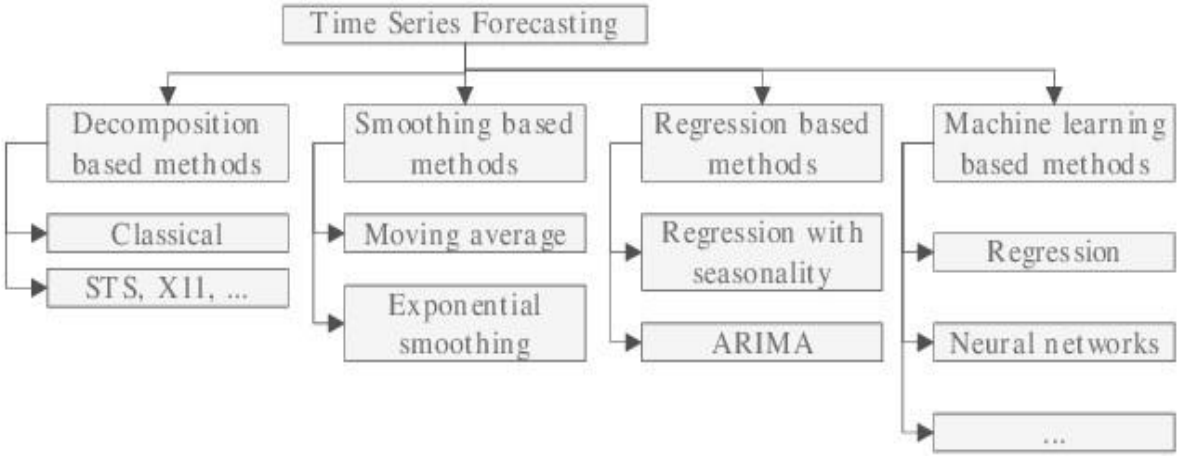


TIME SETIES FORECASTING

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.

PARAKANDLA MANISHA
PGP-DSBA GREATLEARNING | ONLINE

Contents

List of Figures:

TIME SERIES ANALYSIS ON SHOESALES DATA

PROBLEM STATEMENT:

You are an analyst in the IJK shoe company and you are expected to forecast the sales of the pairs of shoes for the upcoming 12 months from where the data ends. The data for the pair of shoe sales have been given to you from January 1980 to July 1995.

Data Set: **Shoesales.csv**

First, we import all the necessary libraries such as seaborn, pandas, sklearn etc to perform our analysis.
Next, we import the dataset **Shoesales.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	Shoe_Sales
0	1980-01	85
1	1980-02	89
2	1980-03	109
3	1980-04	95
4	1980-05	91

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 date column and making it as index.

The Time series is having monthly series.

We need to transform this data into appropriate time series data using pandas date_range function with (start='1/1/1980', end='8/1/1995', freq='M').

After that, I have created timestamp for this data and made timestamp as index for the data and dropped the YearMonth column. Now, the data looks like:

	Shoe_Sales
Time_Stamp	
1980-01-31	85
1980-02-29	89
1980-03-31	109
1980-04-30	95
1980-05-31	91

TIME SERIES PLOT FOR SHOESALES DATA:

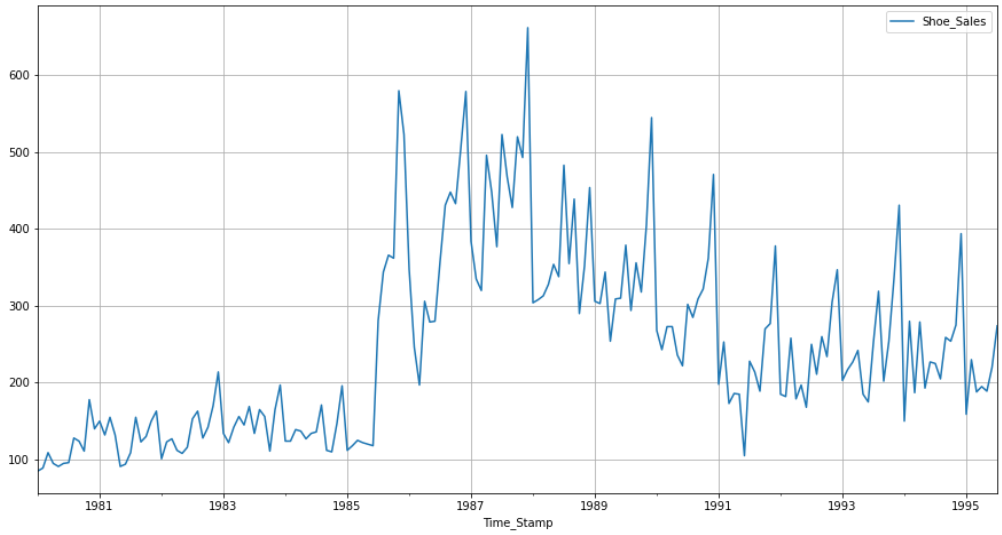


Figure 1. Time series plot

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 is having the Sudden increase in the trend and seasonality and slow decrease in terms of both trend and seasonality.

→Hence, the time series data shows both trend & seasonality.

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

→We can also observe the gradual slow decrease in the trend from 1987.

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, 1980-01-31 to 1995-07-31
Data columns (total 1 columns):
 #   Column      Non-Null Count  Dtype
---  ------      --------------  -----
 0   Shoe_Sales  187 non-null    int64
dtypes: int64(1)
memory usage: 2.9 KB
```

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

5.Description of the data:

	count	mean	std	min	25%	50%	75%	max
Shoe_Sales	187.0	245.636364	121.390804	85.0	143.5	220.0	315.5	662.0

Boxplot for yearly/Monthly sales for Shoe sales:

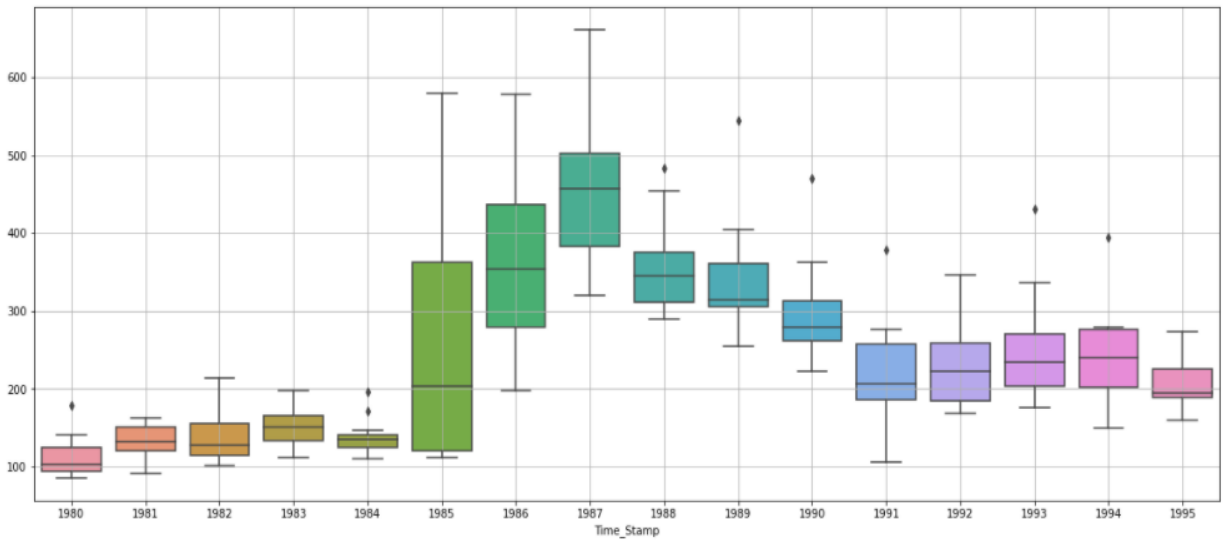


Figure 2 Yearly Boxplot

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

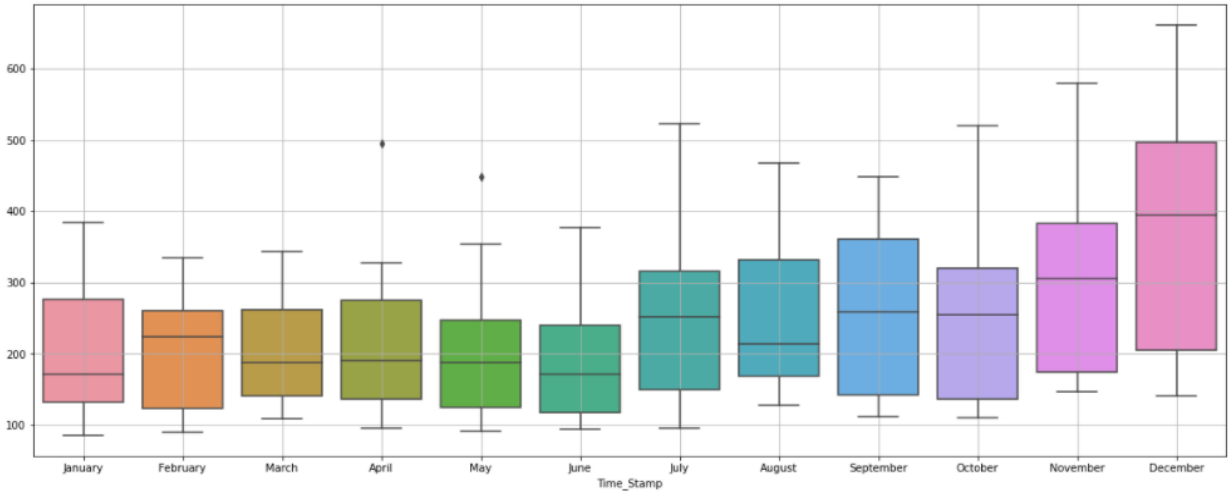


Figure 3 Monthly Boxplot

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 and May months.

- We can observe the spike of sales is in September, November, December for all the years.

- 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

Time_Stamp	1	2	3	4	5	6	7	8	9	10	11	12
Time_Stamp												
1980	85.0	89.0	109.0	95.0	91.0	95.0	96.0	128.0	124.0	111.0	178.0	140.0
1981	150.0	132.0	155.0	132.0	91.0	94.0	109.0	155.0	123.0	130.0	150.0	163.0
1982	101.0	123.0	127.0	112.0	108.0	116.0	153.0	163.0	128.0	142.0	170.0	214.0
1983	134.0	122.0	142.0	156.0	145.0	169.0	134.0	165.0	156.0	111.0	165.0	197.0
1984	124.0	124.0	139.0	137.0	127.0	134.0	136.0	171.0	112.0	110.0	147.0	196.0
1985	112.0	118.0	125.0	122.0	120.0	118.0	281.0	344.0	366.0	362.0	580.0	523.0
1986	348.0	246.0	197.0	306.0	279.0	280.0	358.0	431.0	448.0	433.0	504.0	579.0
1987	384.0	335.0	320.0	496.0	448.0	377.0	523.0	468.0	428.0	520.0	493.0	662.0
1988	304.0	308.0	313.0	328.0	354.0	338.0	483.0	355.0	439.0	290.0	352.0	454.0
1989	306.0	303.0	344.0	254.0	309.0	310.0	379.0	294.0	356.0	318.0	405.0	545.0
1990	268.0	243.0	273.0	273.0	236.0	222.0	302.0	285.0	309.0	322.0	362.0	471.0
1991	198.0	253.0	173.0	186.0	185.0	105.0	228.0	214.0	189.0	270.0	277.0	378.0
1992	185.0	182.0	258.0	179.0	197.0	168.0	250.0	211.0	260.0	234.0	305.0	347.0
1993	203.0	217.0	227.0	242.0	185.0	175.0	252.0	319.0	202.0	254.0	336.0	431.0
1994	150.0	280.0	187.0	279.0	193.0	227.0	225.0	205.0	259.0	254.0	275.0	394.0
1995	159.0	230.0	188.0	195.0	189.0	220.0	274.0	NaN	NaN	NaN	NaN	NaN

Cumulative % and Month on Month % sales plots of Shoe sales:

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

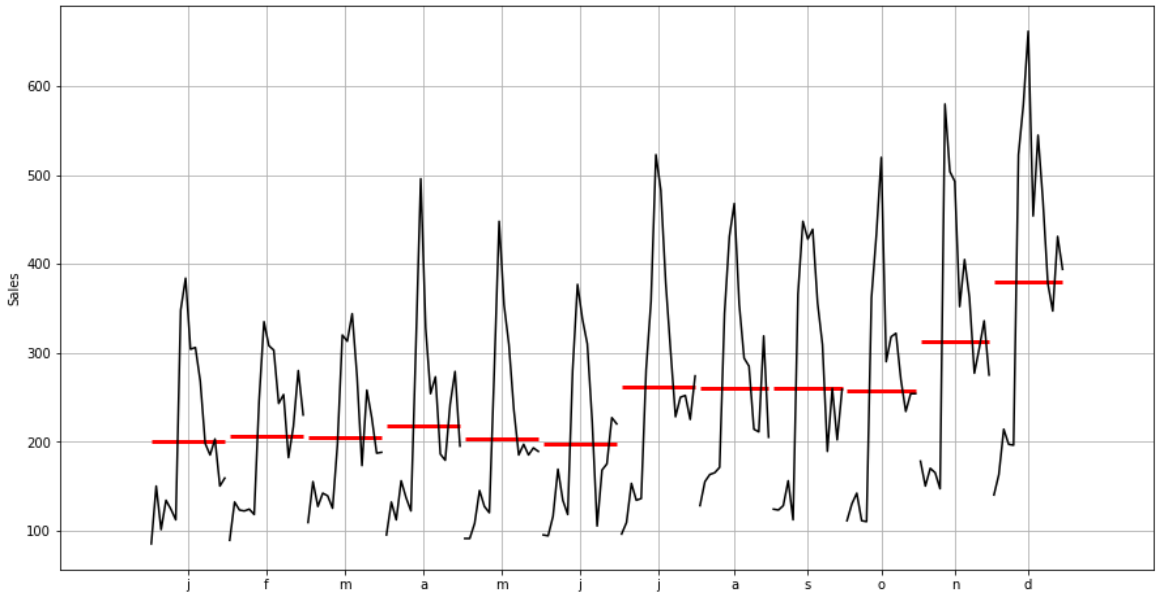


Figure 4 Timeseries month plot for spread of sales

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*.

we can observe that, the median is constant in the first few months and there is a slight variation from July to October and November and December months.

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.

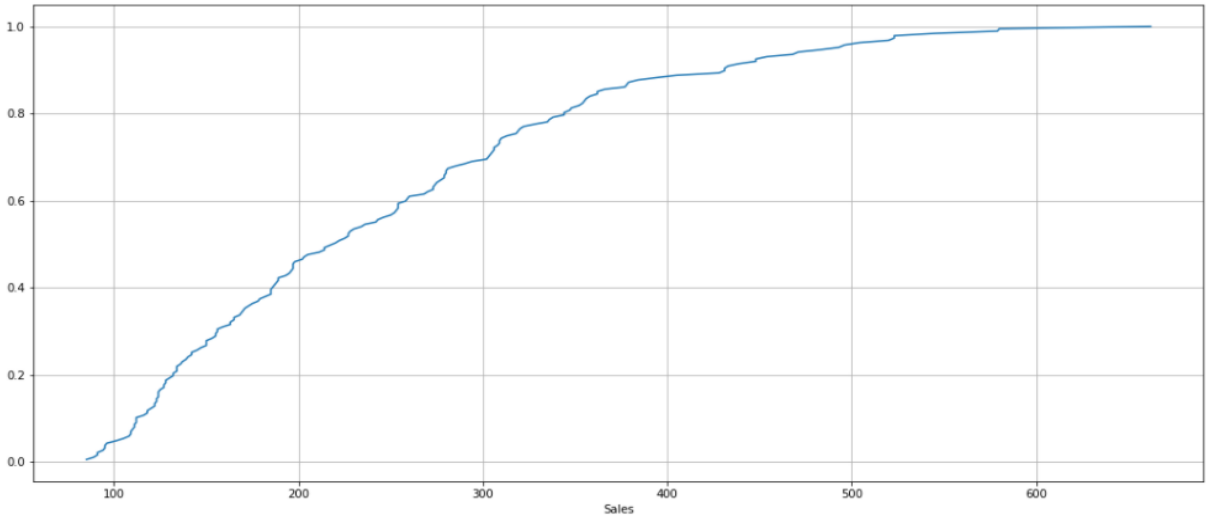


Figure 5 Empirical Cumulative Distribution Curve

Line plot for monthly sales for Shoe sales:

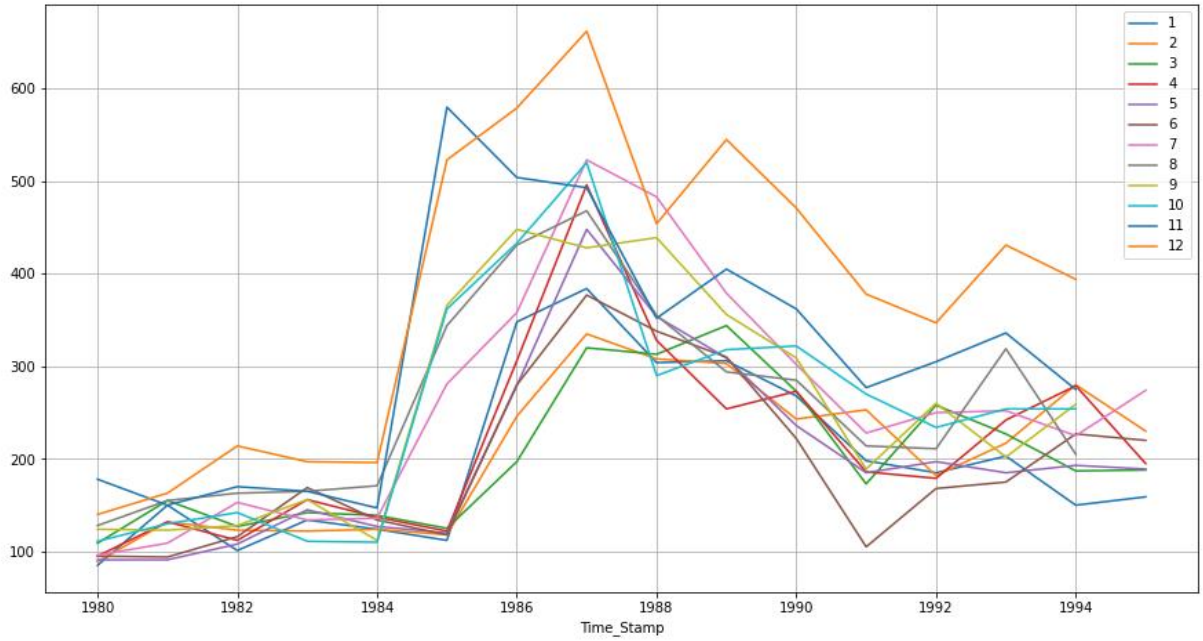


Figure 6 Line plot for monthly sales

The above line plot shoes the monthly sales performance of Shoe Sales across every year. Generally, November and December months are having the highest sales compared to others. The shoe sales are highest and almost in peak stage in the year 1987. December month is having the huge spike of sales in 1987. Also, for November month we can see the highest peak sales in 1985.

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

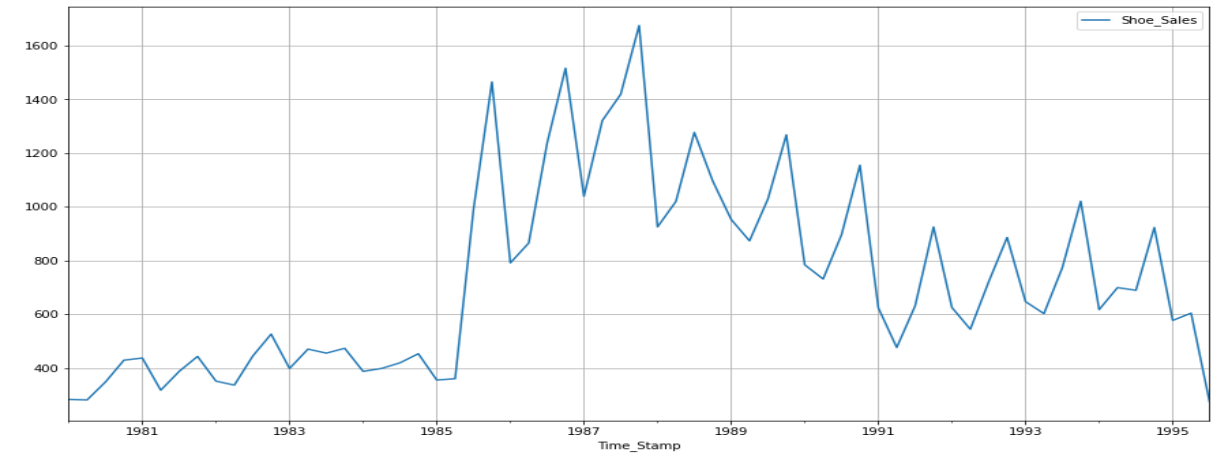
Figure 7 Yearly Plot - sum

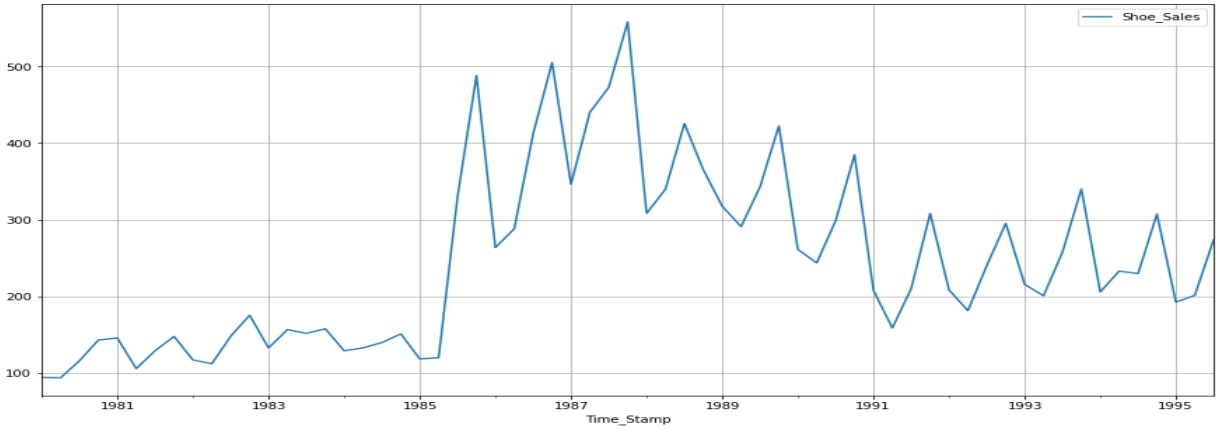
From the above plot, we can observe there is seasonality and trend. The sales across years for all the months are displayed in the graph. we can observe a clear highest spike from 1984 to 1987. Also, the year 1987 has the highest peak point in sales.

Figure 8 Yearly plot – Mean

From the above plot, we can observe the trend, seasonality and the average sales across each year. The plot for sum and mean observations is almost same.

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:

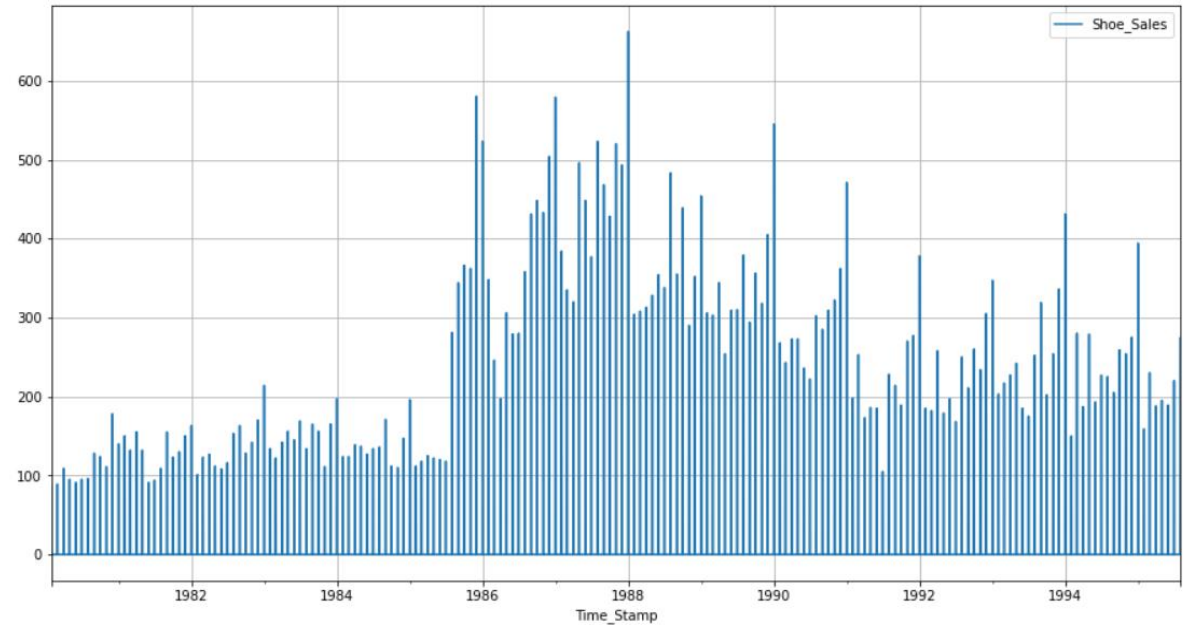


Figure 9 Daily Plot

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

Decade Plot:

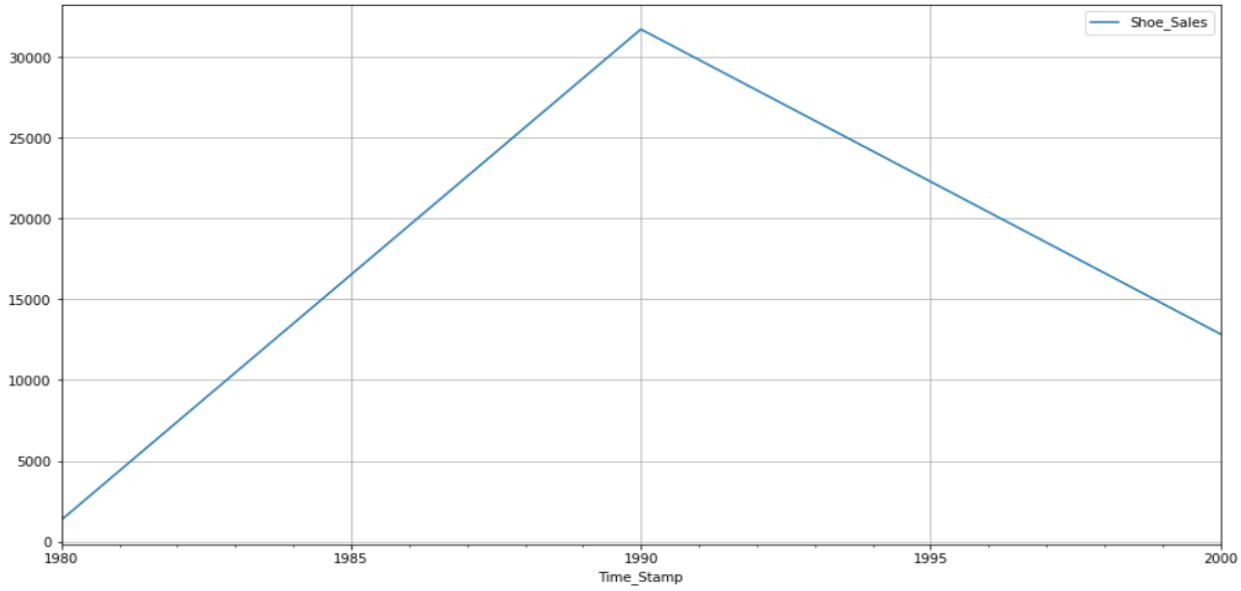


Figure 10 Decade Plot

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

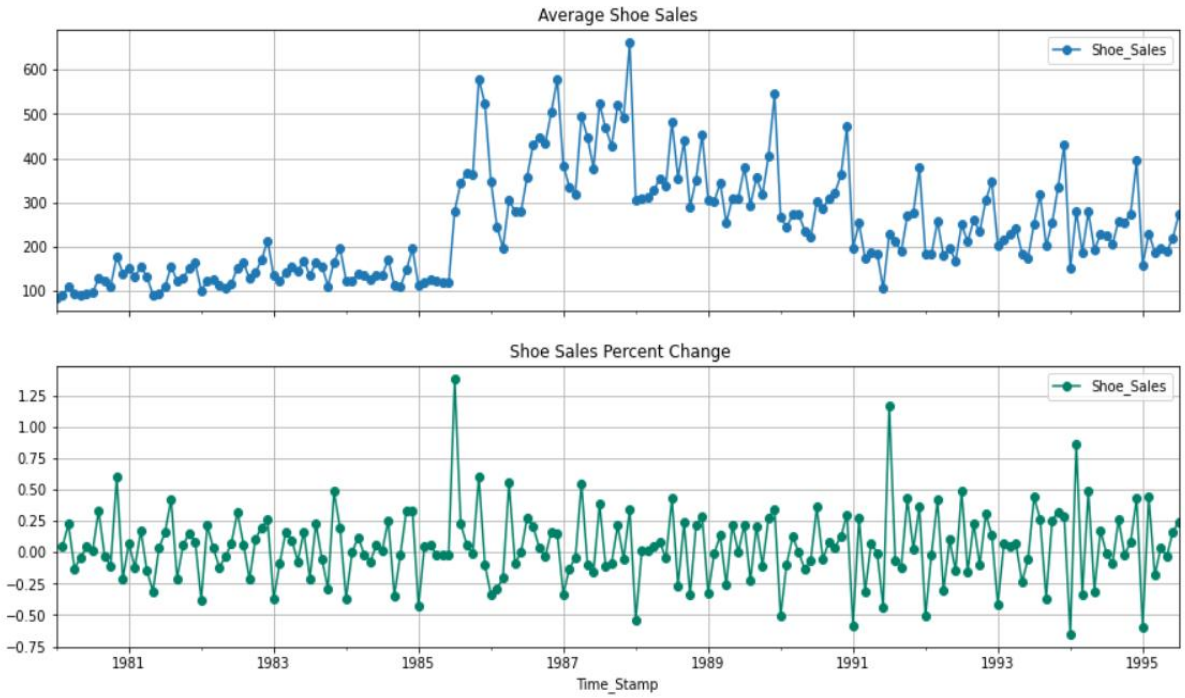


Figure 11 Average sales & Percentage change of Sales

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:

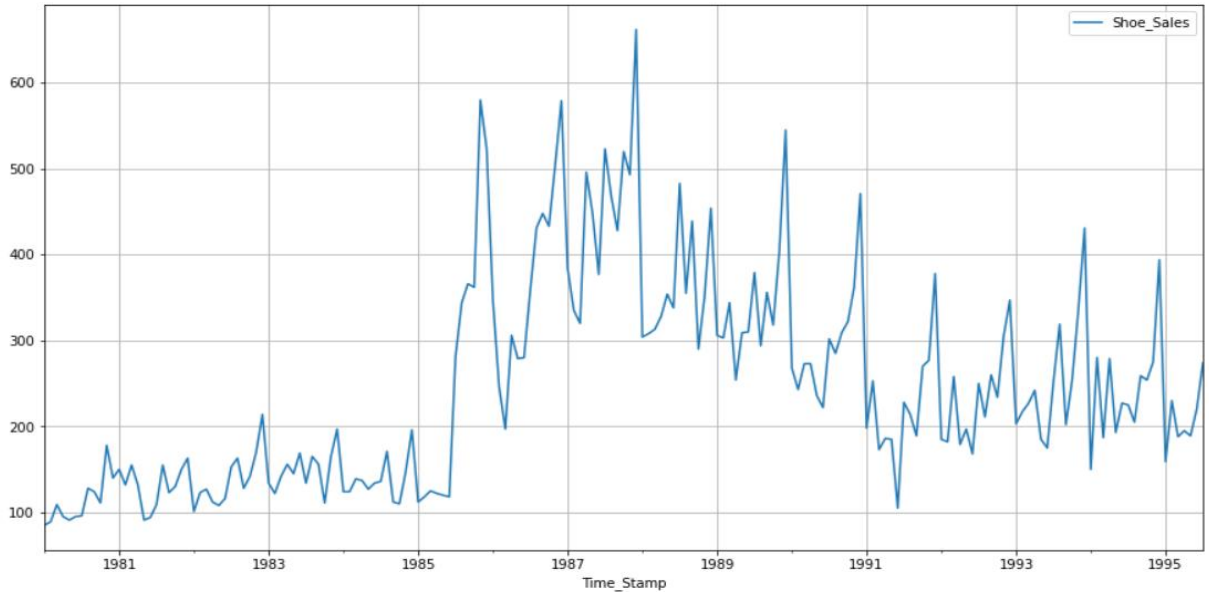


Figure 12 Time series plot -2

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:

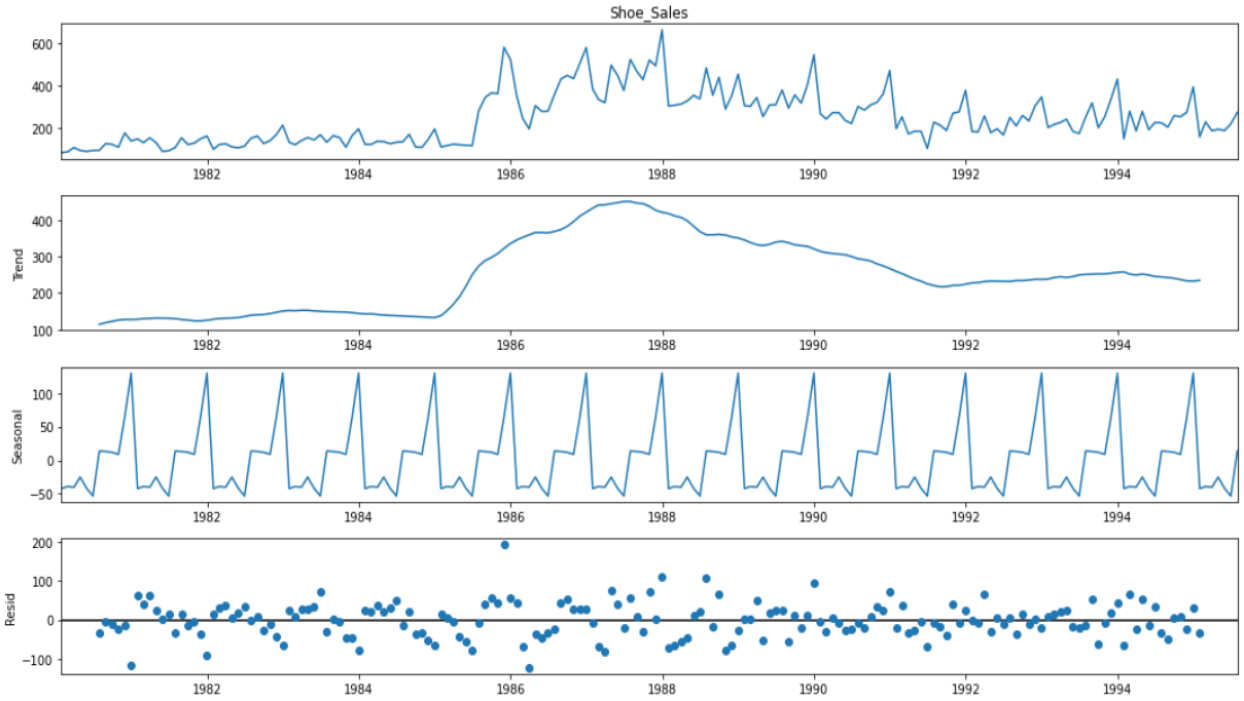


Figure 13 Additive Decomposition

From the above decomposition plot, we can see that the time series data is having the trend, seasonality in both increasing and decreasing fashion. we can see very high residual from -100 to +100 on Y- axis.

Hence, we can clearly say that the time series is not additive.

Multiplicative Decomposition:

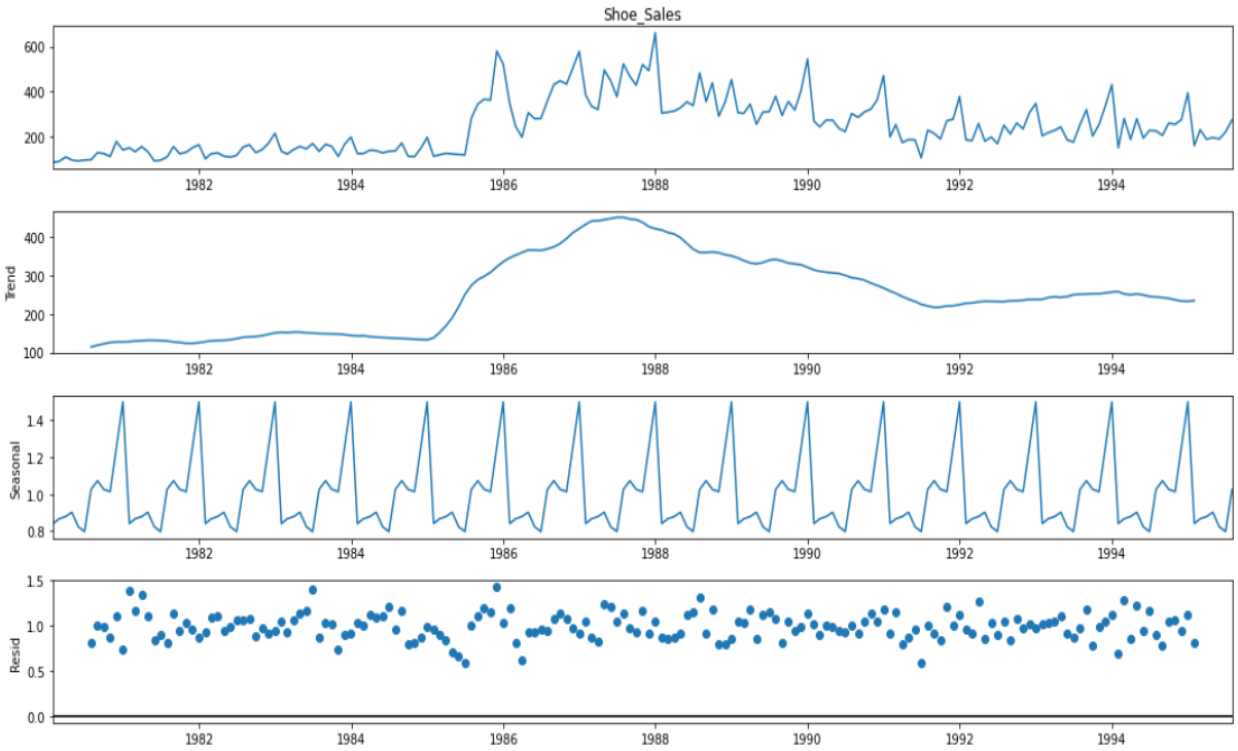


Figure 14 Multiplicative Decomposition

From the above decomposition plot, we can see that the time series data is having the trend, seasonality in both increasing and decreasing fashion.

The residual is lying from 0.5 to 1.5 on Y- axis which is very low when compared to the additive model.

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

Hence, we can clearly say 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 Shoe_Sales
Time_Stamp
1980-01-31 85
1980-02-29 89
1980-03-31 109
1980-04-30 95
1980-05-31 91

Last few rows of Training Data Shoe_Sales
Time_Stamp
1990-08-31 285
1990-09-30 309
1990-10-31 322
1990-11-30 362
1990-12-31 471

First few rows of Test Data Shoe_Sales
Time_Stamp
1991-01-31 198
1991-02-28 253
1991-03-31 173
1991-04-30 186
1991-05-31 185

Last few rows of Test Data Shoe_Sales
Time_Stamp
1995-03-31 188
1995-04-30 195
1995-05-31 189
1995-06-30 220
1995-07-31 274

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:

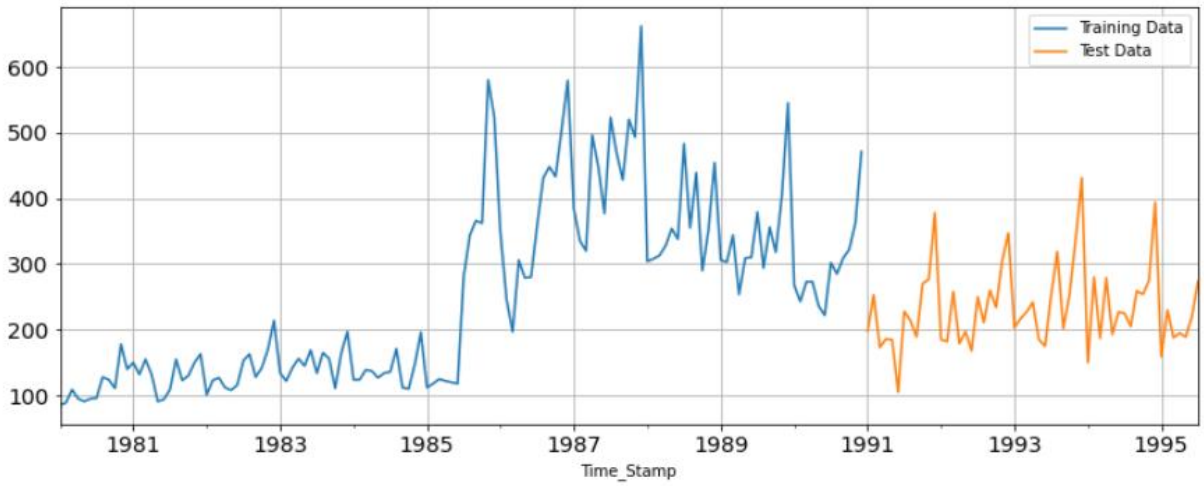


Figure 15 Train - test Split

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.

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 'Shoe_Sales' 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
            Shoe_Sales  Time
Time_Stamp
1980-01-31          85     1
1980-02-29          89     2
1980-03-31         109     3
1980-04-30          95     4
1980-05-31          91     5

Last few rows of Training Data
            Shoe_Sales  Time
Time_Stamp
1990-08-31         285   128
1990-09-30         309   129
1990-10-31         322   130
1990-11-30         362   131
1990-12-31         471   132

First few rows of Test Data
            Shoe_Sales  Time
Time_Stamp
1991-01-31         198    43
1991-02-28         253    44
1991-03-31         173    45
1991-04-30         186    46
1991-05-31         185    47

Last few rows of Test Data
            Shoe_Sales  Time
Time_Stamp
1995-03-31         188    93
1995-04-30         195    94
1995-05-31         189    95
1995-06-30         220    96
1995-07-31         274    97
```

Predictions on Train Data:

```
[array([ 82.06550467,  84.63818029,  87.2108559 ,  89.78353152,
         92.35620714,  94.92888276,  97.50155837, 100.07423399,
        102.64690961, 105.21958522, 107.79226084, 110.36493646,
        112.93761207, 115.51028769, 118.08296331, 120.65563892,
        123.22831454, 125.80099016, 128.37366577, 130.94634139,
        133.51901701, 136.09169263, 138.66436824, 141.23704386,
        143.80971948, 146.38239509, 148.95507071, 151.52774633,
        154.10042194, 156.67309756, 159.24577318, 161.81844879,
        164.39112441, 166.96380003, 169.53647564, 172.10915126,
        174.68182688, 177.25450249, 179.82717811, 182.39985373,
        184.97252935, 187.54520496, 190.11788058, 192.6905562 ,
        195.26323181, 197.83590743, 200.40858305, 202.98125866,
        205.55393428, 208.1266099 , 210.69928551, 213.27196113,
        215.84463675, 218.41731236, 220.98998798, 223.5626636 ,
        226.13533922, 228.70801483, 231.28069045, 233.85336607,
        236.42604168, 238.9987173 , 241.57139292, 244.14406853,
        246.71674415, 249.28941977, 251.86209538, 254.434771  ,
        257.00744662, 259.58012223, 262.15279785, 264.72547347,
        267.29814909, 269.8708247 , 272.44350032, 275.01617594,
        277.58885155, 280.16152717, 282.73420279, 285.3068784 ,
        287.87955402, 290.45222964, 293.02490525, 295.59758087,
        298.17025649, 300.7429321 , 303.31560772, 305.88828334,
        308.46095896, 311.03363457, 313.60631019, 316.17898581,
        318.75166142, 321.32433704, 323.89701266, 326.46968827,
        329.04236389, 331.61503951, 334.18771512, 336.76039074,
        339.33306636, 341.90574197, 344.47841759, 347.05109321,
        349.62376883, 352.19644444, 354.76912006, 357.34179568,
        359.91447129, 362.48714691, 365.05982253, 367.63249814,
        370.20517376, 372.77784938, 375.35052499, 377.92320061,
        380.49587623, 383.06855184, 385.64122746, 388.21390308,
        390.7865787 , 393.35925431, 395.93192993, 398.50460555,
        401.07728116, 403.64995678, 406.2226324 , 408.79530801,
        411.36798363, 413.94065925, 416.51333486, 419.08601048])]
```

Predictions on Test Data:

```
[array([190.11788058, 192.6905562 , 195.26323181, 197.83590743,
        200.40858305, 202.98125866, 205.55393428, 208.1266099 ,
        210.69928551, 213.27196113, 215.84463675, 218.41731236,
        220.98998798, 223.5626636 , 226.13533922, 228.70801483,
        231.28069045, 233.85336607, 236.42604168, 238.9987173 ,
        241.57139292, 244.14406853, 246.71674415, 249.28941977,
        251.86209538, 254.434771  , 257.00744662, 259.58012223,
        262.15279785, 264.72547347, 267.29814909, 269.8708247 ,
        272.44350032, 275.01617594, 277.58885155, 280.16152717,
        282.73420279, 285.3068784 , 287.87955402, 290.45222964,
        293.02490525, 295.59758087, 298.17025649, 300.7429321 ,
        303.31560772, 305.88828334, 308.46095896, 311.03363457,
        313.60631019, 316.17898581, 318.75166142, 321.32433704,
        323.89701266, 326.46968827, 329.04236389])]
```

Linear Regression train test plot:

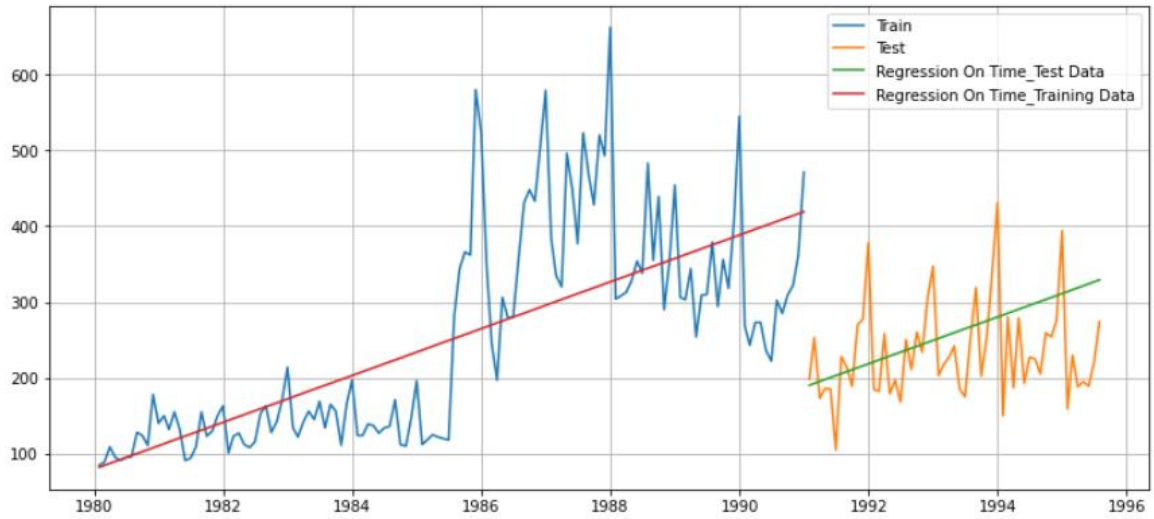


Figure 16 Linear Regression Train - Test Plot

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 regression line to upward peak and downward peak which we predicted is high. From this, we can understand that the error is high.

Hence, we can use the Linear Regression model to predict the trend in the timeseries data.

Regression On Time forecast on the Test Data:

RMSE is 73.112

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.

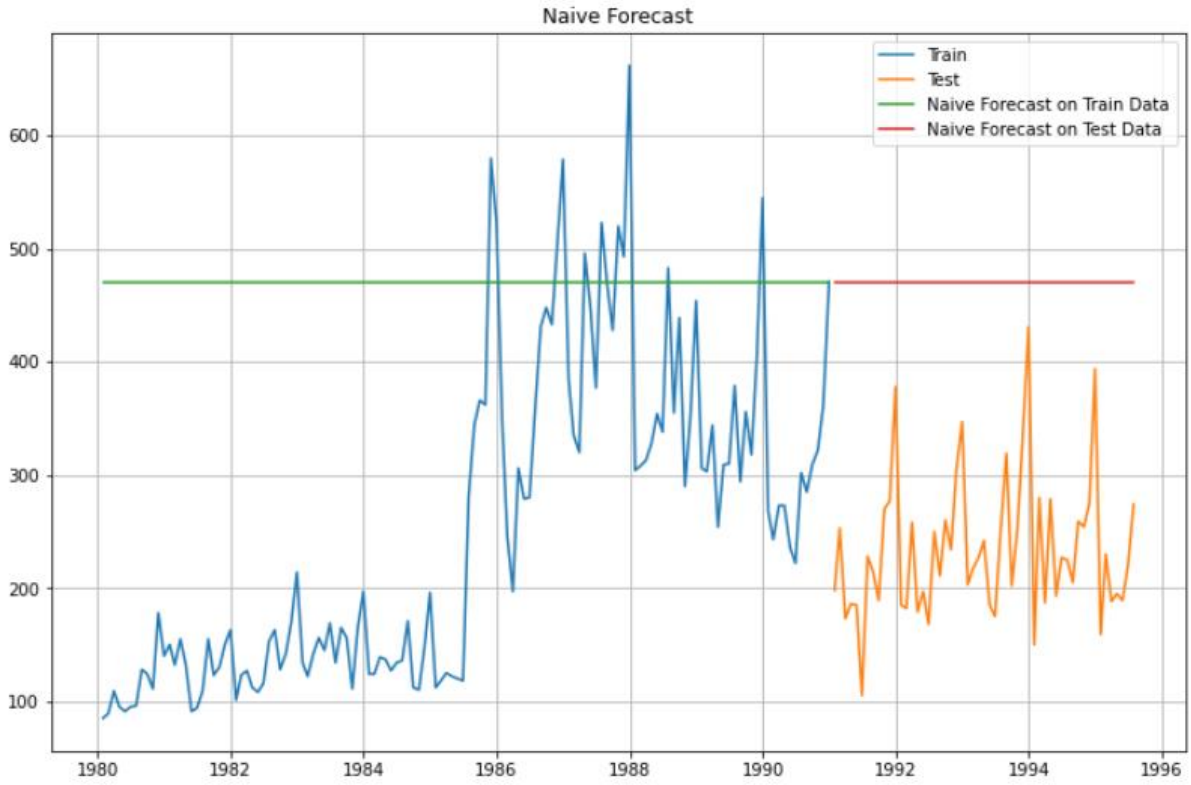


Figure 17 Naive Train- Test Plot

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 245.121

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.

Time_Stamp	Shoe_Sales	mean_forecast
1980-01-31	85	250.575758
1980-02-29	89	250.575758
1980-03-31	109	250.575758
1980-04-30	95	250.575758
1980-05-31	91	250.575758

Time_Stamp	Shoe_Sales	mean_forecast
1991-01-31	198	250.575758
1991-02-28	253	250.575758
1991-03-31	173	250.575758
1991-04-30	186	250.575758
1991-05-31	185	250.575758

Simple Average Train – Test plot:

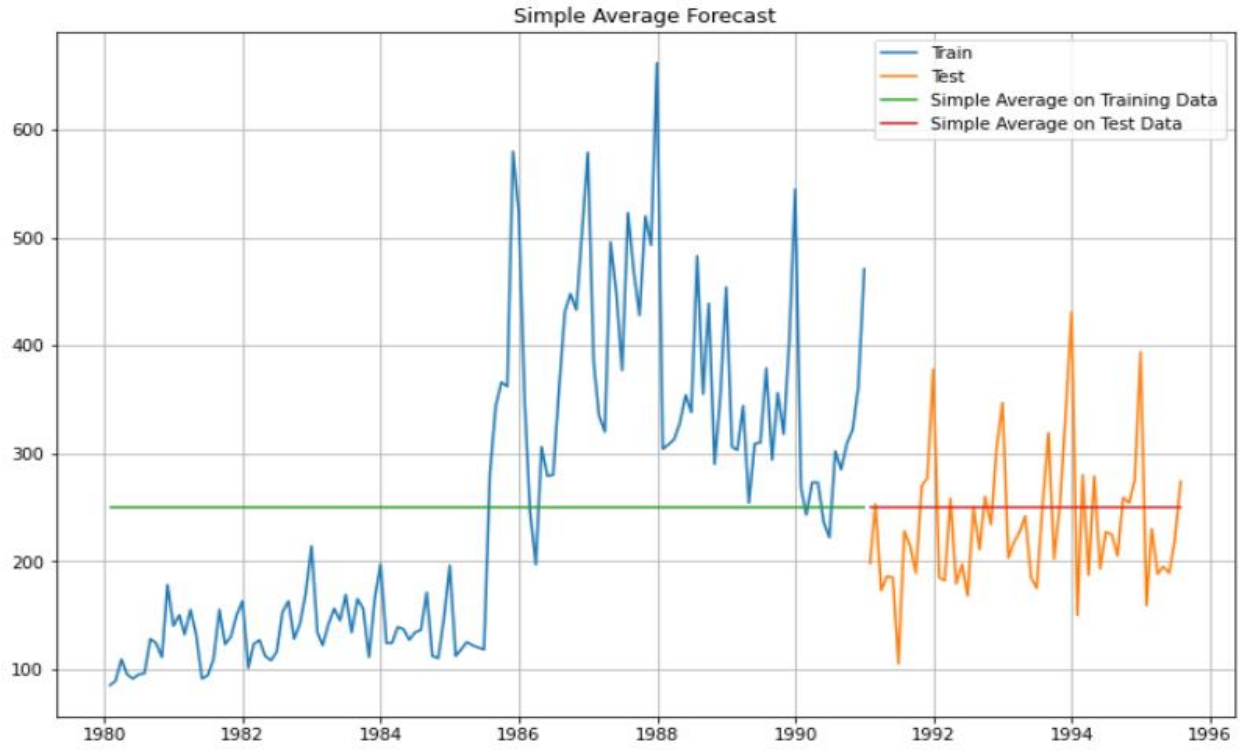


Figure 18 Simple Average Train – Test

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 63.985**

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.

Time_Stamp	Shoe_Sales	Trailing_2	Trailing_4	Trailing_6	Trailing_9
1980-01-31	85	NaN	NaN	NaN	NaN
1980-02-29	89	87.0	NaN	NaN	NaN
1980-03-31	109	99.0	NaN	NaN	NaN
1980-04-30	95	102.0	94.5	NaN	NaN
1980-05-31	91	93.0	96.0	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.

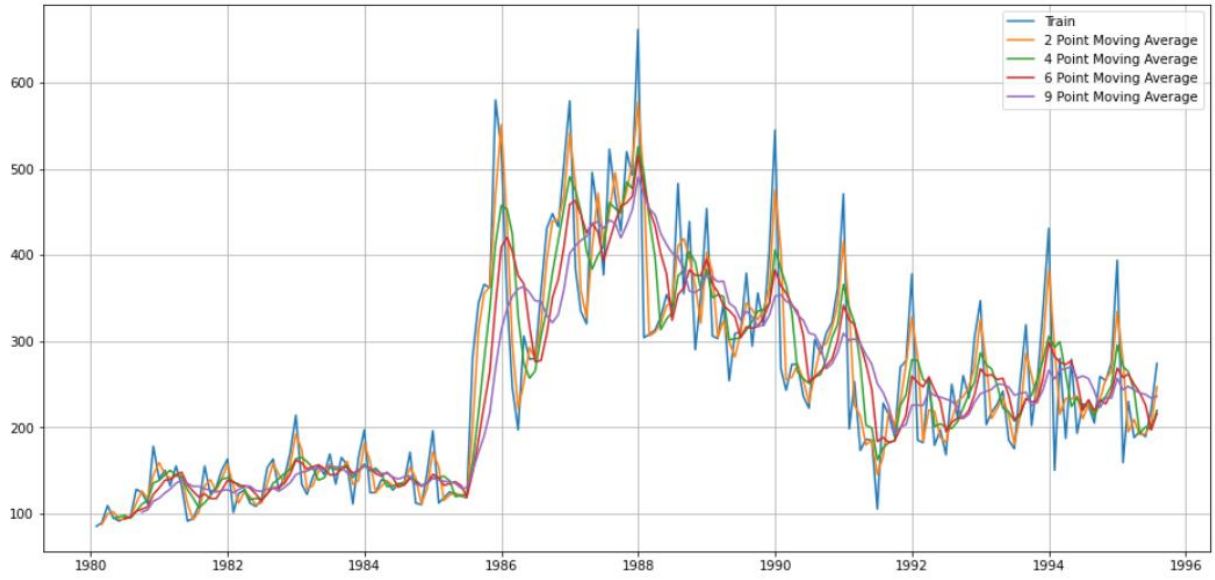


Figure 19 Simple Average model plot

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.

Time_Stamp	Shoe_Sales	Trailing_2	Trailing_4	Trailing_6	Trailing_9
1980-01-31	85	NaN	NaN	NaN	NaN
1980-02-29	89	87.0	NaN	NaN	NaN
1980-03-31	109	99.0	NaN	NaN	NaN
1980-04-30	95	102.0	94.5	NaN	NaN
1980-05-31	91	93.0	96.0	NaN	NaN

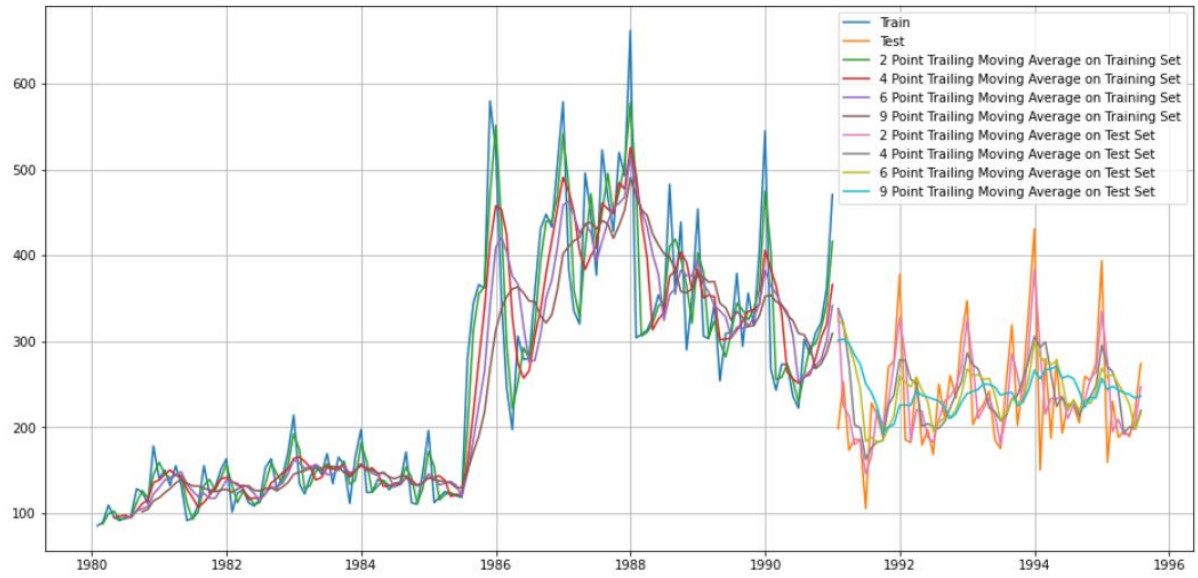


Figure 20 Simple Average Model Train- Test Plot

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.

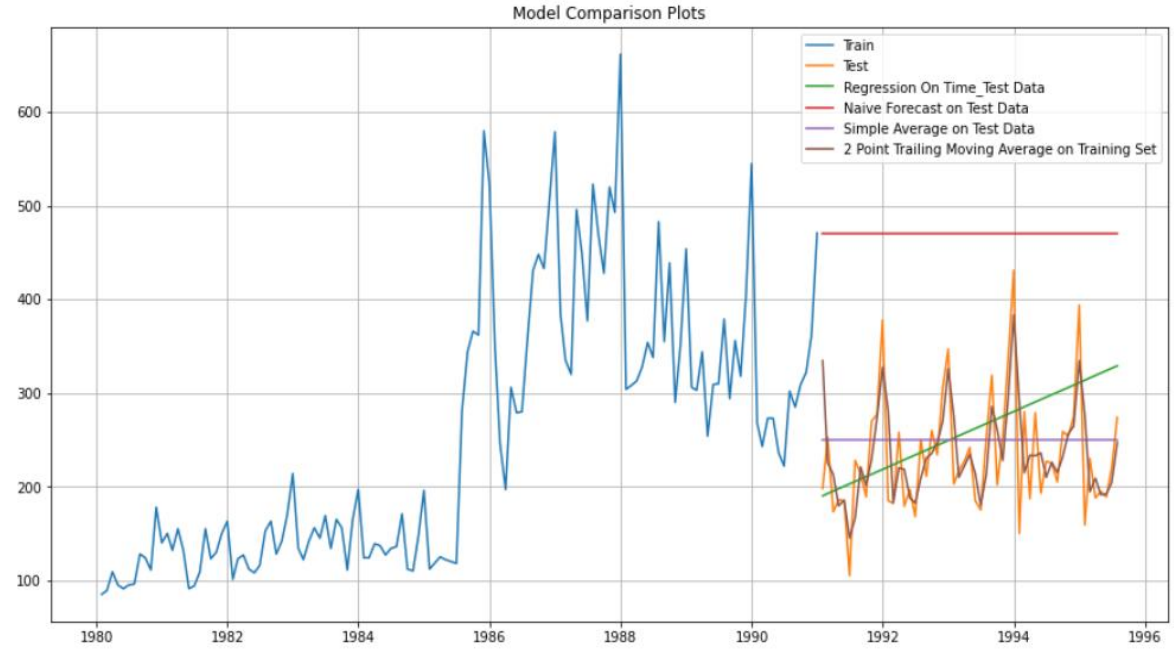
2 point Moving Average Model forecast on the Testing Data:
RMSE is 45.949

4 point Moving Average Model forecast on the Testing Data:
RMSE is 57.873

6 point Moving Average Model forecast on the Testing Data:
RMSE is 63.457

9 point Moving Average Model forecast on the Testing Data:
RMSE is 67.724

Before we go on to build the various Exponential Smoothing models, let us look at all the models and compare the Time Series plots:



From the plot, we can observe that the 2 point 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.

Summary:

SimpleExpSmoothing Model Results

Dep. Variable:	Shoe_Sales	No. Observations:	132
Model:	SimpleExpSmoothing	SSE	682402.272
Optimized:	True	AIC	1132.676
Trend:	None	BIC	1138.441
Seasonal:	None	AICC	1132.991
Seasonal Periods:	None	Date:	Thu, 31 Mar 2022
Box-Cox:	False	Time:	12:01:34
Box-Cox Coeff.:	None		

	coeff	code	optimized
smoothing_level	0.6050493	alpha	True
initial_level	88.829371	l.0	True

Parameters:

```
{'smoothing_level': 0.60504929557062,
 'smoothing_trend': nan,
 'smoothing_seasonal': nan,
 'damping_trend': nan,
 'initial_level': 88.82937100904118,
 '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:

Time_Stamp	Shoe_Sales	predict
1980-01-31	85	88.829371
1980-02-29	89	86.512413
1980-03-31	109	88.017526
1980-04-30	95	100.712957
1980-05-31	91	97.256336

Time_Stamp	Shoe_Sales	predict
1991-01-31	198	420.229868
1991-02-28	253	420.229868
1991-03-31	173	420.229868
1991-04-30	186	420.229868
1991-05-31	185	420.229868

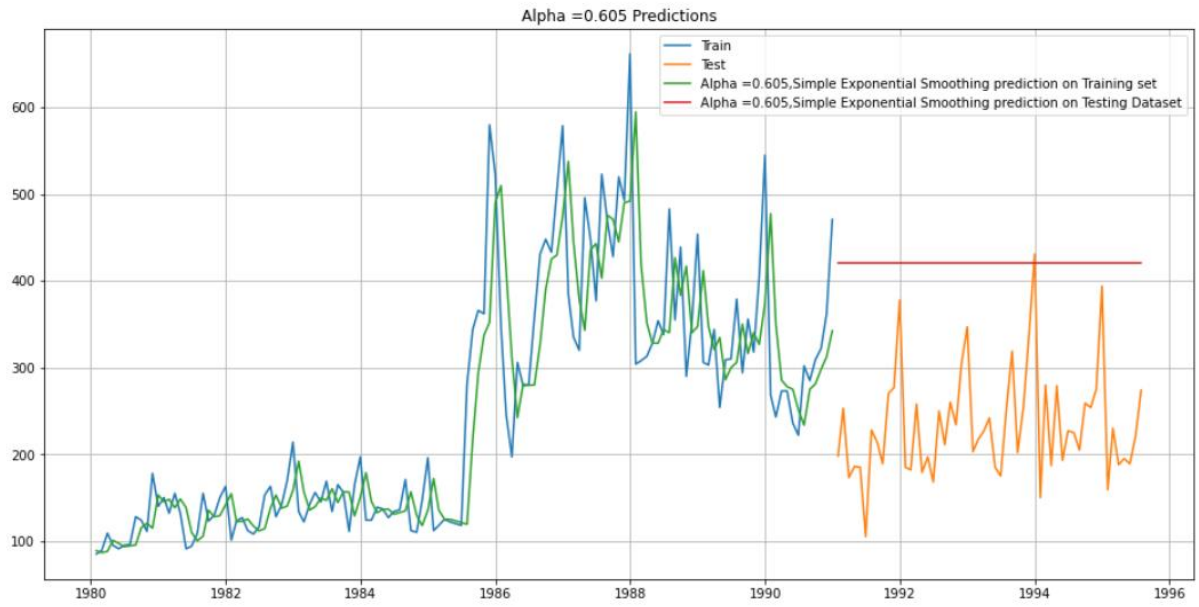


Figure 21 Simple Exponential Smoothing Train - Test Plot

Alpha =0.605 Simple Exponential Smoothing Model forecast on the Training Data,

RMSE is 196.405

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	115.874466	84.870519
1	0.2	124.976820	77.171203
2	0.3	143.400350	74.544186
3	0.4	162.553211	73.058312
4	0.5	180.072484	72.198579
5	0.6	195.663327	71.901396
6	0.7	209.658339	72.131315
7	0.8	222.417584	72.846836
8	0.9	234.188166	74.023411

After sorting the values with respect to test RMSE, I'm choosing alpha with 0.1 since it is having low **RMSE 115.874466.**

SimpleExpSmoothing Model Results

Dep. Variable:	Shoe_Sales	No. Observations:	132
Model:	SimpleExpSmoothing	SSE	950796.654
Optimized:	True	AIC	1176.457
Trend:	None	BIC	1182.223
Seasonal:	None	AICC	1176.772
Seasonal Periods:	None	Date:	Thu, 31 Mar 2022
Box-Cox:	False	Time:	12:01:40
Box-Cox Coeff.:	None		

	coeff	code	optimized
smoothing_level	0.1000000	alpha	False
initial_level	112.84699	l.0	True

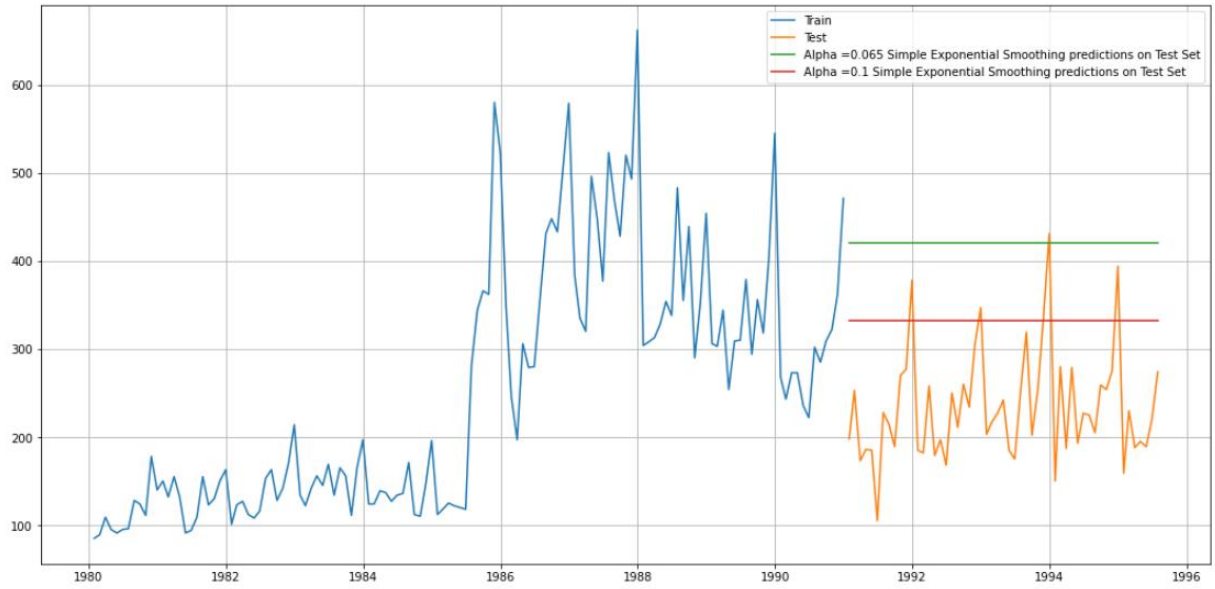


Figure 22 SES train-test plot

We can observe 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:	Shoe_Sales	No. Observations:	132
Model:	Holt	SSE	680501.962
Optimized:	True	AIC	1136.307
Trend:	Additive	BIC	1147.839
Seasonal:	None	AICC	1136.979
Seasonal Periods:	None	Date:	Thu, 31 Mar 2022
Box-Cox:	False	Time:	12:01:42
Box-Cox Coeff.:	None		

	coeff	code	optimized
smoothing_level	0.5982578	alpha	True
smoothing_trend	0.0006281	beta	True
initial_level	84.982346	l.0	True
initial_trend	2.5474367	b.0	True

Predict on train-test:

Time_Stamp	Shoe_Sales	(predict, 0.59, 0.0)
1980-01-31	85	87.529783
1980-02-29	89	88.562807
1980-03-31	109	91.371011
1980-04-30	95	104.470966
1980-05-31	91	101.354602

Time_Stamp	Shoe_Sales	(predict, 0.59, 0.0)
1991-01-31	198	423.398462
1991-02-28	253	425.945842
1991-03-31	173	428.493222
1991-04-30	186	431.040601
1991-05-31	185	433.587981

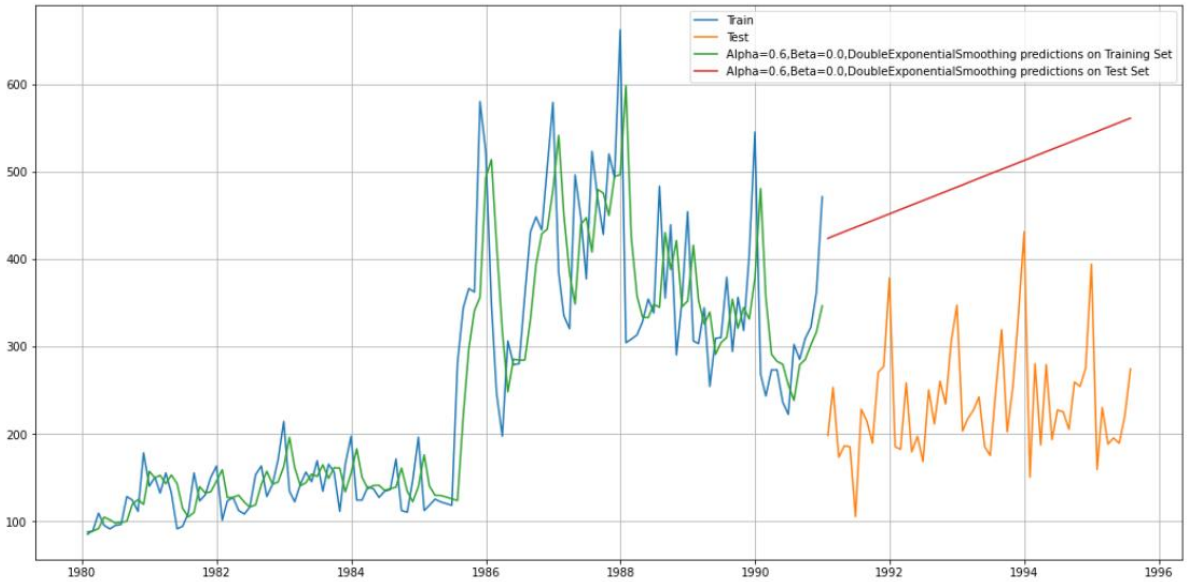


Figure 23 Double Exponential Smoothing Train- Test plot

Alpha=0.59 and Beta=0 Double Exponential Smoothing Model forecast on the Testing Data,

RMSE is 267.252

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	86.606563	76.884339
8	0.2	0.1	79.155470	163.808760
1	0.1	0.2	84.427567	209.883100
2	0.1	0.3	84.183322	225.100422
3	0.1	0.4	86.094141	229.327873
...	...	...	...	...
62	0.8	0.7	91.971921	2645.253070
55	0.7	0.8	91.120054	2663.039985
70	0.9	0.7	96.422783	2821.724663
63	0.8	0.8	95.113247	2889.230467
71	0.9	0.8	100.280707	3077.279937

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:             Shoe_Sales   No. Observations:                  132
Model:                           Holt   SSE                         990091.973
Optimized:                       True   AIC                           1185.803
Trend:                       Additive   BIC                           1197.334
Seasonal:                        None   AICC                          1186.475
Seasonal Periods:                None   Date:                Thu, 31 Mar 2022
Box-Cox:                        False   Time:                        12:01:48
Box-Cox Coeff.:                  None
================================================================================
                         coeff                 code              optimized
--------------------------------------------------------------------------------
smoothing_level          0.1000000                alpha                 False
smoothing_trend          0.1000000                 beta                 False
initial_level            110.73901                  1.0                  True
initial_trend            1.6579527                  b.0                  True
--------------------------------------------------------------------------------
```

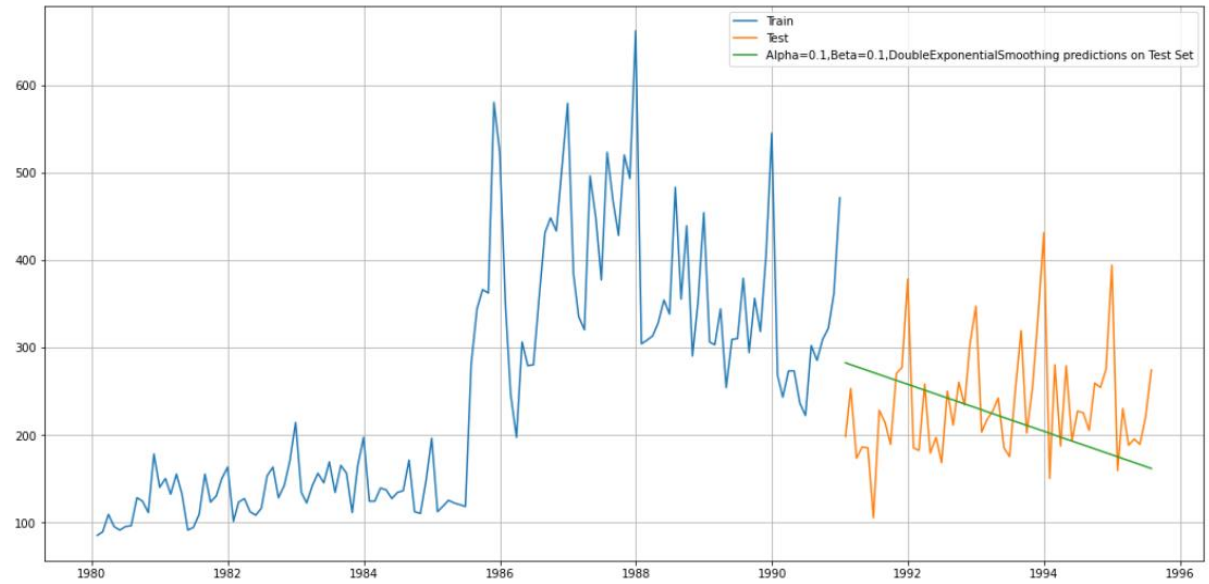


Figure 24 DES train-test plot

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': 0.5355789211755425,
 'smoothing_trend': 0.0005989974704412877,
 'smoothing_seasonal': 0.24421956531721212,
 'damping_trend': nan,
 'initial_level': 210.49455796111337,
 'initial_trend': 1.0641531153308956,
 'initial_seasons': array([0.51446756, 0.47849783, 0.60495344, 0.66658425, 0.57530128,
       0.51581614, 0.55731175, 0.75364195, 0.82116511, 0.69364706,
       0.85108384, 0.85085833]),
 'use_boxcox': False,
 'lamda': None,
 'remove_bias': False}
```

Time_Stamp	Shoe_Sales	auto_predict
1980-01-31	85	108.840094
1980-02-29	89	89.856925
1980-03-31	109	113.658157
1980-04-30	95	123.185737
1980-05-31	91	93.882150

Time_Stamp	Shoe_Sales	auto_predict
1991-01-31	198	261.204815
1991-02-28	253	245.321919
1991-03-31	173	261.538070
1991-04-30	186	272.150881
1991-05-31	185	270.396132

Summary:

ExponentialSmoothing Model Results

Dep. Variable:	Shoe_Sales	No. Observations:	132
Model:	ExponentialSmoothing	SSE	297665.163
Optimized:	True	AIC	1051.162
Trend:	Additive	BIC	1097.287
Seasonal:	Multiplicative	AICC	1057.215
Seasonal Periods:	12	Date:	Thu, 31 Mar 2022
Box-Cox:	False	Time:	12:01:50
Box-Cox Coeff.:	None		

	coeff	code	optimized
smoothing_level	0.5355789	alpha	True
smoothing_trend	0.0005990	beta	True
smoothing_seasonal	0.2442196	gamma	True
initial_level	210.49456	l.0	True
initial_trend	1.0641531	b.0	True
initial_seasons.0	0.5144676	s.0	True
initial_seasons.1	0.4784978	s.1	True
initial_seasons.2	0.6049534	s.2	True
initial_seasons.3	0.6665842	s.3	True
initial_seasons.4	0.5753013	s.4	True
initial_seasons.5	0.5158161	s.5	True
initial_seasons.6	0.5573117	s.6	True
initial_seasons.7	0.7536420	s.7	True
initial_seasons.8	0.8211651	s.8	True
initial_seasons.9	0.6936471	s.9	True
initial_seasons.10	0.8510838	s.10	True
initial_seasons.11	0.8508583	s.11	True

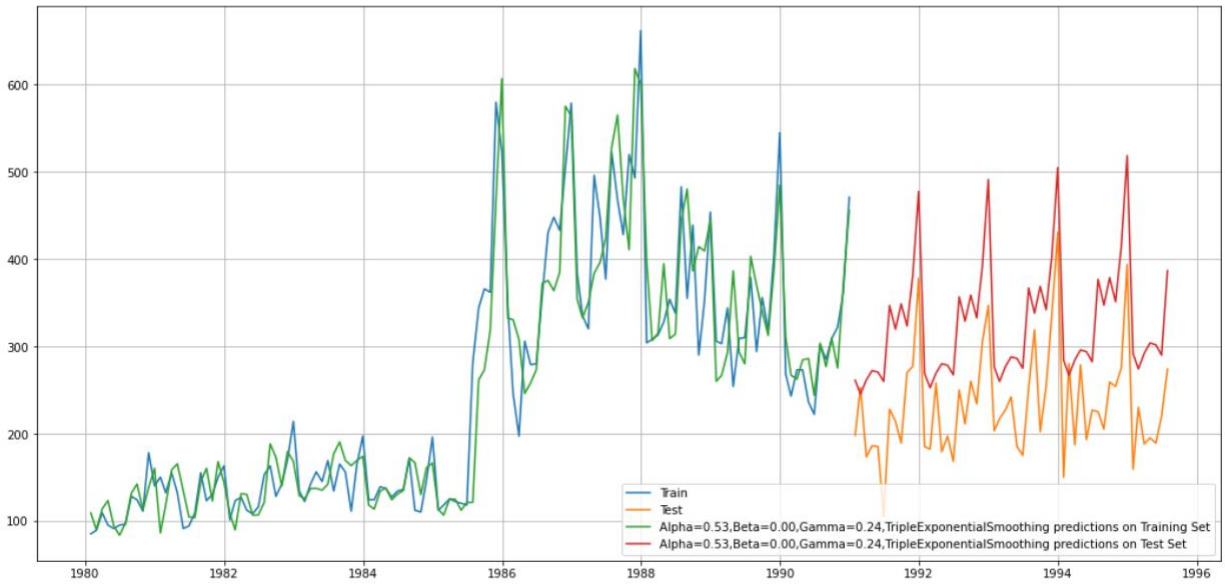


Figure 25 Triple Exponential Smoothing Train – Test

Alpha=0.53, Beta=0.00, Gamma=0.24, Triple Exponential Smoothing Model forecast on the T est Data,

RMSE is 99.428

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
46	0.1	0.6	0.2	7.206017e+01	4.131490e+01
40	0.1	0.5	0.5	9.340528e+01	4.384285e+01
34	0.1	0.4	0.8	1.190352e+02	4.513769e+01
93	0.2	0.2	0.4	6.627422e+01	4.700507e+01
10	0.1	0.2	0.2	6.535062e+01	4.720283e+01
...	...	...	...	...	...
638	0.8	0.8	0.9	1.326806e+04	3.109094e+06
319	0.4	0.9	0.5	3.602118e+05	3.157562e+06
146	0.2	0.8	0.3	7.683824e+04	4.415493e+06
276	0.4	0.4	0.7	6.651619e+10	5.136528e+06
78	0.1	0.9	0.7	2.146187e+15	1.456081e+07

729 rows × 5 columns

After sorting the values with respect to low RMSE value I'm choosing alpha = 0.1, beta = 0.6, gamma = 0.2.

```
                       ExponentialSmoothing Model Results
================================================================================
Dep. Variable:                Shoe_Sales   No. Observations:                  132
Model:            ExponentialSmoothing   SSE                         685432.249
Optimized:                        True   AIC                           1161.260
Trend:                        Additive   BIC                           1207.385
Seasonal:                Multiplicative   AICC                          1167.313
Seasonal Periods:                    12   Date:                 Thu, 31 Mar 2022
Box-Cox:                          False   Time:                         12:50:32
Box-Cox Coeff.:                    None
================================================================================
                          coeff                 code              optimized
--------------------------------------------------------------------------------
smoothing_level         0.1000000                alpha                  False
smoothing_trend         0.6000000                 beta                  False
smoothing_seasonal      0.2000000                gamma                  False
initial_level           155.09653                  l.0                   True
initial_trend           1.7443254                  b.0                   True
initial_seasons.0       0.5742793                  s.0                   True
initial_seasons.1       0.4176835                  s.1                   True
initial_seasons.2       0.4487717                  s.2                   True
initial_seasons.3       0.6438386                  s.3                   True
initial_seasons.4       0.6111264                  s.4                   True
initial_seasons.5       0.5436801                  s.5                   True
initial_seasons.6       0.5751512                  s.6                   True
initial_seasons.7       0.5755828                  s.7                   True
initial_seasons.8       0.7256223                  s.8                   True
initial_seasons.9       0.8113247                  s.9                   True
initial_seasons.10      0.9504114                 s.10                   True
initial_seasons.11      1.0144198                 s.11                   True
```

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

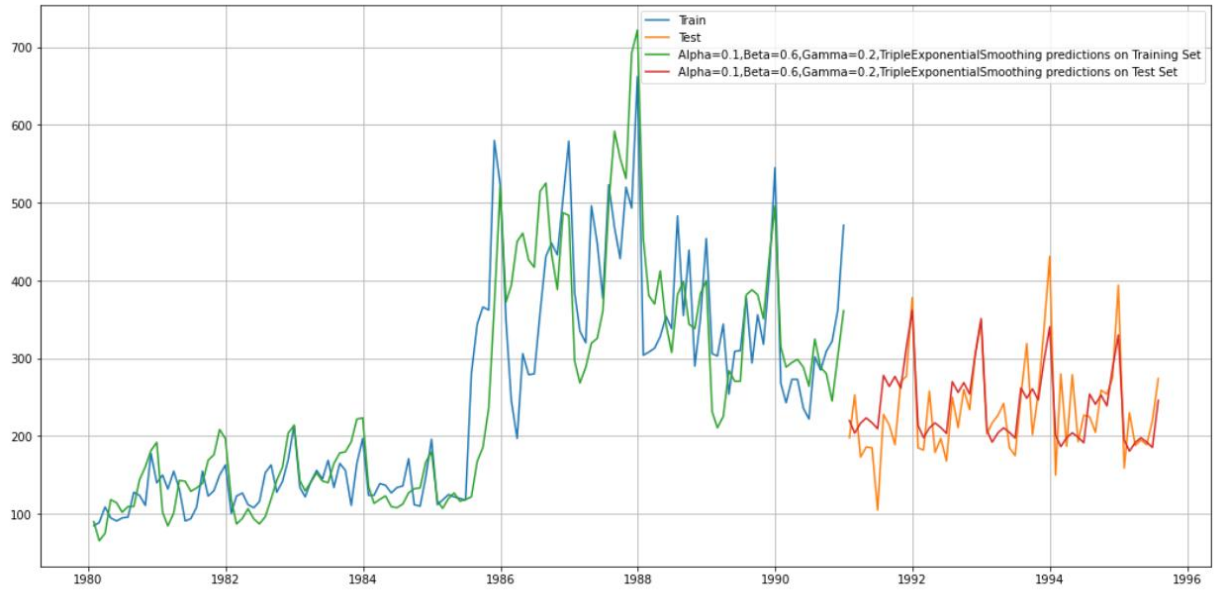


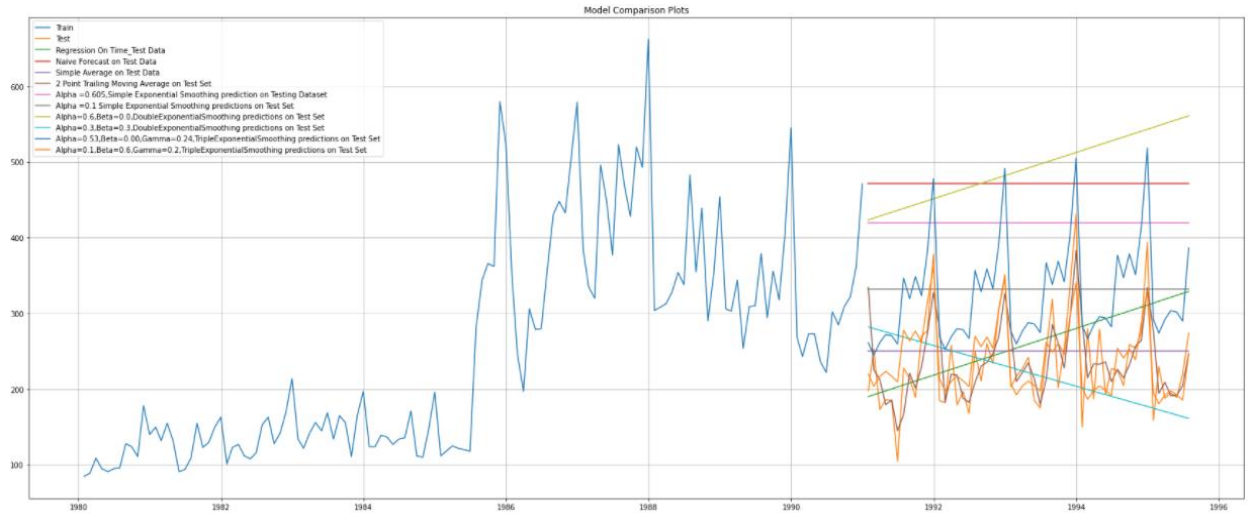
Figure 26 TES train-test plot

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.

For Alpha=0.1, Beta=0.6, Gamma=0.2, Triple Exponential Smoothing **RMSE = 41.314898**

We can now see that the best optimized model is the Triple Exponential Smoothing with multiplicative seasonality with the parameters, alpha = 0.1, beta = 0.6 and gamma = 0.2.The Triple exponential model is having both the trend and seasonality, it can work better compared to the other models.

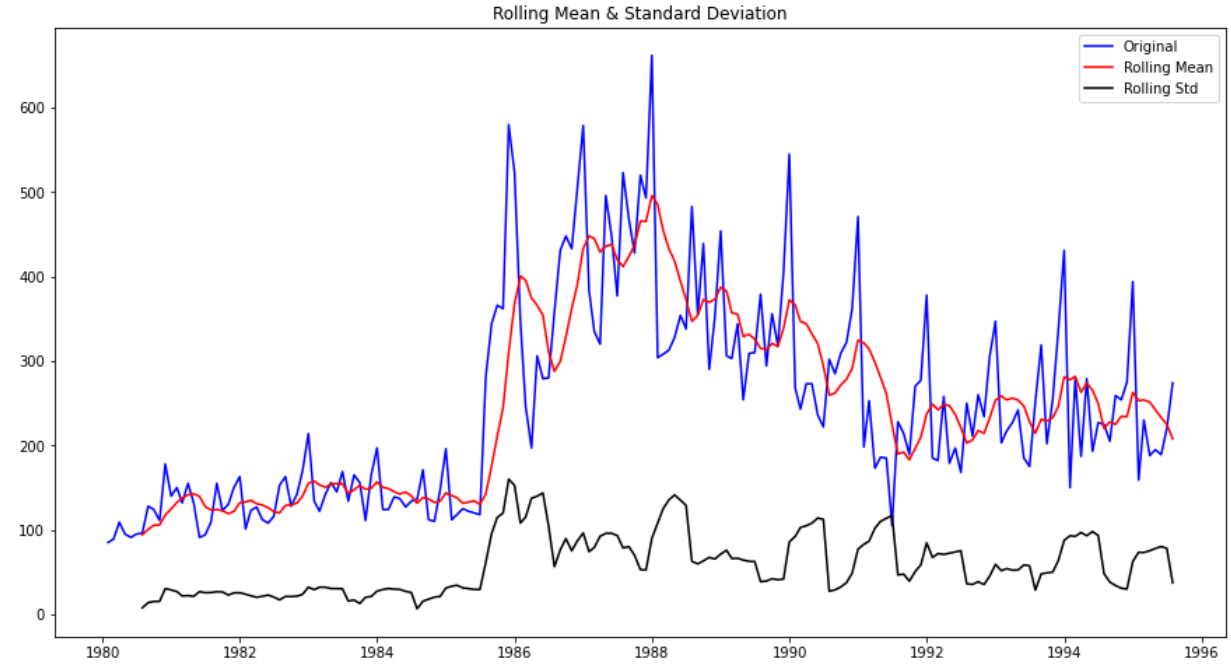
For the purpose of better understanding we are performing all the models and comparing them with respect to the RMSE values. Before moving to the ARIMA models, let us plot all the models and compare the Time Series plots.



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 nonstationary, take appropriate steps to make it stationary. Check the new data for stationarity and comment. Note: Stationarity should be checked at alpha = 0.05.

Check for stationarity of the series at α = 0.05 using appropriate statistical tests:




```
Results of Dickey-Fuller Test:
Test Statistic                  -1.717397
p-value                          0.422172
#Lags Used                      13.000000
Number of Observations Used    173.000000
Critical Value (1%)             -3.468726
Critical Value (5%)             -2.878396
Critical Value (10%)            -2.575756
dtype: float64
```

From the above 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.

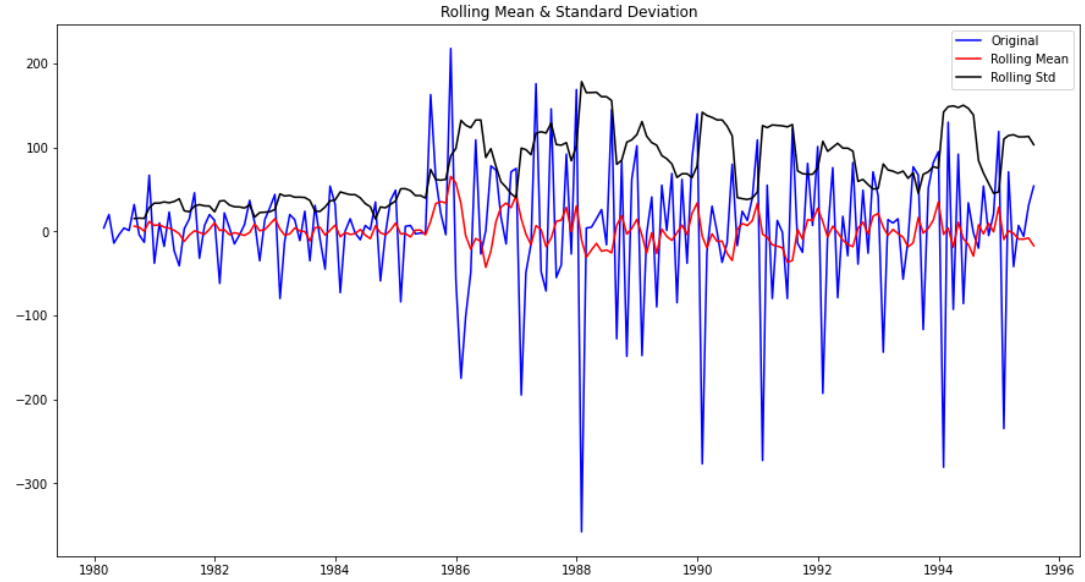
So, Let's take a difference of order 1 and check whether the Time Series is stationary or not.

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

since the p-value of the ADF test is greater than the critical value at 5%, we cannot reject the null hypothesis.

Hence, the given time given series is non stationary.




```
Results of Dickey-Fuller Test:
Test Statistic                  -3.479160
p-value                          0.008539
#Lags Used                      12.000000
Number of Observations Used    173.000000
Critical Value (1%)             -3.468726
Critical Value (5%)             -2.878396
Critical Value (10%)            -2.575756
dtype: float64
```

Now, we can see that after taking a difference of order 1, at $\alpha = 0.05$ the Time Series is indeed stationary.

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.

Here, for q we can take lag 0 i.e., q=0.

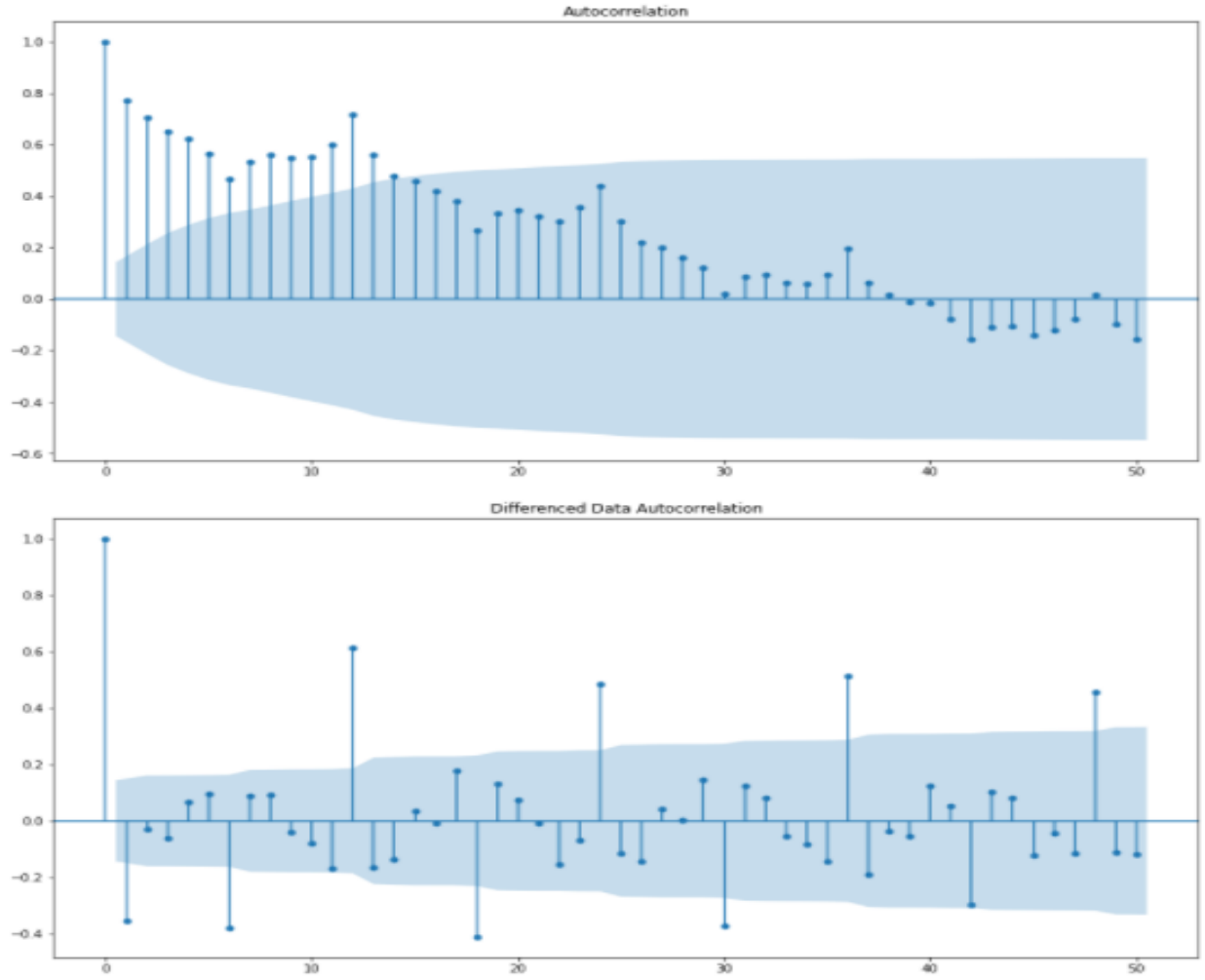


Figure 27 ACF Plots

From the above PACF Plot below,

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.

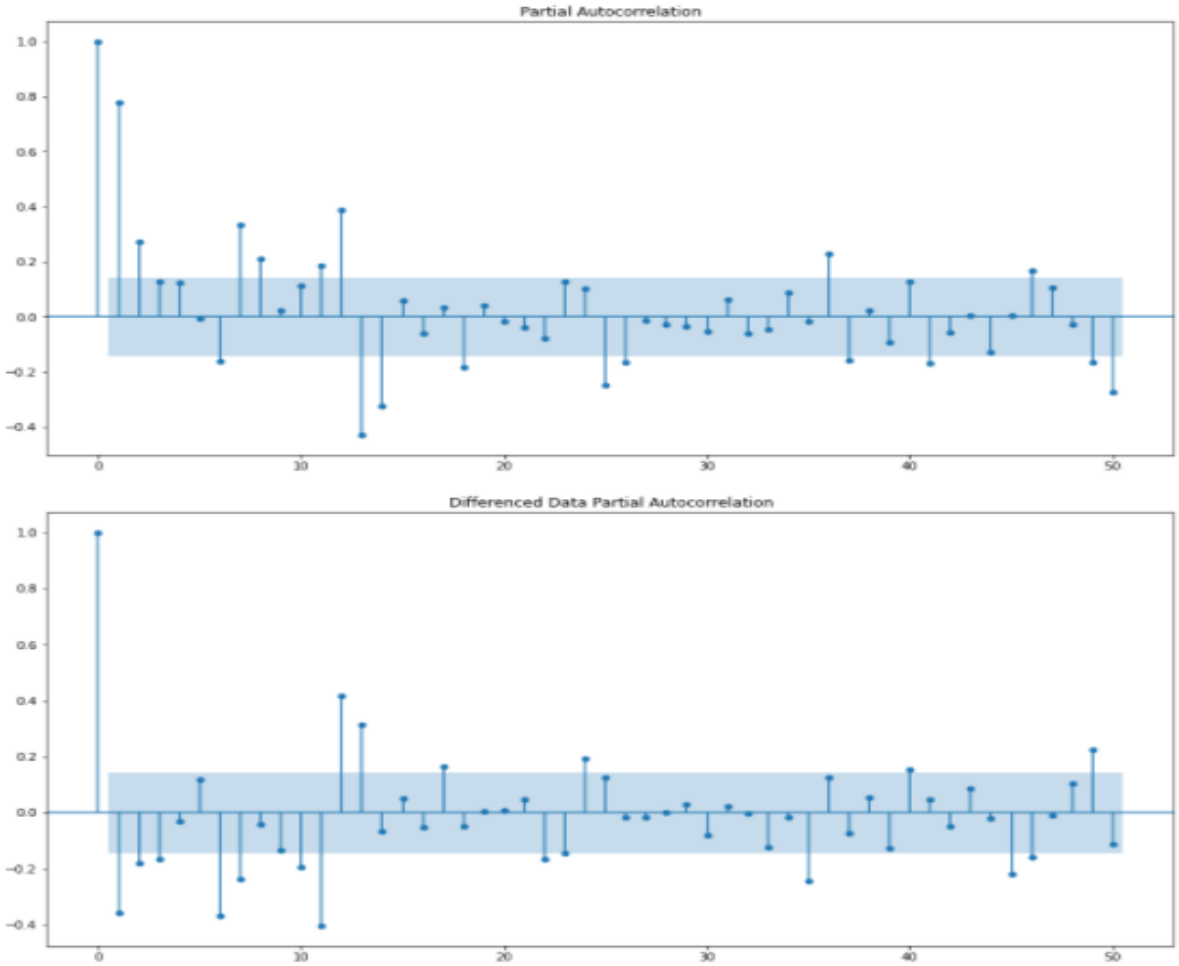
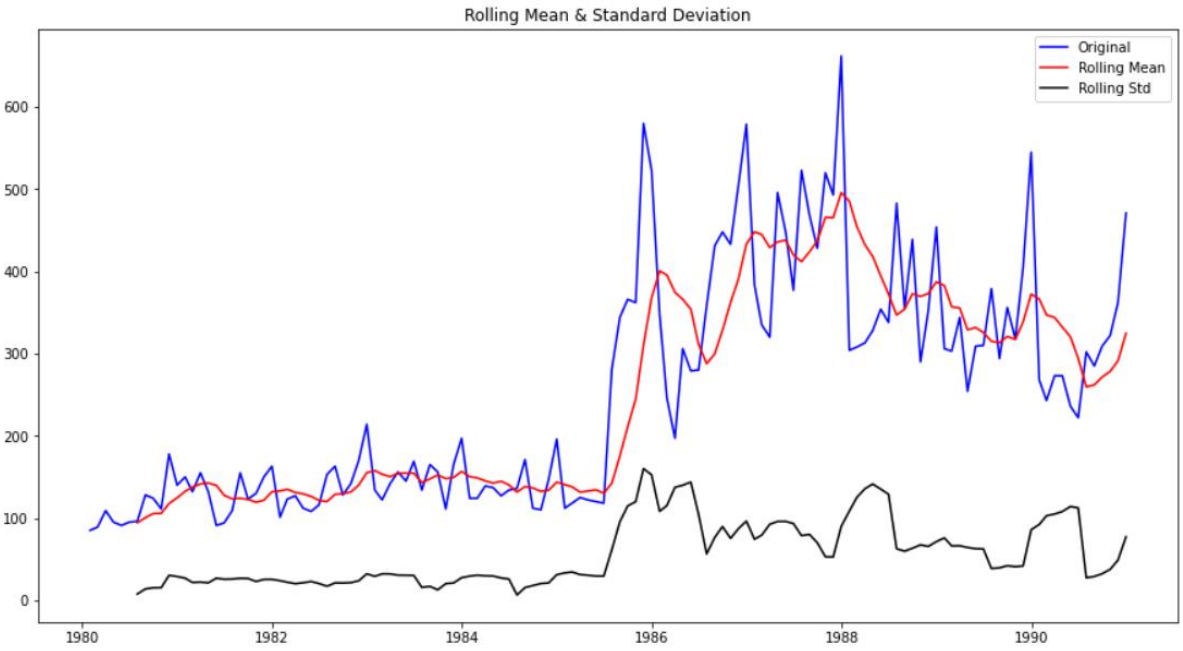


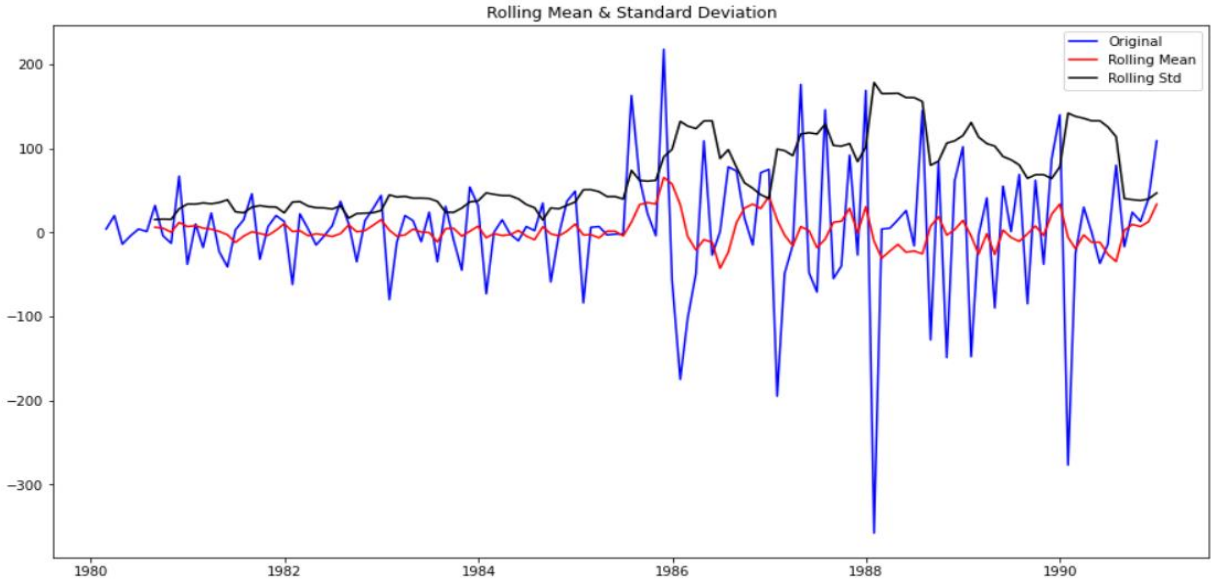
Figure 28 PACF plots

Check for stationarity of the Training Data Time Series:



```
Results of Dickey-Fuller Test:
Test Statistic                  -1.361129
p-value                          0.600763
#Lags Used                      13.000000
Number of Observations Used    118.000000
Critical Value (1%)             -3.487022
Critical Value (5%)             -2.886363
Critical Value (10%)            -2.580009
dtype: float64
```

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




```
Results of Dickey-Fuller Test:
Test Statistic                  -3.144211
p-value                          0.023450
#Lags Used                      13.000000
Number of Observations Used    117.000000
Critical Value (1%)             -3.487517
Critical Value (5%)             -2.886578
Critical Value (10%)            -2.580124
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 it 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.

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 have created an empty data frame to store the values. we are passing the values to model with the values of train data of Shoe 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:1508.2837722095956
ARIMA(0, 1, 1) - AIC:1497.0503224187926
ARIMA(0, 1, 2) - AIC:1494.9646053663441
ARIMA(1, 1, 0) - AIC:1501.64312420119
ARIMA(1, 1, 1) - AIC:1492.487186507898
ARIMA(1, 1, 2) - AIC:1494.4238594575386
ARIMA(2, 1, 0) - AIC:1498.9504830258975
ARIMA(2, 1, 1) - AIC:1494.4314983035438
ARIMA(2, 1, 2) - AIC:1496.4107391784296
```

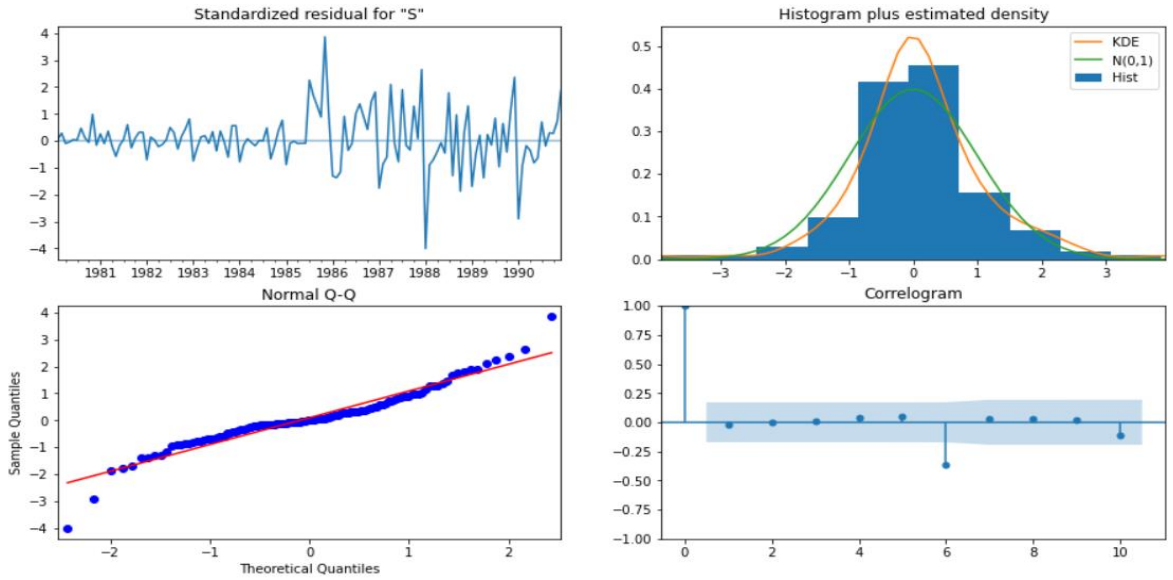
Sort the above AIC values in the ascending order to get the parameters for the minimum AIC value.

	param	AIC
4	(1, 1, 1)	1492.487187
5	(1, 1, 2)	1494.423859
7	(2, 1, 1)	1494.431498
2	(0, 1, 2)	1494.964605
8	(2, 1, 2)	1496.410739
1	(0, 1, 1)	1497.050322
6	(2, 1, 0)	1498.950483
3	(1, 1, 0)	1501.643124
0	(0, 1, 0)	1508.283772

After sorting the values I'm choosing (1,1,1) with AIC = 1492.487187. we can now pass these parameters and see the statistical summary.

```
==============================================================================
Dep. Variable:              Shoe_Sales   No. Observations:                  132
Model:                 ARIMA(1, 1, 1)   Log Likelihood                -743.244
Date:                Thu, 31 Mar 2022   AIC                           1492.487
Time:                        13:52:04   BIC                           1501.113
Sample:                    01-31-1980   HQIC                          1495.992
                         - 12-31-1990
Covariance Type:                  opg
==============================================================================
                 coef    std err          z      P>|z|      [0.025      0.975]
------------------------------------------------------------------------------
ar.L1          0.4699      0.111      4.235      0.000       0.252       0.687
ma.L1         -0.8347      0.068    -12.261      0.000      -0.968      -0.701
sigma2      4944.0869    405.583     12.190      0.000    4149.158    5739.015
===================================================================================
Ljung-Box (L1) (Q):                   0.05   Jarque-Bera (JB):                57.30
Prob(Q):                              0.83   Prob(JB):                         0.00
Heteroskedasticity (H):              12.81   Skew:                             0.01
Prob(H) (two-sided):                  0.00   Kurtosis:                         6.24
===================================================================================

Warnings:
[1] Covariance matrix calculated using the outer product of gradients (complex-step).
```



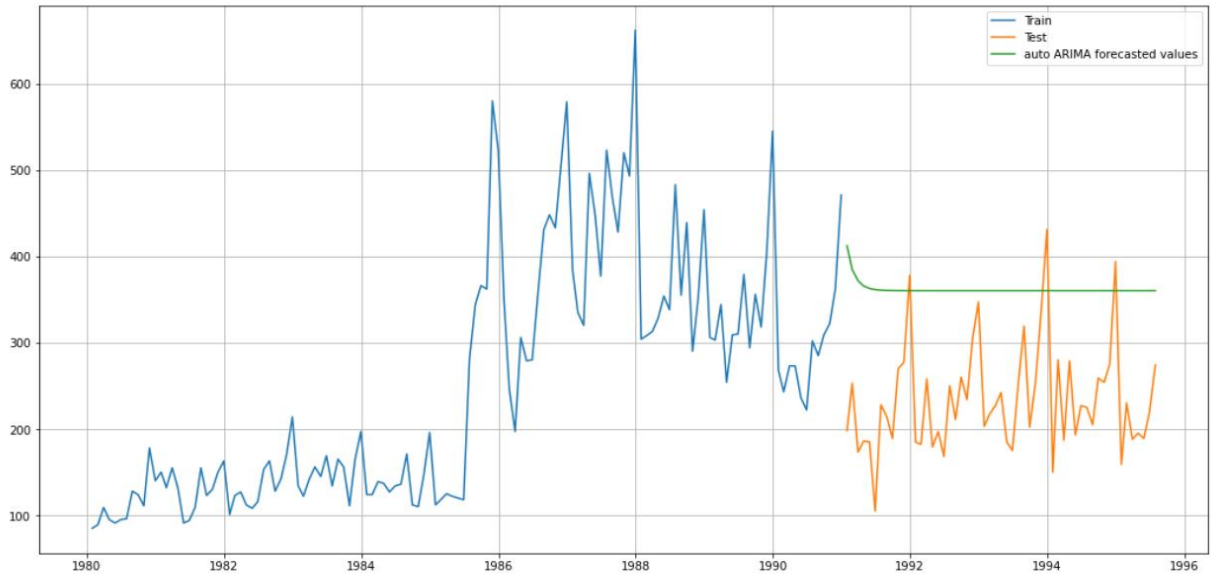


Figure 29 Autofit ARIMA plot

RMSE for the autofit ARIMA model: 142.82073046247484

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.

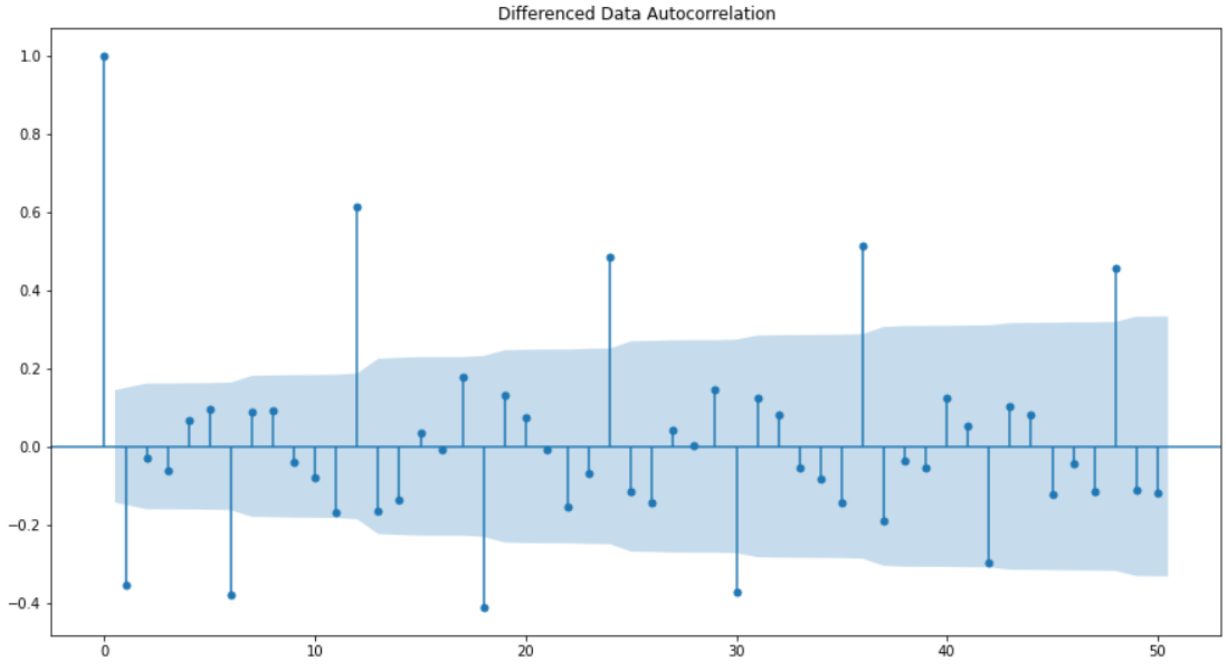
They are the same as the ARIMA model;

- **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.

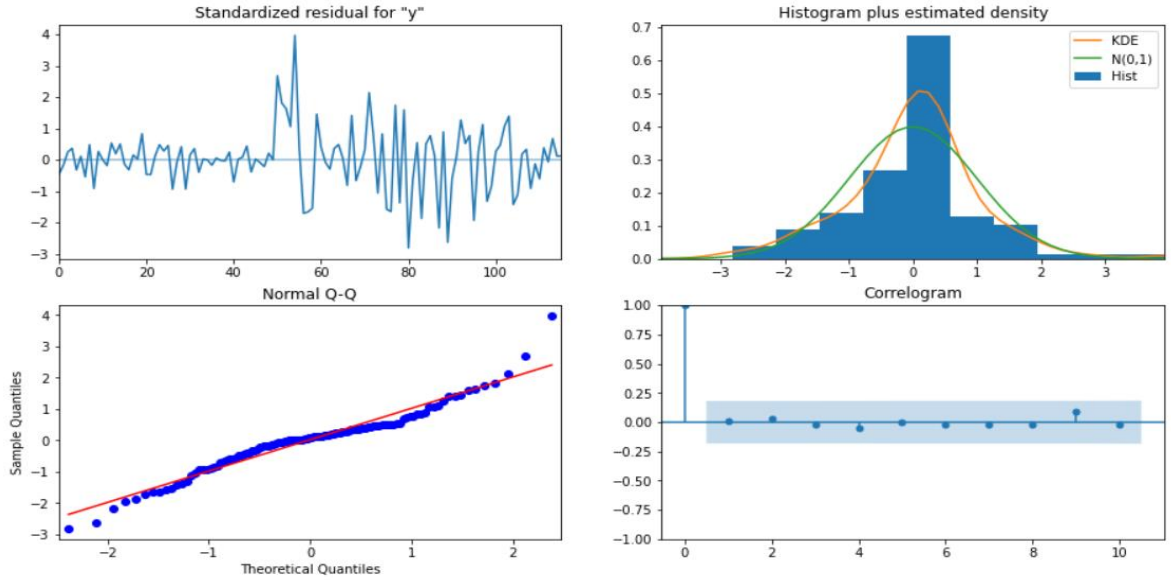
	param	seasonal	AIC
80	(2, 1, 2)	(2, 0, 2, 6)	1280.778664
26	(0, 1, 2)	(2, 0, 2, 6)	1281.026602
53	(1, 1, 2)	(2, 0, 2, 6)	1282.065372
17	(0, 1, 1)	(2, 0, 2, 6)	1288.975663
50	(1, 1, 2)	(1, 0, 2, 6)	1289.791748

After sorting, the combination (2, 1, 2) (2, 0, 2, 6) have less AIC = 1280.778664.

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(2, 1, 2)x(2, 0, 2, 6)   Log Likelihood                -631.389
Date:                           Thu, 31 Mar 2022   AIC                           1280.779
Time:                                   19:51:16   BIC                           1305.561
Sample:                                        0   HQIC                          1290.839
                                             - 132
Covariance Type:                             opg
==============================================================================
                 coef    std err          z      P>|z|      [0.025      0.975]
------------------------------------------------------------------------------
ar.L1          0.0601      0.474      0.127      0.899      -0.869       0.989
ar.L2          0.3977      0.161      2.464      0.014       0.081       0.714
ma.L1         -0.4650      0.492     -0.946      0.344      -1.429       0.498
ma.L2         -0.3238      0.295     -1.096      0.273      -0.903       0.255
ar.S.L6       -0.1732      0.136     -1.278      0.201      -0.439       0.092
ar.S.L12       0.7913      0.130      6.102      0.000       0.537       1.046
ma.S.L6        0.1151      0.184      0.626      0.531      -0.245       0.475
ma.S.L12      -0.3010      0.179     -1.683      0.092      -0.652       0.050
sigma2      3080.4611    348.196      8.847      0.000    2398.010    3762.912
==========================================================================================
Ljung-Box (L1) (Q):                   0.01   Jarque-Bera (JB):                21.55
Prob(Q):                              0.92   Prob(JB):                         0.00
Heteroskedasticity (H):               7.76   Skew:                             0.18
Prob(H) (two-sided):                  0.00   Kurtosis:                         5.08
==========================================================================================

Warnings:
[1] Covariance matrix calculated using the outer product of gradients (complex-step).
```



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

y	mean	mean_se	mean_ci_lower	mean_ci_upper
0	257.242984	55.501902	148.461256	366.024713
1	257.175466	64.584738	130.591706	383.759226
2	265.714757	73.831752	121.007182	410.422331
3	263.007825	78.614949	108.925355	417.090294
4	240.045923	83.309217	76.762857	403.328989

RMSE for the autofit SARIMA model: 57.031260189099534

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
23	(0, 1, 2)	(1, 0, 2, 12)	1156.165429
50	(1, 1, 2)	(1, 0, 2, 12)	1157.082589
26	(0, 1, 2)	(2, 0, 2, 12)	1157.772313
77	(2, 1, 2)	(1, 0, 2, 12)	1158.490998
80	(2, 1, 2)	(2, 0, 2, 12)	1158.630324

After sorting, the combination (0, 1, 2) (1, 0, 2, 12) have less AIC = 1156.165429

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

```
                                 SARIMAX Results
==========================================================================================
Dep. Variable:                        Shoe_Sales   No. Observations:                  132
Model:             SARIMAX(0, 1, 2)x(1, 0, 2, 12)   Log Likelihood                -572.083
Date:                            Thu, 31 Mar 2022   AIC                           1156.165
Time:                                    19:59:01   BIC                           1172.032
Sample:                                01-31-1980   HQIC                          1162.593
                                     - 12-31-1990
Covariance Type:                              opg
==============================================================================
                 coef    std err          z      P>|z|      [0.025      0.975]
------------------------------------------------------------------------------
ma.L1         -0.3742      0.081     -4.632      0.000      -0.533      -0.216
ma.L2          0.0616      0.077      0.803      0.422      -0.089       0.212
ar.S.L12       1.0635      0.051     21.042      0.000       0.964       1.163
ma.S.L12      -0.7636    292.486     -0.003      0.998    -574.025     572.498
ma.S.L24      -0.2365     69.082     -0.003      0.997    -135.634     135.161
sigma2      2818.0945   8.24e+05      0.003      0.997   -1.61e+06    1.62e+06
===================================================================================
Ljung-Box (L1) (Q):                   0.01   Jarque-Bera (JB):                 8.60
Prob(Q):                              0.91   Prob(JB):                         0.01
Heteroskedasticity (H):               7.92   Skew:                            -0.15
Prob(H) (two-sided):                  0.00   Kurtosis:                         4.38
===================================================================================

Warnings:
[1] Covariance matrix calculated using the outer product of gradients (complex-step).
[2] Covariance matrix is singular or near-singular, with condition number 8.67e+14. Standard errors may be unstable.
```

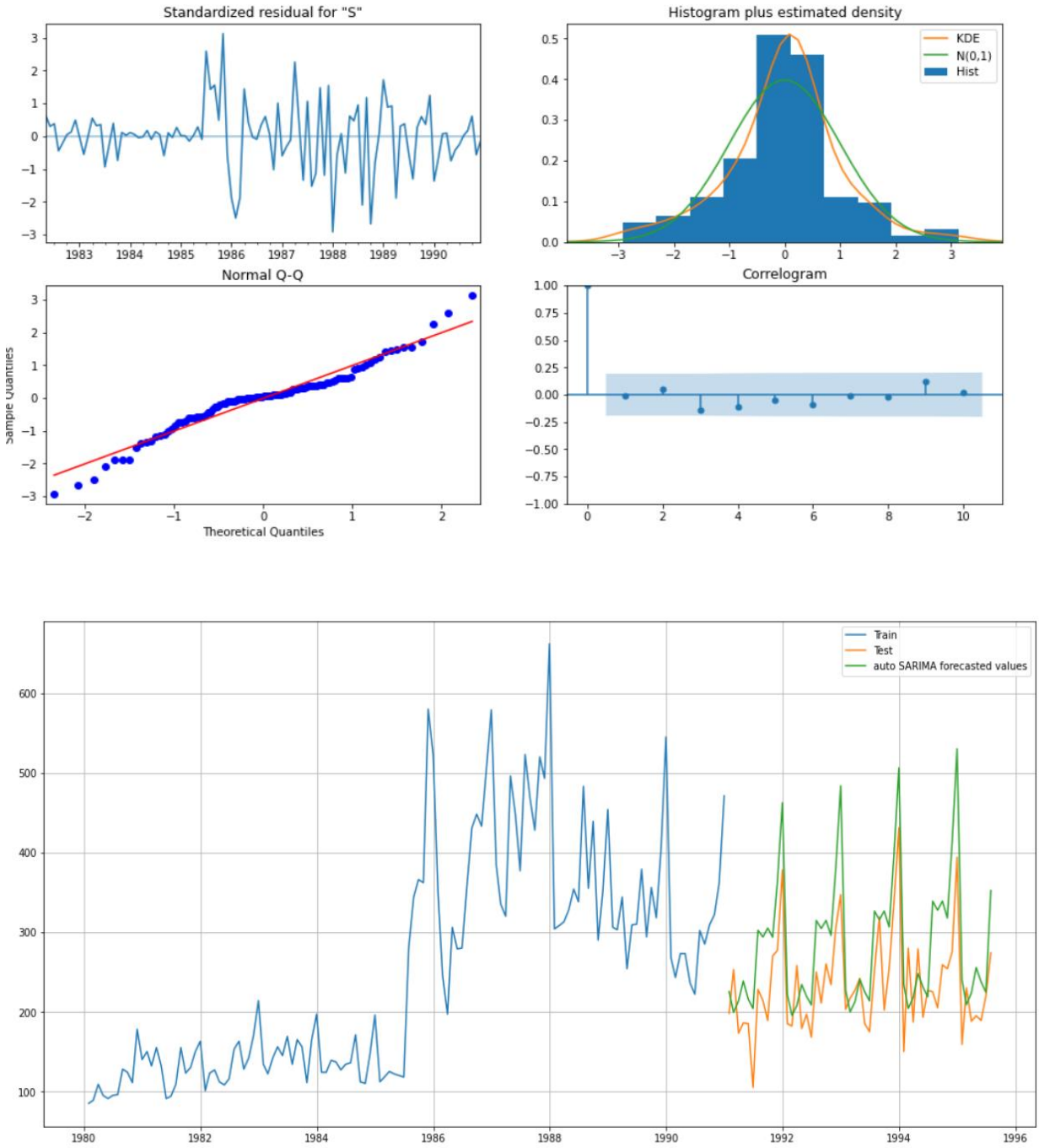


Figure 30 Autofit SARIMA

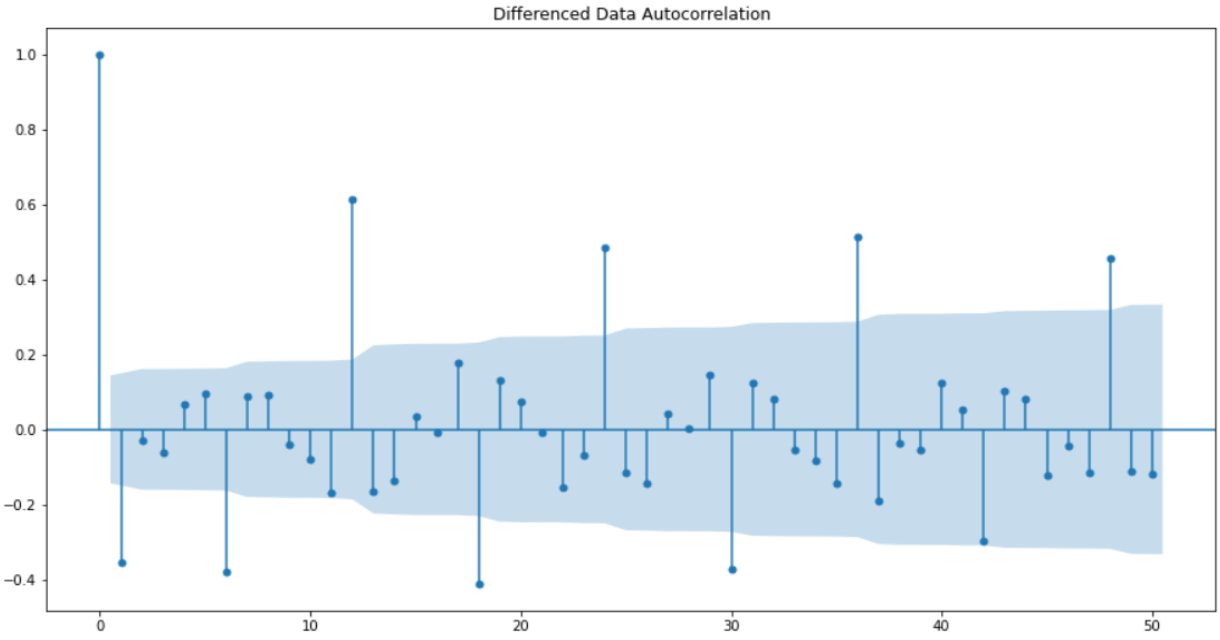
RMSE for the autofit SARIMA model: 69.03066432738571

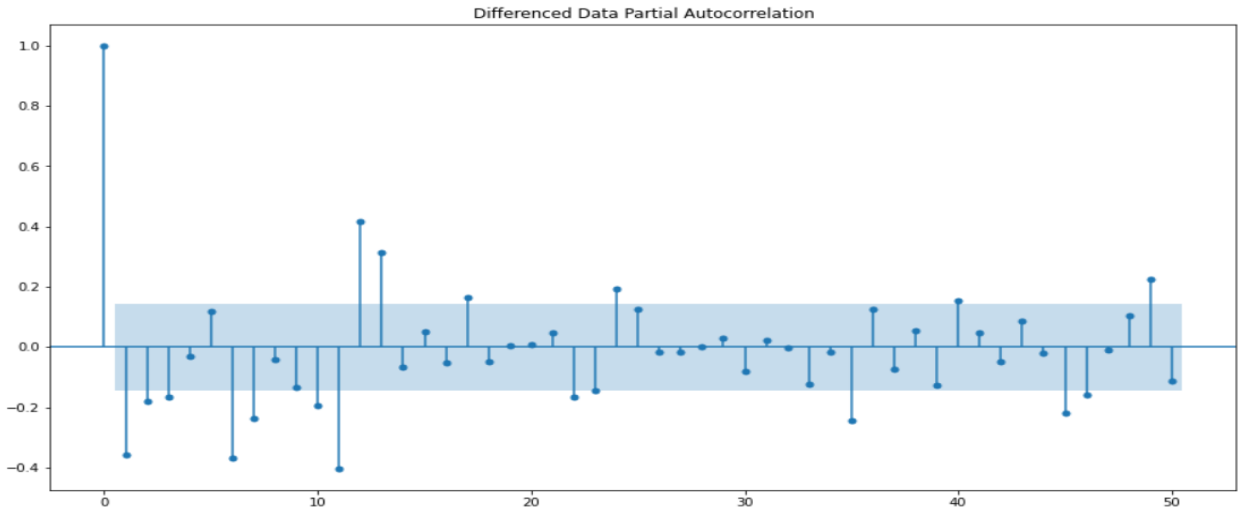
	Test RMSE
RegressionOnTime	73.111522
NaiveModel	245.121306
SimpleAverageModel	63.984570
2pointTrailingMovingAverage	45.948736
4pointTrailingMovingAverage	57.872686
6pointTrailingMovingAverage	63.456893
9pointTrailingMovingAverage	67.723648
Alpha=0,SimpleExponentialSmoothing	196.404847
Alpha=0.1,SimpleExponentialSmoothing	115.874466
Alpha=0.59 and Beta=0,DoubleExponentialSmoothing	267.251583
Alpha=0.1,Beta=0.1,DoubleExponentialSmoothing	76.884339
Alpha: 0.112,Beta: 0.037 and Gamma:0.493,TripleExponentialSmoothing	99.428217
Alpha=0.1,Beta=0.6,Gamma=0.2,TripleExponentialSmoothing	41.314898
automated ARIMA(1,1,1)	142.820730
automatedSARIMA(2,1,2)(2,0,2,6)	57.031260
automated SARIMA(0,1,2)*(1,0,2,12)	69.030664

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





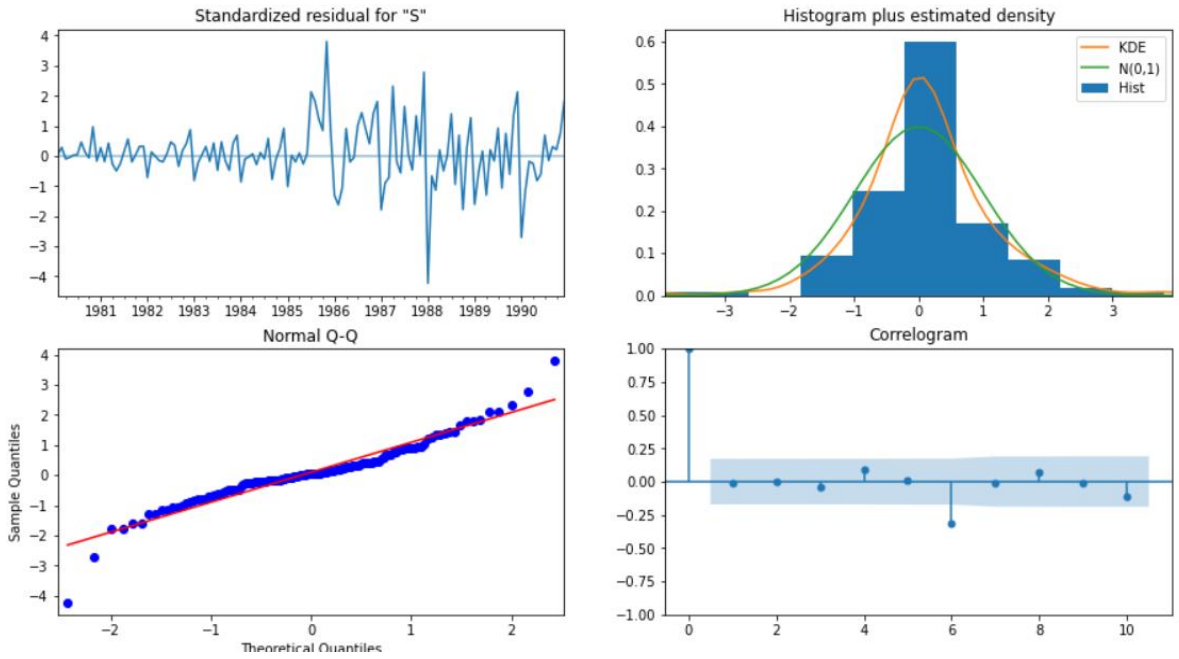
SARIMAX Results

Dep. Variable:	Shoe_Sales	No. Observations:	132
Model:	ARIMA(3, 1, 2)	Log Likelihood	-741.828
Date:	Thu, 31 Mar 2022	AIC	1495.656
Time:	20:31:38	BIC	1512.907
Sample:	01-31-1980 - 12-31-1990	HQIC	1502.666
Covariance Type:	opg		

	coef	std err	z	P>|z|	[0.025	0.975]
ar.L1	-0.4389	0.146	-3.009	0.003	-0.725	-0.153
ar.L2	0.4463	0.157	2.850	0.004	0.139	0.753
ar.L3	0.0809	0.115	0.701	0.484	-0.145	0.307
ma.L1	0.0853	0.119	0.714	0.475	-0.149	0.319
ma.L2	-0.8011	0.093	-8.640	0.000	-0.983	-0.619
sigma2	4832.0089	415.928	11.617	0.000	4016.804	5647.213

Ljung-Box (L1) (Q):	0.01	Jarque-Bera (JB):	66.57
Prob(Q):	0.92	Prob(JB):	0.00
Heteroskedasticity (H):	11.39	Skew:	-0.09
Prob(H) (two-sided):	0.00	Kurtosis:	6.49

Warnings:
[1] Covariance matrix calculated using the outer product of gradients (complex-step).



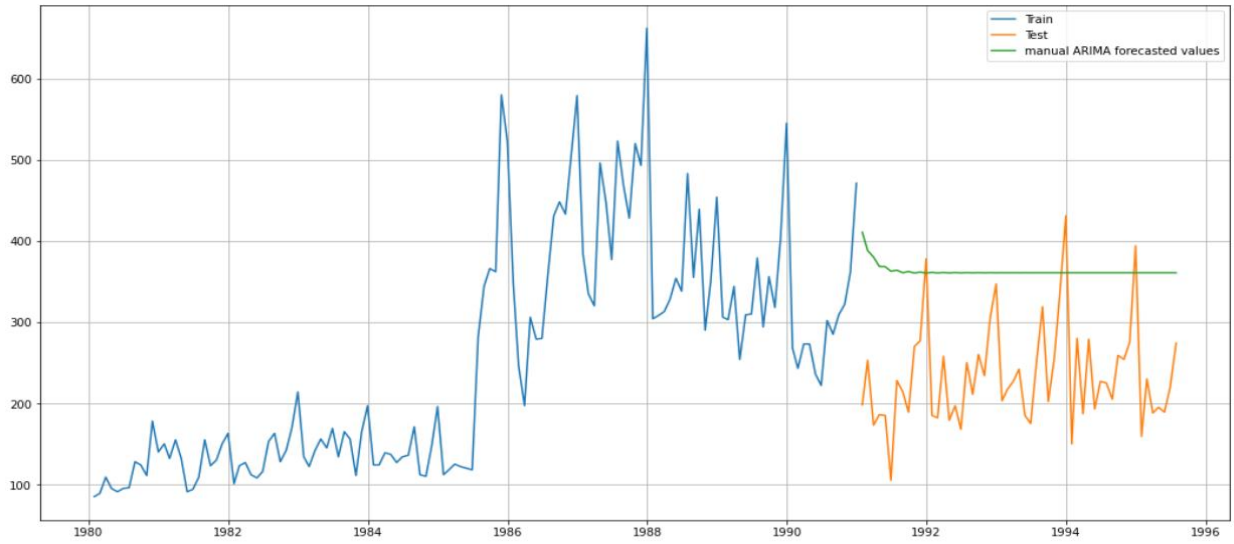
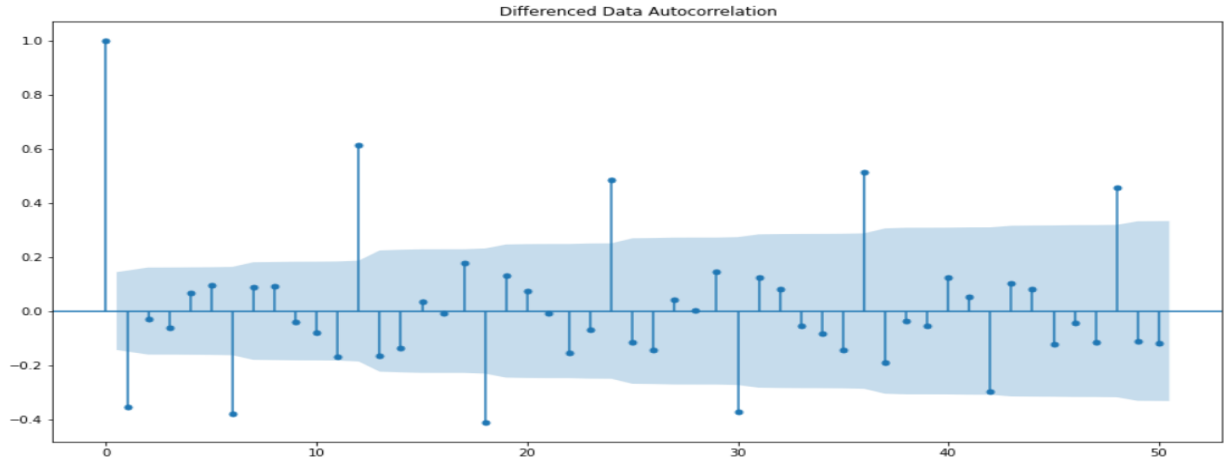
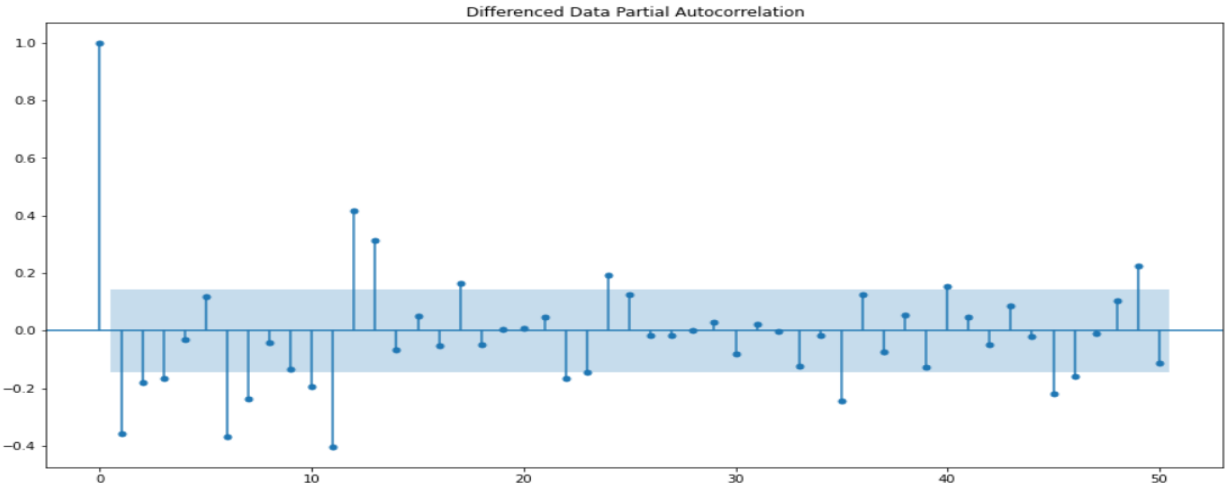


Figure 31 Manual ARIMA

Model14: manual SARIMA

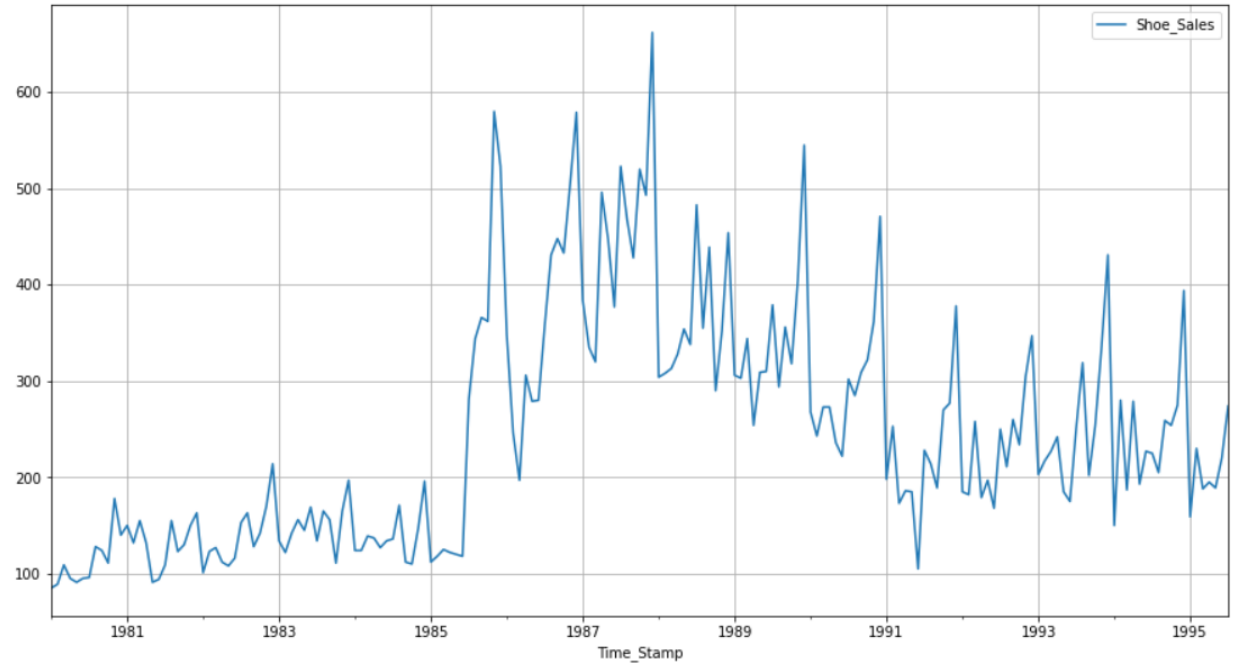




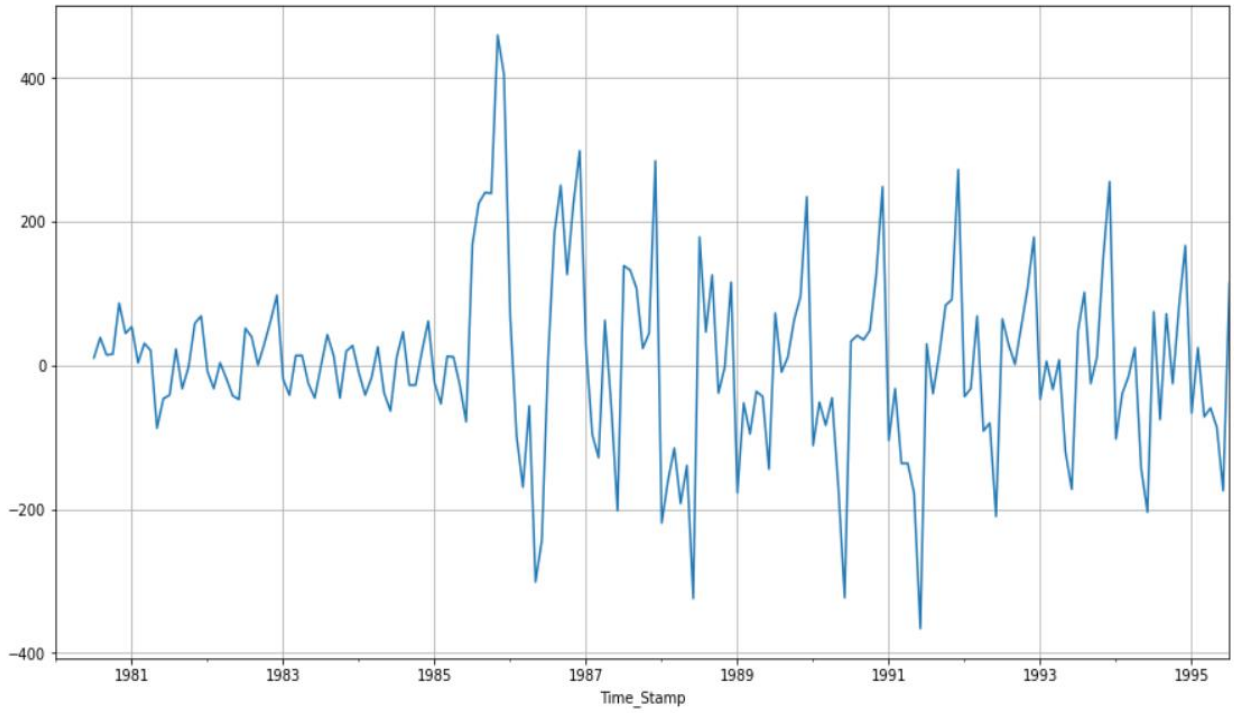
We can observe that the above ACF plot at the seasonal interval (6) does not taper off.

Hence, we are taking a seasonal differencing of original series.

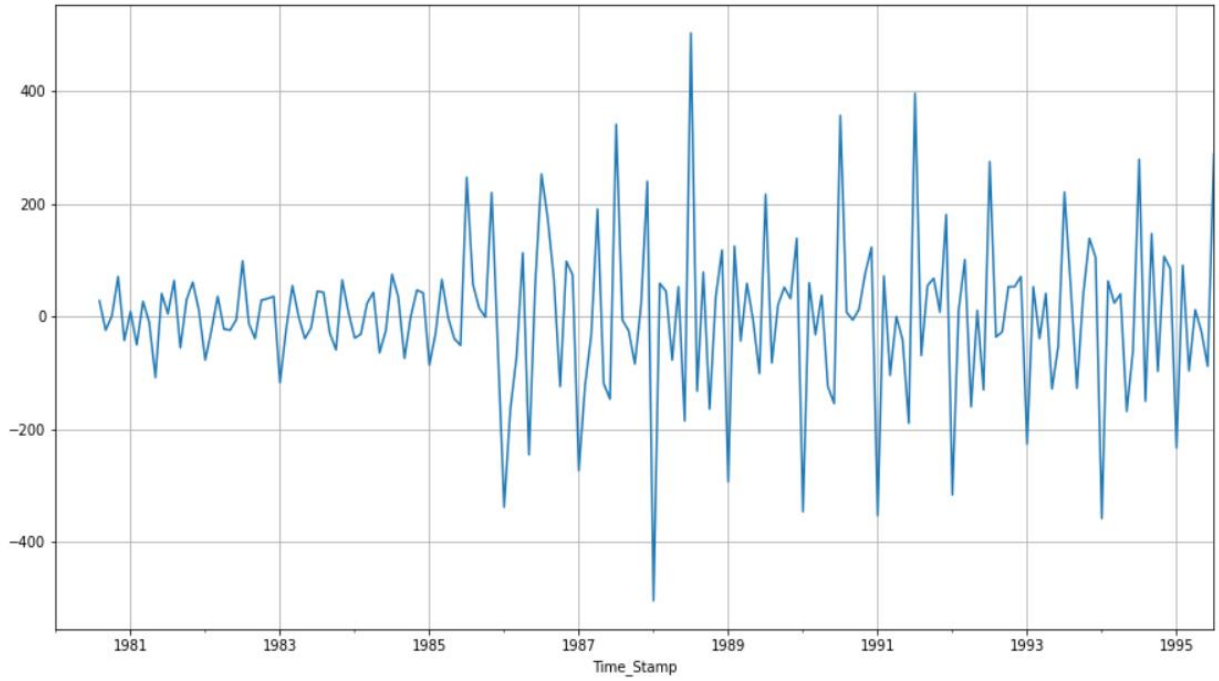
Before that let us look at the original series.



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

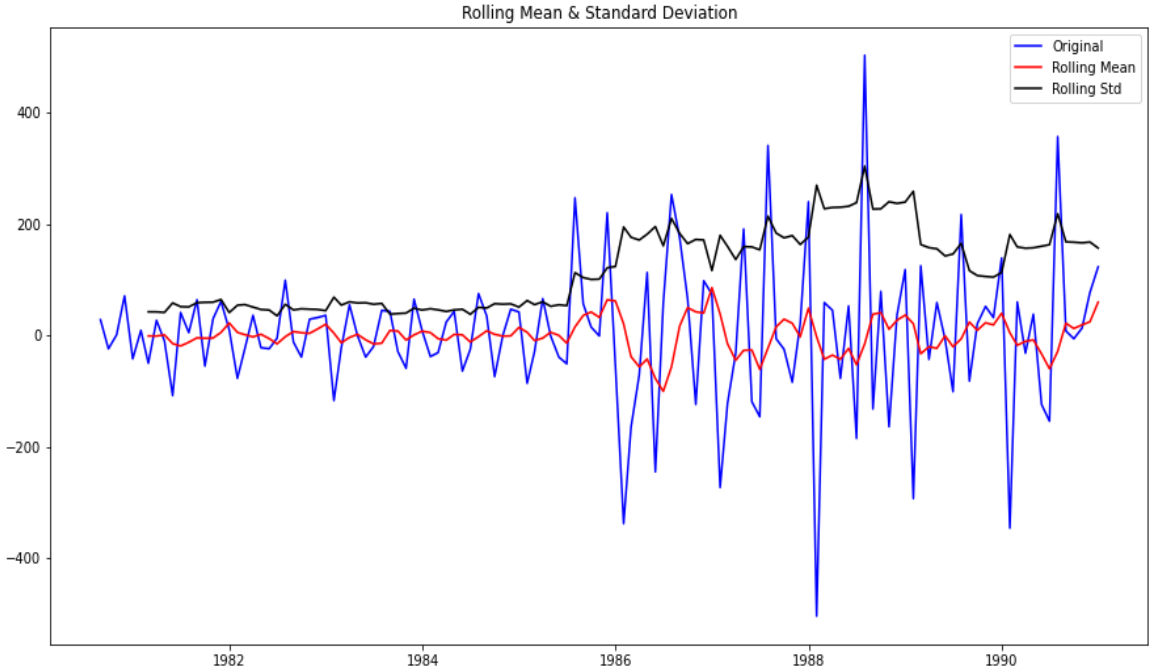


There is a slight trend which we can observe in the data. So, we take a differencing of first order on the seasonally differenced series.



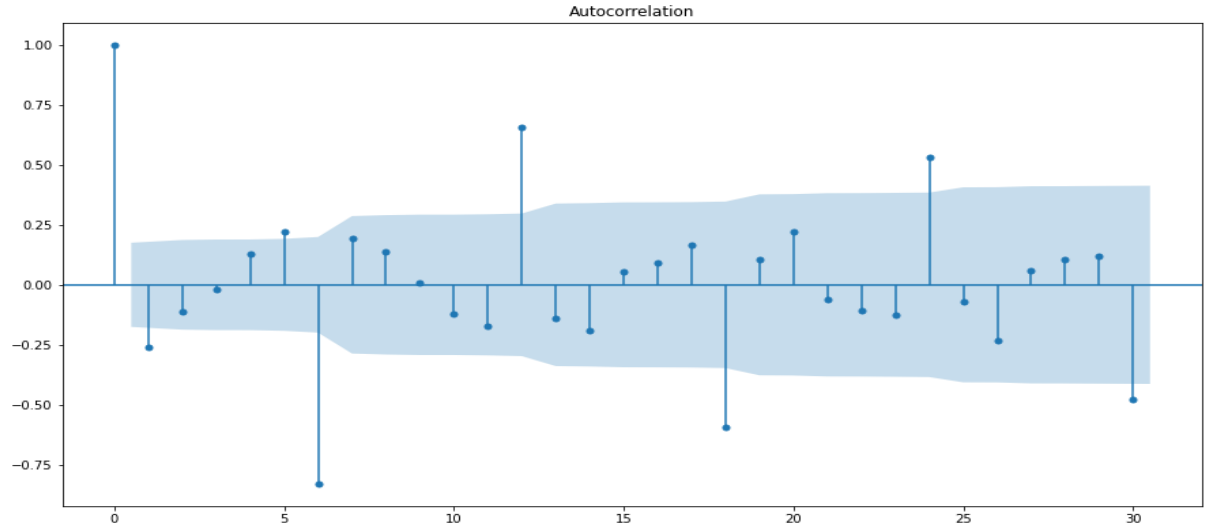
Here, There is almost no trend present in the data. Only the seasonality is present in the data.

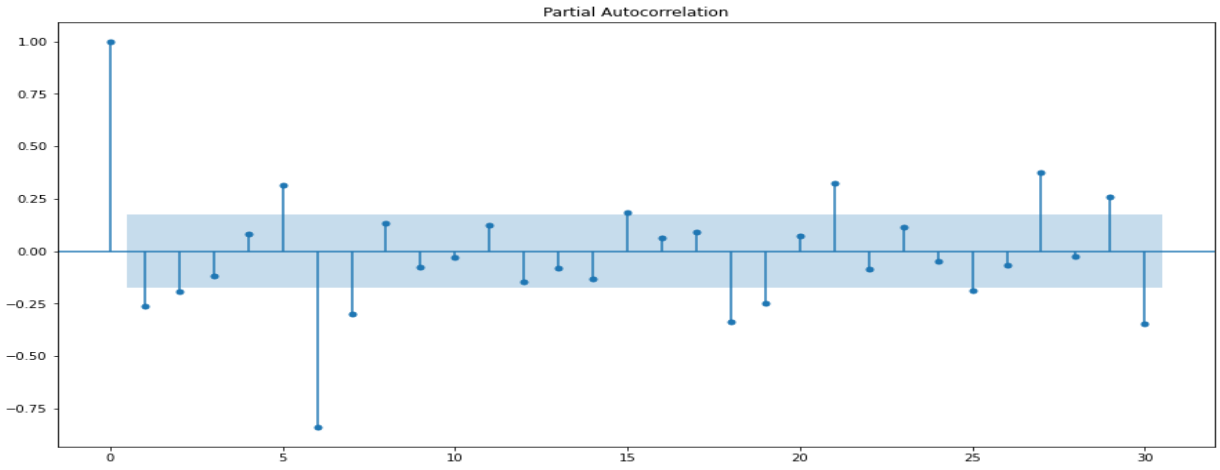
Check the stationarity of the above series before fitting the SARIMA model.




```
Results of Dickey-Fuller Test:
Test Statistic                -1.133336e+01
p-value                        1.098825e-20
#Lags Used                     6.000000e+00
Number of Observations Used    1.180000e+02
Critical Value (1%)           -3.487022e+00
Critical Value (5%)           -2.886363e+00
Critical Value (10%)          -2.580009e+00
dtype: float64
```

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





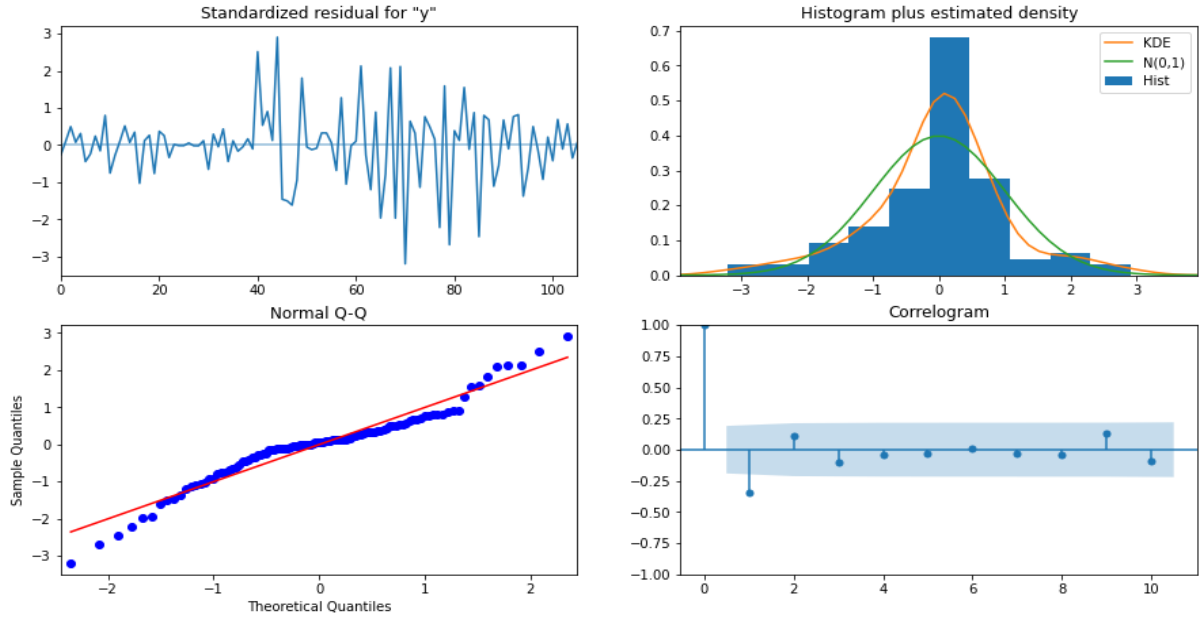
SARIMAX Results

Dep. Variable:	y	No. Observations:	132
Model:	SARIMAX(0, 1, 0)x(1, 1, [1, 2, 3], 6)	Log Likelihood	-589.644
Date:	Thu, 31 Mar 2022	AIC	1189.288
Time:	20:38:57	BIC	1202.605
Sample:	0	HQIC	1194.685
	- 132		
Covariance Type:	opg		

	coef	std err	z	P>|z|	[0.025	0.975]
ar.S.L6	-0.9858	0.047	-20.957	0.000	-1.078	-0.894
ma.S.L6	-0.1187	0.119	-1.000	0.317	-0.351	0.114
ma.S.L12	-0.5640	0.115	-4.883	0.000	-0.790	-0.338
ma.S.L18	-0.0688	0.099	-0.695	0.487	-0.263	0.125
sigma2	3836.2658	439.146	8.736	0.000	2975.555	4696.977

Ljung-Box (L1) (Q):	13.23	Jarque-Bera (JB):	11.20
Prob(Q):	0.00	Prob(JB):	0.00
Heteroskedasticity (H):	6.58	Skew:	-0.22
Prob(H) (two-sided):	0.00	Kurtosis:	4.53

Warnings:
[1] Covariance matrix calculated using the outer product of gradients (complex-step).



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

```
array([254.89133878, 248.92147552, 259.29271667, 260.77384175,
       256.43438182, 232.66162163, 329.06292284, 290.73929366,
       316.10849605, 309.62629596, 365.18709816, 468.96180254,
       259.08889548, 249.71510041, 260.74988546, 259.68413865,
       257.69042988, 235.43227917, 327.48619923, 289.57349582,
       314.73922964, 308.3339129 , 363.07776287, 465.06350546,
       259.47627104, 249.69738305, 260.93274855, 259.79120971,
       258.60285701, 238.10827484, 325.93735485, 288.4239943 ,
       313.39199478, 307.06139426, 361.01131972, 461.25852657,
       259.8361631 , 249.66360068, 261.09389282, 259.87869831,
       259.47300084, 240.69227692, 324.41560386, 287.29032976,
       312.06617048, 305.80818012, 358.98655967, 457.54423549,
       260.16934635, 249.61420614, 261.23393044, 259.94715643,
       260.30205323, 243.18687845, 322.92018256])
```

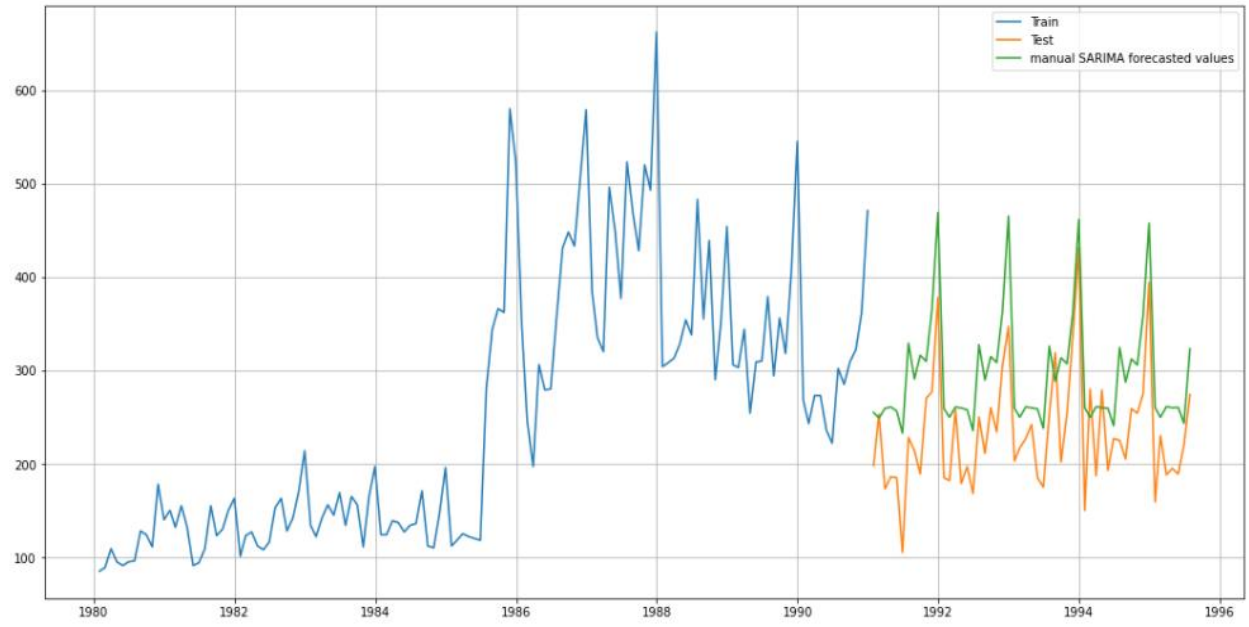


Figure 32 Manual SARIMA

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.

	Test RMSE
Alpha=0.1,Beta=0.6,Gamma=0.2,TripleExponentialSmoothing	41.314898
2pointTrailingMovingAverage	45.948736
automatedSARIMA(2,1,2)(2,0,2,6)	57.031260
4pointTrailingMovingAverage	57.872686
6pointTrailingMovingAverage	63.456893
SimpleAverageModel	63.984570
9pointTrailingMovingAverage	67.723648
automated SARIMA(0,1,2)*(1,0,2,12)	69.030664
manual SARIMA(0,1,0)(1,1,3,6)	70.531485
RegressionOnTime	73.111522
Alpha=0.1,Beta=0.1,DoubleExponentialSmoothing	76.884339
Alpha: 0.112,Beta: 0.037 and Gamma:0.493,TripleExponentialSmoothing	99.428217
Alpha=0.1,SimpleExponentialSmoothing	115.874466
automated ARIMA(1,1,1)	142.820730
manual ARIMA(3,1,2)	143.834590
Alpha=0,SimpleExponentialSmoothing	196.404847
NaiveModel	245.121306
Alpha=0.59 and Beta=0,DoubleExponentialSmoothing	267.251583

Alpha=0.1,Beta=0.6,Gamma=0.2, TripleExponentialSmoothing 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: 65.72957832870476

	lower_CI	prediction	upper_ci
1995-08-31	93.929888	223.070877	352.211866
1995-09-30	97.890459	227.031448	356.172437
1995-10-31	113.444192	242.585181	371.726170
1995-11-30	164.320184	293.461173	422.602162
1995-12-31	254.106202	383.247191	512.388180

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

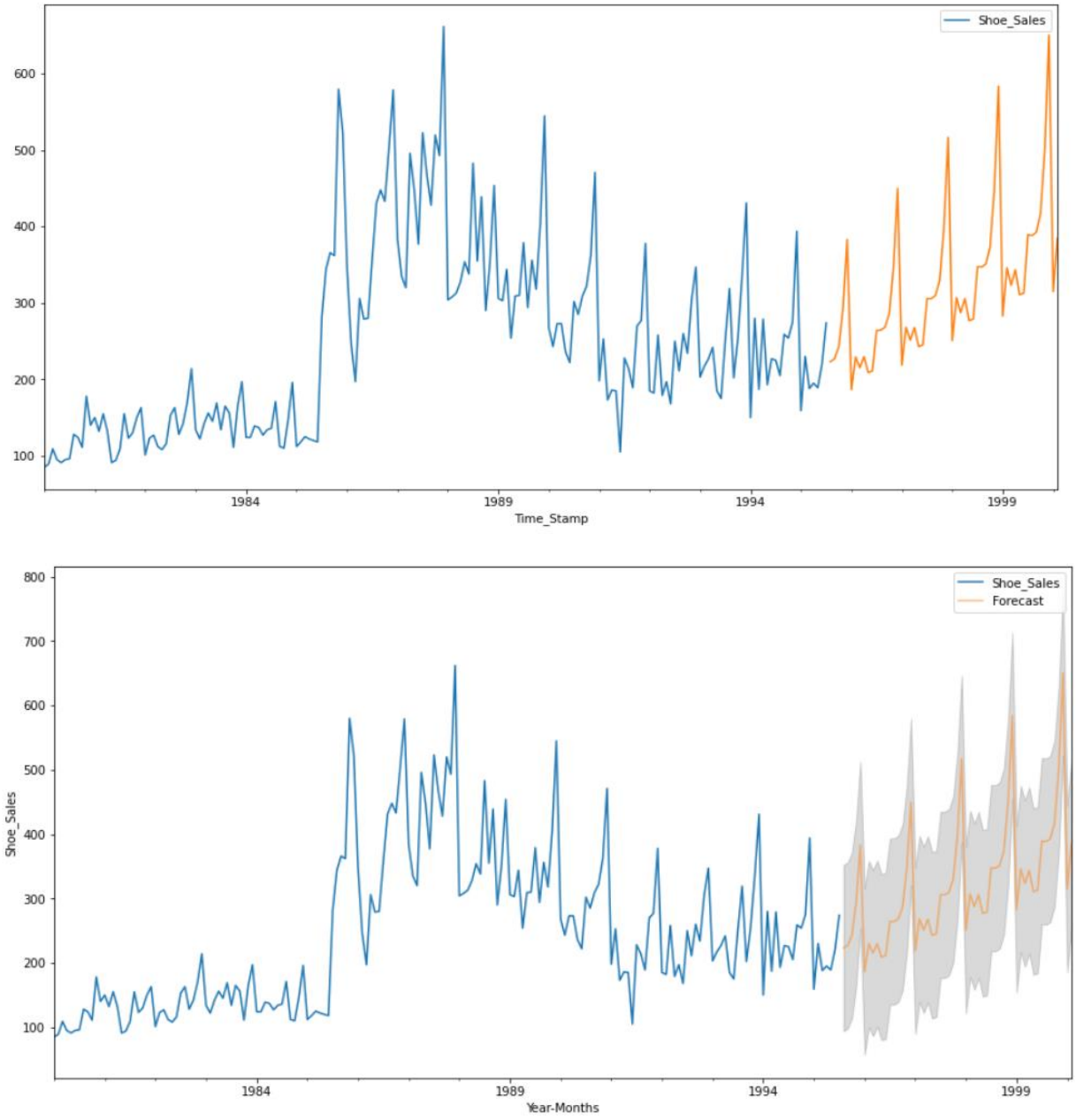


Figure 33 Final Model

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.

Any time series may be split into the following components:

Base Level + Trend + Seasonality + Error

Observations:

→Shoe Sales is higher in November and December months.

→The 1987 year having the peak of shoe sales in the month of December.

→The Time series data is having both increasing and decreasing trend as well as seasonality in the data.

→The Shoe sales were in high spike from 1984 to 1987. After that we can see a gradual decreasing and slight increasing fluctuations.

→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 mostly in November and December months 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.Need to introduce the new designs and brands of shoes according to the marketing trends and customer choices.

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

6.Displays need to look fresh and interesting and tell a compelling story about why the consumer should purchase the product.

7.Generally, Consumers get very excited about savings and appreciate discounts being passed on.

These changes can help to improve the sales in low demand seasons and also to maintain the same spike of sales during the high demand seasons.