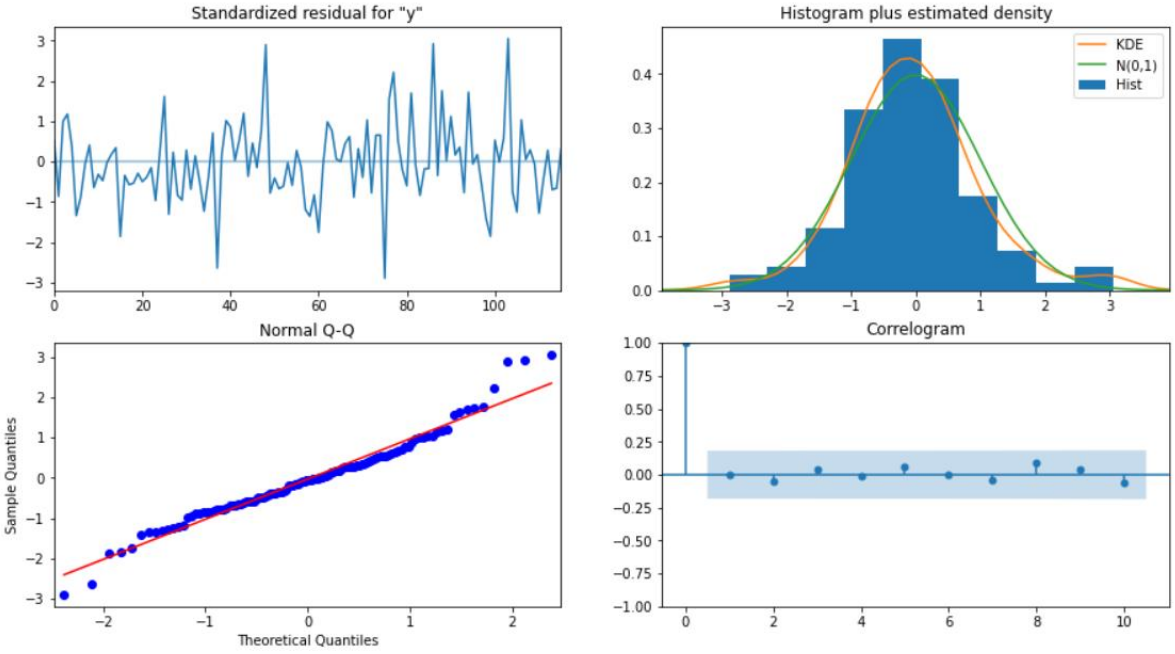
Now we will build a model using this combination of values and check the RMSE values.

SARIMAX Results

Dep. Variable:	y	No. Observations:	132
Model:	SARIMAX(0, 1, 2)x(2, 0, 2, 6)	Log Likelihood	-836.086
Date:	Sun, 03 Apr 2022	AIC	1686.172
Time:	17:34:40	BIC	1705.447
Sample:	0	HQIC	1693.997
	- 132		
Covariance Type:	opg		

	coef	std err	z	P>|z|	[0.025	0.975]
ma.L1	-0.7796	0.116	-6.725	0.000	-1.007	-0.552
ma.L2	-0.0866	0.094	-0.926	0.354	-0.270	0.097
ar.S.L6	0.0053	0.023	0.228	0.820	-0.040	0.051
ar.S.L12	0.9803	0.029	33.262	0.000	0.922	1.038
ma.S.L6	-0.1663	0.113	-1.469	0.142	-0.388	0.056
ma.S.L12	-0.6096	0.098	-6.203	0.000	-0.802	-0.417
sigma2	1.012e+05	1.11e+04	9.126	0.000	7.94e+04	1.23e+05

Ljung-Box (L1) (Q):	0.00	Jarque-Bera (JB):	11.94
Prob(Q):	0.96	Prob(JB):	0.00
Heteroskedasticity (H):	1.67	Skew:	0.41
Prob(H) (two-sided):	0.12	Kurtosis:	4.34



Predict on the Test Set using this model and evaluate the model:

	y	mean	mean_se	mean_ci_lower	mean_ci_upper
0		2716.673970	318.172396	2093.067532	3340.280407
1		3110.199714	325.816233	2471.611631	3748.787796
2		3344.806215	328.585028	2700.791394	3988.821035
3		3103.976723	331.330127	2454.581607	3753.371840
4		3290.402851	334.053454	2635.670113	3945.135589

RMSE for the autofit SARIMA model: 447.94274294699886

Setting the seasonality as 12 for the second iteration of the auto SARIMA model:

```
Examples of some parameter combinations for Model...
Model: (0, 1, 1)(0, 0, 1, 12)
Model: (0, 1, 2)(0, 0, 2, 12)
Model: (1, 1, 0)(1, 0, 0, 12)
Model: (1, 1, 1)(1, 0, 1, 12)
Model: (1, 1, 2)(1, 0, 2, 12)
Model: (2, 1, 0)(2, 0, 0, 12)
Model: (2, 1, 1)(2, 0, 1, 12)
Model: (2, 1, 2)(2, 0, 2, 12)
```

	param	seasonal	AIC
26	(0, 1, 2)	(2, 0, 2, 12)	1517.207903
23	(0, 1, 2)	(1, 0, 2, 12)	1518.229381
53	(1, 1, 2)	(2, 0, 2, 12)	1518.328978
50	(1, 1, 2)	(1, 0, 2, 12)	1519.197012
80	(2, 1, 2)	(2, 0, 2, 12)	1520.313657

After sorting, the combination (0, 1, 2) (2, 0, 2, 12) have less AIC = 1517.207903

Now we will build a model using this combination of values and check the RMSE values.

```
                                 SARIMAX Results
==========================================================================================
Dep. Variable:                SoftDrinkProduction   No. Observations:                  132
Model:             SARIMAX(0, 1, 2)x(2, 0, 2, 12)   Log Likelihood                -751.604
Date:                            Sun, 03 Apr 2022   AIC                            1517.208
Time:                                    17:39:46   BIC                            1535.719
Sample:                                01-01-1980   HQIC                           1524.707
                                     - 12-01-1990
Covariance Type:                              opg
==============================================================================
                 coef    std err          z      P>|z|      [0.025      0.975]
------------------------------------------------------------------------------
ma.L1         -0.9981      0.141     -7.092      0.000      -1.274      -0.722
ma.L2         -0.1064      0.121     -0.879      0.379      -0.344       0.131
ar.S.L12       0.6097      0.437      1.396      0.163      -0.246       1.466
ar.S.L24       0.3745      0.438      0.855      0.392      -0.484       1.233
ma.S.L12      -0.2179      0.438     -0.498      0.619      -1.076       0.640
ma.S.L24      -0.2300      0.273     -0.841      0.400      -0.766       0.306
sigma2      9.002e+04    1.8e+04      4.993      0.000    5.47e+04    1.25e+05
===================================================================================
Ljung-Box (L1) (Q):                   0.00   Jarque-Bera (JB):                12.45
Prob(Q):                              0.99   Prob(JB):                         0.00
Heteroskedasticity (H):               1.74   Skew:                             0.49
Prob(H) (two-sided):                  0.10   Kurtosis:                         4.39
===================================================================================
```

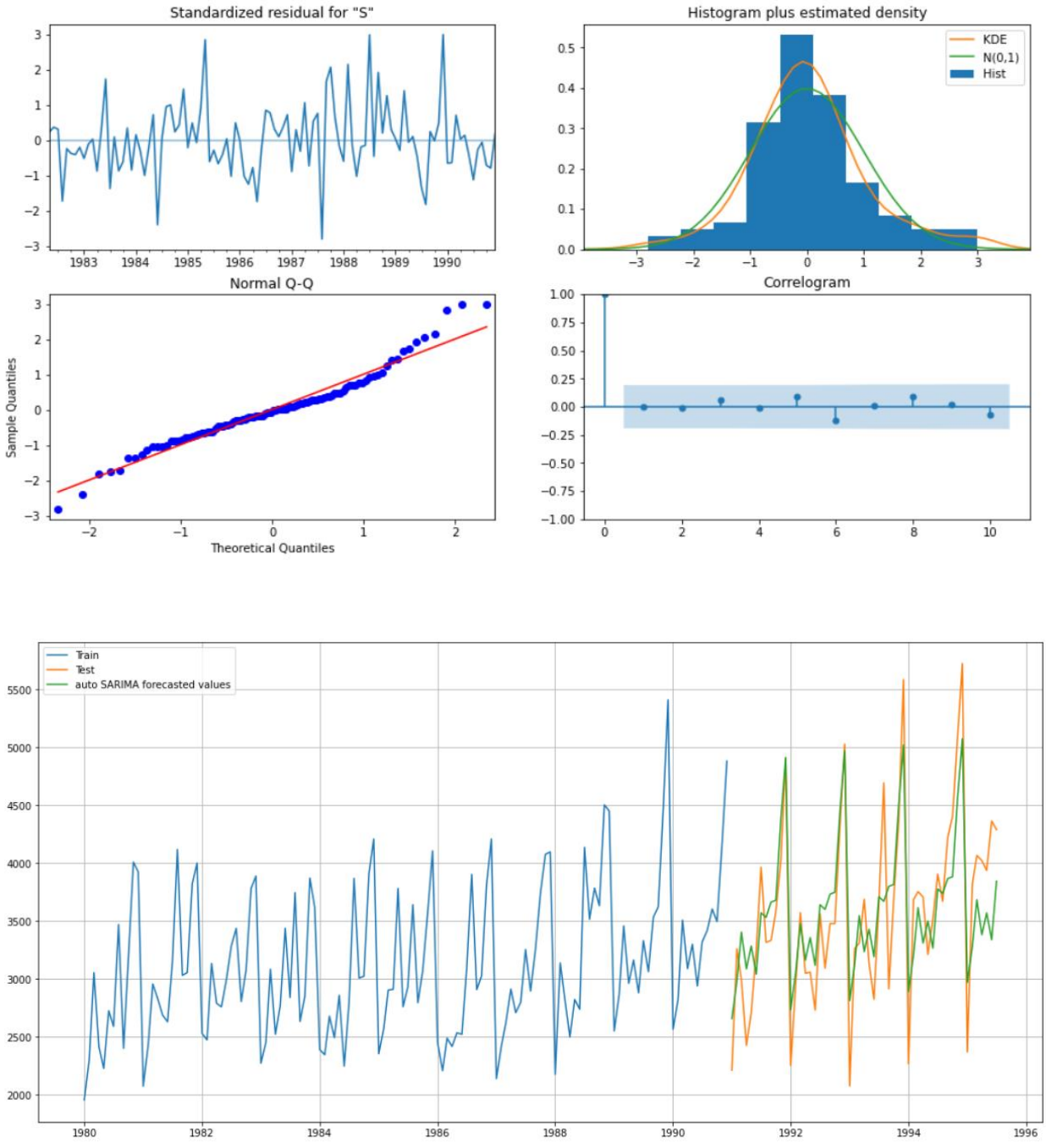


Figure 30 Autofit SARIMA

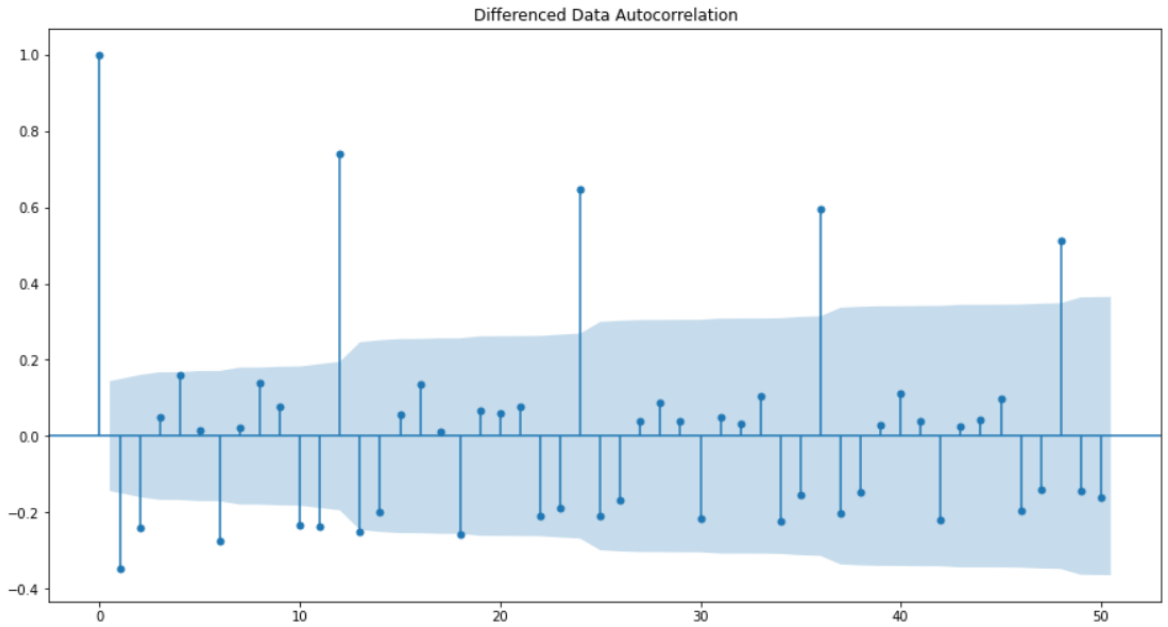
RMSE for the autofit SARIMA model: 437.70656001343895

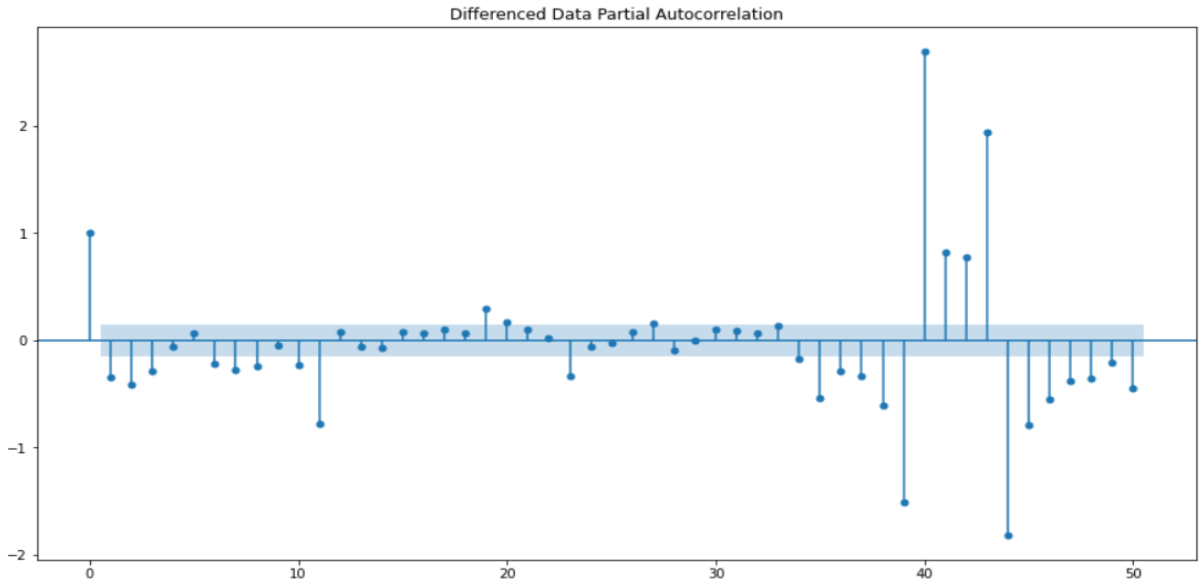
	Test RMSE
RegressionOnTime	898.172528
NaiveModel	1519.259233
SimpleAverageModel	934.353358
2pointTrailingMovingAverage	556.725418
4pointTrailingMovingAverage	687.181726
6pointTrailingMovingAverage	710.513877
9pointTrailingMovingAverage	735.889827
Alpha=0.216,SimpleExponentialSmoothing	847.635259
Alpha=0.1,SimpleExponentialSmoothing	807.346881
Alpha=0.39 and Beta=0,DoubleExponentialSmoothing	2301.542724
Alpha=0.1,Beta=0.1,DoubleExponentialSmoothing	985.423307
Alpha=0.11,Beta=0.04,Gamma=0.39,TripleExponentialSmoothing	460.437728
Alpha=0.4,Beta=0.4,Gamma=0.4,TripleExponentialSmoothing	427.331067
automated ARIMA(0,1,2)	831.615850
automatedSARIMA(0,1,2)(2,0,2,6)	447.942743
automated SARIMA(0,1,2)*(2,0,2,12)	437.706560

We can observe that the RMSE value have not reduced further when the seasonality parameter was changed to 12.

7. Build ARIMA/SARIMA models based on the cut-off points of ACF and PACF on the training data and evaluate this model on the test data using RMSE.

Model13: Manual ARIMA with cut-off values from ACF and PACF graphs





For alpha=0.05.

The Auto-Regressive parameter in an ARIMA model is 'p' from the significant lag before which the PACF plot cuts-off to 0.

The Moving-Average parameter in an ARIMA model is 'q' from the significant lag before the ACF plot cuts-off to 0.

From above plots, we can say that both the PACF and ACF plot cuts-off at lag 0.

SARIMAX Results

Dep. Variable:	SoftDrinkProduction	No. Observations:	132
Model:	ARIMA(3, 1, 2)	Log Likelihood	-1024.340
Date:	Sun, 03 Apr 2022	AIC	2060.680
Time:	17:45:17	BIC	2077.931
Sample:	01-01-1980	HQIC	2067.690
	- 12-01-1990		
Covariance Type:	opg		

	coef	std err	z	P>|z|	[0.025	0.975]
ar.L1	-0.3216	0.460	-0.700	0.484	-1.223	0.579
ar.L2	-0.0001	0.189	-0.001	0.999	-0.371	0.370
ar.L3	-0.0227	0.218	-0.104	0.917	-0.449	0.404
ma.L1	-0.2633	0.453	-0.581	0.561	-1.151	0.624
ma.L2	-0.6405	0.400	-1.599	0.110	-1.425	0.144
sigma2	3.514e+05	4.97e+04	7.066	0.000	2.54e+05	4.49e+05

Ljung-Box (L1) (Q):	0.07	Jarque-Bera (JB):	0.43
Prob(Q):	0.80	Prob(JB):	0.81
Heteroskedasticity (H):	1.29	Skew:	0.04
Prob(H) (two-sided):	0.40	Kurtosis:	2.73

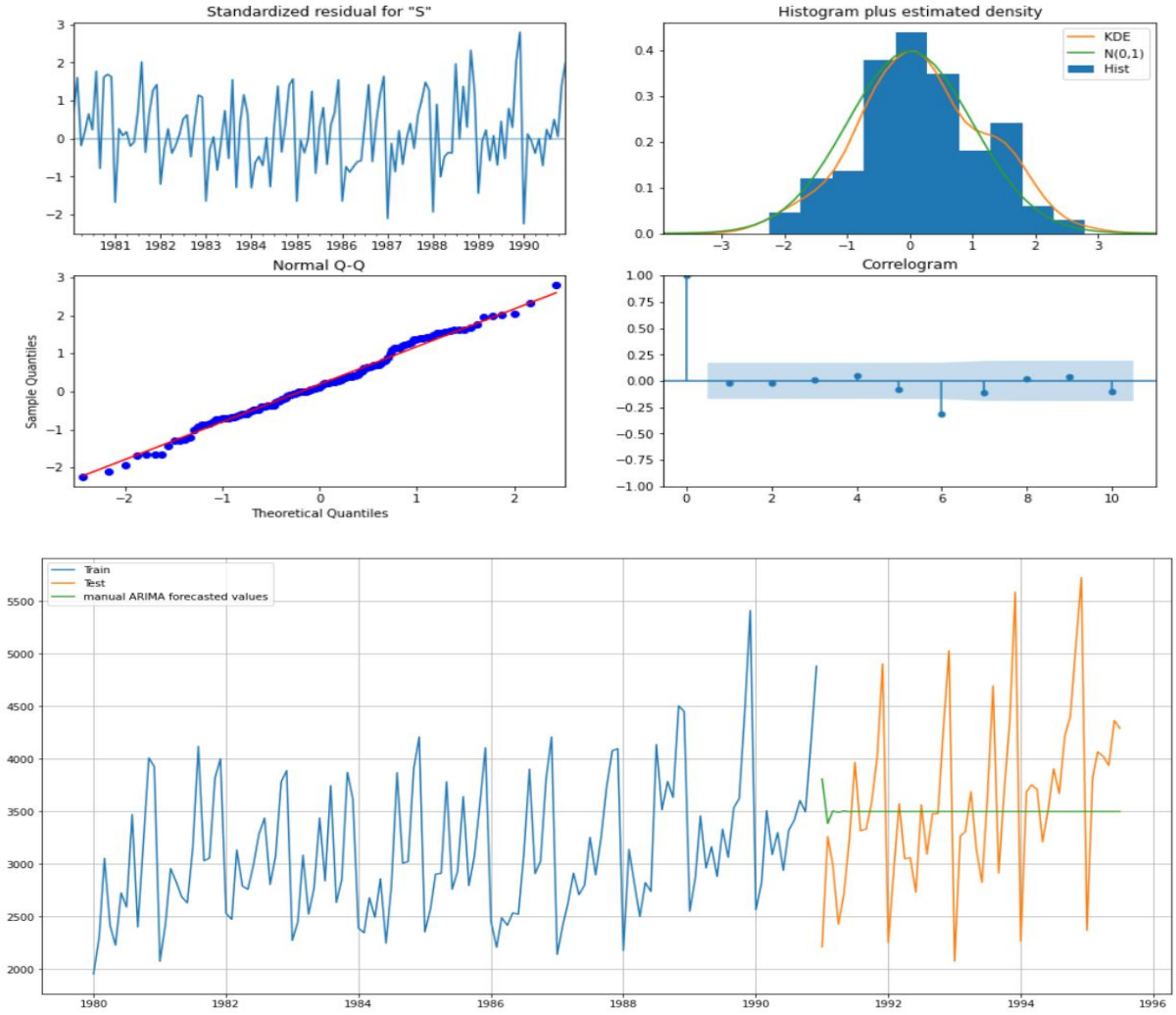
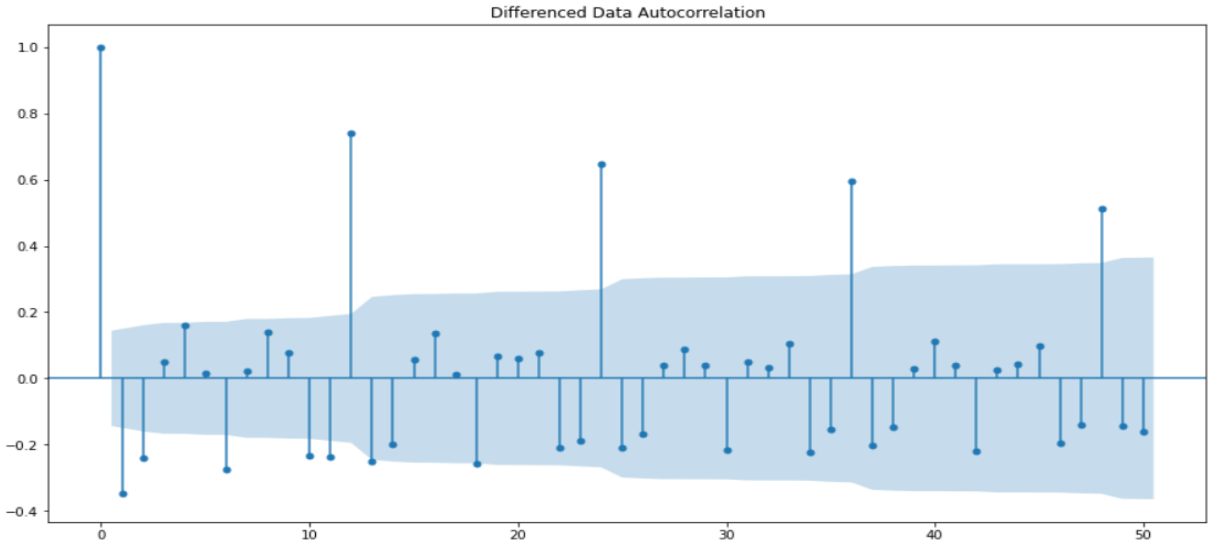
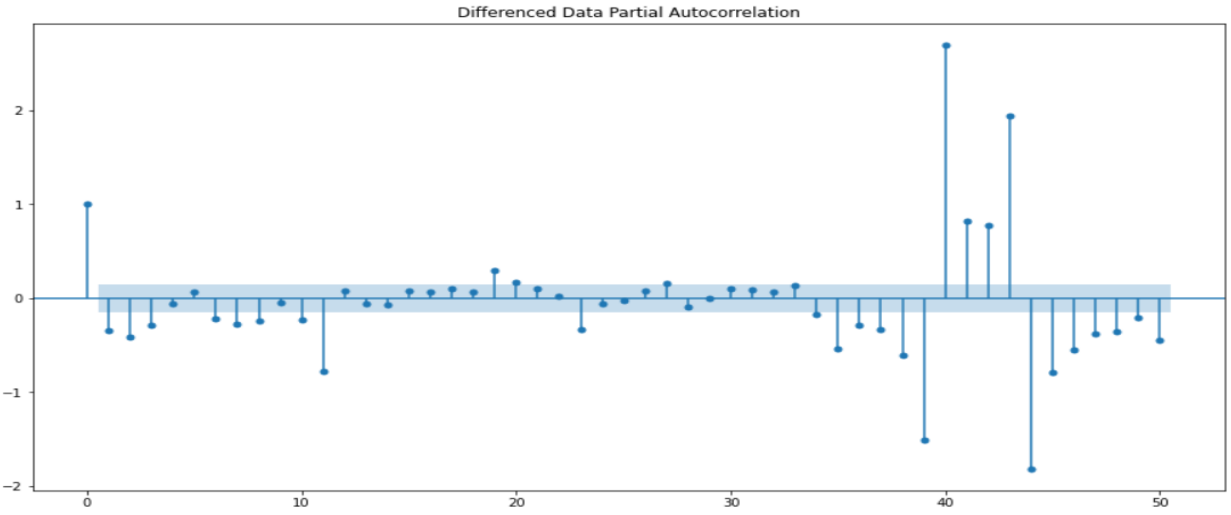


Figure 31 Manual ARIMA

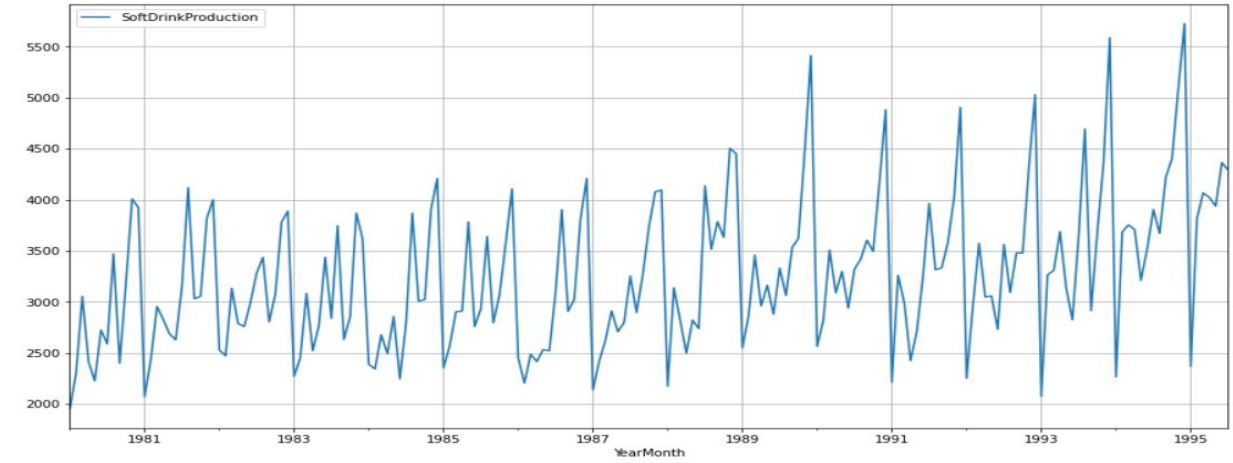
RMSE for the manual ARIMA model: 822.2174069221138

Model14: manual SARIMA

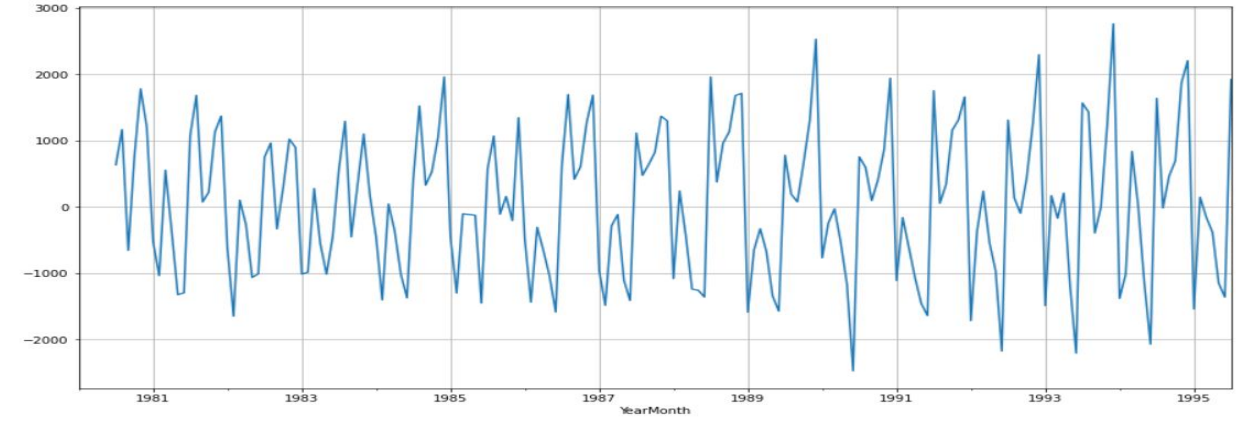




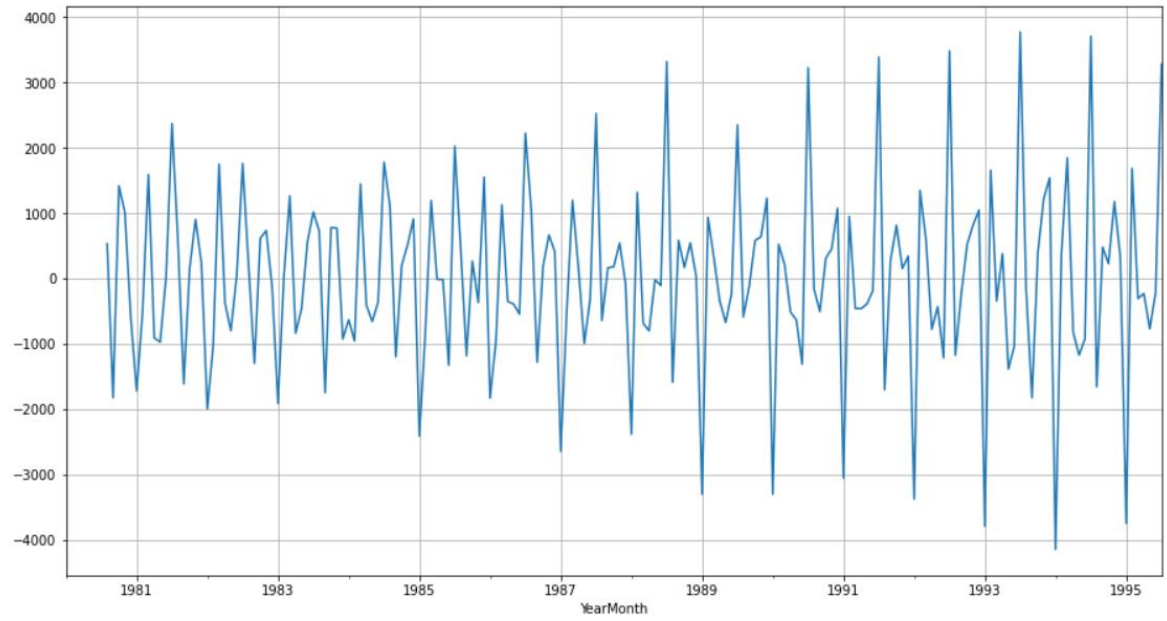
We can observe that the above ACF plot at the seasonal interval (6) does not taper off. Hence, we are taking a seasonal differencing of original series.
Before that let us look at the original series.



Here, we can observe that there is a trend and a seasonality. So, we can take a seasonal diff erencing and check the series.

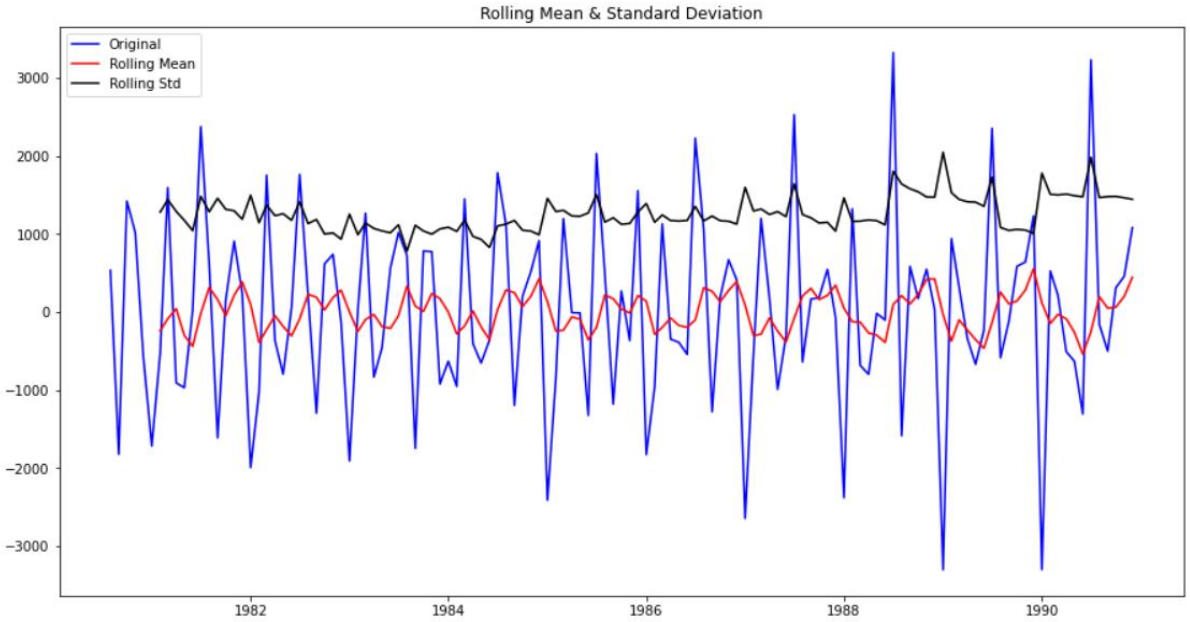


There is a slight trend which we can observe in the data. So, we take a differencing of first order on the seasonally differenced series.



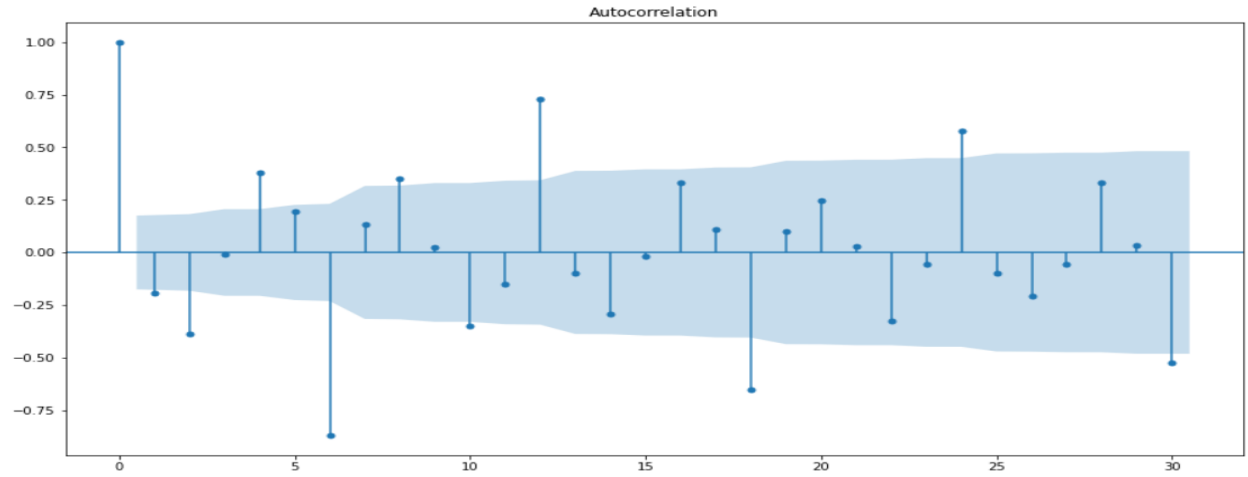
Here, There is almost no trend present in the data. Only the seasonality is present in the data.

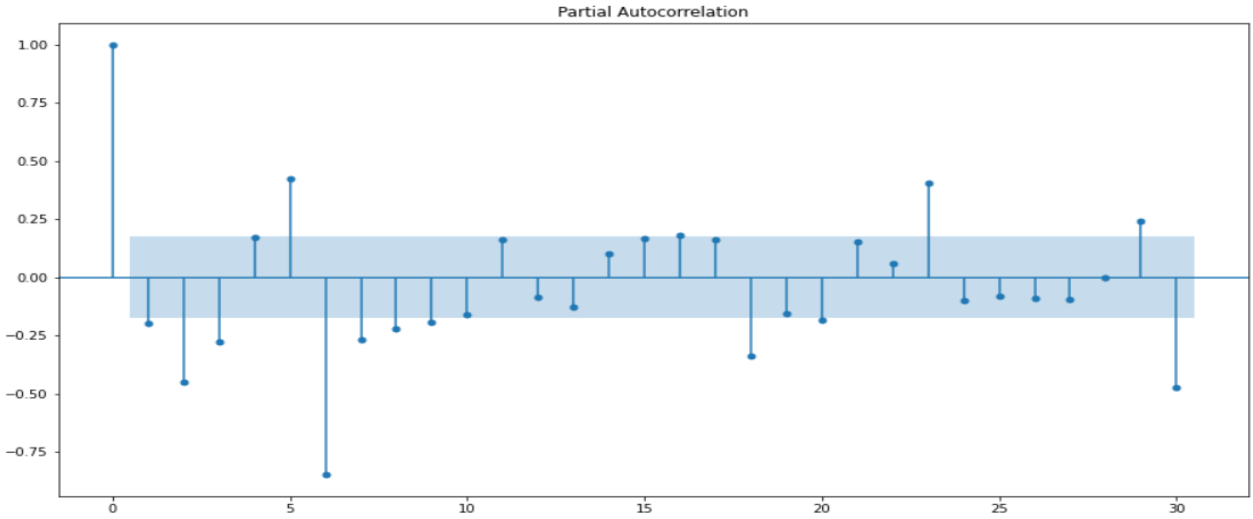
check the stationarity of the above series before fitting the SARIMA model.




```
Results of Dickey-Fuller Test:
Test Statistic                -7.167226e+00
p-value                        2.865418e-10
#Lags Used                     1.200000e+01
Number of Observations Used    1.120000e+02
Critical Value (1%)           -3.490131e+00
Critical Value (5%)           -2.887712e+00
Critical Value (10%)          -2.580730e+00
dtype: float64
```

Check the ACF and the PACF plots for the new modified Time Series.



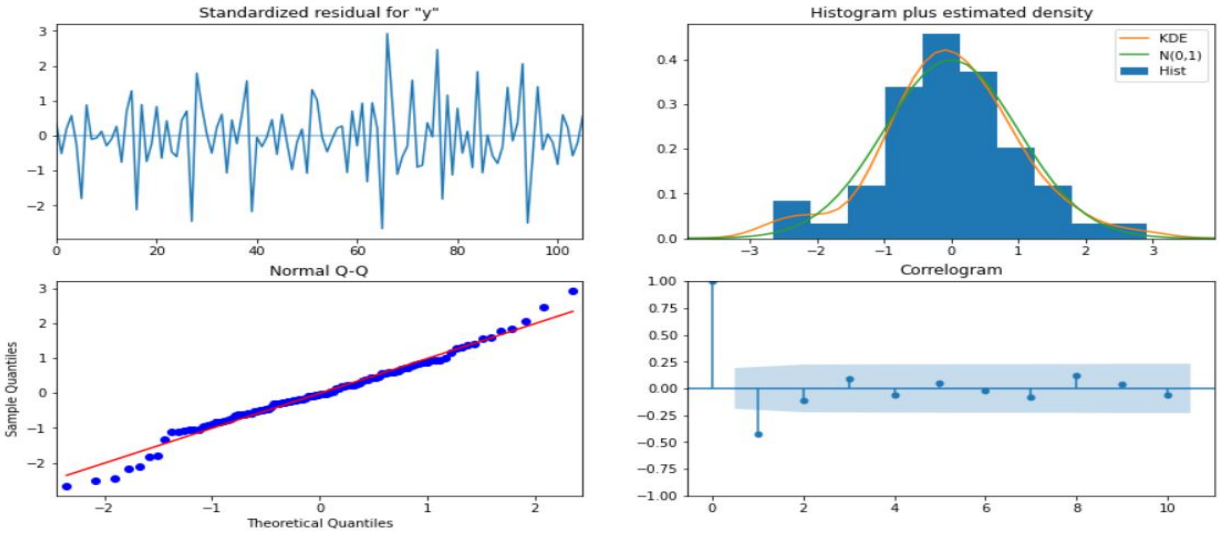


SARIMAX Results

Dep. Variable:	y	No. Observations:	132
Model:	SARIMAX(0, 1, 0)x(1, 1, [1, 2, 3], 6)	Log Likelihood	-790.890
Date:	Sun, 03 Apr 2022	AIC	1591.780
Time:	17:45:24	BIC	1605.098
Sample:	0	HQIC	1597.178
	- 132		
Covariance Type:	opg		

	coef	std err	z	P>|z|	[0.025	0.975]
ar.S.L6	-0.9773	0.026	-37.900	0.000	-1.028	-0.927
ma.S.L6	-0.1686	0.108	-1.558	0.119	-0.381	0.044
ma.S.L12	-0.5156	0.091	-5.683	0.000	-0.693	-0.338
ma.S.L18	0.0145	0.113	0.128	0.898	-0.208	0.237
sigma2	1.74e+05	2.13e+04	8.156	0.000	1.32e+05	2.16e+05

Ljung-Box (L1) (Q):	19.81	Jarque-Bera (JB):	2.23
Prob(Q):	0.00	Prob(JB):	0.33
Heteroskedasticity (H):	1.44	Skew:	-0.05
Prob(H) (two-sided):	0.28	Kurtosis:	3.70



Predict on the Test Set using this model and evaluate the model

```
array([2699.45013997, 3039.54421884, 3407.91702693, 3121.31119147,
       3348.4197238 , 3027.65243853, 3512.20526785, 3483.56750488,
       3633.52567894, 3641.76842203, 4340.22802006, 4914.05701732,
       2787.16866311, 3122.48479998, 3487.25699131, 3203.88249857,
       3440.5770243 , 3143.52129432, 3568.71449645, 3548.34962431,
       3703.19065006, 3704.80475815, 4392.76815863, 4946.80142277,
       2877.92341576, 3205.15470959, 3565.15503938, 3288.25850878,
       3535.21064717, 3257.5006814 , 3626.00214639, 3613.53834396,
       3773.04276395, 3768.32615789, 4446.2651074 , 4981.39234817,
       2967.91744106, 3287.42726124, 3642.87019854, 3372.16048116,
       3628.90920873, 3369.67552106, 3684.0332319 , 3679.11538937,
       3843.07360959, 3832.31082028, 4500.67586298, 5017.74680256,
       3057.18492953, 3369.320314  , 3720.41068863, 3455.60972112,
       3721.71473485, 3480.1269178 , 3742.77433961])
```

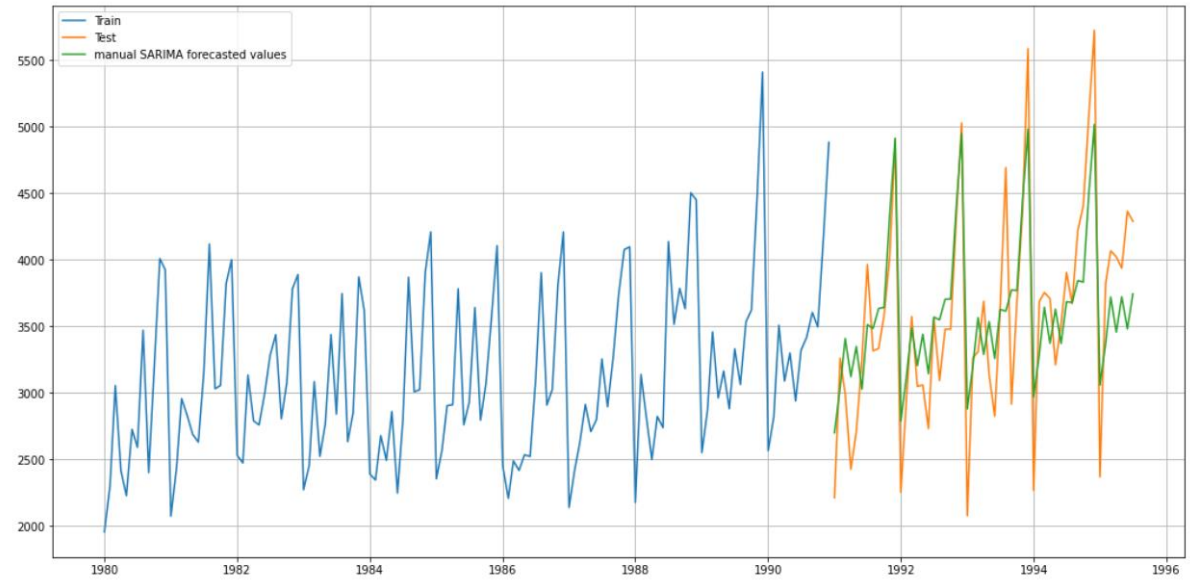


Figure 32 Manual SARIMA

RMSE for the manual SARIMA model: 445.42649206281226

8. Build a table (create a data frame) with all the models built along with their corresponding parameters and the respective RMSE values on the test data.

The data frame "results_df" which I have created to store the RMSE test values for all the models,

Let's now sort the dataframe with RMSE test score to find the best optimized model.

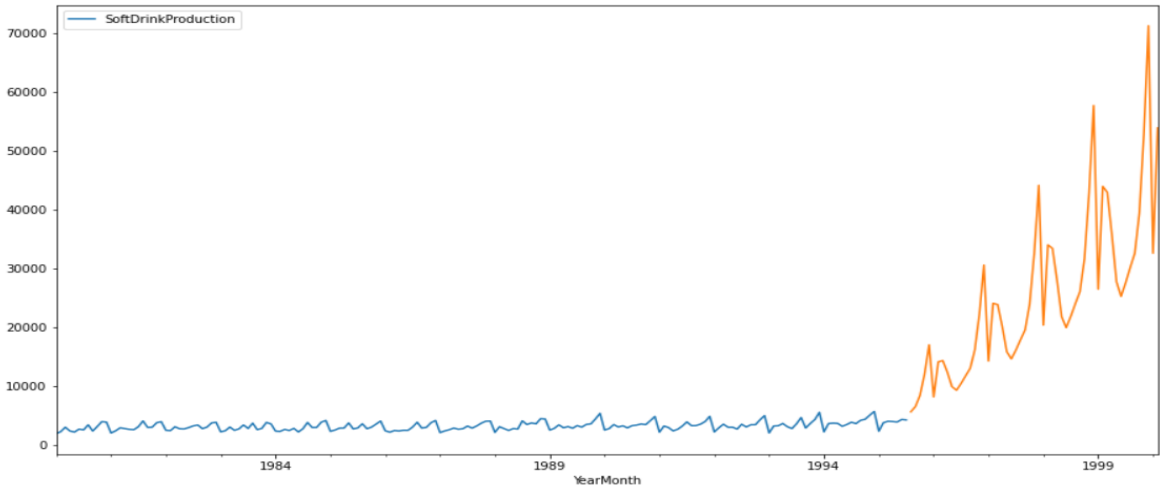
	Test RMSE
Alpha=0.4,Beta=0.4,Gamma=0.4,TripleExponentialSmoothing	427.331067
automated SARIMA(0,1,2)*(2,0,2,12)	437.706560
manual SARIMA(0,1,0)(1,1,3,6)	445.426492
automatedSARIMA(0,1,2)(2,0,2,6)	447.942743
Alpha=0.11,Beta=0.04,Gamma=0.39,TripleExponentialSmoothing	460.437728
2pointTrailingMovingAverage	556.725418
4pointTrailingMovingAverage	687.181726
6pointTrailingMovingAverage	710.513877
9pointTrailingMovingAverage	735.889827
Alpha=0.1,SimpleExponentialSmoothing	807.346881
manual ARIMA(3,1,2)	822.217407
automated ARIMA(0,1,2)	831.615850
Alpha=0.216,SimpleExponentialSmoothing	847.635259
RegressionOnTime	898.172528
SimpleAverageModel	934.353358
Alpha=0.1,Beta=0.1,DoubleExponentialSmoothing	985.423307
NaiveModel	1519.259233
Alpha=0.39 and Beta=0,DoubleExponentialSmoothing	2301.542724

Alpha=0.4, Beta=0.4, Gamma=0.4, Triple Exponential Smoothing is the best optimised model with the least RMSE value.

9. Based on the model-building exercise, build the most optimum model(s) on the complete data and predict 12 months into the future with appropriate confidence intervals/bands.

Optimum Model on Complete Dataset:

RMSE: 672.6828581062896



We assume that while calculating the confidence bands, the standard deviation of the forecast distribution is almost equal to the residual standard deviation.

	lower_CI	prediction	upper_ci
1995-08-01	4339.555941	5661.546954	6983.537966
1995-09-01	5235.539508	6557.530521	7879.521534
1995-10-01	7121.179397	8443.170409	9765.161422
1995-11-01	10707.951881	12029.942894	13351.933906
1995-12-01	15719.140912	17041.131925	18363.122937

plotting the forecast along with the confidence band:

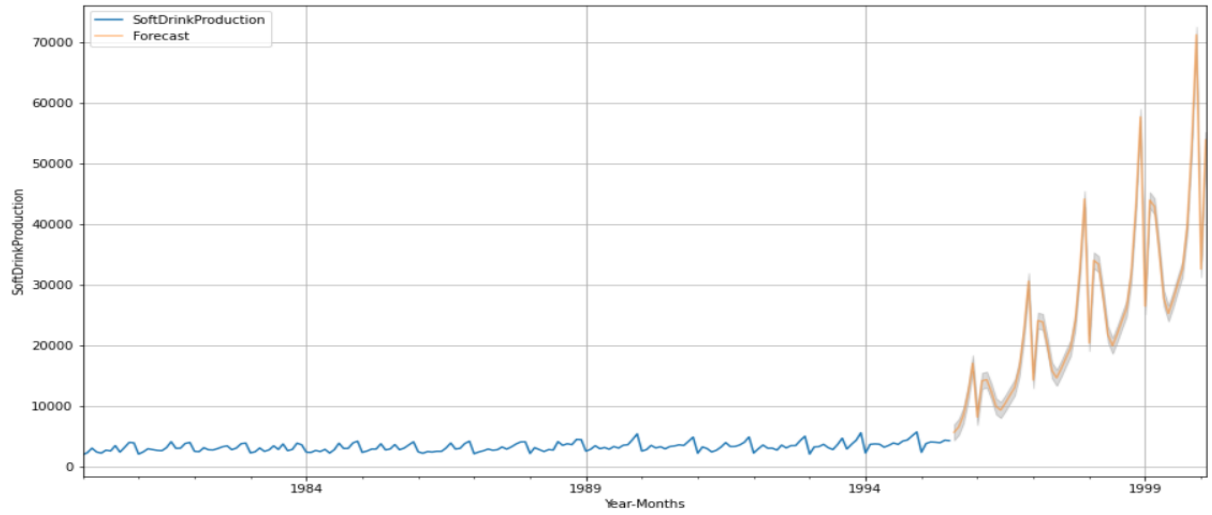


Figure 33 Final Model

10. Comment on the model thus built and report your findings and suggest the measures that the company should be taking for future sales.

Time series analysis involves understanding various aspects about the inherent nature of the series so that you are better informed to create meaningful and accurate forecasts.

Observations:

The soft drink production dataset is having both trend and seasonality.

The sales shows stabilized values.December month shows the highest sales.

The sales of soft drink is seasonal and also had trend. Therefore, the company cannot have the same stock throughout the year.

The models are built and are chosen based on the least RMSE score.

Insights and Recommendations:

The models are built considering the Trend and Seasonality into account and we can see from the output plot that the future prediction is in line with the trend and seasonality in the previous years.

1.The company should use the prediction results and capitalize on the high demand seasons and ensure to source and supply the high demand.

2.The company need to focus on the low demand seasons and can introduce new plans to improve the sales such as Discounts sales & Seasonal memberships.

3.Products that are discounted should be highlighted. So that the consumers can see the savings prominently and the discounts can compel consumers to buy.

4.The Company has to create a dynamic consumer experience with fresh point-of-sale materials and well stocked displays.

5.Seasonal memberships and discounts can be introduced. Many prominent retailers also have loyalty programs that create excitement.

6.Advertising events and exciting discounts can help draw consumers to your store and generate sales. Retailers with economies of scale successfully sample consumers on more profitable soft drink sales.

7.Comparison of products with the national brands that are more expensive to demonstrate the company is offering a less expensive but superior product. And bringing in celebrities, sommeliers or trade reps for advertising and promotion campaigns can help create excitement and drive traffic.