Problem:

For this particular assignment, the data of different types of wine sales in the 20th century is to be analysed. Both of these data are from the same company but of different wines. As an analyst in the ABC Estate Wines, you are tasked to analyse and forecast Wine Sales in the 20th century.

Please do perform the following questions on each of these two data sets separately.

DataSet – Rose.csv

The dataset consists of 2 columns – Year-Month(date) and the sales for that particular month (continuous variable).We have to analyse the data, prepare models and predict the sales for next 12 months based on the Time Series.

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

Initially we import all the libraries : -

```
import pandas                           as      pd
import numpy                            as      np
import matplotlib.pyplot                as      plt
import statsmodels.tools.eval_measures  as      em
from   sklearn.metrics                  import  mean_squared_error
from   statsmodels.tsa.api              import ExponentialSmoothing, SimpleExpSmoothing, Holt
from   IPython.display                  import display
from   pylab                            import rcParams
import warnings
warnings.filterwarnings("ignore")
```

Now using the pandas library, we use the read_csv() to read the data.

	YearMonth	Rose
0	1980-01	112.0
1	1980-02	118.0
2	1980-03	129.0
3	1980-04	99.0
4	1980-05	116.0

But, since this is a time-series data, we need to change the format and it can be done either using parsedate or the other way is –

```
1  date = pd.date_range(start='1/1/1980', end='8/1/1995', freq='M')
2  date
```

```
DatetimeIndex(['1980-01-31', '1980-02-29', '1980-03-31', '1980-04-30',
               '1980-05-31', '1980-06-30', '1980-07-31', '1980-08-31',
               '1980-09-30', '1980-10-31',
               ...
               '1994-10-31', '1994-11-30', '1994-12-31', '1995-01-31',
               '1995-02-28', '1995-03-31', '1995-04-30', '1995-05-31',
               '1995-06-30', '1995-07-31'],
              dtype='datetime64[ns]', length=187, freq='M')
```

Now we add this date range as a column TimeStamp and then use that column for indexing.

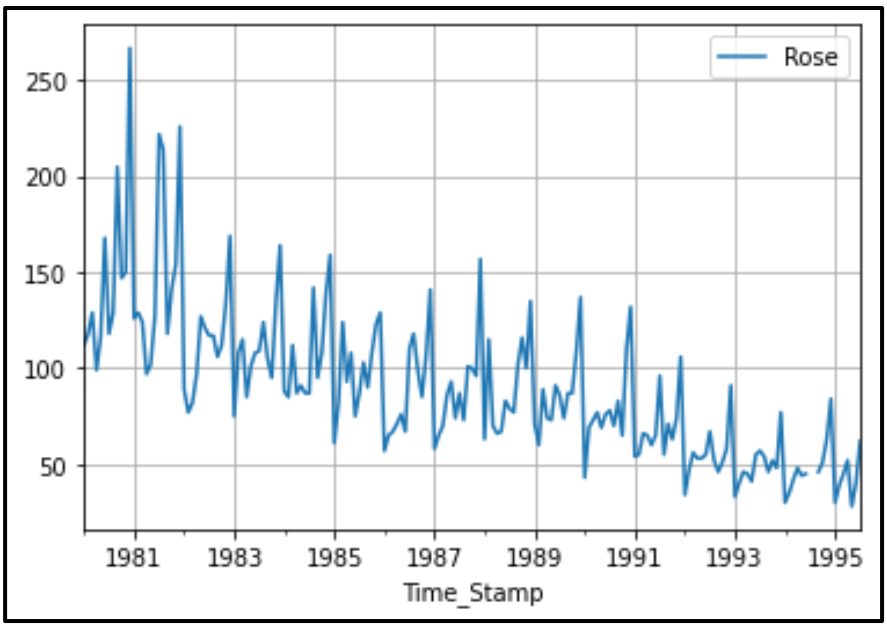
```
df['Time_Stamp'] = pd.DataFrame(date,columns=['Month'])
df.head()
```

	YearMonth	Rose	Time_Stamp
0	1980-01	112.0	1980-01-31
1	1980-02	118.0	1980-02-29
2	1980-03	129.0	1980-03-31
3	1980-04	99.0	1980-04-30
4	1980-05	116.0	1980-05-31

```
df['Time_Stamp'] = pd.to_datetime(df['Time_Stamp'])
df = df.set_index('Time_Stamp')
df.drop(['YearMonth'], axis=1, inplace=True)
df.head()
```

Time_Stamp	Rose
1980-01-31	112.0
1980-02-29	118.0
1980-03-31	129.0
1980-04-30	99.0
1980-05-31	116.0

Now we have our data in a time-series format. So we can plot it.



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

Firstly, we check the 5 number summary and number of missing values in our data using describe () and isnull().sum()

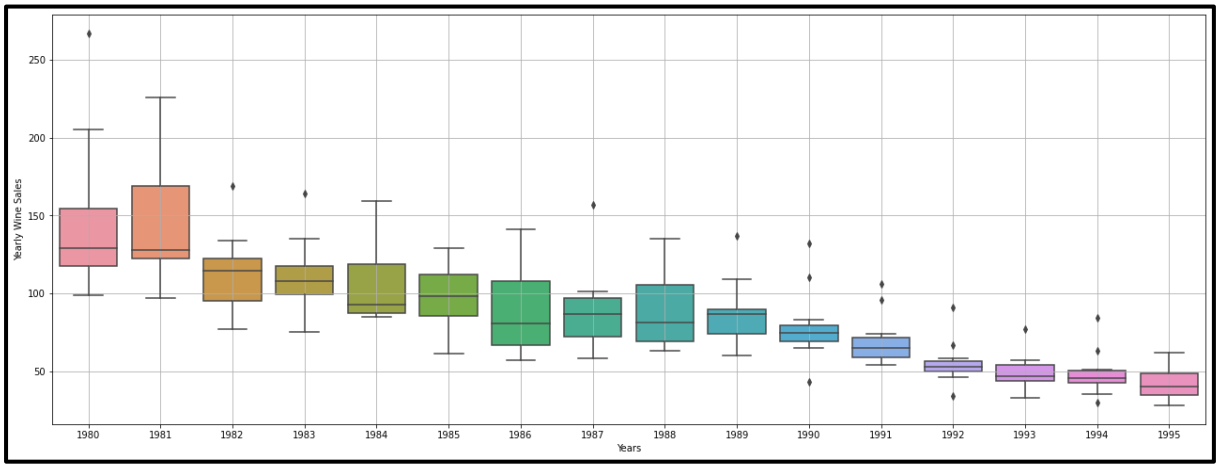
```
df.describe()
```

	Rose
count	185.000000
mean	90.394595
std	39.175344
min	28.000000
25%	63.000000
50%	86.000000
75%	112.000000
max	267.000000

```
df.isnull().sum()
```

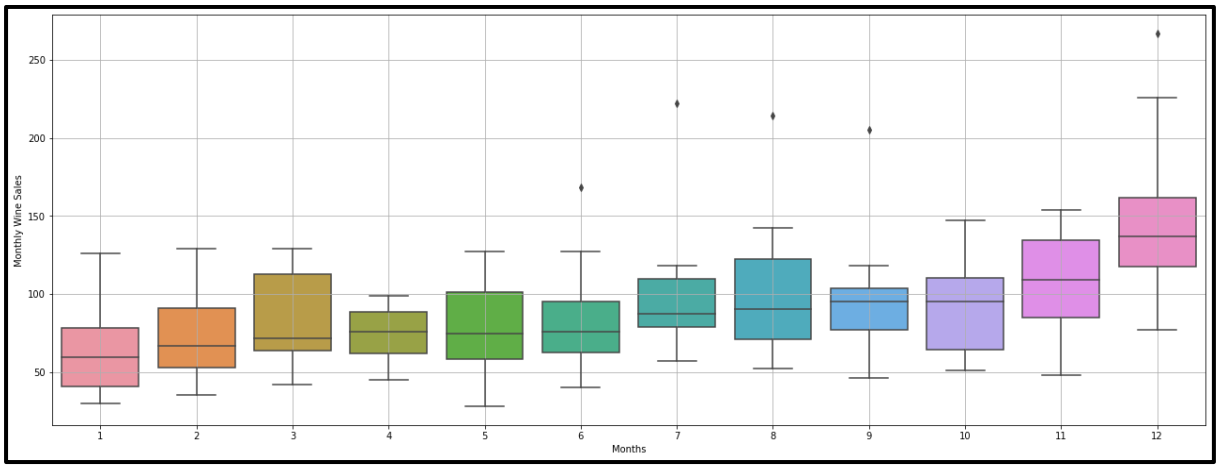
```
Rose    2
dtype: int64
```

BoxPlots by Year



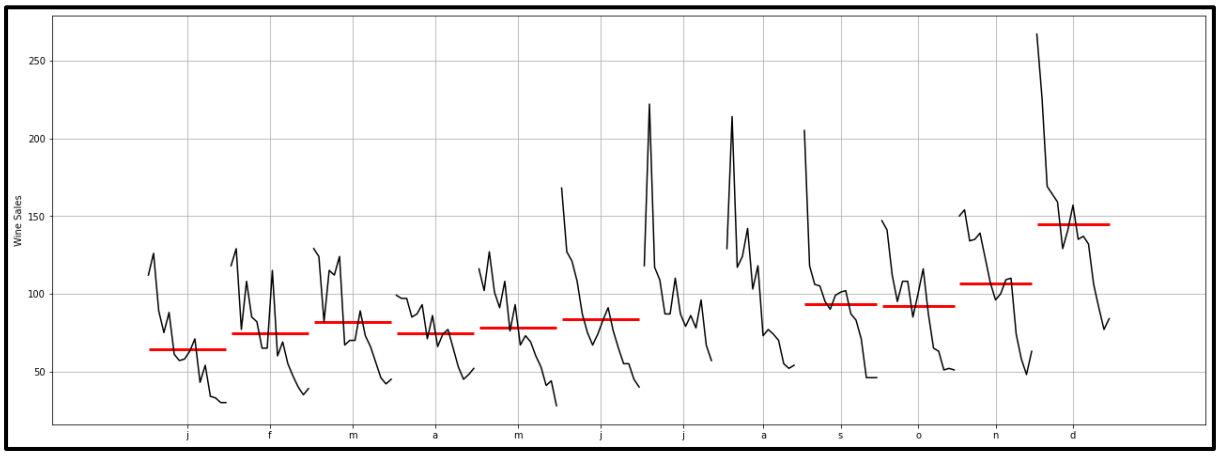
Here, we can see that sales were higher initially and then it started dropping down.

BoxPlots by Month



Here, we can see that sales are generally high towards the year's end.

Monthly Plot for TimeSeries

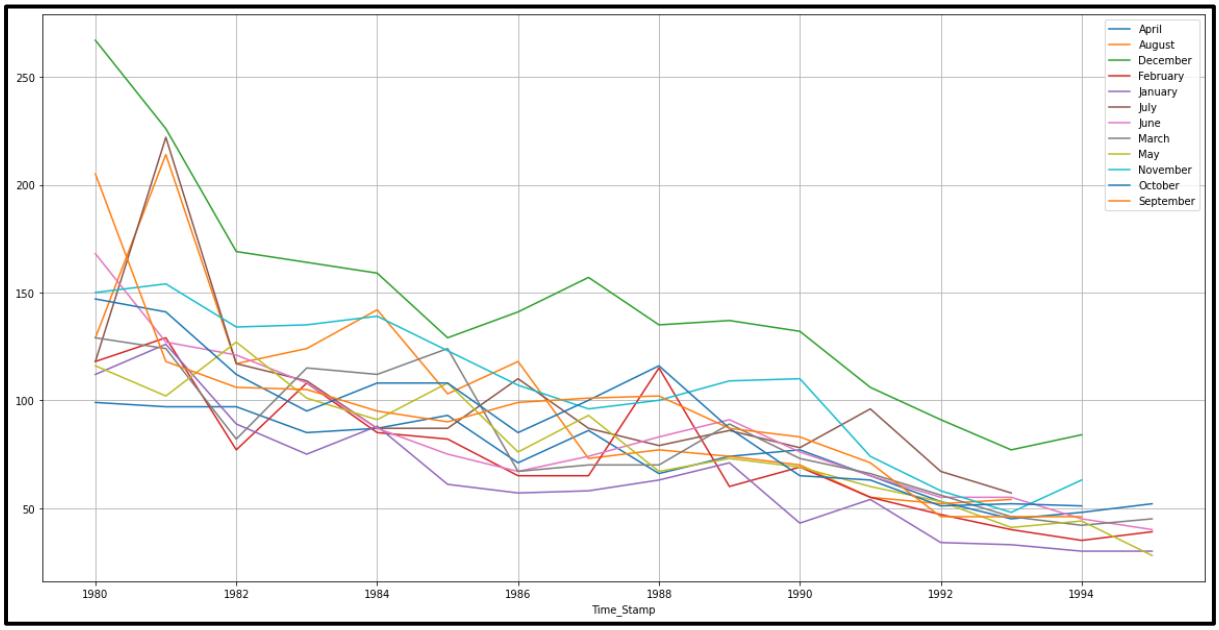


The red lines indicate the average sales for the month.

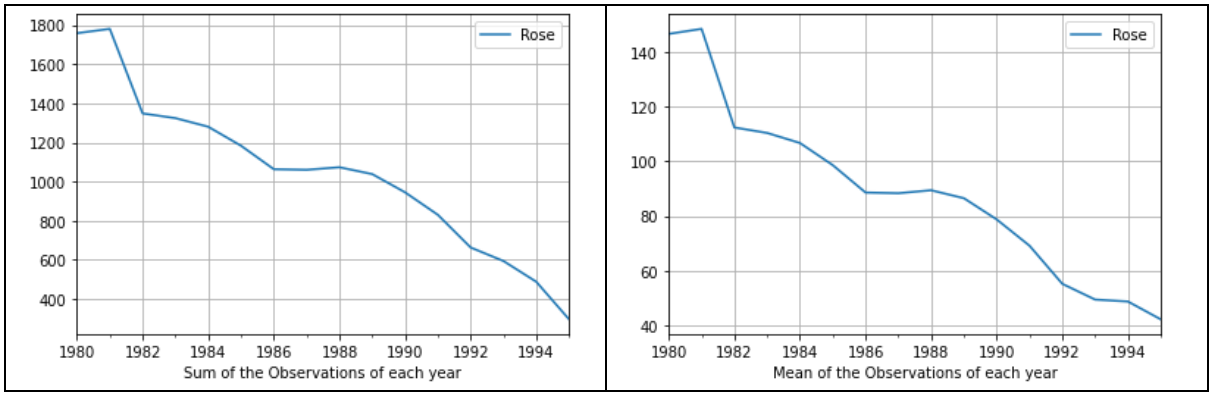
Pivot Table Months vs Year

YearMonth	April	August	December	February	January	July	June	March	May	November	October	September
Time_Stamp												
1980	99.0	129.0	267.0	118.0	112.0	118.0	168.0	129.0	116.0	150.0	147.0	205.0
1981	97.0	214.0	226.0	129.0	126.0	222.0	127.0	124.0	102.0	154.0	141.0	118.0
1982	97.0	117.0	169.0	77.0	89.0	117.0	121.0	82.0	127.0	134.0	112.0	106.0
1983	85.0	124.0	164.0	108.0	75.0	109.0	108.0	115.0	101.0	135.0	95.0	105.0
1984	87.0	142.0	159.0	85.0	88.0	87.0	87.0	112.0	91.0	139.0	108.0	95.0
1985	93.0	103.0	129.0	82.0	61.0	87.0	75.0	124.0	108.0	123.0	108.0	90.0
1986	71.0	118.0	141.0	65.0	57.0	110.0	67.0	67.0	76.0	107.0	85.0	99.0
1987	86.0	73.0	157.0	65.0	58.0	87.0	74.0	70.0	93.0	96.0	100.0	101.0
1988	66.0	77.0	135.0	115.0	63.0	79.0	83.0	70.0	67.0	100.0	116.0	102.0
1989	74.0	74.0	137.0	60.0	71.0	86.0	91.0	89.0	73.0	109.0	87.0	87.0
1990	77.0	70.0	132.0	69.0	43.0	78.0	76.0	73.0	69.0	110.0	65.0	83.0
1991	65.0	55.0	106.0	55.0	54.0	96.0	65.0	66.0	60.0	74.0	63.0	71.0
1992	53.0	52.0	91.0	47.0	34.0	67.0	55.0	56.0	53.0	58.0	51.0	46.0
1993	45.0	54.0	77.0	40.0	33.0	57.0	55.0	46.0	41.0	48.0	52.0	46.0
1994	48.0	NaN	84.0	35.0	30.0	NaN	45.0	42.0	44.0	63.0	51.0	46.0
1995	52.0	NaN	NaN	39.0	30.0	62.0	40.0	45.0	28.0	NaN	NaN	NaN

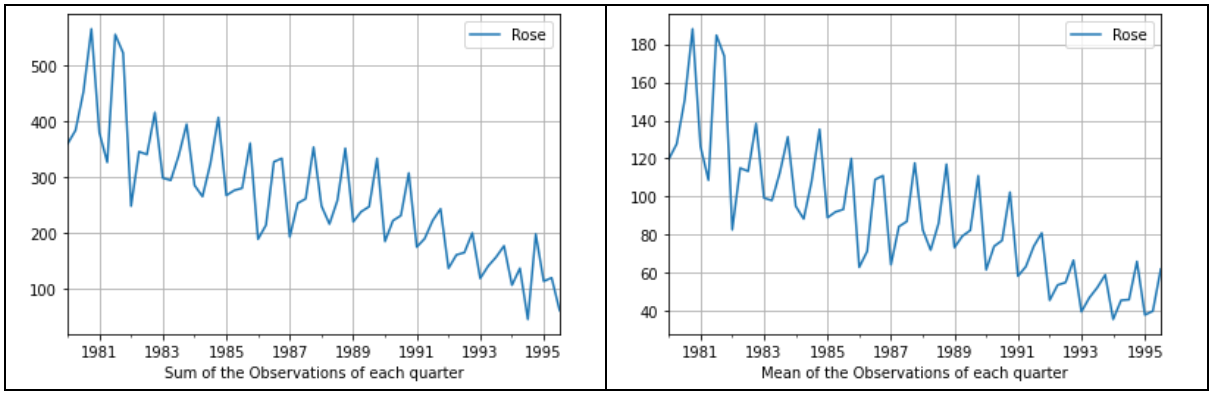
Monthly Sales across Years



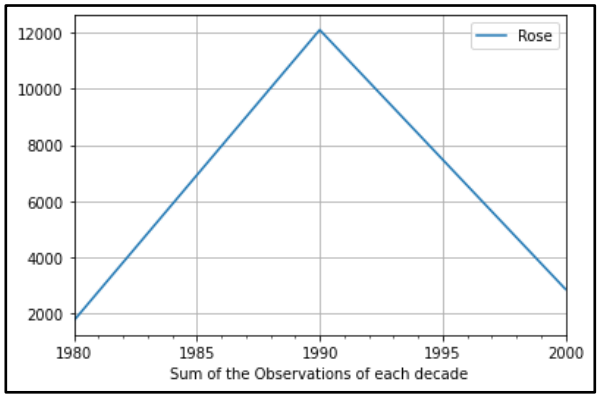
Yearly Plots (Sum and Mean Sales)



Quarterly Plots (Sum and Mean Sales)



Decade Plot (Sum Sales)



Now we have to replace the null values, so we use the interpolate function and use the method as spline to generate values and replace them with missing values, As we know, we had 2 missing values in 1994, we have to replace them.

```
1 df.interpolate(method= 'spline',order=3,inplace=True)
```

```
1 df['1994']
```

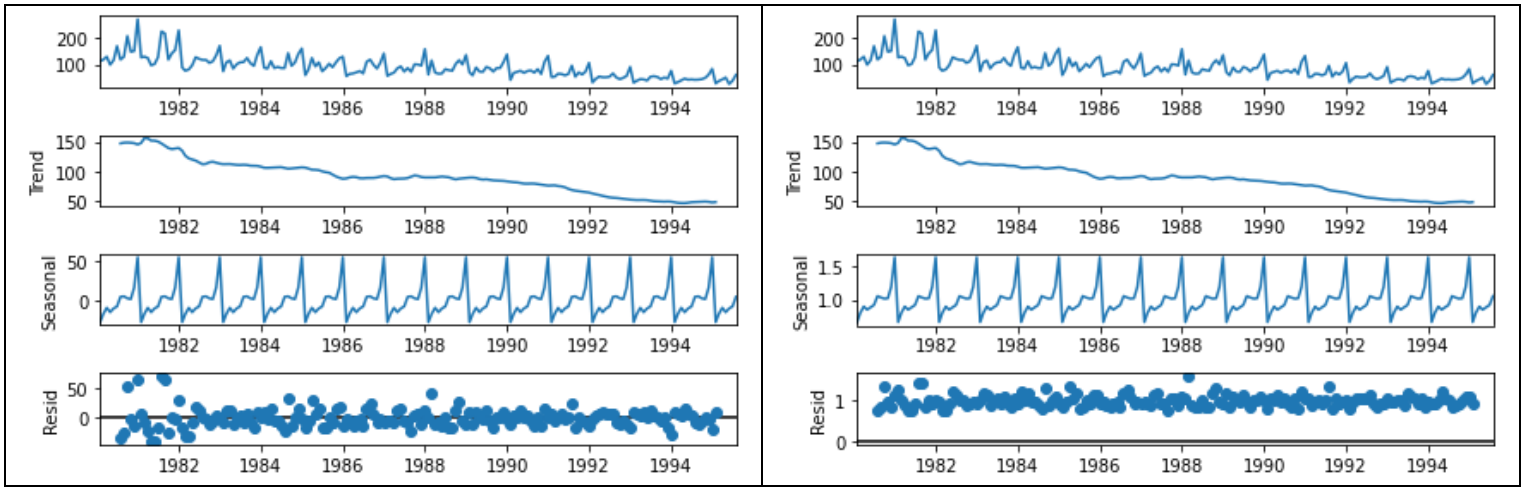
Time_Stamp	Rose
1994-01-31	30.000000
1994-02-28	35.000000
1994-03-31	42.000000
1994-04-30	48.000000
1994-05-31	44.000000
1994-06-30	45.000000
1994-07-31	43.693064
1994-08-31	44.326877
1994-09-30	46.000000
1994-10-31	51.000000
1994-11-30	63.000000
1994-12-31	84.000000

```
1 df.isna().sum()
```

```
Rose    0
dtype: int64
```

We now have all the values, so we can decompose our series to check the seasonality, trend and residual components. There are 2 methods- additive and multiplicative.

Additive	Multiplicative



Looking at the decompositions, we can see that both the methods show similar trends and seasonality i.e downward trend and a repetitive seasonality. But the residuals vary. In additive method, the residuals are scattered at level 0 and go till 50 whereas in multiplicative, we can observe the residuals are more varied around level 1. Hence, we conclude that we have a multiplicative model.

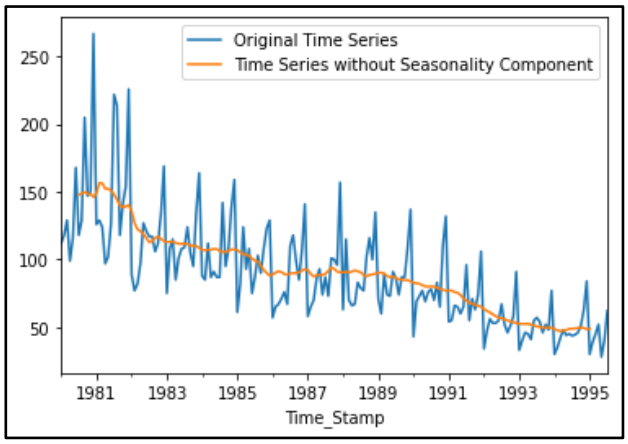
Now we further split the series so we can look at these components individually.

```
Trend
 Time_Stamp
1980-01-31           NaN
1980-02-29           NaN
1980-03-31           NaN
1980-04-30           NaN
1980-05-31           NaN
1980-06-30           NaN
1980-07-31    147.083333
1980-08-31    148.125000
1980-09-30    148.375000
1980-10-31    148.083333
1980-11-30    147.416667
1980-12-31    145.125000
Name: trend, dtype: float64

Seasonality
 Time_Stamp
1980-01-31    0.670320
1980-02-29    0.806375
1980-03-31    0.901501
1980-04-30    0.854411
1980-05-31    0.889760
1980-06-30    0.924324
1980-07-31    1.056071
1980-08-31    1.034338
1980-09-30    1.017959
1980-10-31    1.022915
1980-11-30    1.192781
1980-12-31    1.629244
Name: seasonal, dtype: float64

Residual
 Time_Stamp
1980-01-31         NaN
1980-02-29         NaN
1980-03-31         NaN
1980-04-30         NaN
1980-05-31         NaN
1980-06-30         NaN
1980-07-31    0.759671
1980-08-31    0.841974
1980-09-30    1.357259
1980-10-31    0.970446
1980-11-30    0.853069
1980-12-31    1.129231
Name: resid, dtype: float64
```

```
deaseasonalized_ts = trend + residual
deaseasonalized_ts.head(12)
```

```
Time_Stamp
1980-01-31           NaN
1980-02-29           NaN
1980-03-31           NaN
1980-04-30           NaN
1980-05-31           NaN
1980-06-30           NaN
1980-07-31    147.843004
1980-08-31    148.966974
1980-09-30    149.732259
1980-10-31    149.053780
1980-11-30    148.269735
1980-12-31    146.254231
dtype: float64
```

We can also plot the time series without the seasonality component.



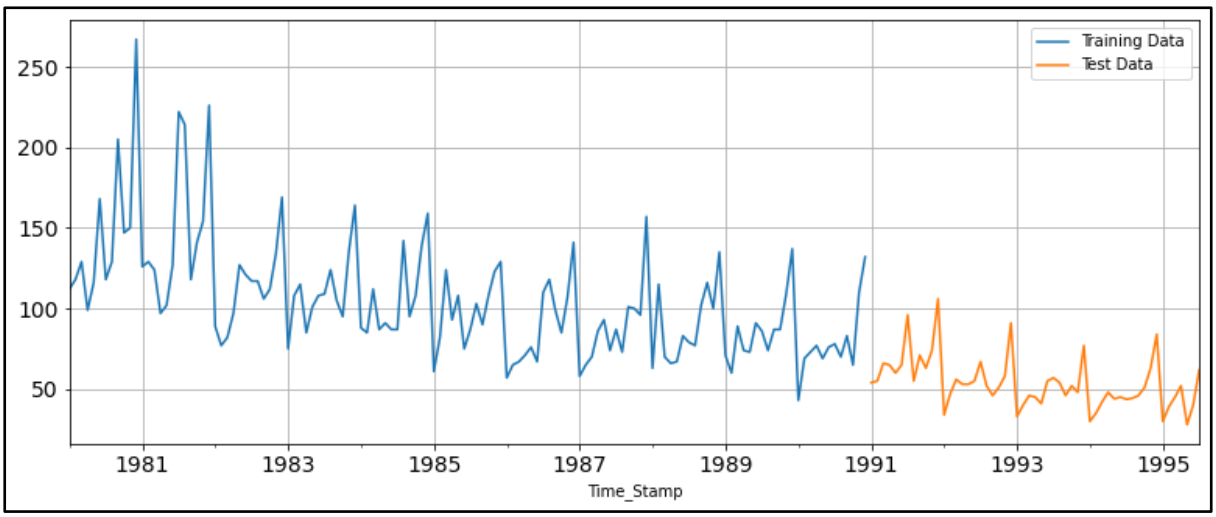
3.Split the data into training and test. The test data should start in 1991.

```
train=df[df.index.year < 1991]
test=df[df.index.year >= 1991]
```

```
print(train.shape)
print (test.shape)
```

```
(132, 1)
(55, 1)
```

We can also see the training and test set data visually in form of plot.



4.Build various 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, simple average models etc. should also be built on the training data and check the performance on the test data using RMSE.

- **Linear Regression**

This model is based on Linear Regression method to forecast the data.

```
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
 [133, 134, 135, 136, 137, 138, 139, 140, 141, 142, 143, 144, 145, 146, 147, 148, 149, 150, 151, 152, 153, 154, 155, 156, 157,
158, 159, 160, 161, 162, 163, 164, 165, 166, 167, 168, 169, 170, 171, 172, 173, 174, 175, 176, 177, 178, 179, 180, 181, 182, 18
3, 184, 185, 186, 187]
```

We have now created time instances for train and test set.

First few rows of Training Data

Time_Stamp	Rose	time
1980-01-31	112.0	1
1980-02-29	118.0	2
1980-03-31	129.0	3
1980-04-30	99.0	4
1980-05-31	116.0	5

Last few rows of Training Data

Time_Stamp	Rose	time
1990-08-31	70.0	128
1990-09-30	83.0	129
1990-10-31	65.0	130
1990-11-30	110.0	131
1990-12-31	132.0	132

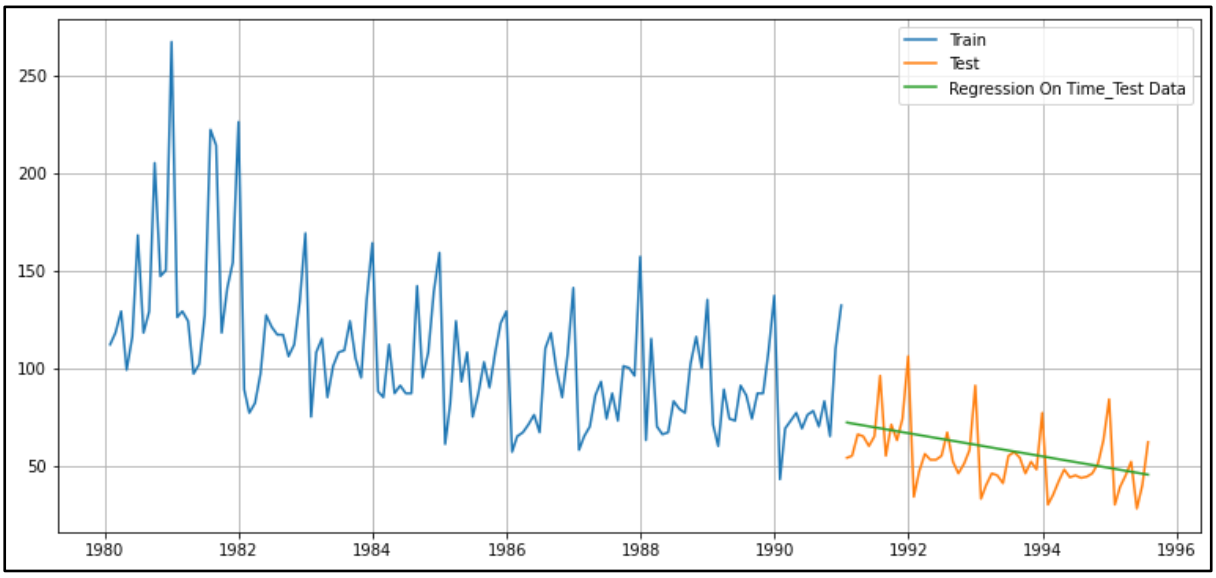
First few rows of Test Data

Time_Stamp	Rose	time
1991-01-31	54.0	133
1991-02-28	55.0	134
1991-03-31	66.0	135
1991-04-30	65.0	136
1991-05-31	60.0	137

Last few rows of Test Data

Time_Stamp	Rose	time
1995-03-31	45.0	183
1995-04-30	52.0	184
1995-05-31	28.0	185
1995-06-30	40.0	186
1995-07-31	62.0	187

Now we build a LinearRegression model on the train set -



The green line represents the forecasted data, which takes care of the trend but the seasonality component seems to be overlooked here.

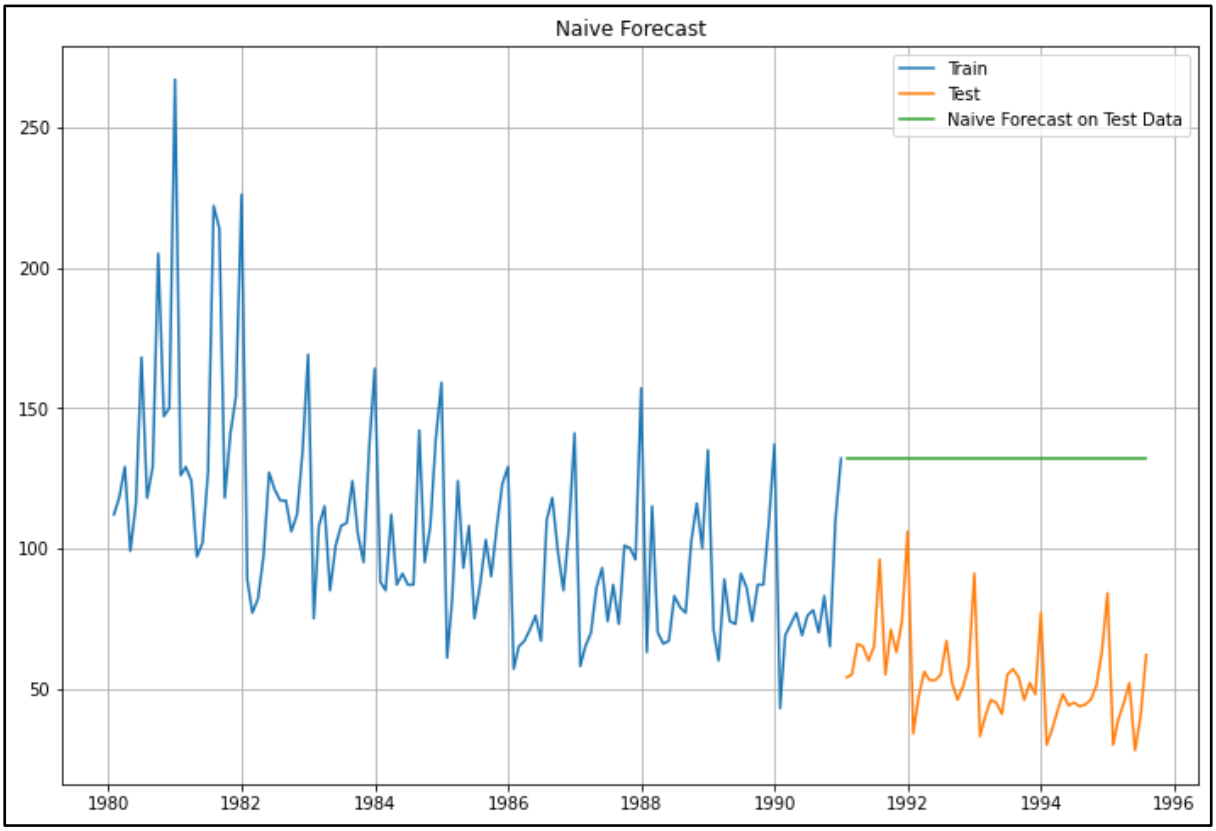
We now check the RMSE on test data- **RMSE= 15.291**

- **Naïve Approach-**

This approach forecasts the data by simply copying the last occurred value.

```
Time_Stamp
1991-01-31    132.0
1991-02-28    132.0
1991-03-31    132.0
1991-04-30    132.0
1991-05-31    132.0
Name: naive, dtype: float64
```

We can observe the plot after building the Naïve Approach model-



The green line represents the data forecasted using the Naïve Approach, the straight line clearly shows the same point being duplicated for all the timestamps to be forecasted. The last data point in the train set is taken and used as forecasting data on the whole test set and the observations are far away from what we need/expect the forecasting to be.

Now we calculate the RMSE to check the accuracy of forecasted data –

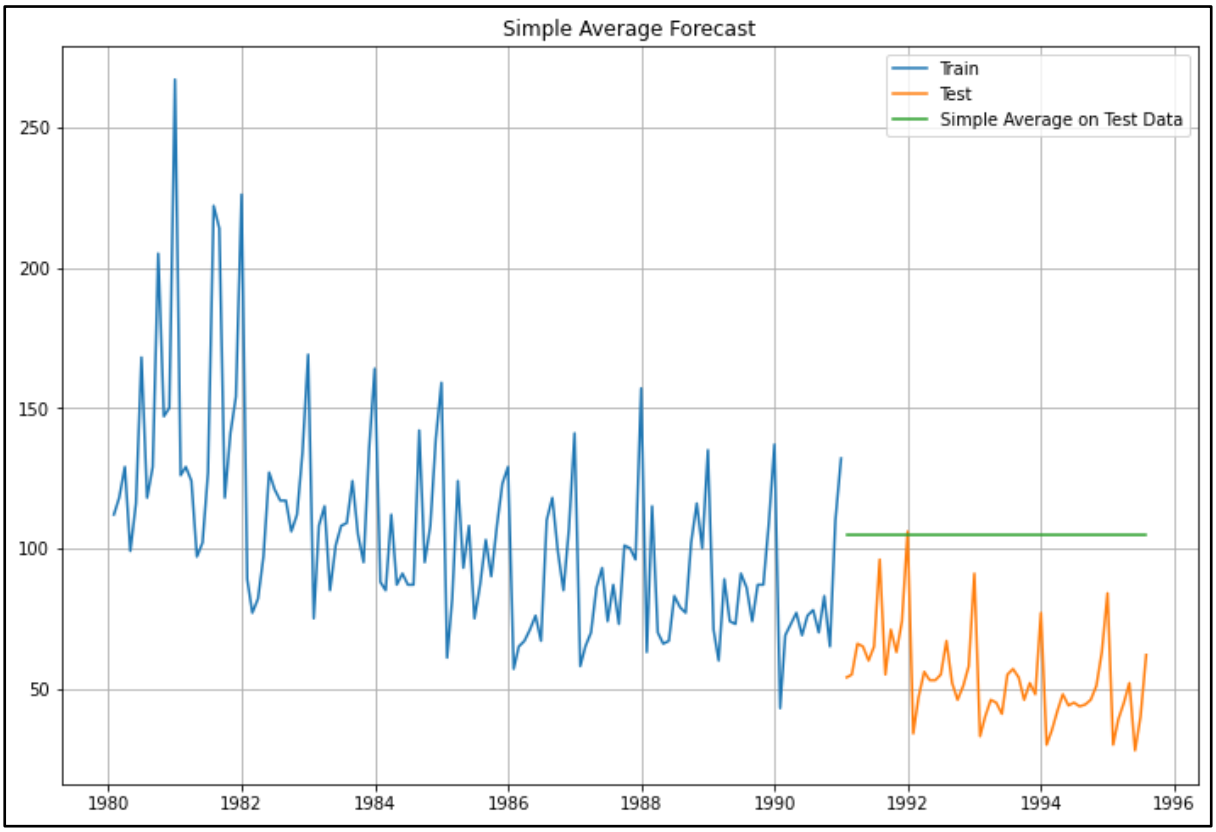
RMSE= 79.778

- **Simple Average –**

This model is similar to Naïve Approach, but the only difference is that it used the average of all the data points and then uses them as forecasted values.

Time_Stamp	Rose	mean_forecast
1991-01-31	54.0	104.939394
1991-02-28	55.0	104.939394
1991-03-31	66.0	104.939394
1991-04-30	65.0	104.939394
1991-05-31	60.0	104.939394

Now we build a model based on it and visualise it using plot –



We can see that the green list is the forecast given by the model and it is not what the required forecast should be. It is better than Naïve Forecast and it can be seen by looking at the RMSE value.

RSME on test set- **RMSE=53.522**

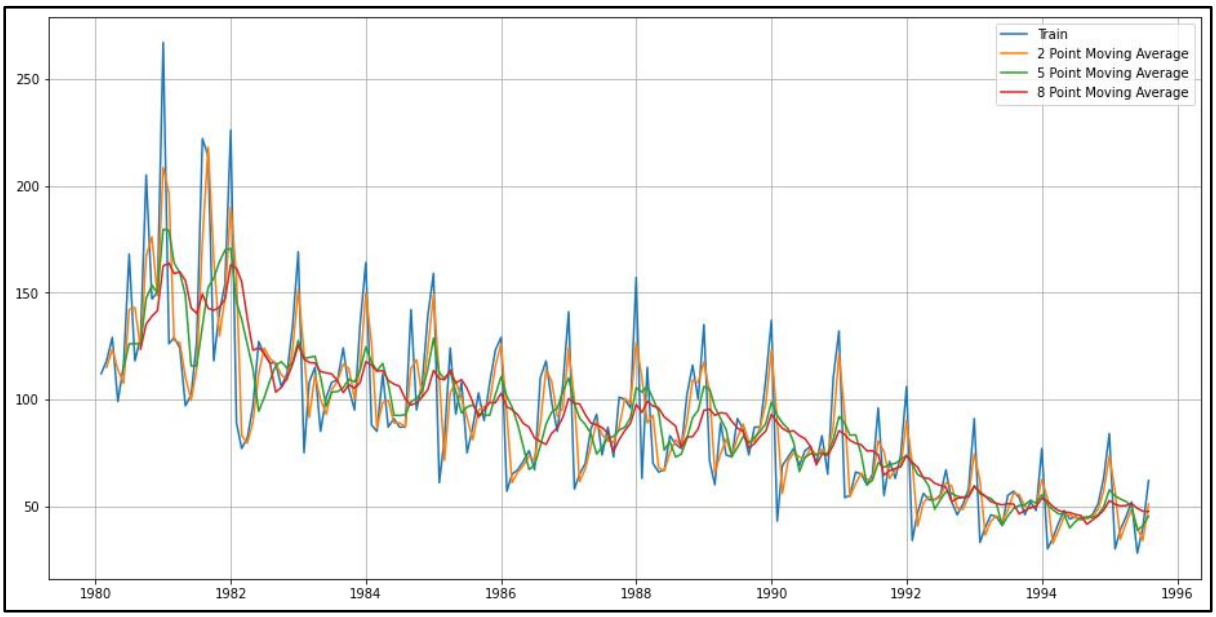
- **Moving Average-**

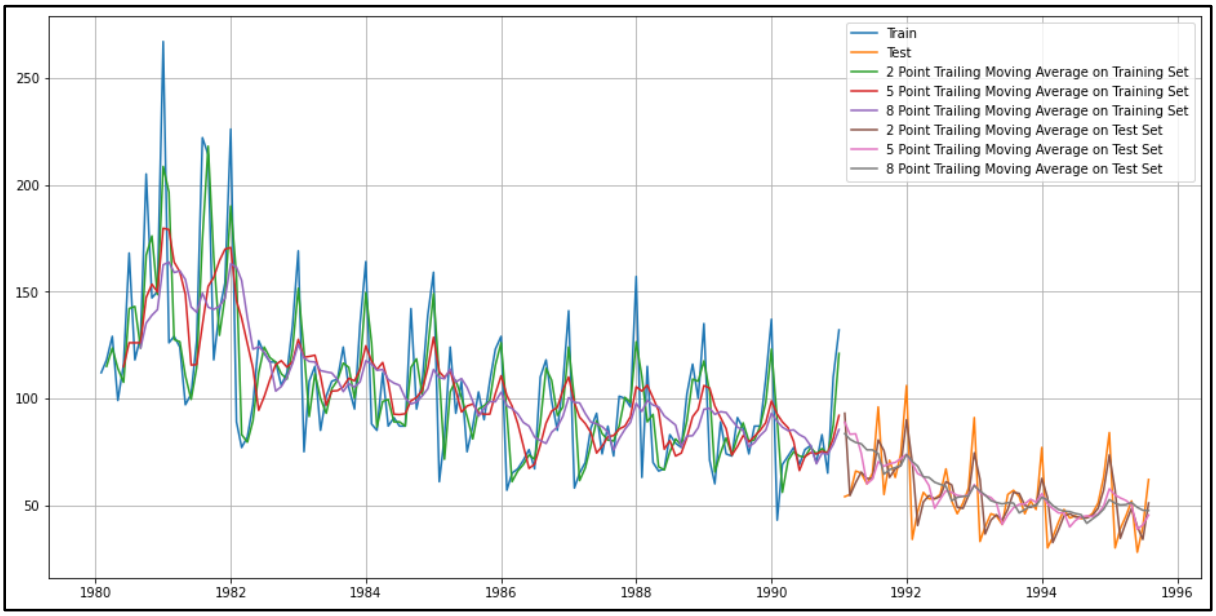
This method uses averaging to forecast the values based on window sizes ie. The window keeps on moving with the size constant for newer points to be forecasted.

Considering window size as 2,5 and 8 –

Time_Stamp	Rose	Trailing_2	Trailing_5	Trailing_8
1980-01-31	112.0	NaN	NaN	NaN
1980-02-29	118.0	115.0	NaN	NaN
1980-03-31	129.0	123.5	NaN	NaN
1980-04-30	99.0	114.0	NaN	NaN
1980-05-31	116.0	107.5	114.8	NaN
1980-06-30	168.0	142.0	126.0	NaN
1980-07-31	118.0	143.0	126.0	NaN
1980-08-31	129.0	123.5	126.0	123.625
1980-09-30	205.0	167.0	147.2	135.250
1980-10-31	147.0	176.0	153.4	138.875

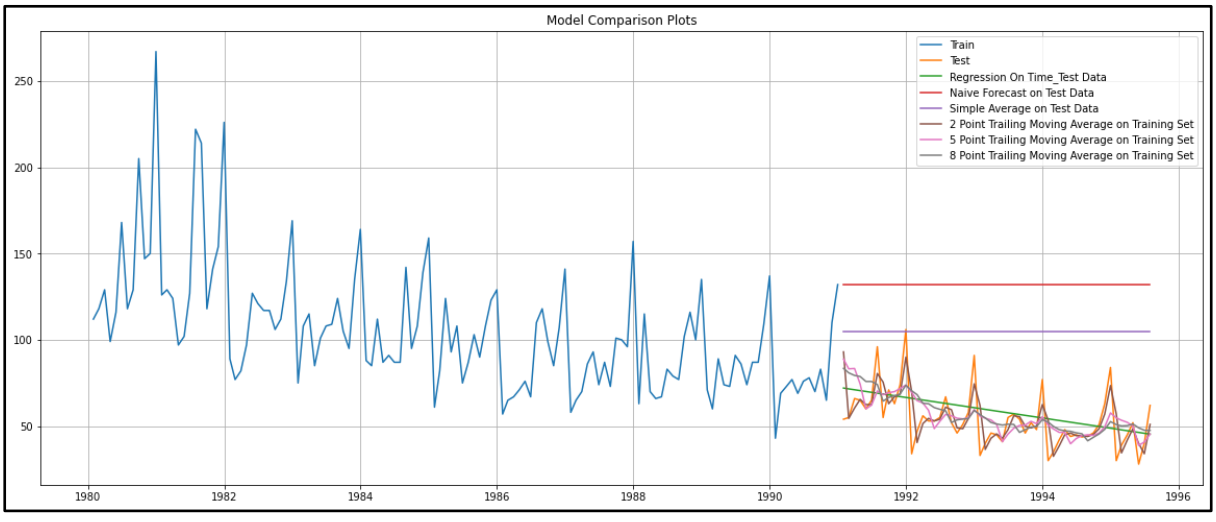
Now we build a model on train set to forecast using this approach and visualize it using plot-





We now calculate the accuracy using the RMSE-

```
For 2 point Moving Average Model forecast on the Training Data,  RMSE is 11.530
For 5 point Moving Average Model forecast on the Training Data,  RMSE is 14.491
For 8 point Moving Average Model forecast on the Training Data,  RMSE is 14.812
```



Now we use different Exponential Smoothing models

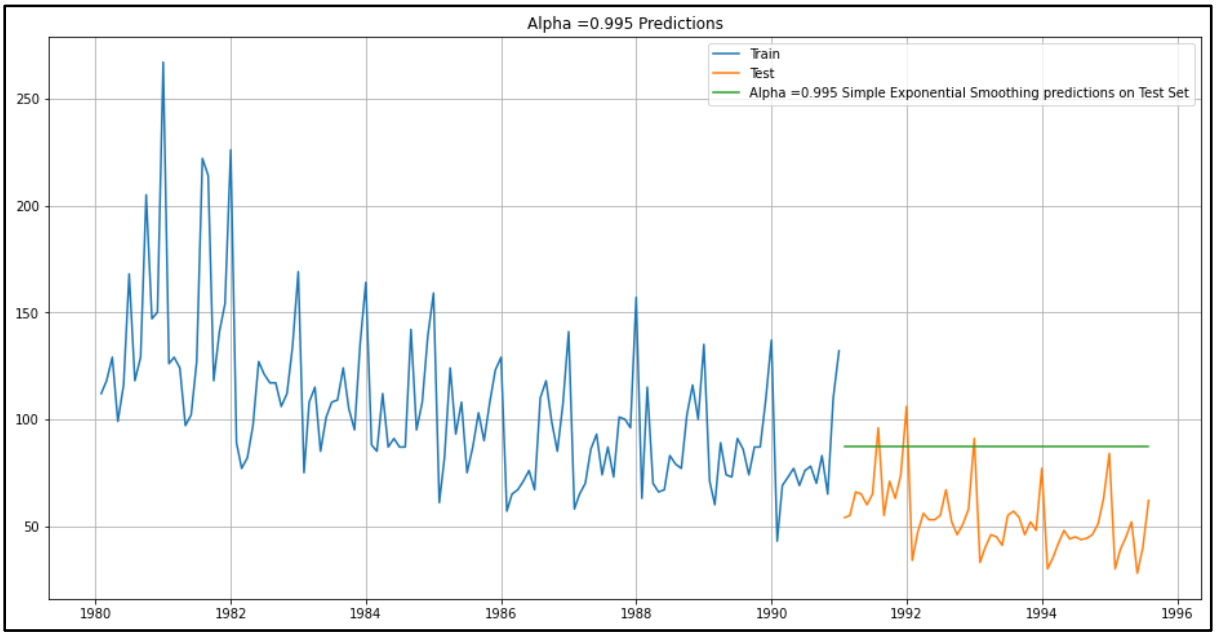
- **Simple Exponential Smoothing-**
  This model considers only level as a parameter.

The model uses the following parameters-

```
{'smoothing_level': 0.0987493111726833,
 'smoothing_trend': nan,
 'smoothing_seasonal': nan,
 'damping_trend': nan,
 'initial_level': 134.38720226208358,
 'initial_trend': nan,
 'initial_seasons': array([], dtype=float64),
 'use_boxcox': False,
 'lamda': None,
 'remove_bias': False}
```

Time_Stamp	Rose	predict
1991-01-31	54.0	87.104983
1991-02-28	55.0	87.104983
1991-03-31	66.0	87.104983
1991-04-30	65.0	87.104983
1991-05-31	60.0	87.104983

We build a model on these parameters on train set and forecast on test set. Plot looks like -

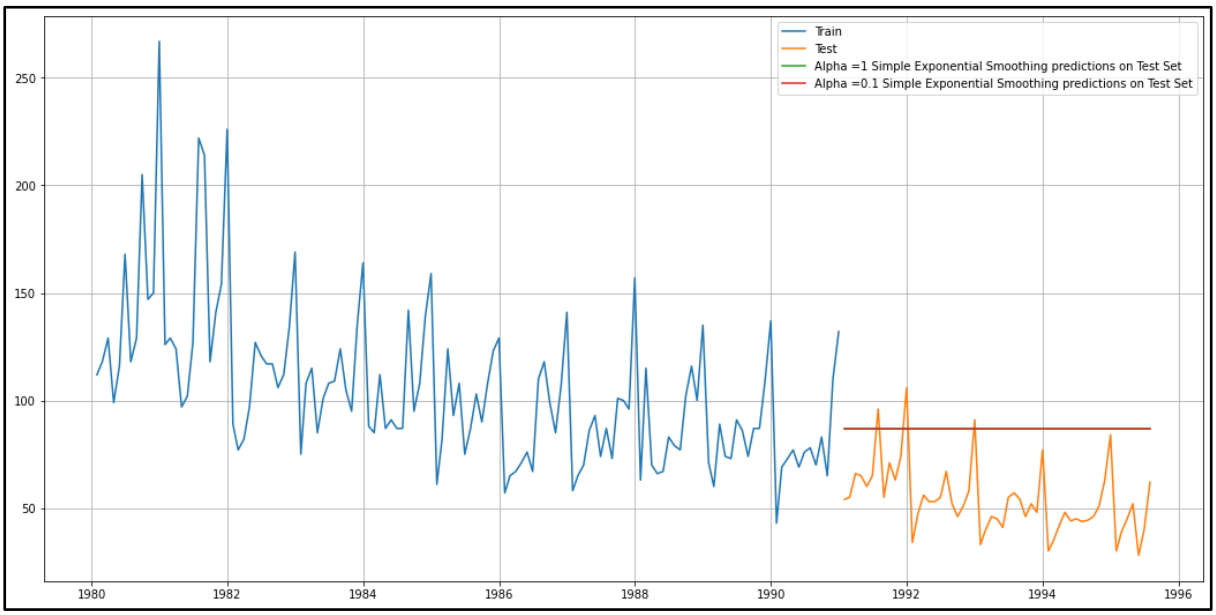


The predicted RMSE on test set is, **RMSE=36.859**

We can also look at different alpha values and RMSE at those values-

	Alpha Values	Train RMSE	Test RMSE
0	0.1	31.815610	36.890375
1	0.2	31.979391	41.423877
2	0.3	32.470164	47.566302
3	0.4	33.035130	53.828369
4	0.5	33.682839	59.702301
5	0.6	34.441171	65.031433
6	0.7	35.323261	69.758009
7	0.8	36.334596	73.833601
8	0.9	37.482782	77.198702

So, when we plot at alpha at 0.1, we get a similar graph as RMSE is same-

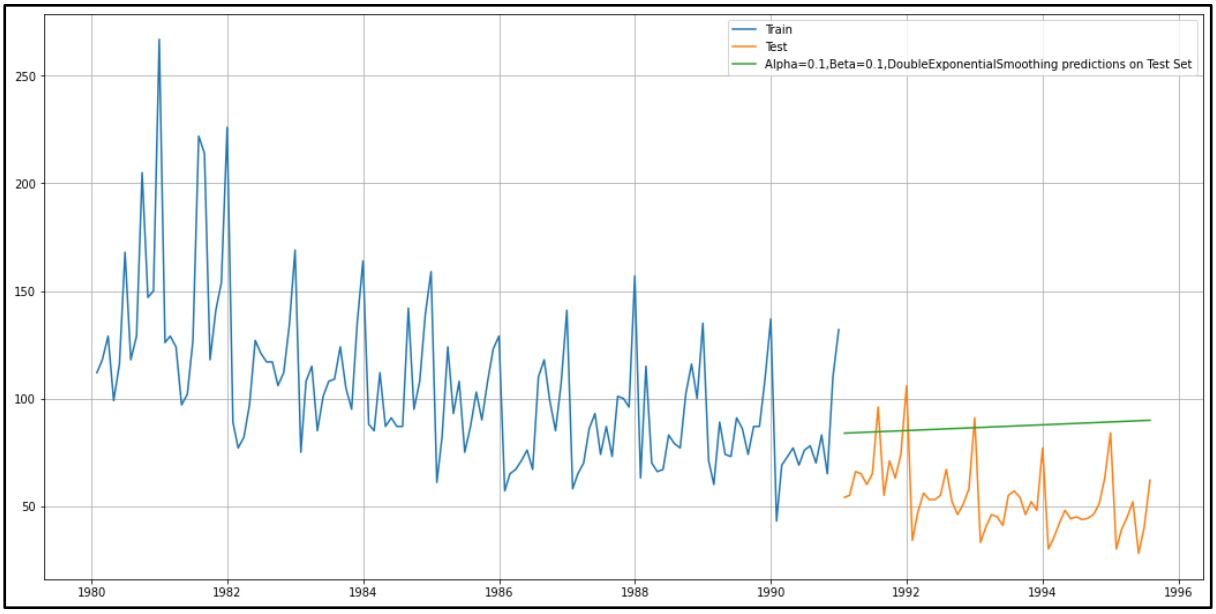


- **Double Exponential Smoothing (Holt's Model) –**
  This model considers level and trend while forecasting values.

We use a combination of different alpha and beta values and consider the best value and build a model on it.

	Alpha Values	Beta Values	Train RMSE	Test RMSE
0	0.1	0.1	34.439111	36.987695
1	0.1	0.2	33.450729	48.757725
10	0.2	0.1	33.097427	65.800617
2	0.1	0.3	33.145789	78.228265
20	0.3	0.1	33.611269	98.723180

We build a model considering alpha as 0.1 and beta as 0.1 and visualise the forecast on plot-



We check the accuracy of the model using RMSE, **RMSE=36.987**

- **Triple Exponential Smoothing (Holt-Winter's Model) –**
  This model considers level, trend and seasonality while forecasting values.

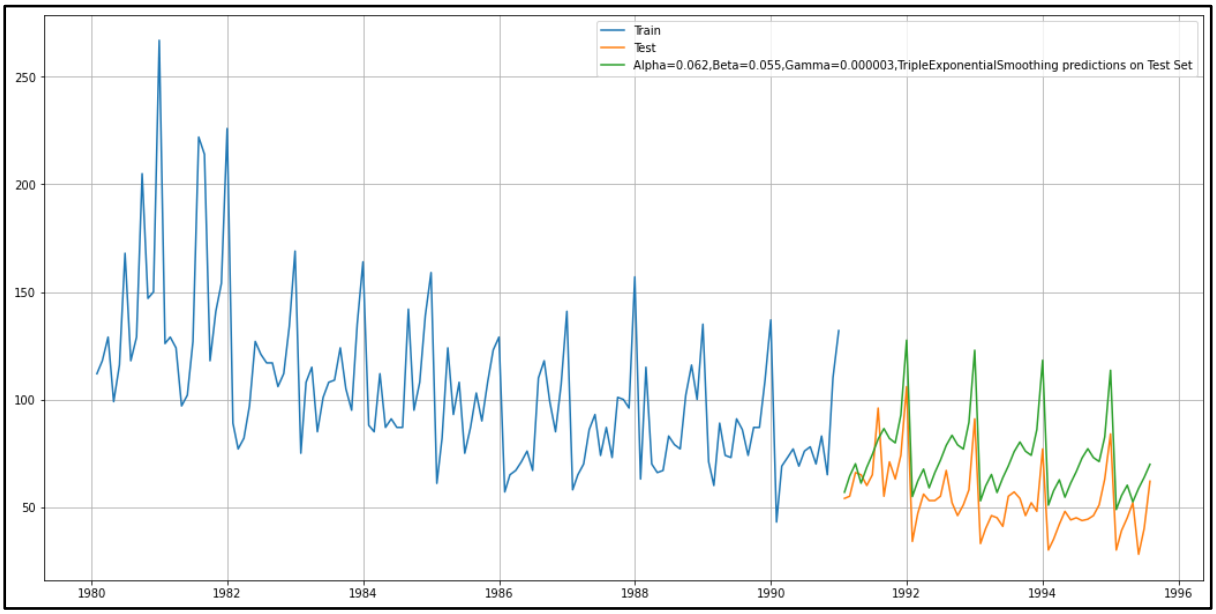
The parameters used for building model-

```
{'smoothing_level': 0.06280372101991354,
 'smoothing_trend': 0.05568813542468586,
 'smoothing_seasonal': 3.115268099923303e-06,
 'damping_trend': nan,
 'initial_level': 59.29348777160217,
 'initial_trend': -0.3727281817131398,
 'initial_seasons': array([1.90362937, 2.16072498, 2.36051494, 2.06290154, 2.31908662,
        2.52928387, 2.77978951, 2.95507465, 2.80602228, 2.74429704,
        3.19876277, 4.41211721]),
 'use_boxcox': False,
 'lamda': None,
 'remove_bias': False}
```

Forecasted values at different timestamps for the above mentioned alpha, beta and gamma values-

Time_Stamp	Rose	auto_predict
1991-01-31	54.0	56.879265
1991-02-28	55.0	64.371067
1991-03-31	66.0	70.115332
1991-04-30	65.0	61.093709
1991-05-31	60.0	68.476551

Now, we visualise the model build on train set and forecast on test set, plot is given as-

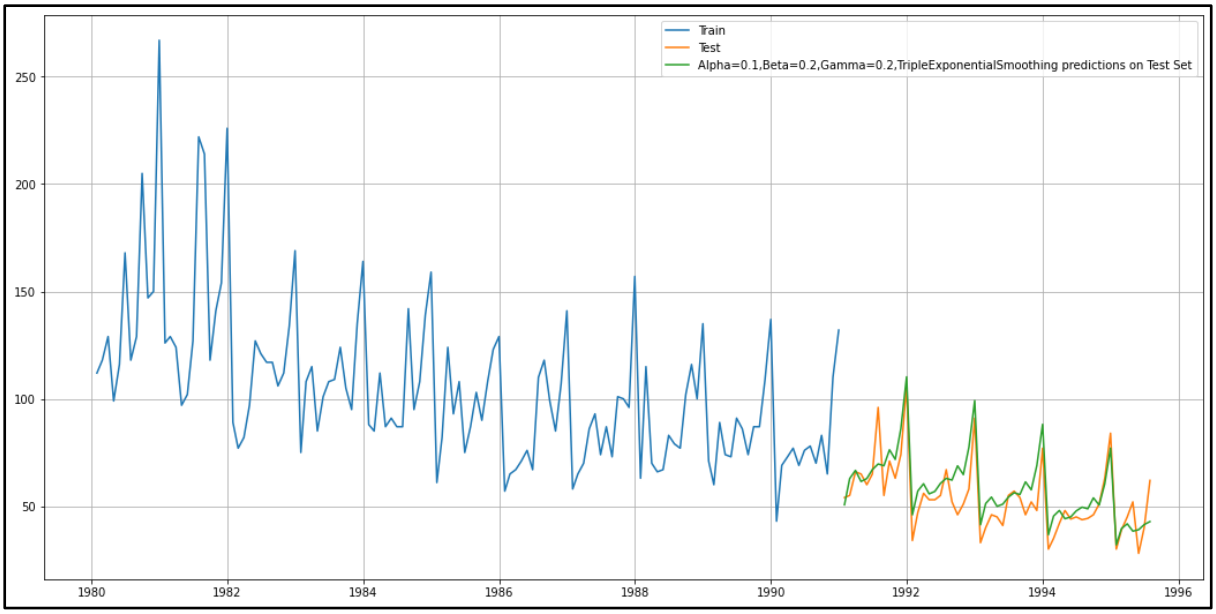


We check the accuracy using RMSE, **RMSE=21.459**

We also build a Triple Exponential model considering the best values for alpha, beta and gamma based on RMSE.

	Alpha Values	Beta Values	Gamma Values	Train RMSE	Test RMSE
11	0.1	0.2	0.2	24.365597	9.665739
10	0.1	0.2	0.1	25.529854	9.954136
12	0.1	0.2	0.3	23.969166	9.961876
142	0.2	0.5	0.3	27.631767	10.002930
151	0.2	0.6	0.2	28.289836	10.005753

So we consider alpha, beta, gamma as 0.1, 0.2, 0.2 respectively and build a model on train set and forecast on test set.

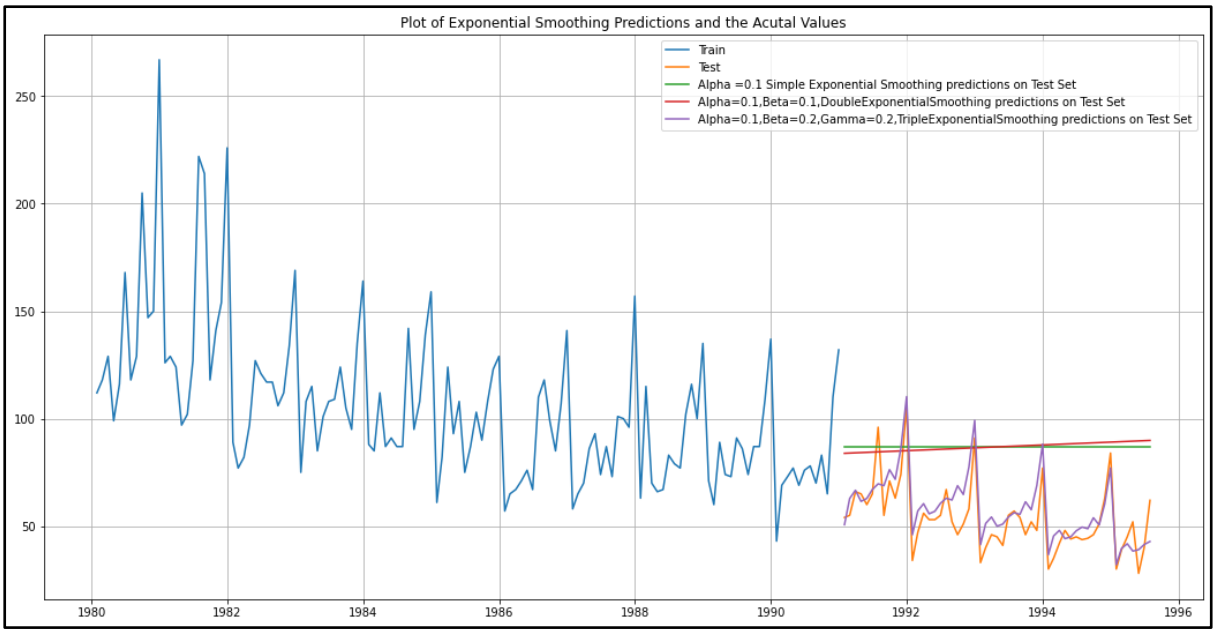


The accuracy of model is, **RMSE=9.665**

All the model that we build and their accuracies in terms of RMSE-

	RMSE
LinearRegression	15.291460
NaiveModel	79.778066
SimpleAverageModel	53.521557
2pointTrailingMovingAverage	11.530180
5pointTrailingMovingAverage	14.491131
8pointTrailingMovingAverage	14.811615
SimpleExponentialSmoothing @Alpha=0.995	36.858571
SimpleExponentialSmoothing @Alpha=0.1	36.890375
DoubleExponentialSmoothing @Alpha=0.1,Beta=0.1	36.987695
TripleExponentialSmoothing @Alpha=0.062,Beta=0.055,Gamma=3.115	21.458971
TripleExponentialSmoothing @Alpha=0.1,Beta=0.2,Gamma=0.2	9.665739

We now plot all the Exponential Smoothing models-



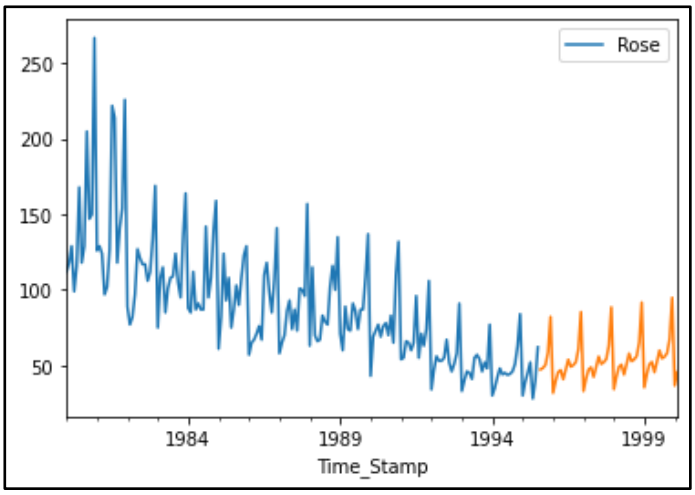
Looking at the RMSE values for all the models, we can conclude that the Triple Exponential Model with alpha, beta, gamma as 0.1,0.2 and 0.2 respectively performs the best. So we build a complete model on Rose dataset on these parameters.

```
fullmodel1 = ExponentialSmoothing(df,
                                  trend='additive',
                                  seasonal='multiplicative').fit(smoothing_level=0.1,
                                                                 smoothing_trend=0.2,
                                                                 smoothing_seasonal=0.2)
```

```
RMSE_fullmodel1 = metrics.mean_squared_error(df['Rose'],fullmodel1.fittedvalues,squared=False)
print('RMSE:',RMSE_fullmodel1)
```

```
RMSE: 17.411770315489036
```

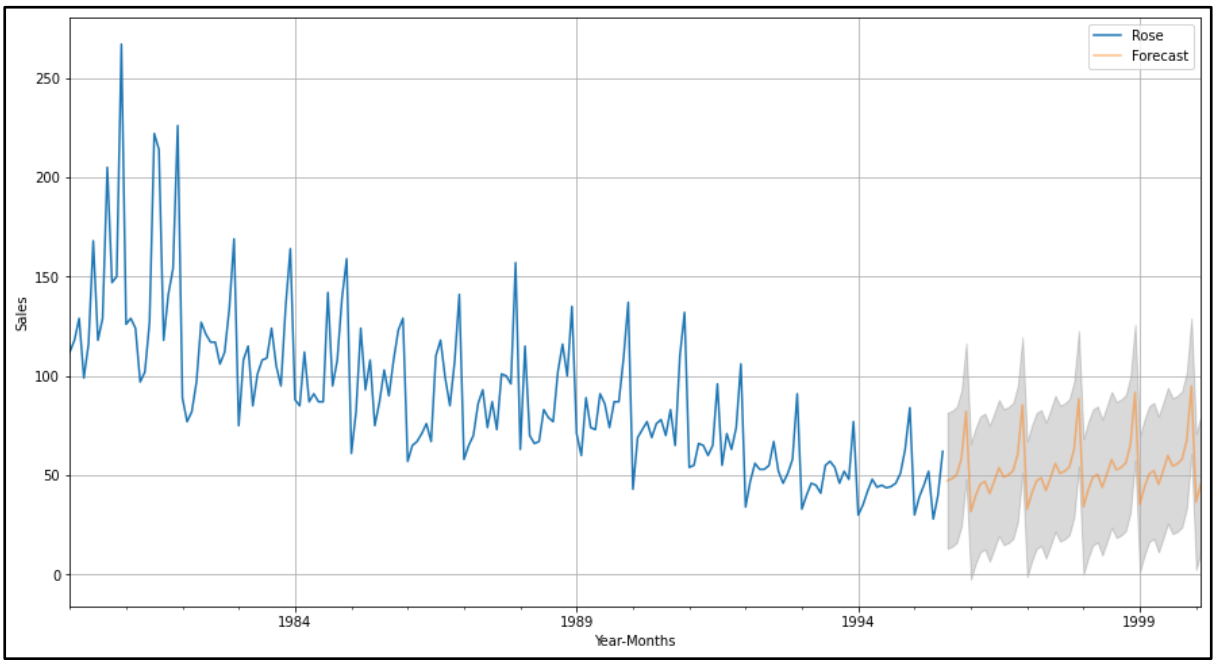
The following is how it predicts-



Considering the confidence levels at alpha at 0.05, we compute the predictions and upper and lower confidence levels-

	lower_CI	prediction	upper_ci
1995-08-31	13.049823	47.250342	81.450862
1995-09-30	14.070165	48.270685	82.471204
1995-10-31	16.095596	50.296116	84.496635
1995-11-30	24.311272	58.511791	92.712310
1995-12-31	48.008626	82.209146	116.409665

We have plotted our forecasting considering the error -



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.)

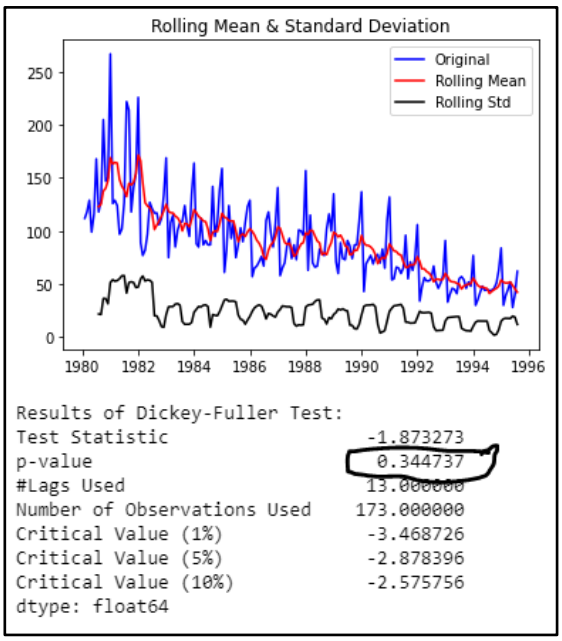
We are checking the Stationarity at alpha = 0.05 for the Time Series data. This will be done using Dicky Fuller Test.

Hypothesis for Dicky Fuller Test is as follows

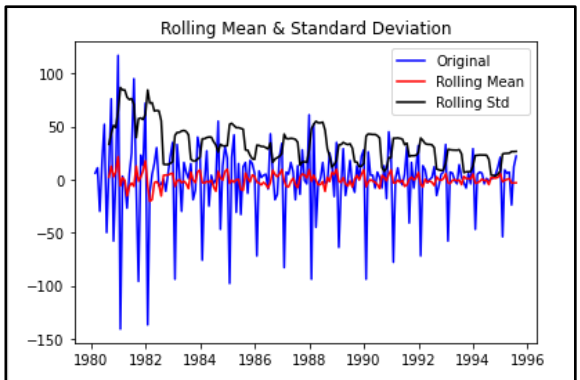
Ho: Time Series is Non-Stationary

Ha: Time Series is Stationary

The result of Dicky fuller Test is



The p-value is more than 0.05, hence we cannot reject the null. The data is not stationary, which means have to differentiate the time series.




```
Results of Dickey-Fuller Test:
Test Statistic                -8.043385e+00
p-value                        1.821604e-12
#Lags Used                     1.200000e+01
Number of Observations Used    1.730000e+02
Critical Value (1%)           -3.468726e+00
Critical Value (5%)           -2.878396e+00
Critical Value (10%)          -2.575756e+00
dtype: float64
```

Now our p-value is < 0.05, hence our data is stationary.

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.

- **ARIMA model based on AIC**

```
import itertools
p = q = range(0, 3)
d= range(1,2)
pdq = list(itertools.product(p, d, q))
print('Some parameter combinations for the Model...')
for i in range(1,len(pdq)):
    print('Model: {}'.format(pdq[i]))
```

```
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)
```

```
ARIMA_AIC = pd.DataFrame(columns=['param', 'AIC'])

from statsmodels.tsa.arima_model import ARIMA

for param in pdq:
    ARIMA_model = ARIMA(train.values,order=param).fit()
    print('ARIMA{} - AIC:{}'.format(param,ARIMA_model.aic))
    ARIMA_AIC = ARIMA_AIC.append({'param':param, 'AIC': ARIMA_model.aic}, ignore_index=True)
```

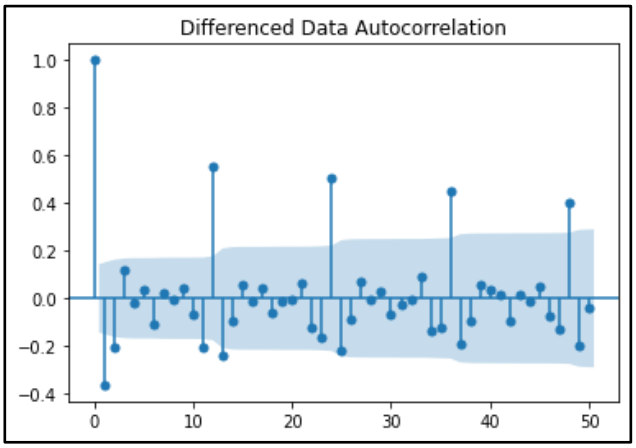
```
ARIMA(0, 1, 0) - AIC:1335.1526583086775
ARIMA(0, 1, 1) - AIC:1280.7261830464035
ARIMA(0, 1, 2) - AIC:1276.8353724115532
ARIMA(1, 1, 0) - AIC:1319.3483105803125
ARIMA(1, 1, 1) - AIC:1277.7757493733707
ARIMA(1, 1, 2) - AIC:1277.3592229395217
ARIMA(2, 1, 0) - AIC:1300.609261174427
ARIMA(2, 1, 1) - AIC:1279.0456894093113
ARIMA(2, 1, 2) - AIC:1279.298693936556
```

Since the best performance is at (0,1,2) we build a model on that –

```
                             ARIMA Model Results
==============================================================================
Dep. Variable:                 D.Rose   No. Observations:                  131
Model:                 ARIMA(0, 1, 2)   Log Likelihood                -634.418
Method:                       css-mle   S.D. of innovations             30.167
Date:                Sat, 24 Apr 2021   AIC                           1276.835
Time:                        17:08:09   BIC                           1288.336
Sample:                    02-29-1980   HQIC                          1281.509
                         - 12-31-1990
==============================================================================
                 coef    std err          z      P>|z|      [0.025      0.975]
------------------------------------------------------------------------------
const          -0.4885      0.085     -5.742      0.000      -0.655      -0.322
ma.L1.D.Rose   -0.7601      0.101     -7.499      0.000      -0.959      -0.561
ma.L2.D.Rose   -0.2398      0.095     -2.518      0.012      -0.427      -0.053
                                    Roots
=============================================================================
                  Real          Imaginary           Modulus         Frequency
-----------------------------------------------------------------------------
MA.1            1.0000           +0.0000j            1.0000            0.0000
MA.2           -4.1695           +0.0000j            4.1695            0.5000
-----------------------------------------------------------------------------
```

Now, we predict accuracy on the test set by RMSE- **RMSE= 15.64**

- **SARIMA model based on AIC**



Looking at this we can see that the seasonality is repeated 6 and 12 months. So we build SARIMA models for 6 and 12 months.

- **6-month seasonality-**

```
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)
```

	param	seasonal	AIC
53	(1, 1, 2)	(2, 0, 2, 6)	1041.655818
26	(0, 1, 2)	(2, 0, 2, 6)	1043.600261
80	(2, 1, 2)	(2, 0, 2, 6)	1045.220571
71	(2, 1, 1)	(2, 0, 2, 6)	1051.673461
44	(1, 1, 1)	(2, 0, 2, 6)	1052.778470

Based on the AIC values, we build model on parameter- (1,1,2) seasonal- (2,0,2,6)

```
                                 SARIMAX Results
==========================================================================================
Dep. Variable:                                 y   No. Observations:                  132
Model:             SARIMAX(1, 1, 2)x(2, 0, 2, 6)   Log Likelihood                -512.828
Date:                           Sat, 24 Apr 2021   AIC                           1041.656
Time:                                   17:08:35   BIC                           1063.685
Sample:                                        0   HQIC                          1050.598
                                             - 132
Covariance Type:                             opg
==============================================================================
                 coef    std err          z      P>|z|      [0.025      0.975]
------------------------------------------------------------------------------
ar.L1         -0.5939      0.152     -3.899      0.000      -0.892      -0.295
ma.L1         -0.1954    799.740     -0.000      1.000   -1567.656    1567.265
ma.L2         -0.8046    643.516     -0.001      0.999   -1262.072    1260.463
ar.S.L6       -0.0626      0.035     -1.764      0.078      -0.132       0.007
ar.S.L12       0.8451      0.039     21.883      0.000       0.769       0.921
ma.S.L6        0.2226    532.437      0.000      1.000   -1043.334    1043.780
ma.S.L12      -0.7774    413.876     -0.002      0.999    -811.960     810.406
sigma2       335.1994   3.09e+05      0.001      0.999   -6.04e+05    6.05e+05
===================================================================================
Ljung-Box (L1) (Q):                   0.07   Jarque-Bera (JB):                56.68
Prob(Q):                              0.78   Prob(JB):                         0.00
Heteroskedasticity (H):               0.47   Skew:                             0.52
Prob(H) (two-sided):                  0.02   Kurtosis:                         6.26
===================================================================================
```

Now, we predict on the test set-

	y	mean	mean_se	mean_ci_lower	mean_ci_upper
0	62.841435	18.848416	25.899217	99.783652
1	67.630543	19.300215	29.802817	105.458270
2	74.747073	19.412764	36.698756	112.795391
3	71.325789	19.475703	33.154112	109.497465
4	76.018296	19.483981	37.830395	114.206197

The accuracy is given by RMSE, **RMSE= 26.20**

- **12-month seasonality-**

```
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)	887.937509
53	(1, 1, 2)	(2, 0, 2, 12)	889.903097
80	(2, 1, 2)	(2, 0, 2, 12)	890.668798
69	(2, 1, 1)	(2, 0, 0, 12)	896.518161
78	(2, 1, 2)	(2, 0, 0, 12)	897.346444

Based on the AIC values, we build model on parameter- (0,1,2) seasonal- (2,0,2,12)

```
                                 SARIMAX Results
==========================================================================================
Dep. Variable:                                 y   No. Observations:                  132
Model:             SARIMAX(0, 1, 2)x(2, 0, 2, 12)   Log Likelihood                -436.969
Date:                            Sat, 24 Apr 2021   AIC                            887.938
Time:                                    17:09:11   BIC                            906.448
Sample:                                         0   HQIC                           895.437
                                              - 132
Covariance Type:                              opg
==============================================================================
                 coef    std err          z      P>|z|      [0.025      0.975]
------------------------------------------------------------------------------
ma.L1         -0.8427    189.952     -0.004      0.996    -373.141     371.455
ma.L2         -0.1573     29.842     -0.005      0.996     -58.647      58.333
ar.S.L12       0.3467      0.079      4.375      0.000       0.191       0.502
ar.S.L24       0.3023      0.076      3.996      0.000       0.154       0.451
ma.S.L12       0.0767      0.133      0.577      0.564      -0.184       0.337
ma.S.L24      -0.0726      0.146     -0.498      0.618      -0.358       0.213
sigma2       251.3137   4.77e+04      0.005      0.996   -9.33e+04    9.38e+04
===================================================================================
Ljung-Box (L1) (Q):                   0.10   Jarque-Bera (JB):                 2.33
Prob(Q):                              0.75   Prob(JB):                         0.31
Heteroskedasticity (H):               0.88   Skew:                             0.37
Prob(H) (two-sided):                  0.70   Kurtosis:                         3.03
===================================================================================
```

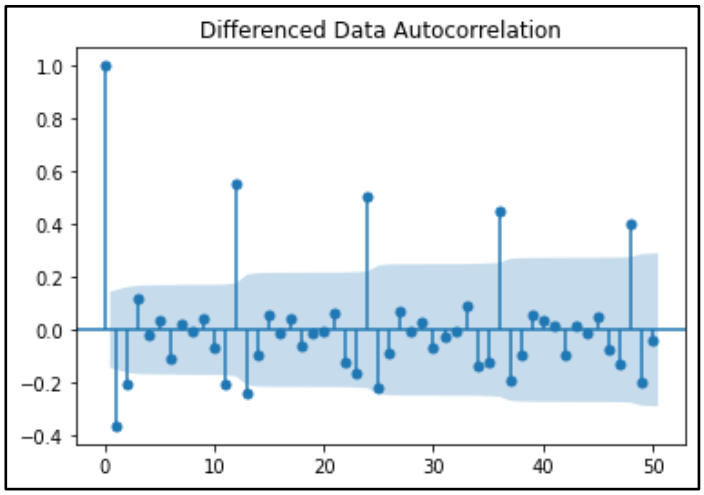
Now, we predict on the test set-

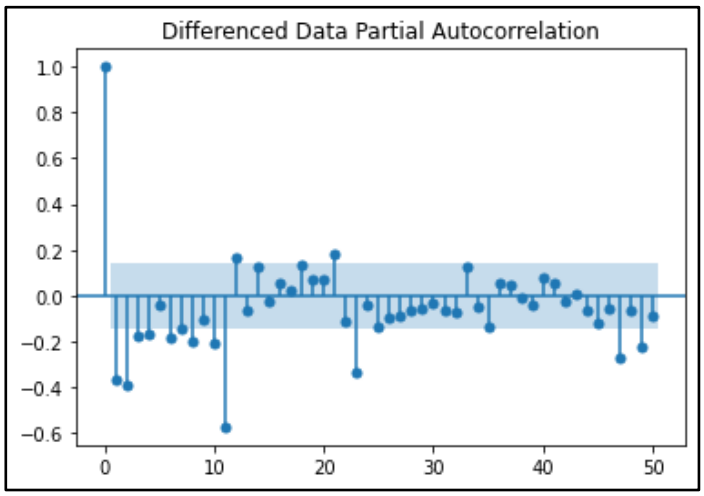
	y	mean	mean_se	mean_ci_lower	mean_ci_upper
0	62.867264	15.928501	31.647976	94.086553
1	70.541190	16.147659	38.892360	102.190020
2	77.356411	16.147657	45.707585	109.005236
3	76.208814	16.147657	44.559988	107.857639
4	72.747398	16.147657	41.098573	104.396223

The accuracy is given by RMSE, **RMSE= 26.99**

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.

- **ARIMA based on ACF and PACF cut-off points**





The Moving Average i.e q comes from lag before ACF cut-off i.e 1 and Auto-Regressive parameter i.e p is also 1 in our case.

We build model on (1,1,1)

ARIMA Model Results

Dep. Variable:	D.Rose	No. Observations:	131
Model:	ARIMA(1, 1, 1)	Log Likelihood	-634.888
Method:	css-mle	S.D. of innovations	30.279
Date:	Sat, 24 Apr 2021	AIC	1277.776
Time:	17:09:12	BIC	1289.277
Sample:	02-29-1980 - 12-31-1990	HQIC	1282.449

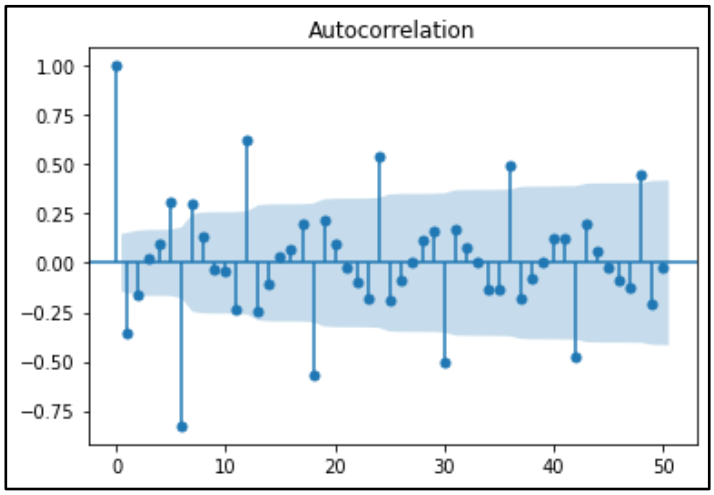
	coef	std err	z	P>|z|	[0.025	0.975]
const	-0.4871	0.086	-5.656	0.000	-0.656	-0.318
ar.L1.D.Rose	0.2006	0.087	2.293	0.022	0.029	0.372
ma.L1.D.Rose	-0.9999	0.035	-28.646	0.000	-1.068	-0.932

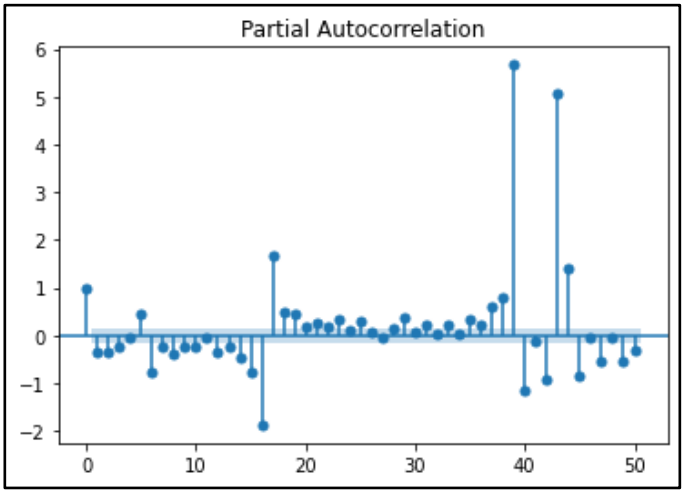
Roots

	Real	Imaginary	Modulus	Frequency
AR.1	4.9856	+0.0000j	4.9856	0.0000
MA.1	1.0001	+0.0000j	1.0001	0.0000

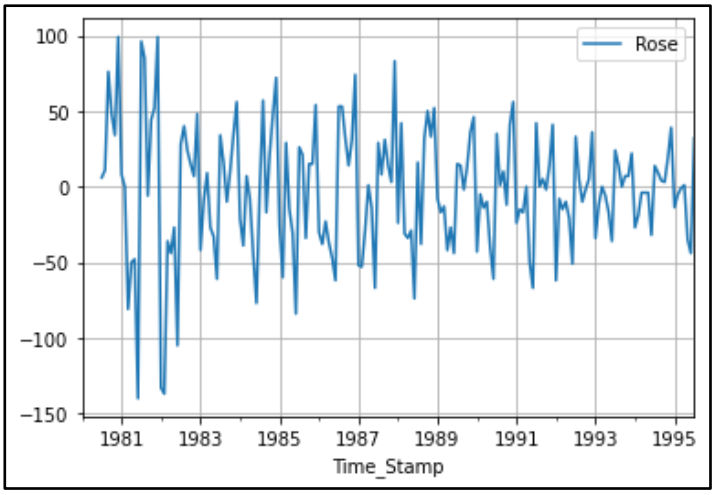
The accuracy is given by RMSE, **RMSE=15.75**

- **SARIMA based on ACF and PACF cut-off points**
  - **6-month seasonality-**



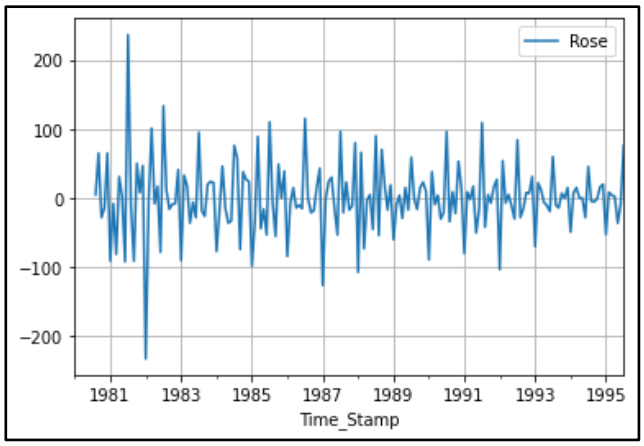


Now we will make a plot for data set considering seasonal difference of 6. Graph is represented as below-

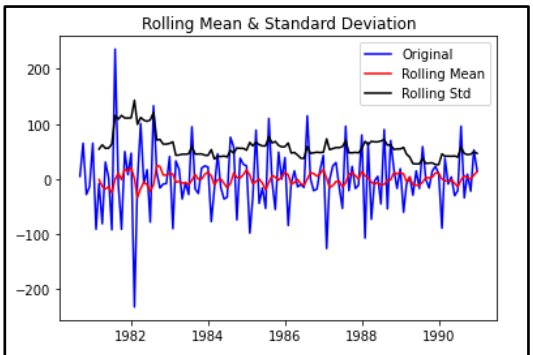


The lag we have used in our model is 50.

We see that there might be a slight trend which can be noticed in the data. So we take a differencing of first order on the seasonally differenced series. Graph is represented as below-



We check the stationarity of data-




```
Results of Dickey-Fuller Test:
Test Statistic                -6.882869e+00
p-value                        1.418693e-09
#Lags Used                     1.300000e+01
Number of Observations Used    1.110000e+02
Critical Value (1%)           -3.490683e+00
Critical Value (5%)           -2.887952e+00
Critical Value (10%)          -2.580857e+00
dtype: float64
```

The data is stationary as p-value is less than 0.05 , We take p,d,q as 2,1,2 and build the model.

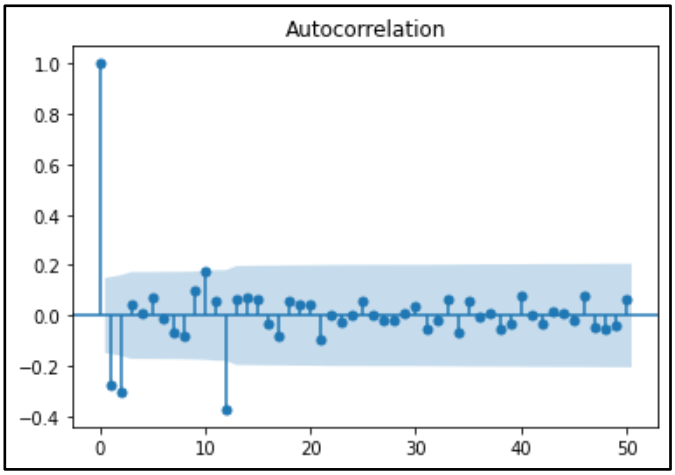
```
                                SARIMAX Results
==========================================================================================
Dep. Variable:                                  y   No. Observations:                  132
Model:             SARIMAX(2, 1, 2)x(1, 1, [1, 2, 3], 6)   Log Likelihood                -454.294
Date:                                Sat, 24 Apr 2021   AIC                            926.588
Time:                                        17:09:16   BIC                            950.387
Sample:                                             0   HQIC                           936.229
                                                  - 132
Covariance Type:                                  opg
==============================================================================
                 coef    std err          z      P>|z|      [0.025      0.975]
------------------------------------------------------------------------------
ar.L1         -0.6050      0.210     -2.877      0.004      -1.017      -0.193
ar.L2          0.0478      0.131      0.365      0.715      -0.209       0.304
ma.L1         -0.1463      0.155     -0.943      0.346      -0.450       0.158
ma.L2         -0.6924      0.138     -5.016      0.000      -0.963      -0.422
ar.S.L6       -0.9050      0.028    -32.557      0.000      -0.959      -0.851
ma.S.L6        0.0536      0.141      0.379      0.705      -0.224       0.331
ma.S.L12      -0.4565      0.117     -3.889      0.000      -0.686      -0.226
ma.S.L18       0.0730      0.113      0.649      0.517      -0.148       0.294
sigma2       356.9421     47.872      7.456      0.000     263.114     450.770
===================================================================================
Ljung-Box (L1) (Q):                   0.01   Jarque-Bera (JB):                 2.23
Prob(Q):                              0.94   Prob(JB):                         0.33
Heteroskedasticity (H):               0.44   Skew:                             0.03
Prob(H) (two-sided):                  0.02   Kurtosis:                         3.72
===================================================================================
```

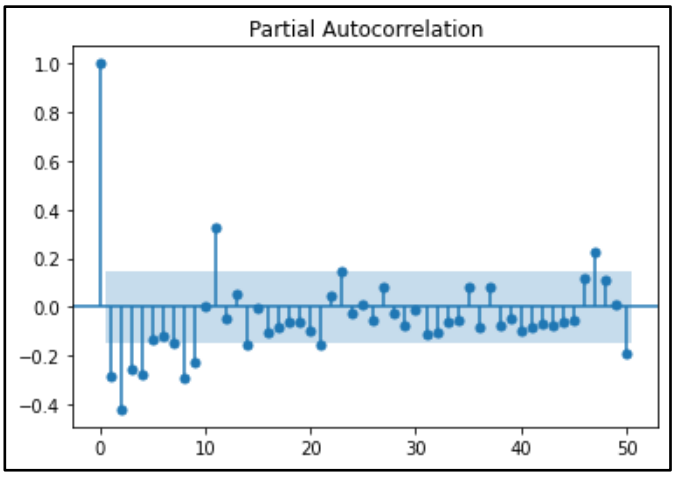
We compute the mean and confidence intervals-

y	mean	mean_se	mean_ci_lower	mean_ci_upper
0	52.681904	18.893025	15.652255	89.711553
1	65.638232	19.468441	27.480789	103.795676
2	72.769530	19.499975	34.550281	110.988779
3	70.096433	19.672851	31.538353	108.654513
4	74.654779	19.732031	35.980709	113.328849

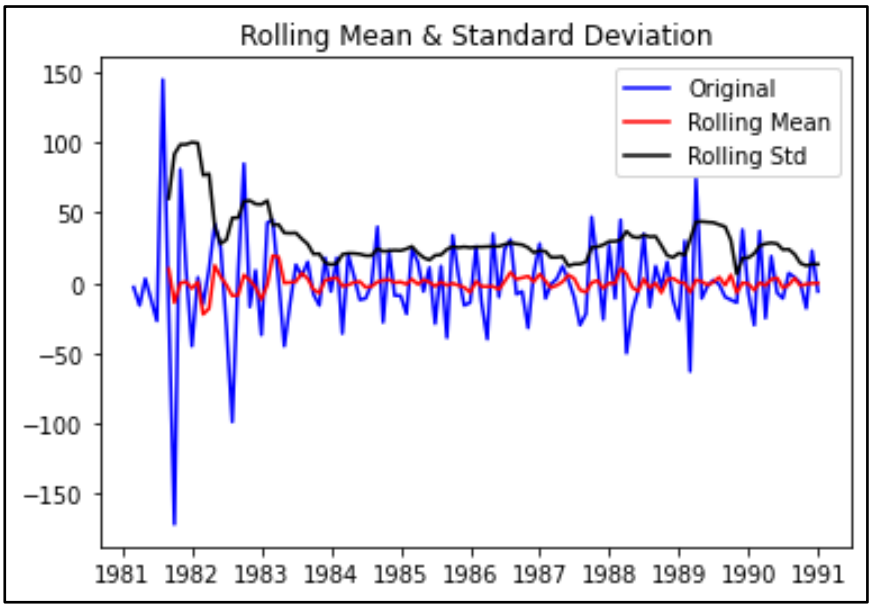
The accuracy is given by **RMSE=20.35**

- **12-month seasonality-**





We check the stationarity of data-



The data is stationary as p-value is less than 0.05 ,

We take p,d,q as 3,1,2 and build the model.

SARIMAX Results

Dep. Variable:	y	No. Observations:	132
Model:	SARIMAX(3, 1, 2)x(1, 1, [1, 2, 3], 12)	Log Likelihood	-1693.730
Date:	Sat, 24 Apr 2021	AIC	3407.459
Time:	17:09:23	BIC	3431.280
Sample:	0	HQIC	3417.009
	- 132		
Covariance Type:	opg		

	coef	std err	z	P>|z|	[0.025	0.975]
ar.L1	-0.4102	-0	inf	0.000	-0.410	-0.410
ar.L2	-0.1670	-0	inf	0.000	-0.167	-0.167
ar.L3	-0.1333	2.17e-34	-6.13e+32	0.000	-0.133	-0.133
ma.L1	-0.1592	3.15e-32	-5.05e+30	0.000	-0.159	-0.159
ma.L2	-0.4595	4.04e-32	-1.14e+31	0.000	-0.459	-0.459
ar.S.L12	-0.2848	1.33e-33	-2.14e+32	0.000	-0.285	-0.285
ma.S.L12	2.46e+14	1.65e-33	1.5e+47	0.000	2.46e+14	2.46e+14
ma.S.L24	1.31e+13	8.69e-48	1.51e+60	0.000	1.31e+13	1.31e+13
ma.S.L36	9.855e+13	6.69e-46	1.47e+59	0.000	9.86e+13	9.86e+13
sigma2	6.447e-12	2.13e-10	0.030	0.976	-4.1e-10	4.23e-10

Ljung-Box (L1) (Q):	2.44	Jarque-Bera (JB):	0.20
Prob(Q):	0.12	Prob(JB):	0.90
Heteroskedasticity (H):	0.80	Skew:	0.12
Prob(H) (two-sided):	0.58	Kurtosis:	3.08

We compute the mean and confidence intervals-

y	mean	mean_se	mean_ci_lower	mean_ci_upper
0	40.998880	6.246373e+08	-1.224267e+09	1.224267e+09
1	34.142372	6.800742e+08	-1.332921e+09	1.332921e+09
2	74.503649	6.804799e+08	-1.333716e+09	1.333716e+09
3	67.332349	6.878389e+08	-1.348139e+09	1.348140e+09
4	59.984549	7.055792e+08	-1.382910e+09	1.382910e+09

The accuracy is given by **RMSE=20.47**

So, all the ARIMA and SARIMA models i.e based on AIC values and ACF/PACF cut-off points with its accuracy are given as –

	RMSE
ARIMA_AIC(0,1,2)	15.640546
SARIMA_AIC(1,1,2)(2,0,2,6)	26.209519
SARIMA_AIC(0,1,2)(2,0,2,12)	26.992038
ARIMA_ACF/PACF(1,1,1)	15.756589
SARIMA_ACF/PACF(2,1,2)(1,1,3,6)	20.350502
SARIMA_ACF/PACF(3,1,2)(1,1,3,12)	20.470001

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

	RMSE
LinearRegression	15.291460
NaiveModel	79.778066
SimpleAverageModel	53.521557
2pointTrailingMovingAverage	11.530180
5pointTrailingMovingAverage	14.491131
8pointTrailingMovingAverage	14.811615
SimpleExponentialSmoothing @Alpha=0.995	36.858571
SimpleExponentialSmoothing @Alpha=0.1	36.890375
DoubleExponentialSmoothing @Alpha=0.1,Beta=0.1	36.987695
TripleExponentialSmoothing @Alpha=0.062,Beta=0.055,Gamma=3.115	21.458971
TripleExponentialSmoothing @Alpha=0.1,Beta=0.2,Gamma=0.2	9.665739
ARIMA_AIC(0,1,2)	15.640546
SARIMA_AIC(1,1,2)(2,0,2,6)	26.209519
SARIMA_AIC(0,1,2)(2,0,2,12)	26.992038
ARIMA_ACF/PACF(1,1,1)	15.756589
SARIMA_ACF/PACF(2,1,2)(1,1,3,6)	20.350502
SARIMA_ACF/PACF(3,1,2)(1,1,3,12)	20.470001

When we sort the values by RMSE scores,

	RMSE
TripleExponentialSmoothing @Alpha=0.1,Beta=0.2,Gamma=0.2	9.665739
2pointTrailingMovingAverage	11.530180
5pointTrailingMovingAverage	14.491131
8pointTrailingMovingAverage	14.811615
LinearRegression	15.291460
ARIMA_AIC(0,1,2)	15.640546
ARIMA_ACF/PACF(1,1,1)	15.756589
SARIMA_ACF/PACF(2,1,2)(1,1,3,6)	20.350502
SARIMA_ACF/PACF(3,1,2)(1,1,3,12)	20.470001
TripleExponentialSmoothing @Alpha=0.062,Beta=0.055,Gamma=3.115	21.458971
SARIMA_AIC(1,1,2)(2,0,2,6)	26.209519
SARIMA_AIC(0,1,2)(2,0,2,12)	26.992038
SimpleExponentialSmoothing @Alpha=0.995	36.858571
SimpleExponentialSmoothing @Alpha=0.1	36.890375
DoubleExponentialSmoothing @Alpha=0.1,Beta=0.1	36.987695
SimpleAverageModel	53.521557
NaiveModel	79.778066

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.

Considering the best model i.e SARIMA with order (2,1,2) at 6 month seasonality, we build a model to predict the next 12 months.

```
full_data_model = sm.tsa.statespace.SARIMAX(df,
                                order=(2,1,2),
                                seasonal_order=(1, 1, 3, 6),
                                enforce_stationarity=False,
                                enforce_invertibility=False)
results_full_data_model = full_data_model.fit(maxiter=1000)
print(results_full_data_model.summary())
```

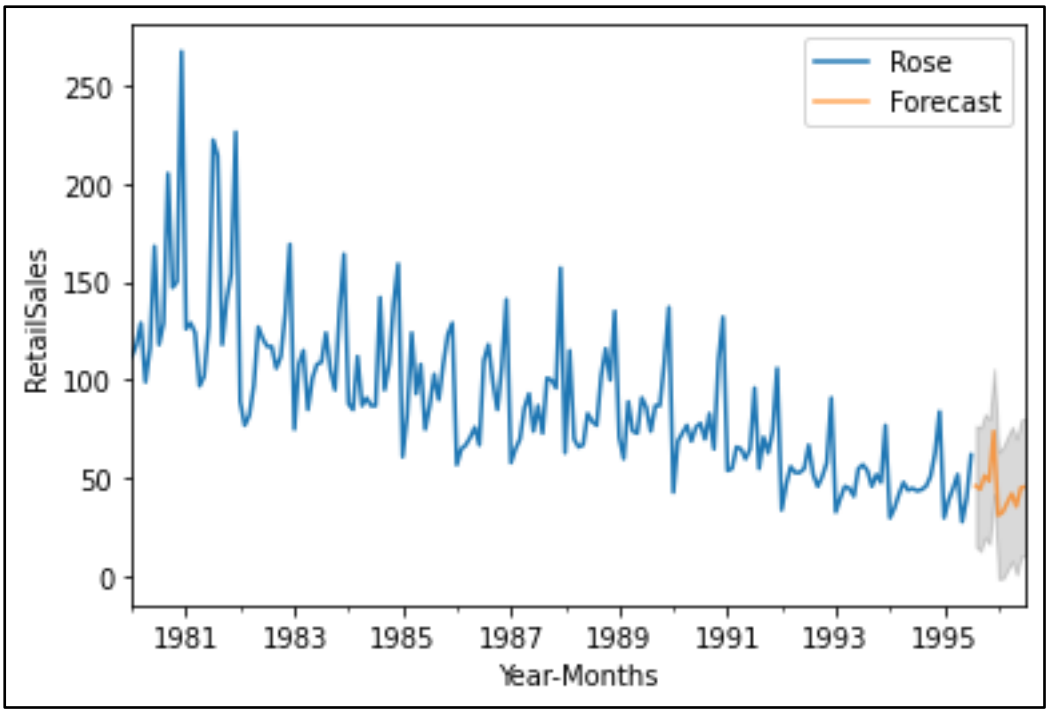
```
                                     SARIMAX Results
==========================================================================================
Dep. Variable:                                Rose   No. Observations:                  187
Model:             SARIMAX(2, 1, 2)x(1, 1, [1, 2, 3], 6)   Log Likelihood                -669.097
Date:                                   Sat, 24 Apr 2021   AIC                           1356.195
Time:                                           17:09:27   BIC                           1383.815
Sample:                                     01-31-1980   HQIC                          1367.411
                                          - 07-31-1995
Covariance Type:                                  opg
==============================================================================
                 coef    std err          z      P>|z|      [0.025      0.975]
------------------------------------------------------------------------------
ar.L1         -0.8956      0.103     -8.675      0.000      -1.098      -0.693
ar.L2          0.0426      0.101      0.422      0.673      -0.155       0.240
ma.L1          0.1064     90.198      0.001      0.999    -176.678     176.891
ma.L2         -0.8936     80.587     -0.011      0.991    -158.840     157.053
ar.S.L6       -0.9549      0.011    -89.857      0.000      -0.976      -0.934
ma.S.L6        0.2107     90.227      0.002      0.998    -176.631     177.052
ma.S.L12      -0.6842     71.196     -0.010      0.992    -140.225     138.857
ma.S.L18       0.1050      9.475      0.011      0.991     -18.465      18.675
sigma2       236.0546      0.621    379.968      0.000     234.837     237.272
==============================================================================
Ljung-Box (L1) (Q):                   0.04   Jarque-Bera (JB):                22.37
Prob(Q):                              0.83   Prob(JB):                         0.00
Heteroskedasticity (H):               0.17   Skew:                             0.34
Prob(H) (two-sided):                  0.00   Kurtosis:                         4.71
==============================================================================

Warnings:
[1] Covariance matrix calculated using the outer product of gradients (complex-step).
[2] Covariance matrix is singular or near-singular, with condition number 7.56e+20. Standard errors may be unstable.
```

Now , forecasting the values with confidence intervals-

Rose	mean	mean_se	mean_ci_lower	mean_ci_upper
1995-08-31	46.262616	15.671262	15.547507	76.977725
1995-09-30	44.592673	15.994434	13.244158	75.941187
1995-10-31	51.479750	16.033458	20.054751	82.904750
1995-11-30	48.639325	16.219823	16.849057	80.429593
1995-12-31	73.950231	16.261941	42.077413	105.823049

RMSE of the full model - **RMSE= 33.958**

The graph showing the predictions for 12 months period from start='1995-08-31',end='1996-7-31' shows that predicted numbers will follow the past trend years . highest sales will be ~80 units and lowest sales will be ~40 units over a period of 12 months.



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

The model that we have built is based on a SARIMA model that has repeated seasonality every 6 months, and the p,d,q as 2,1,2 respectively. Based on the model forecasting , we can observe that the sales range between 40 to 80 units being sold, which is quite less compared to what the company sold in its peak, but as we have seen the pattern/ trend decreasing over years, this doesn't look like an anomaly.

What we suggest is that the company should encourage sales in the summer and other seasons by putting some discounts and other offers which encourages more customers to buy mainly during the first 3 months and July-August-September.

Problem:

For this particular assignment, the data of different types of wine sales in the 20th century is to be analysed. Both of these data are from the same company but of different wines. As an analyst in the ABC Estate Wines, you are tasked to analyse and forecast Wine Sales in the 20th century.

Please do perform the following questions on each of these two data sets separately.

DataSet – Sparkling.csv

The dataset consists of 2 columns – Year-Month(date) and the sales for that particular month (continuous variable).We have to analyse the data, prepare models and predict the sales for next 12 months based on the Time Series.

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

Initially we import all the libraries : -

```
import pandas                           as      pd
import numpy                            as      np
import matplotlib.pyplot                as      plt
import statsmodels.tools.eval_measures  as      em
from   sklearn.metrics                  import  mean_squared_error
from   statsmodels.tsa.api              import ExponentialSmoothing, SimpleExpSmoothing, Holt
from   IPython.display                  import display
from   pylab                            import rcParams
import warnings
warnings.filterwarnings("ignore")
```

Now using the pandas library, we use the read_csv() to read the data.

	YearMonth	Sparkling
0	1980-01	1686
1	1980-02	1591
2	1980-03	2304
3	1980-04	1712
4	1980-05	1471

But, since this is a time-series data, we need to change the format and it can be done either using parsedate or the other way is –

```
1 date = pd.date_range(start='1/1/1980', end='8/1/1995', freq='M')
2 date
```

```
DatetimeIndex(['1980-01-31', '1980-02-29', '1980-03-31', '1980-04-30',
               '1980-05-31', '1980-06-30', '1980-07-31', '1980-08-31',
               '1980-09-30', '1980-10-31',
               ...
               '1994-10-31', '1994-11-30', '1994-12-31', '1995-01-31',
               '1995-02-28', '1995-03-31', '1995-04-30', '1995-05-31',
               '1995-06-30', '1995-07-31'],
              dtype='datetime64[ns]', length=187, freq='M')
```

Now we add this date range as a column TimeStamp and then use that column for indexing.

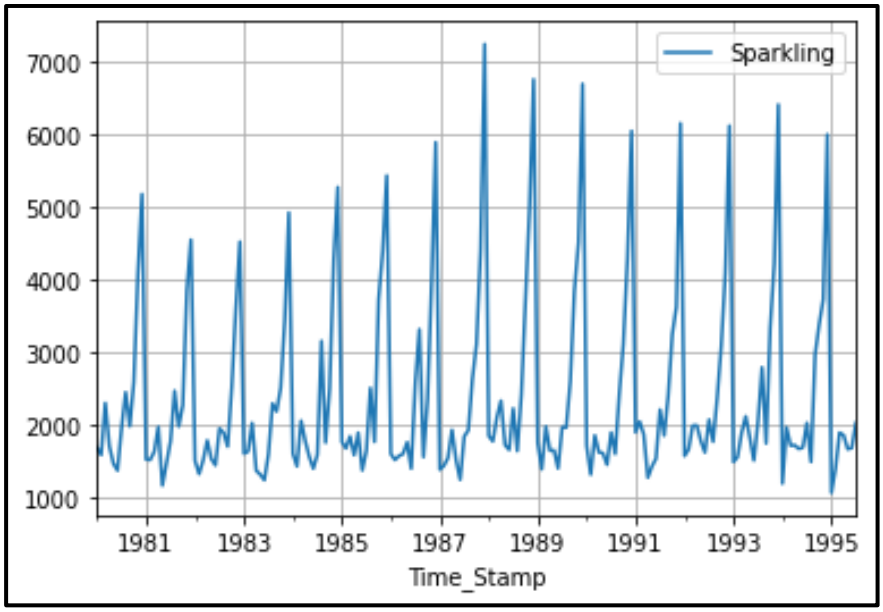
```
df['Time_Stamp'] = pd.DataFrame(date,columns=['Month'])
df.head()
```

	YearMonth	Sparkling	Time_Stamp
0	1980-01	1686	1980-01-31
1	1980-02	1591	1980-02-29
2	1980-03	2304	1980-03-31
3	1980-04	1712	1980-04-30
4	1980-05	1471	1980-05-31

```
df['Time_Stamp'] = pd.to_datetime(df['Time_Stamp'])
df = df.set_index('Time_Stamp')
df.drop(['YearMonth'], axis=1, inplace=True)
df.head()
```

Time_Stamp	Sparkling
1980-01-31	1686
1980-02-29	1591
1980-03-31	2304
1980-04-30	1712
1980-05-31	1471

Now we have our data in a time-series format. So we can plot it.



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

Firstly, we check the 5 number summary and number of missing values in our data using describe () and isnull().sum()

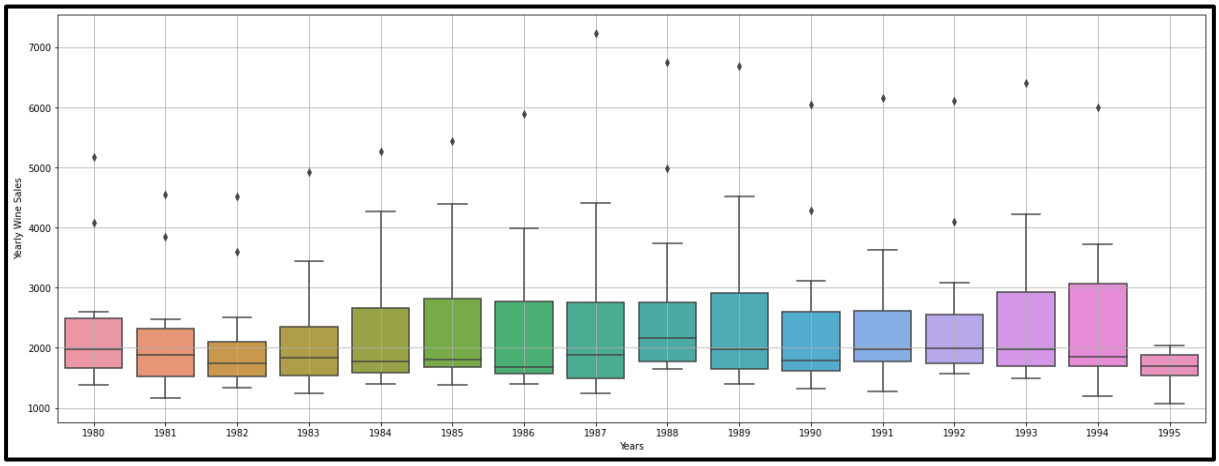
```
df.describe()
```

	Sparkling
count	187.000000
mean	2402.417112
std	1295.111540
min	1070.000000
25%	1605.000000
50%	1874.000000
75%	2549.000000
max	7242.000000

```
df.isnull().sum()
```

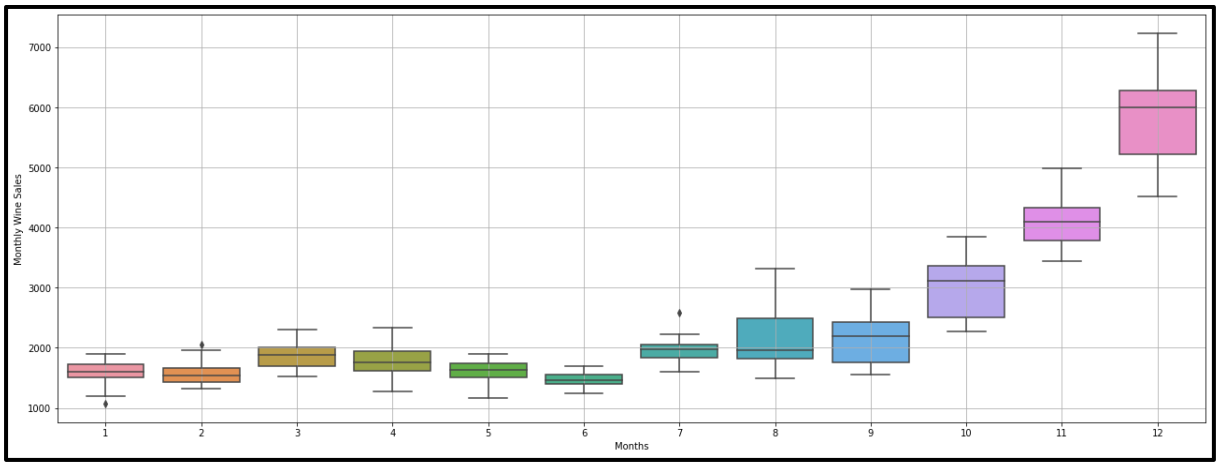
```
Sparkling    0
dtype: int64
```

BoxPlots by Year



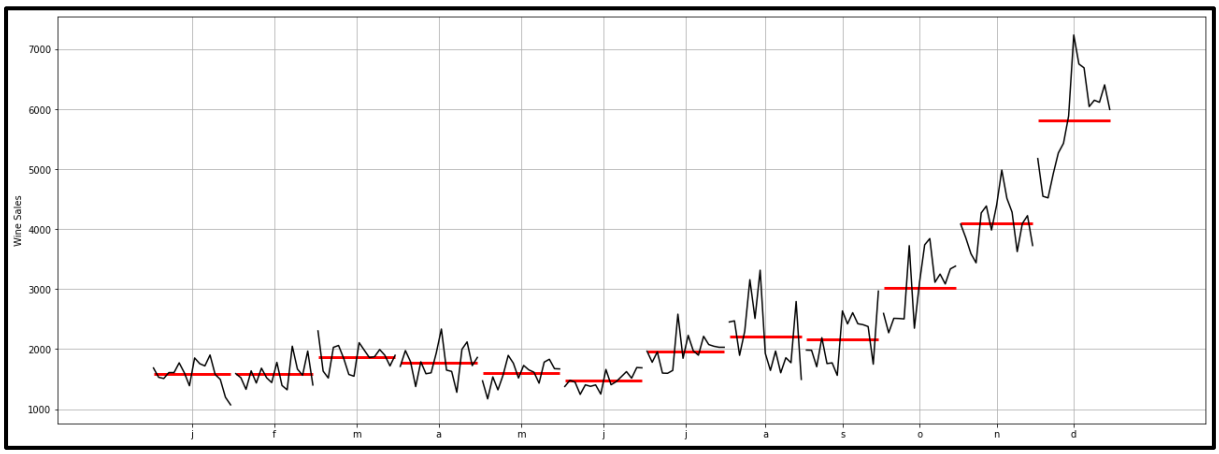
Here, we can see that sales were higher initially and then it started dropping down.

BoxPlots by Month



Here, we can see that sales are generally high towards the year's end.

Monthly Plot for TimeSeries

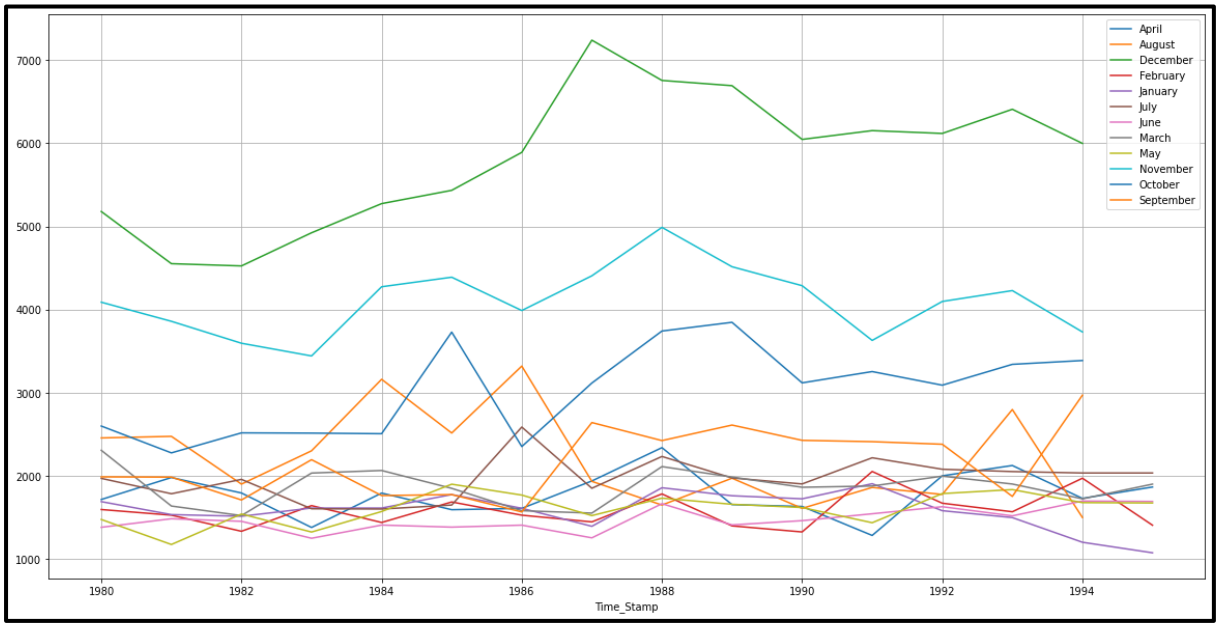


The red lines indicate the average sales for the month.

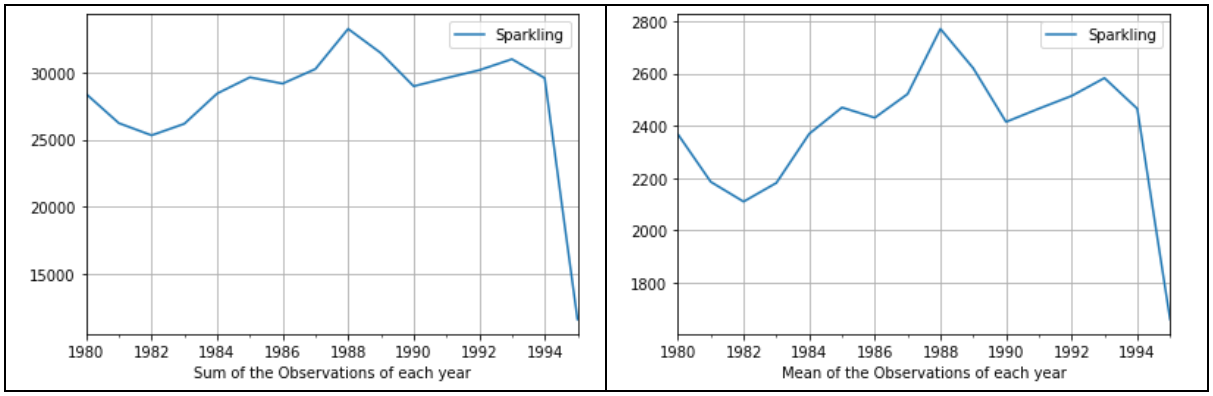
Pivot Table Months vs Year

YearMonth	April	August	December	February	January	July	June	March	May	November	October	September
Time_Stamp												
1980	1712.0	2453.0	5179.0	1591.0	1686.0	1966.0	1377.0	2304.0	1471.0	4087.0	2596.0	1984.0
1981	1976.0	2472.0	4551.0	1523.0	1530.0	1781.0	1480.0	1633.0	1170.0	3857.0	2273.0	1981.0
1982	1790.0	1897.0	4524.0	1329.0	1510.0	1954.0	1449.0	1518.0	1537.0	3593.0	2514.0	1706.0
1983	1375.0	2298.0	4923.0	1638.0	1609.0	1600.0	1245.0	2030.0	1320.0	3440.0	2511.0	2191.0
1984	1789.0	3159.0	5274.0	1435.0	1609.0	1597.0	1404.0	2061.0	1567.0	4273.0	2504.0	1759.0
1985	1589.0	2512.0	5434.0	1682.0	1771.0	1645.0	1379.0	1846.0	1896.0	4388.0	3727.0	1771.0
1986	1605.0	3318.0	5891.0	1523.0	1606.0	2584.0	1403.0	1577.0	1765.0	3987.0	2349.0	1562.0
1987	1935.0	1930.0	7242.0	1442.0	1389.0	1847.0	1250.0	1548.0	1518.0	4405.0	3114.0	2638.0
1988	2336.0	1645.0	6757.0	1779.0	1853.0	2230.0	1661.0	2108.0	1728.0	4988.0	3740.0	2421.0
1989	1650.0	1968.0	6694.0	1394.0	1757.0	1971.0	1406.0	1982.0	1654.0	4514.0	3845.0	2608.0
1990	1628.0	1605.0	6047.0	1321.0	1720.0	1899.0	1457.0	1859.0	1615.0	4286.0	3116.0	2424.0
1991	1279.0	1857.0	6153.0	2049.0	1902.0	2214.0	1540.0	1874.0	1432.0	3627.0	3252.0	2408.0
1992	1997.0	1773.0	6119.0	1667.0	1577.0	2076.0	1625.0	1993.0	1783.0	4096.0	3088.0	2377.0
1993	2121.0	2795.0	6410.0	1564.0	1494.0	2048.0	1515.0	1898.0	1831.0	4227.0	3339.0	1749.0
1994	1725.0	1495.0	5999.0	1968.0	1197.0	2031.0	1693.0	1720.0	1674.0	3729.0	3385.0	2968.0
1995	1862.0	NaN	NaN	1402.0	1070.0	2031.0	1688.0	1897.0	1670.0	NaN	NaN	NaN

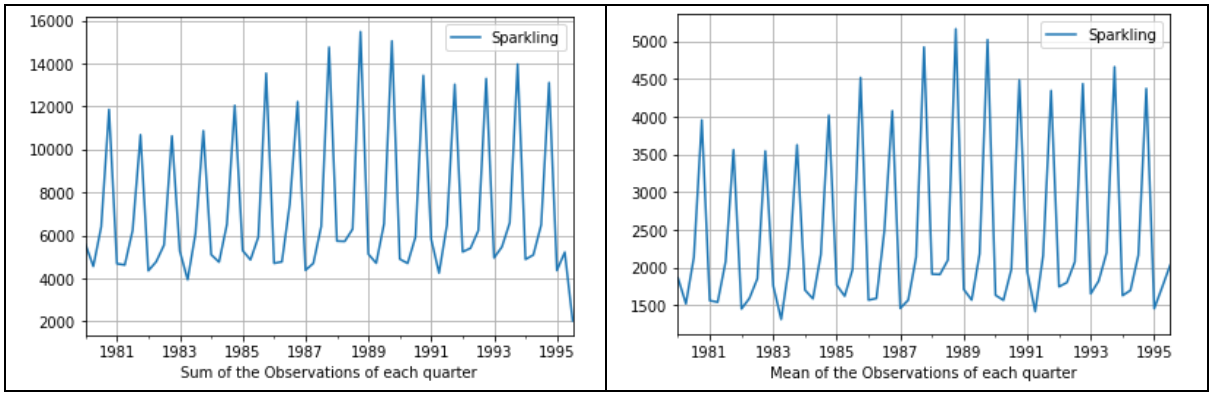
Monthly Sales across Years



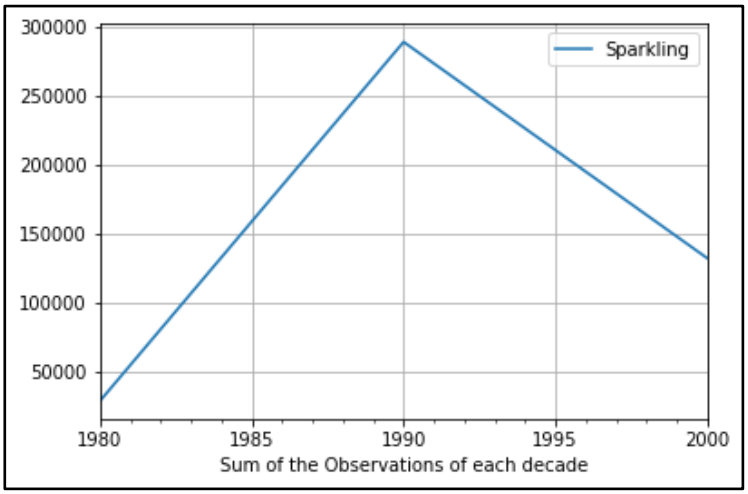
Yearly Plots (Sum and Mean Sales)



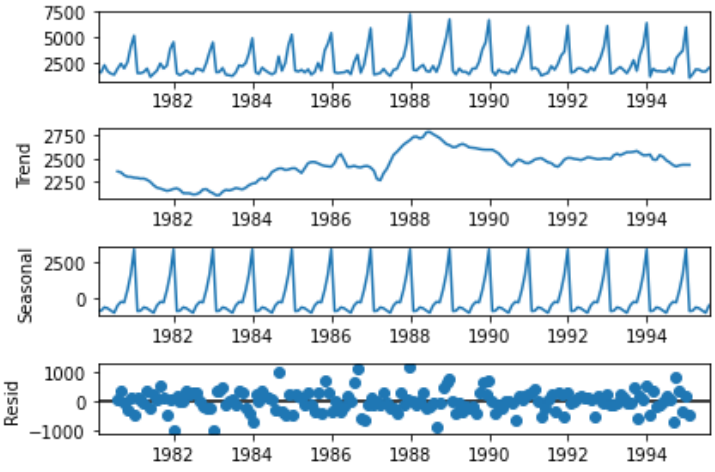
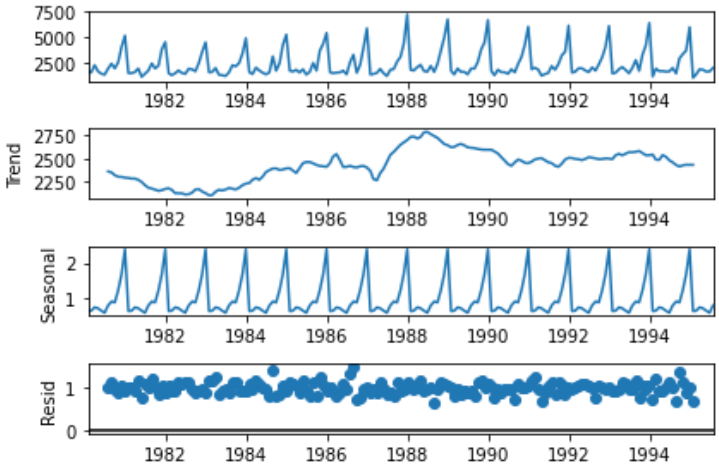
Quarterly Plots (Sum and Mean Sales)



Decade Plot (Sum Sales)



We now have all the values, so we can decompose our series to check the seasonality, trend and residual components. There are 2 methods- additive and multiplicative.

Additive	Multiplicative
	

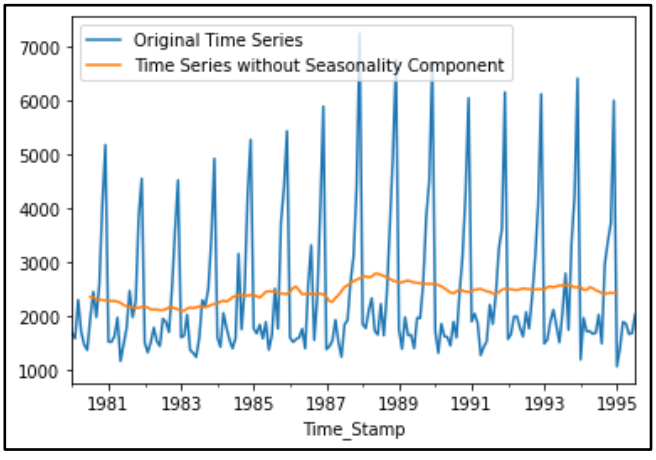
Looking at the decompositions, we can see that both the methods show similar trends and seasonality i.e downward trend and a repetitive seasonality. But the residuals vary. In additive method, the residuals are scattered at level 0 and scatter between -1000 and 1000 whereas in multiplicative, we can observe the residuals are more varied around level 1. Hence, we conclude that we have a multiplicative model.

Now we further split the series so we can look at these components individually.

```
Trend
 Time_Stamp
1980-01-31             NaN
1980-02-29             NaN
1980-03-31             NaN
1980-04-30             NaN
1980-05-31             NaN
1980-06-30             NaN
1980-07-31     2360.666667
1980-08-31     2351.333333
1980-09-30     2320.541667
1980-10-31     2303.583333
1980-11-30     2302.041667
1980-12-31     2293.791667
Name: trend, dtype: float64

Seasonality
 Time_Stamp
1980-01-31     0.649843
1980-02-29     0.659214
1980-03-31     0.757440
1980-04-30     0.730351
1980-05-31     0.660609
1980-06-30     0.603468
1980-07-31     0.809164
1980-08-31     0.918822
1980-09-30     0.894367
1980-10-31     1.241789
1980-11-30     1.690158
1980-12-31     2.384776
Name: seasonal, dtype: float64

Residual
 Time_Stamp
1980-01-31          NaN
1980-02-29          NaN
1980-03-31          NaN
1980-04-30          NaN
1980-05-31          NaN
1980-06-30          NaN
1980-07-31     1.029230
1980-08-31     1.135407
1980-09-30     0.955954
1980-10-31     0.907513
1980-11-30     1.050423
1980-12-31     0.946770
Name: resid, dtype: float64
```

```
Time_Stamp
1980-01-31             NaN
1980-02-29             NaN
1980-03-31             NaN
1980-04-30             NaN
1980-05-31             NaN
1980-06-30             NaN
1980-07-31     2361.695896
1980-08-31     2352.468741
1980-09-30     2321.497620
1980-10-31     2304.490847
1980-11-30     2303.092089
1980-12-31     2294.738436
dtype: float64
```

We can also plot the time series without the seasonality component.



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

```
train=df[df.index.year < 1991]
test=df[df.index.year >= 1991]
```

```
print(train.shape)
print (test.shape)
```

```
(132, 1)
(55, 1)
```

First few rows of Training Data

Time_Stamp	Sparkling
1980-01-31	1686
1980-02-29	1591
1980-03-31	2304
1980-04-30	1712
1980-05-31	1471

Last few rows of Training Data

Time_Stamp	Sparkling
1990-08-31	1605
1990-09-30	2424
1990-10-31	3116
1990-11-30	4286
1990-12-31	6047

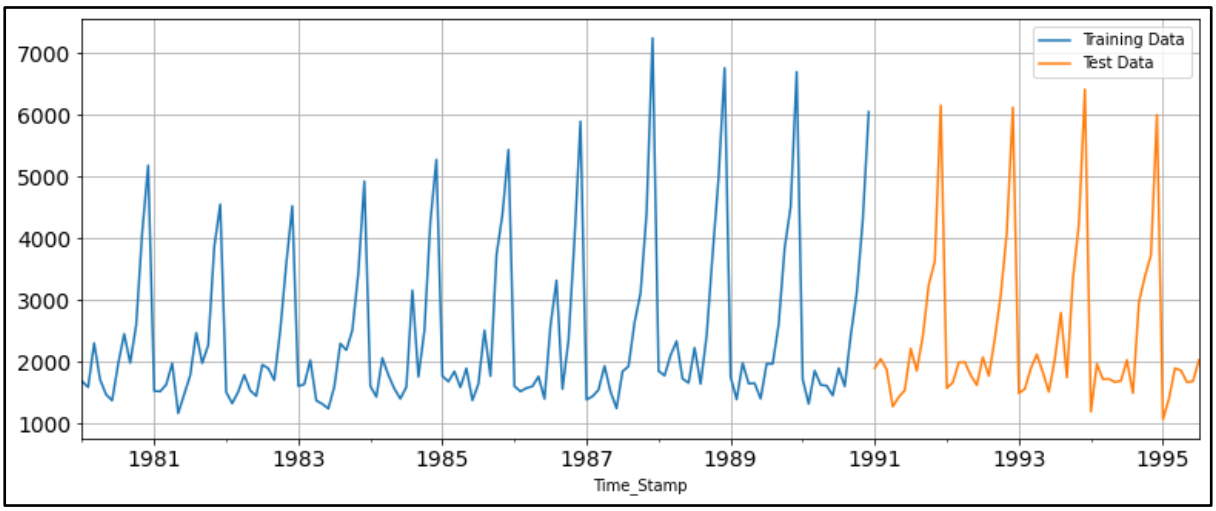
First few rows of Test Data

Time_Stamp	Sparkling
1991-01-31	1902
1991-02-28	2049
1991-03-31	1874
1991-04-30	1279
1991-05-31	1432

Last few rows of Test Data

Time_Stamp	Sparkling
1995-03-31	1897
1995-04-30	1862
1995-05-31	1670
1995-06-30	1688
1995-07-31	2031

We can also see the training and test set data visually in form of plot.



4.Build various 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, simple average models etc. should also be built on the training data and check the performance on the test data using RMSE.

- **Linear Regression**

This model is based on Linear Regression method to forecast the data.

```
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
 [133, 134, 135, 136, 137, 138, 139, 140, 141, 142, 143, 144, 145, 146, 147, 148, 149, 150, 151, 152, 153, 154, 155, 156, 157,
158, 159, 160, 161, 162, 163, 164, 165, 166, 167, 168, 169, 170, 171, 172, 173, 174, 175, 176, 177, 178, 179, 180, 181, 182, 18
3, 184, 185, 186, 187]
```

We have now created time instances for train and test set.

First few rows of Training Data

Time_Stamp	Sparkling	time
1980-01-31	1686	1
1980-02-29	1591	2
1980-03-31	2304	3
1980-04-30	1712	4
1980-05-31	1471	5

Last few rows of Training Data

Time_Stamp	Sparkling	time
1990-08-31	1605	128
1990-09-30	2424	129
1990-10-31	3116	130
1990-11-30	4286	131
1990-12-31	6047	132

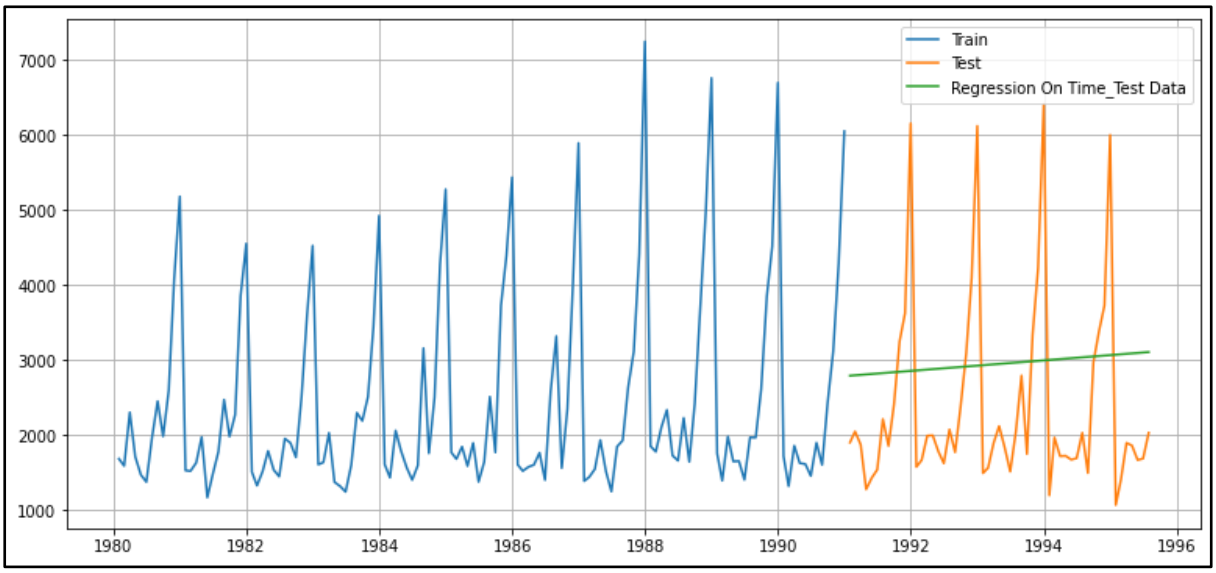
First few rows of Test Data

Time_Stamp	Sparkling	time
1991-01-31	1902	133
1991-02-28	2049	134
1991-03-31	1874	135
1991-04-30	1279	136
1991-05-31	1432	137

Last few rows of Test Data

Time_Stamp	Sparkling	time
1995-03-31	1897	183
1995-04-30	1862	184
1995-05-31	1670	185
1995-06-30	1688	186
1995-07-31	2031	187

Now we build a LinearRegression model on the train set -



The green line represents the forecasted data, which takes care of the trend but the seasonality component seems to be overlooked here.

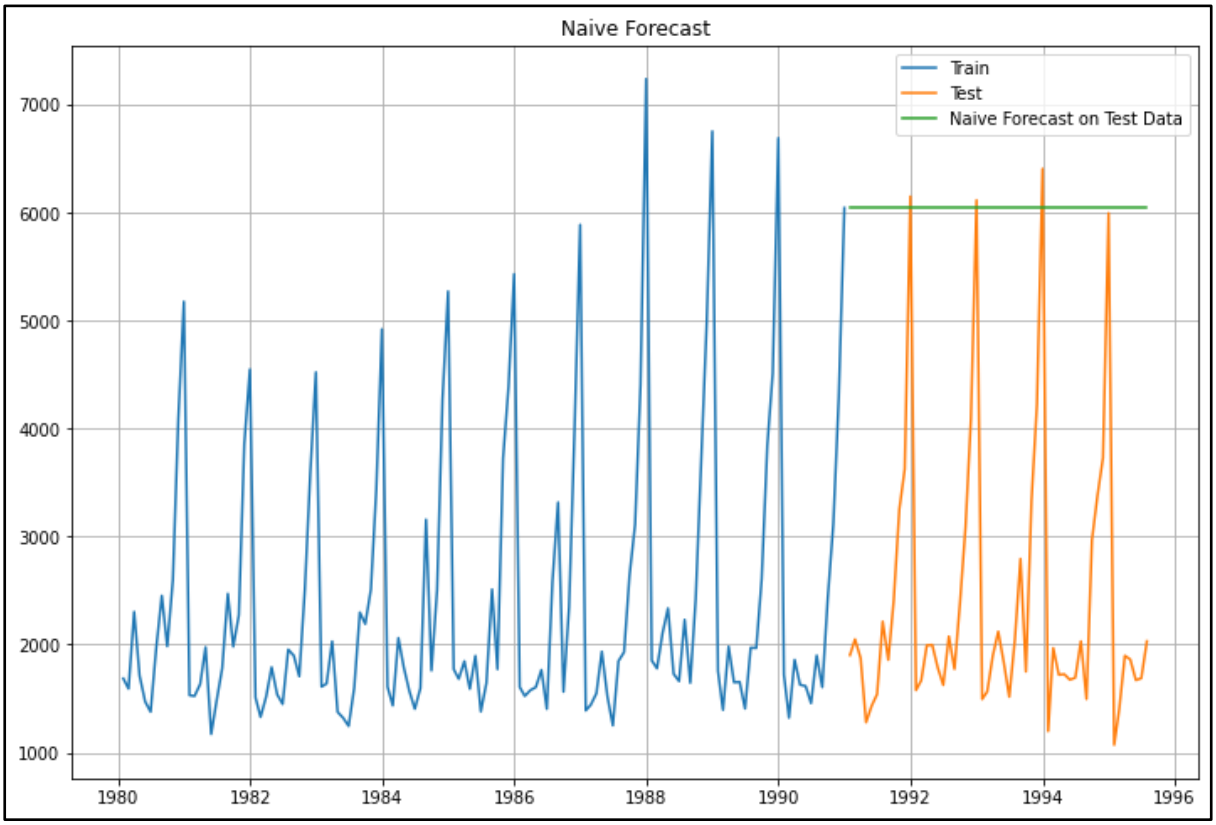
We now check the RMSE on test data- **RMSE= 1389.135**

- **Naïve Approach-**

This approach forecasts the data by simply copying the last occurred value.

```
Time_Stamp
1991-01-31    6047
1991-02-28    6047
1991-03-31    6047
1991-04-30    6047
1991-05-31    6047
Name: naive, dtype: int64
```

We can observe the plot after building the Naïve Approach model-



The green line represents the data forecasted using the Naïve Approach, the straight line clearly shows the same point being duplicated for all the timestamps to be forecasted. The last data point in the train set is taken and used as forecasting data on the whole test set and the observations are far away from what we need/expect the forecasting to be.

Now we calculate the RMSE to check the accuracy of forecasted data –

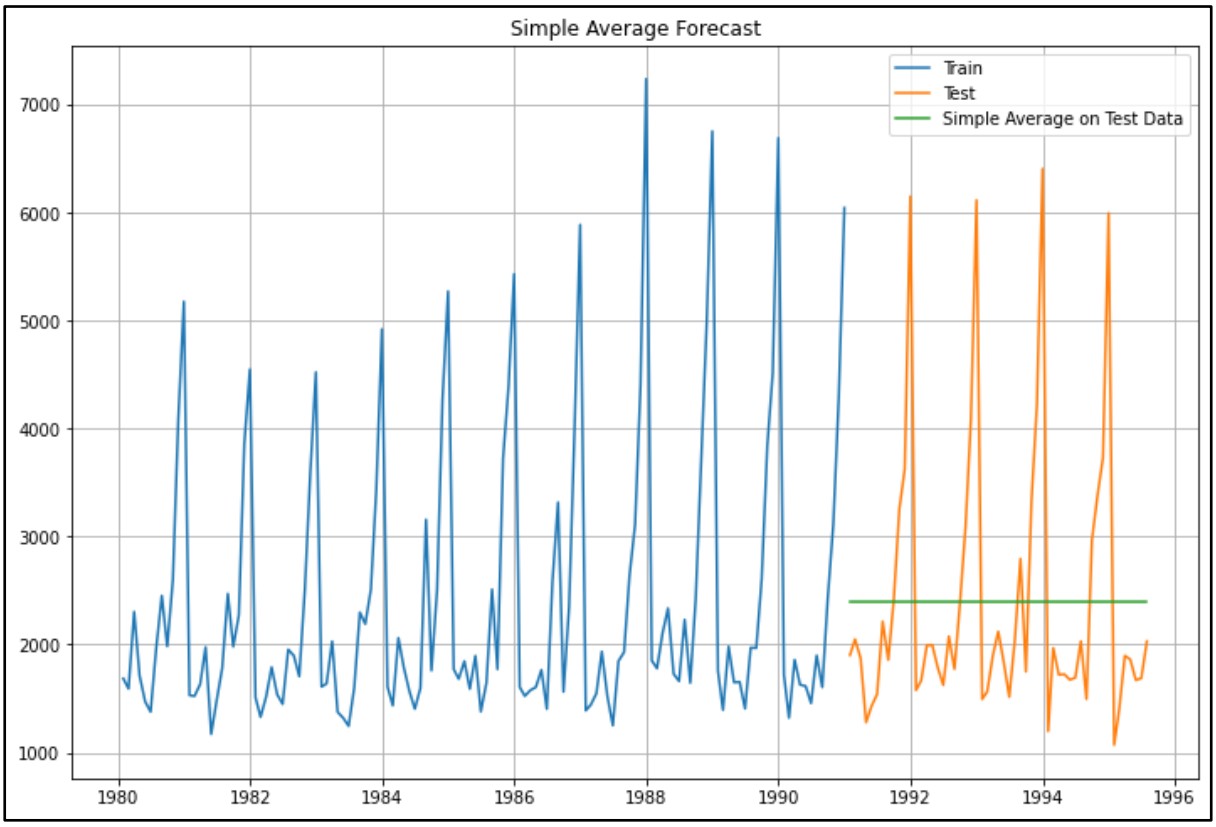
RMSE= 3864.279

- **Simple Average –**

This model is similar to Naïve Approach, but the only difference is that it used the average of all the data points and then uses them as forecasted values.

Time_Stamp	Sparkling	mean_forecast
1991-01-31	1902	2403.780303
1991-02-28	2049	2403.780303
1991-03-31	1874	2403.780303
1991-04-30	1279	2403.780303
1991-05-31	1432	2403.780303

Now we build a model based on it and visualise it using plot –



We can see that the green list is the forecast given by the model and it is not what the required forecast should be. It is better than Naïve Forecast and it can be seen by looking at the RMSE value. RSME on test set- **RMSE=1275.082**

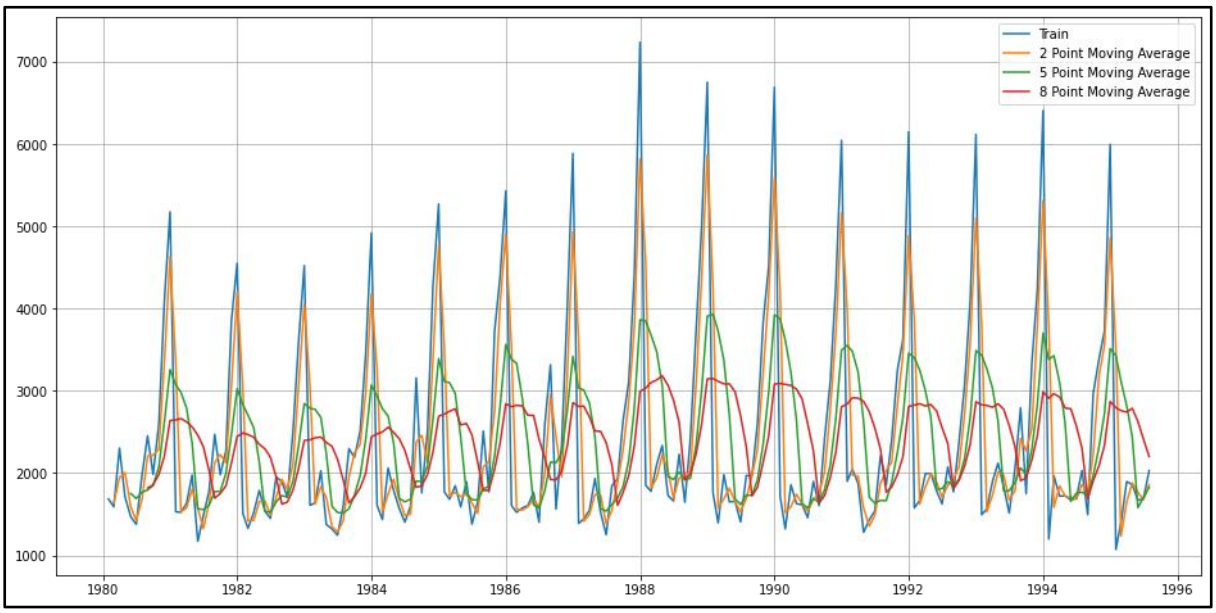
- **Moving Average-**

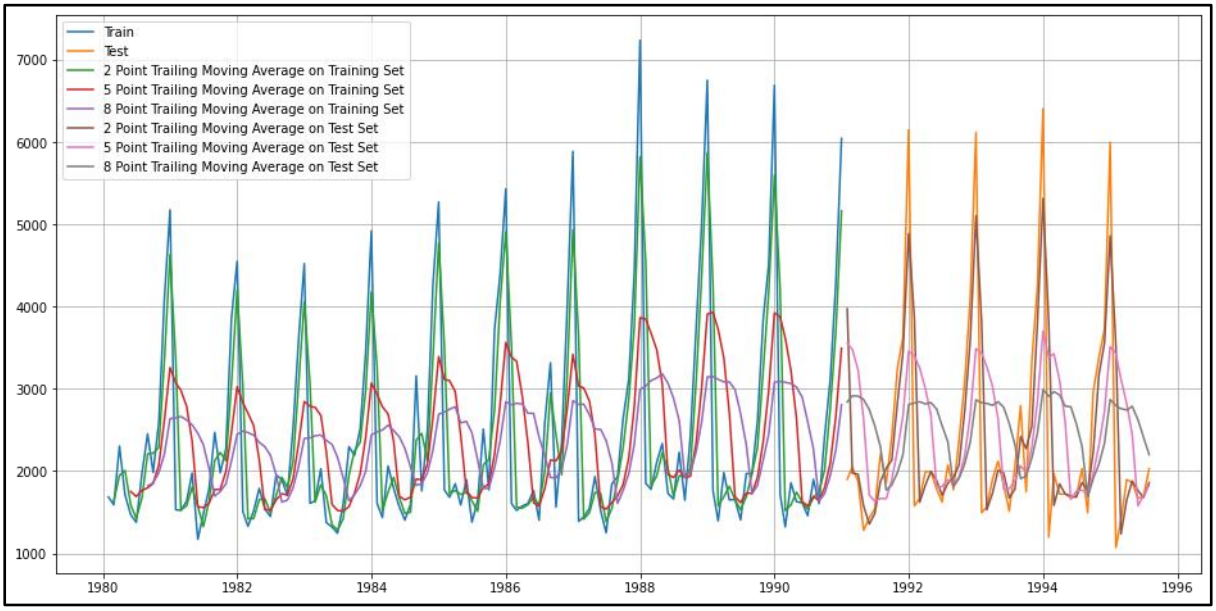
This method uses averaging to forecast the values based on window sizes ie. The window keeps on moving with the size constant for newer points to be forecasted.

Considering window size as 2,5 and 8 –

Time_Stamp	Sparkling	Trailing_2	Trailing_5	Trailing_8
1980-01-31	1686	NaN	NaN	NaN
1980-02-29	1591	1638.5	NaN	NaN
1980-03-31	2304	1947.5	NaN	NaN
1980-04-30	1712	2008.0	NaN	NaN
1980-05-31	1471	1591.5	1752.8	NaN
1980-06-30	1377	1424.0	1691.0	NaN
1980-07-31	1966	1671.5	1766.0	NaN
1980-08-31	2453	2209.5	1795.8	1820.000
1980-09-30	1984	2218.5	1850.2	1857.250
1980-10-31	2596	2290.0	2075.2	1982.875

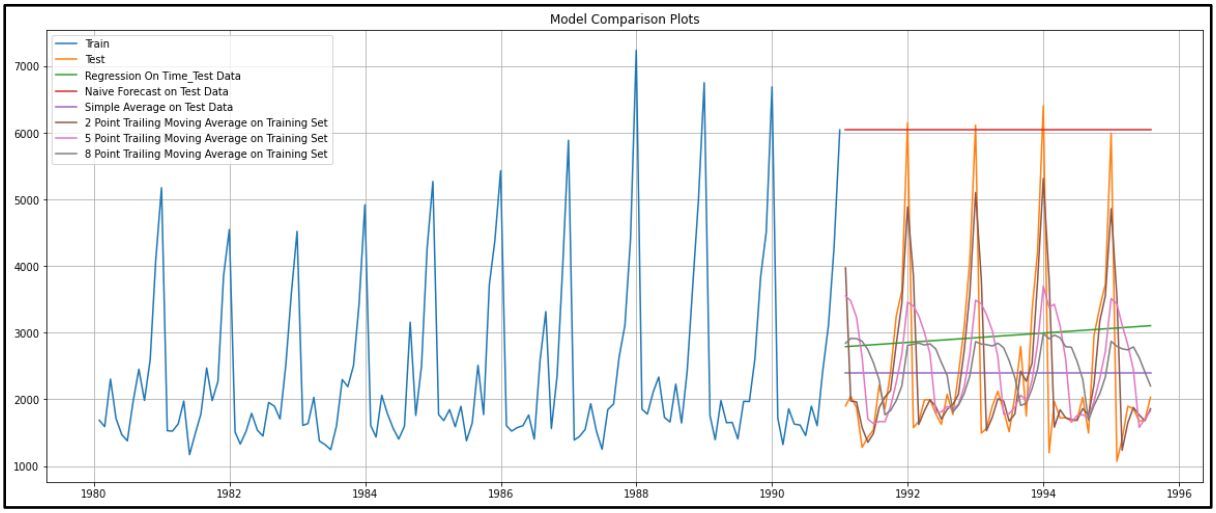
Now we build a model on train set to forecast using this approach and visualize it using plot-





We now calculate the accuracy using the RMSE-

```
For 2 point Moving Average Model forecast on the Training Data,  RMSE is 813.401
For 5 point Moving Average Model forecast on the Training Data,  RMSE is 1234.045
For 8 point Moving Average Model forecast on the Training Data,  RMSE is 1342.568
```



Now we use different Exponential Smoothing models

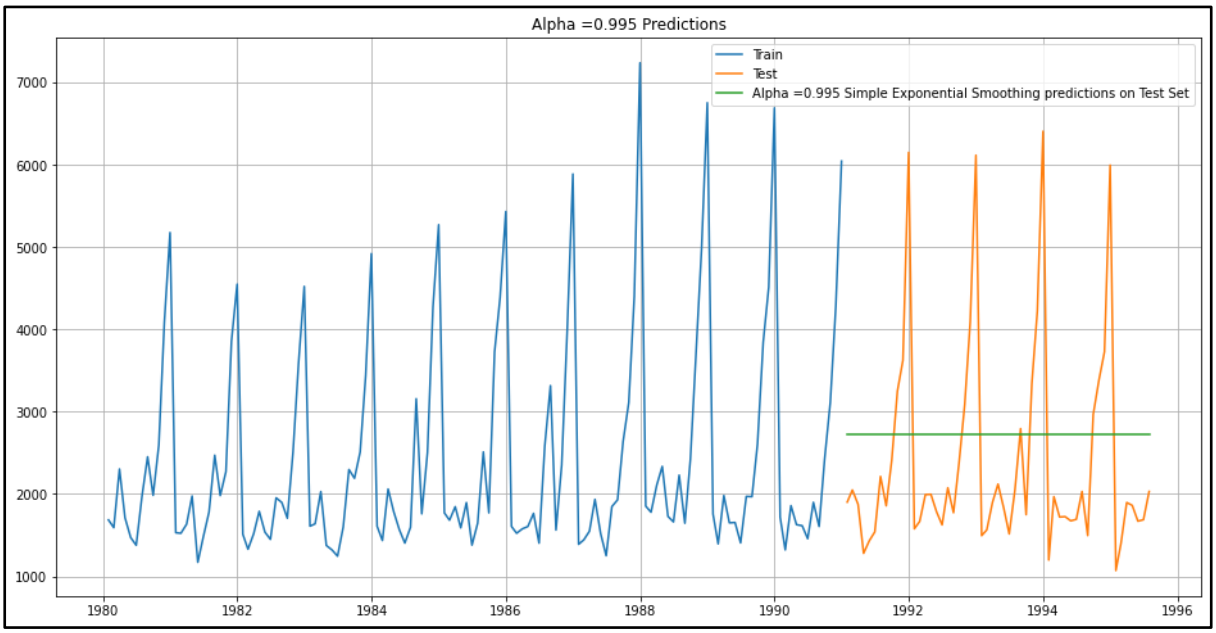
- **Simple Exponential Smoothing-**
  This model considers only level as a parameter.

The model uses the following parameters-

```
{'smoothing_level': 0.049607360581862936,
 'smoothing_trend': nan,
 'smoothing_seasonal': nan,
 'damping_trend': nan,
 'initial_level': 1818.535750008871,
 'initial_trend': nan,
 'initial_seasons': array([], dtype=float64),
 'use_boxcox': False,
 'lamda': None,
 'remove_bias': False}
```

Time_Stamp	Sparkling	predict
1991-01-31	1902	2724.932624
1991-02-28	2049	2724.932624
1991-03-31	1874	2724.932624
1991-04-30	1279	2724.932624
1991-05-31	1432	2724.932624

We build a model on these parameters on train set and forecast on test set. Plot looks like -

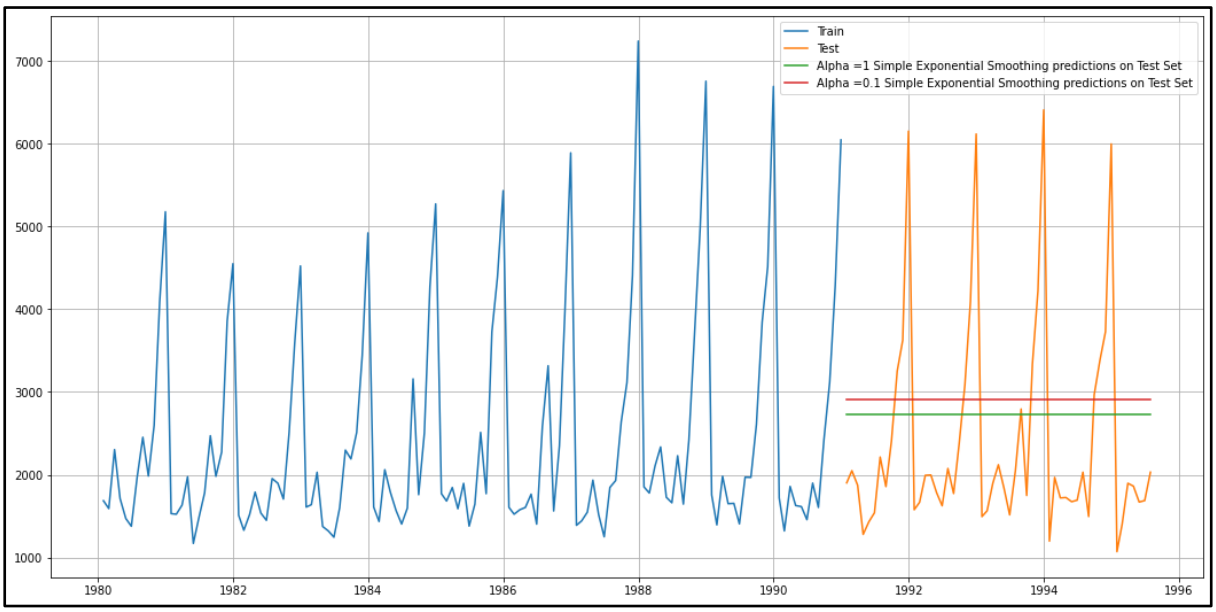


The predicted RMSE on test set is, **RMSE=1316.035**

We can also look at different alpha values and RMSE at those values-

	Alpha Values	Train RMSE	Test RMSE
0	0.1	1333.873836	1375.393398
1	0.2	1356.042987	1595.206839
2	0.3	1359.511747	1935.507132
3	0.4	1352.588879	2311.919615
4	0.5	1344.004369	2666.351413
5	0.6	1338.805381	2979.204388
6	0.7	1338.844308	3249.944092
7	0.8	1344.462091	3483.801006
8	0.9	1355.723518	3686.794285

So, when we plot at alpha at 0.1 and also plot alpha at 0.995, we get a graph as -

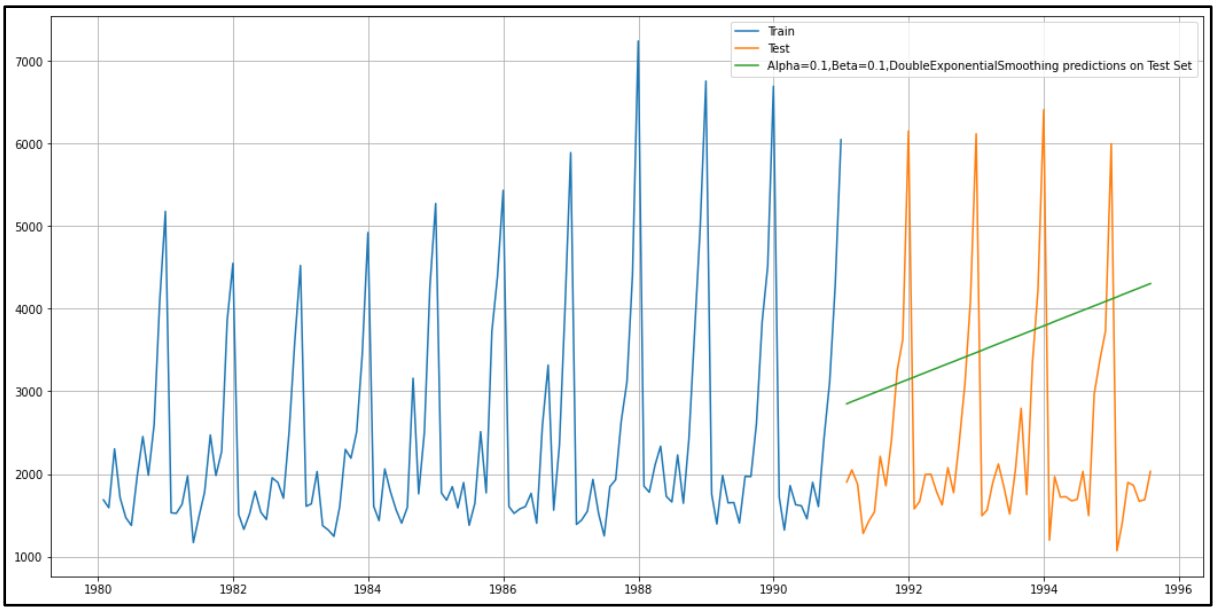


- **Double Exponential Smoothing (Holt's Model) –**
  This model considers level and trend while forecasting values.

We use a combination of different alpha and beta values and consider the best value and build a model on it.

	Alpha Values	Beta Values	Train RMSE	Test RMSE
0	0.1	0.1	1382.520870	1778.564670
1	0.1	0.2	1413.598835	2599.439986
10	0.2	0.1	1418.041591	3611.763322
2	0.1	0.3	1445.762015	4293.084674
20	0.3	0.1	1431.169601	5908.185554

We build a model considering alpha as 0.1 and beta as 0.1 and visualise the forecast on plot-



We check the accuracy of the model using RMSE, **RMSE=1778.564**

- **Triple Exponential Smoothing (Holt-Winter's Model) –**
  This model considers level, trend and seasonality while forecasting values.

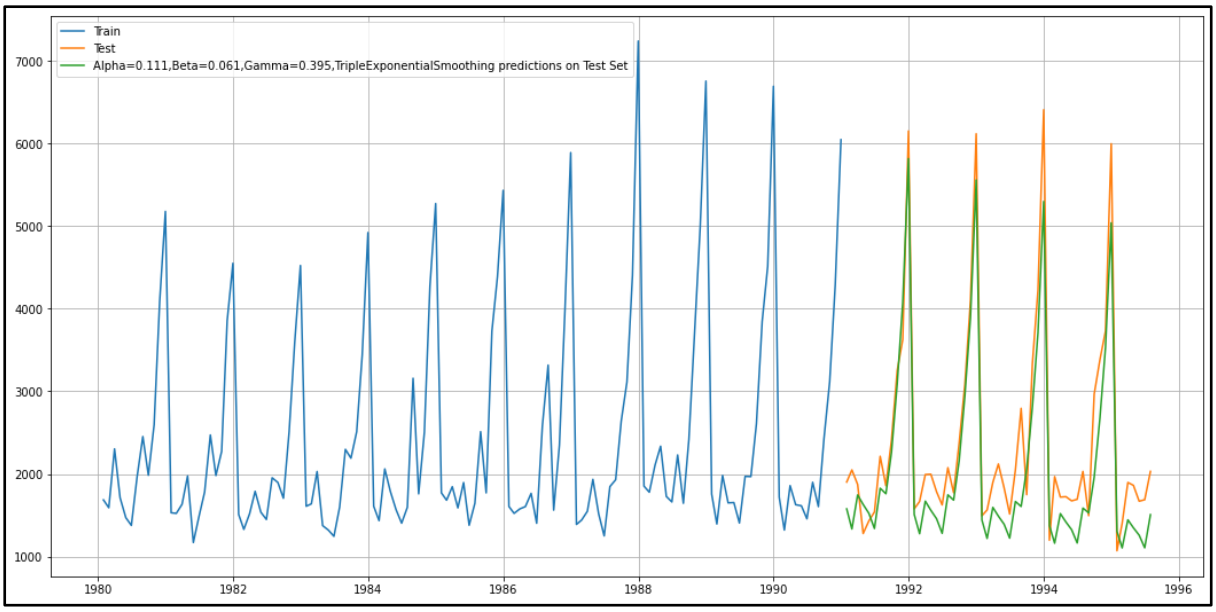
The parameters used for building model-

```
{'smoothing_level': 0.111108139467838,
 'smoothing_trend': 0.06172875597197263,
 'smoothing_seasonal': 0.3950479631147446,
 'damping_trend': nan,
 'initial_level': 1639.9340657558994,
 'initial_trend': -12.22494561218149,
 'initial_seasons': array([1.06402008, 1.02352078, 1.40671876, 1.20165543, 0.97593   ,
       0.97100155, 1.31897446, 1.69588922, 1.3895294 , 1.81476396,
       2.85150039, 3.62470528]),
 'use_boxcox': False,
 'lamda': None,
 'remove_bias': False}
```

Forecasted values at different timestamps for the above mentioned alpha, beta and gamma values-

Time_Stamp	Sparkling	auto_predict
1991-01-31	1902	1577.224489
1991-02-28	2049	1333.677558
1991-03-31	1874	1745.945679
1991-04-30	1279	1630.411925
1991-05-31	1432	1523.289070

Now, we visualise the model build on train set and forecast on test set, plot is given as-

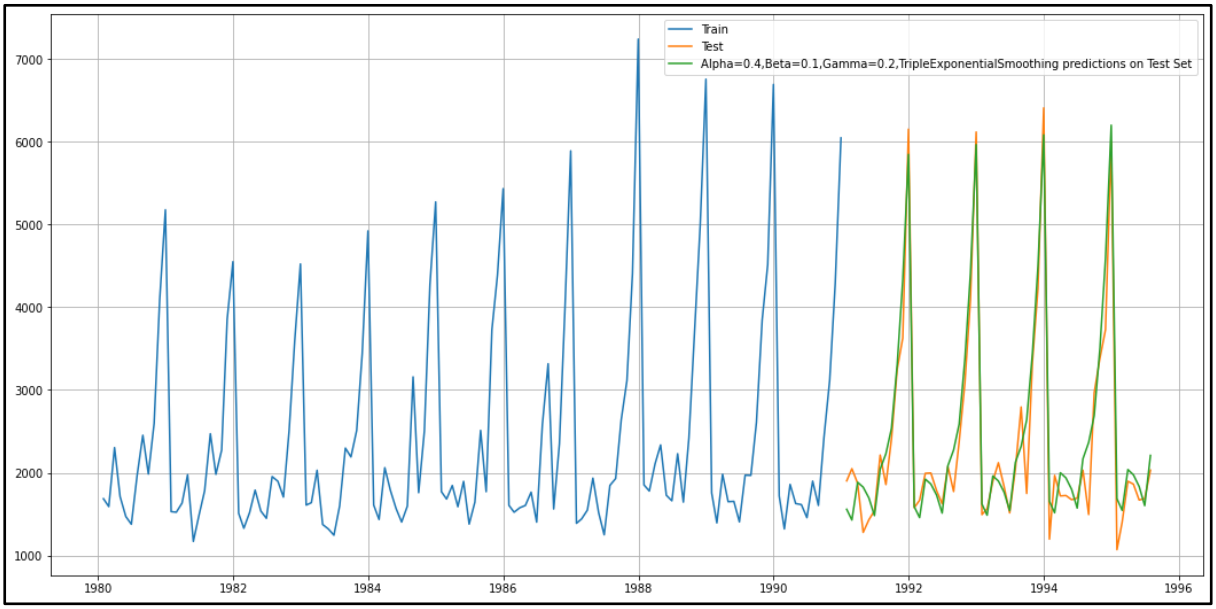


We check the accuracy using RMSE, **RMSE=469.786**

We also build a Triple Exponential model considering the best values for alpha, beta and gamma based on RMSE.

	Alpha Values	Beta Values	Gamma Values	Train RMSE	Test RMSE
301	0.4	0.1	0.2	389.772245	336.715250
211	0.3	0.2	0.2	395.529174	350.145204
110	0.2	0.2	0.1	405.333164	352.571689
200	0.3	0.1	0.1	394.630053	352.607849
20	0.1	0.3	0.1	414.423963	354.534561

So we consider alpha, beta, gamma as 0.4, 0.1, 0.2 respectively and build a model on train set and forecast on test set.

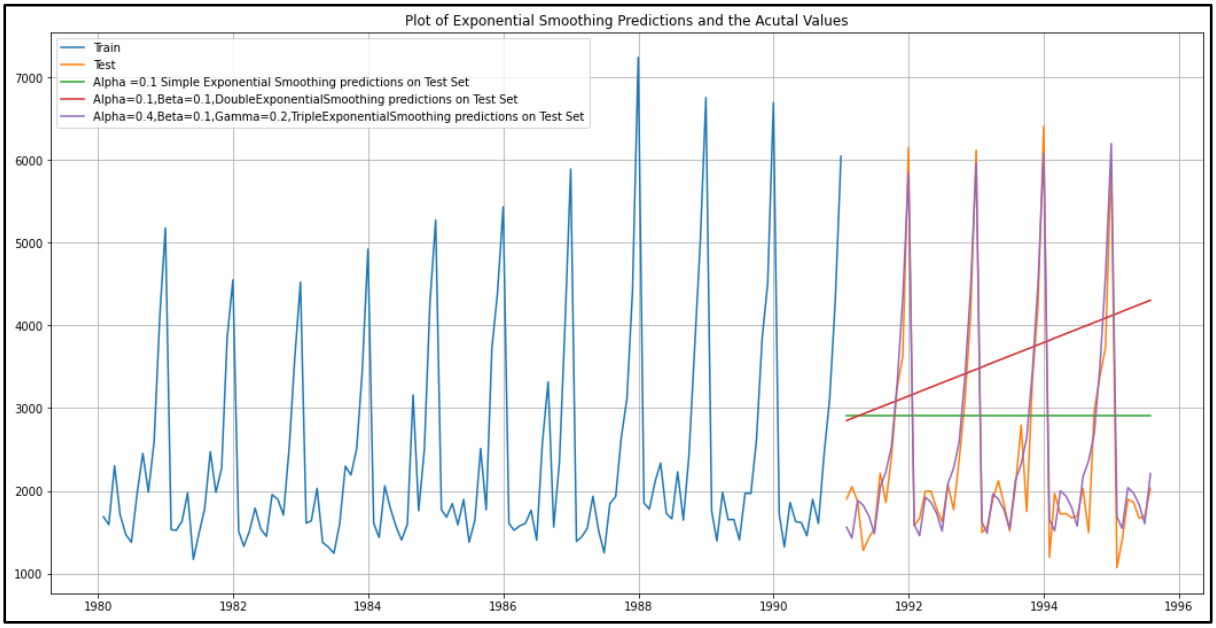


The accuracy of model is, **RMSE=336.715**

All the model that we build and their accuracies in terms of RMSE-

	RMSE
LinearRegression	1389.135175
NaiveModel	3864.279352
SimpleAverageModel	1275.081804
2pointTrailingMovingAverage	813.400684
5pointTrailingMovingAverage	1234.045344
8pointTrailingMovingAverage	1342.567772
SimpleExponentialSmoothing @Alpha=0.995	1316.035487
SimpleExponentialSmoothing @Alpha=0.1	1375.393398
DoubleExponentialSmoothing @Alpha=0.1,Beta=0.1	1778.564670
TripleExponentialSmoothing @Alpha=0.111,Beta=0.061,Gamma=0.395	469.767970
TripleExponentialSmoothing @Alpha=0.4,Beta=0.1,Gamma=0.2	336.715250

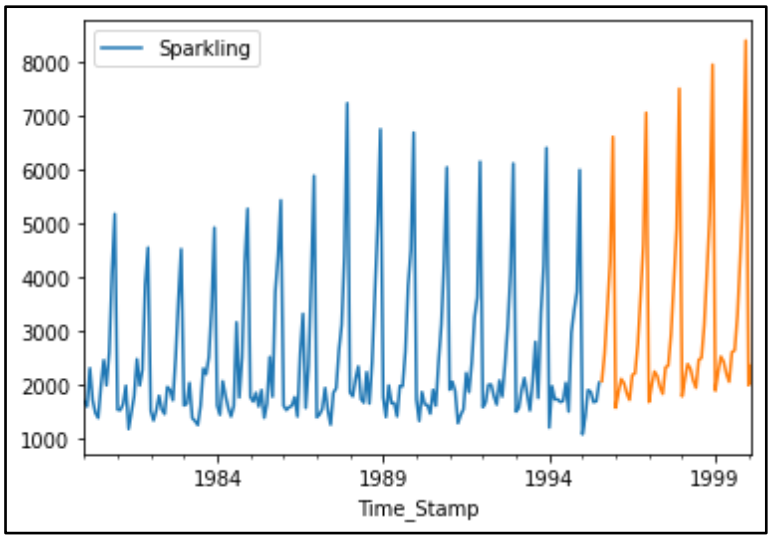
We now plot all the Exponential Smoothing models-



Looking at the RMSE values for all the models, we can conclude that the Triple Exponential Model with alpha, beta, gamma as 0.4,0.1 and 0.2 respectively performs the best. So we build a complete model on Rose dataset on these parameters.

```
fullmodel1 = ExponentialSmoothing(df,
                                  trend='additive',
                                  seasonal='multiplicative').fit(smoothing_level=0.4,
                                                                 smoothing_trend=0.1,
                                                                 smoothing_seasonal=0.2)
```

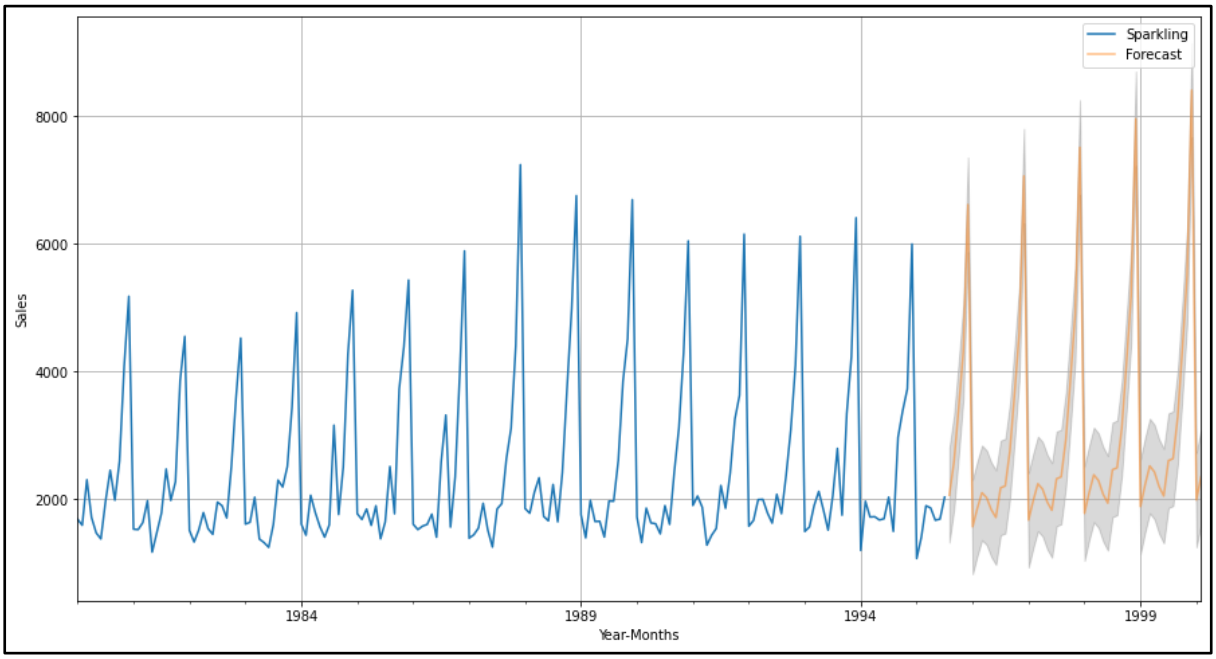
The following is how it predicts-



Considering the confidence levels at alpha at 0.05, we compute the predictions and upper and lower confidence levels-

	lower_CI	prediction	upper_ci
1995-08-31	1321.896024	2063.370030	2804.844037
1995-09-30	1838.303763	2579.777769	3321.251776
1995-10-31	2676.612337	3418.086343	4159.560350
1995-11-30	3567.115379	4308.589385	5050.063392
1995-12-31	5874.310141	6615.784148	7357.258154

We have plotted our forecasting considering the error -



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.)

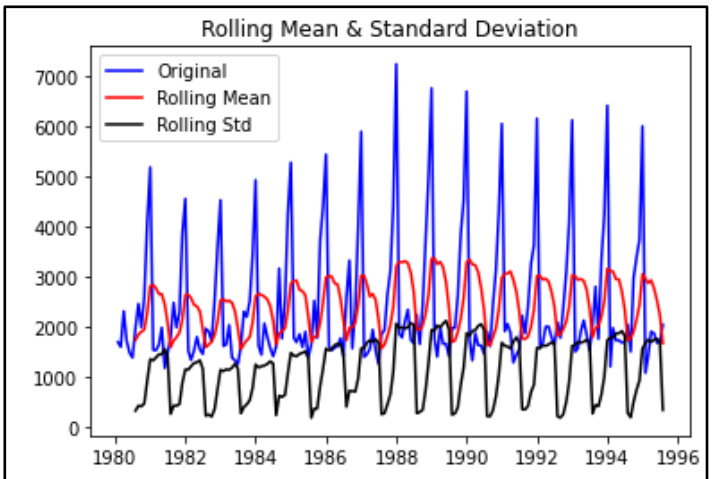
We are checking the Stationarity at alpha = 0.05 for the Time Series data. This will be done using Dicky Fuller Test.

Hypothesis for Dicky Fuller Test is as follows

Ho: Time Series is Non-Stationary

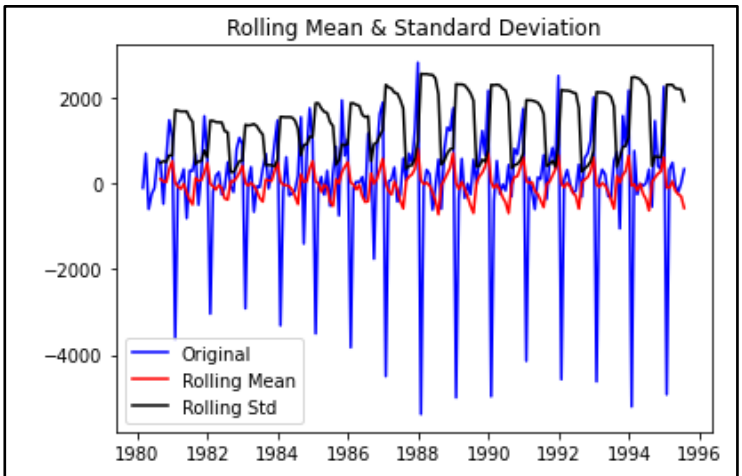
Ha: Time Series is Stationary

The result of Dicky fuller Test is




```
Results of Dickey-Fuller Test:
Test Statistic                  -1.360497
p-value                          0.601061
#Lags Used                      11.000000
Number of Observations Used    175.000000
Critical Value (1%)             -3.468280
Critical Value (5%)             -2.878202
Critical Value (10%)            -2.575653
dtype: float64
```

The p-value is more than 0.05, hence we cannot reject the null. The data is not stationary, which means have to differentiate the time series.




```
Results of Dickey-Fuller Test:
Test Statistic                 -45.050301
p-value                          0.000000
#Lags Used                      10.000000
Number of Observations Used    175.000000
Critical Value (1%)             -3.468280
Critical Value (5%)             -2.878202
Critical Value (10%)            -2.575653
dtype: float64
```

Now our p-value is < 0.05, hence our data is stationary.

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.

- **ARIMA model based on AIC**

```
import itertools
p = q = range(0, 3)
d= range(1,2)
pdq = list(itertools.product(p, d, q))
print('Some parameter combinations for the Model...')
for i in range(1,len(pdq)):
    print('Model: {}'.format(pdq[i]))
```

```
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)
```

```
ARIMA_AIC = pd.DataFrame(columns=['param', 'AIC'])

from statsmodels.tsa.arima_model import ARIMA

for param in pdq:
    ARIMA_model = ARIMA(train.values,order=param).fit()
    print('ARIMA{} - AIC:{}'.format(param,ARIMA_model.aic))
    ARIMA_AIC = ARIMA_AIC.append({'param':param, 'AIC': ARIMA_model.aic}, ignore_index=True)
```

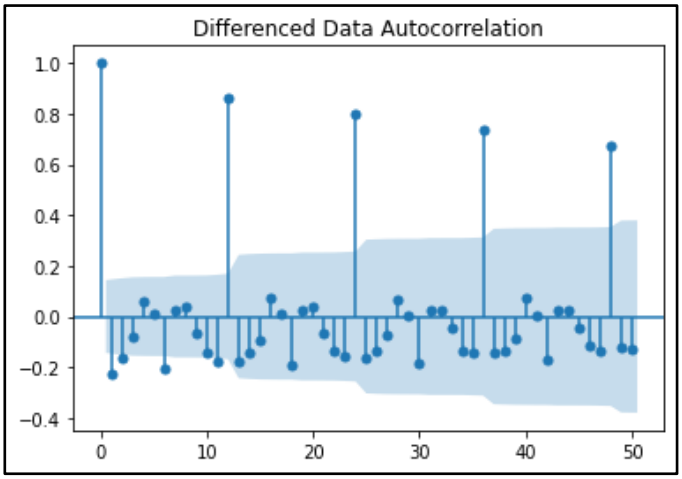
```
ARIMA(0, 1, 0) - AIC:2269.582796371201
ARIMA(0, 1, 1) - AIC:2264.906437115225
ARIMA(0, 1, 2) - AIC:2232.7830976841665
ARIMA(1, 1, 0) - AIC:2268.5280605942976
ARIMA(1, 1, 1) - AIC:2235.0139453494457
ARIMA(1, 1, 2) - AIC:2233.5976471189274
ARIMA(2, 1, 0) - AIC:2262.0356001096925
ARIMA(2, 1, 1) - AIC:2232.360489886391
ARIMA(2, 1, 2) - AIC:2210.6189875760692
```

Since the best performance is at (2,1,2) we build a model on that –

```
                             ARIMA Model Results
==============================================================================
Dep. Variable:            D.Sparkling   No. Observations:                  131
Model:                 ARIMA(2, 1, 2)   Log Likelihood               -1099.309
Method:                       css-mle   S.D. of innovations           1012.929
Date:                Sat, 24 Apr 2021   AIC                           2210.619
Time:                        17:10:28   BIC                           2227.870
Sample:                    02-29-1980   HQIC                          2217.629
                         - 12-31-1990
==========================================================================================
                             coef    std err          z      P>|z|      [0.025      0.975]
------------------------------------------------------------------------------------------
const                     5.5854      0.517     10.806      0.000       4.572       6.598
ar.L1.D.Sparkling         1.2699      0.075     17.046      0.000       1.124       1.416
ar.L2.D.Sparkling        -0.5602      0.074     -7.618      0.000      -0.704      -0.416
ma.L1.D.Sparkling        -1.9974      0.042    -47.112      0.000      -2.080      -1.914
ma.L2.D.Sparkling         0.9974      0.042     23.497      0.000       0.914       1.081
                                    Roots
=============================================================================
                  Real          Imaginary           Modulus         Frequency
-----------------------------------------------------------------------------
AR.1            1.1334           -0.7074j            1.3361           -0.0888
AR.2            1.1334           +0.7074j            1.3361            0.0888
MA.1            1.0006           +0.0000j            1.0006            0.0000
MA.2            1.0020           +0.0000j            1.0020            0.0000
-----------------------------------------------------------------------------
```

Now, we predict accuracy on the test set by RMSE- **RMSE= 1374.484**

- **SARIMA model based on AIC**



Looking at this we can see that the seasonality is repeated 6 and 12 months. So we build SARIMA models for 6 and 12 months.

➢ **6-month seasonality-**

```
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)
```

	param	seasonal	AIC
53	(1, 1, 2)	(2, 0, 2, 6)	1727.678698
26	(0, 1, 2)	(2, 0, 2, 6)	1727.888814
17	(0, 1, 1)	(2, 0, 2, 6)	1741.703707
44	(1, 1, 1)	(2, 0, 2, 6)	1743.330541
71	(2, 1, 1)	(2, 0, 2, 6)	1744.040751

Based on the AIC values, we build model on parameter- (1,1,2) seasonal- (2,0,2,6)

```
                                 SARIMAX Results
==========================================================================================
Dep. Variable:                                 y   No. Observations:                  132
Model:             SARIMAX(1, 1, 2)x(2, 0, 2, 6)   Log Likelihood                -855.839
Date:                           Sat, 24 Apr 2021   AIC                           1727.679
Time:                                   17:11:01   BIC                           1749.707
Sample:                                        0   HQIC                          1736.621
                                             - 132
Covariance Type:                             opg
==============================================================================
                 coef    std err          z      P>|z|      [0.025      0.975]
------------------------------------------------------------------------------
ar.L1         -0.6449      0.286     -2.256      0.024      -1.205      -0.085
ma.L1         -0.1068      0.250     -0.428      0.669      -0.596       0.383
ma.L2         -0.7006      0.202     -3.471      0.001      -1.096      -0.305
ar.S.L6       -0.0045      0.027     -0.165      0.869      -0.057       0.049
ar.S.L12       1.0361      0.018     56.082      0.000       1.000       1.072
ma.S.L6        0.0676      0.152      0.444      0.657      -0.231       0.366
ma.S.L12      -0.6123      0.093     -6.590      0.000      -0.794      -0.430
sigma2      1.448e+05   1.71e+04      8.466      0.000    1.11e+05    1.78e+05
==========================================================================================
Ljung-Box (L1) (Q):                   0.09   Jarque-Bera (JB):                25.24
Prob(Q):                              0.77   Prob(JB):                         0.00
Heteroskedasticity (H):               2.63   Skew:                             0.47
Prob(H) (two-sided):                  0.00   Kurtosis:                         5.09
==========================================================================================
```

Now, we predict on the test set-

	y	mean	mean_se	mean_ci_lower	mean_ci_upper
0		1330.382316	380.556609	584.505068	2076.259563
1		1177.257047	392.107264	408.740930	1945.773163
2		1625.929998	392.301428	857.033328	2394.826668
3		1546.321157	397.701601	766.840342	2325.801971
4		1308.751650	398.920807	526.881235	2090.622066

The accuracy is given by RMSE, **RMSE= 626.931**

- **12-month seasonality-**

```
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
50	(1, 1, 2)	(1, 0, 2, 12)	1555.584247
53	(1, 1, 2)	(2, 0, 2, 12)	1556.076771
26	(0, 1, 2)	(2, 0, 2, 12)	1557.121563
23	(0, 1, 2)	(1, 0, 2, 12)	1557.160507
77	(2, 1, 2)	(1, 0, 2, 12)	1557.340402

Based on the AIC values, we build model on parameter- (1,1,2) seasonal- (1,0,2,12)

```
                                 SARIMAX Results
==========================================================================================
Dep. Variable:                                 y   No. Observations:                  132
Model:             SARIMAX(1, 1, 2)x(1, 0, 2, 12)   Log Likelihood                -770.792
Date:                            Sat, 24 Apr 2021   AIC                           1555.584
Time:                                    17:11:49   BIC                           1574.095
Sample:                                         0   HQIC                          1563.083
                                              - 132
Covariance Type:                              opg
==============================================================================
                 coef    std err          z      P>|z|      [0.025      0.975]
------------------------------------------------------------------------------
ar.L1         -0.6281      0.255     -2.463      0.014      -1.128      -0.128
ma.L1         -0.1041      0.225     -0.463      0.643      -0.545       0.337
ma.L2         -0.7276      0.154     -4.734      0.000      -1.029      -0.426
ar.S.L12       1.0439      0.014     72.842      0.000       1.016       1.072
ma.S.L12      -0.5550      0.098     -5.663      0.000      -0.747      -0.363
ma.S.L24      -0.1355      0.120     -1.134      0.257      -0.370       0.099
sigma2      1.506e+05   2.03e+04      7.401      0.000    1.11e+05     1.9e+05
===================================================================================
Ljung-Box (L1) (Q):                   0.04   Jarque-Bera (JB):                11.72
Prob(Q):                              0.84   Prob(JB):                         0.00
Heteroskedasticity (H):               1.47   Skew:                             0.36
Prob(H) (two-sided):                  0.26   Kurtosis:                         4.48
===================================================================================
```

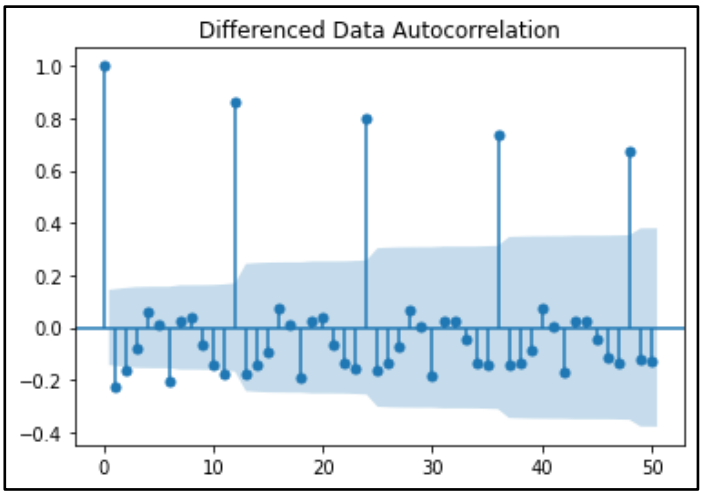
Now, we predict on the test set-

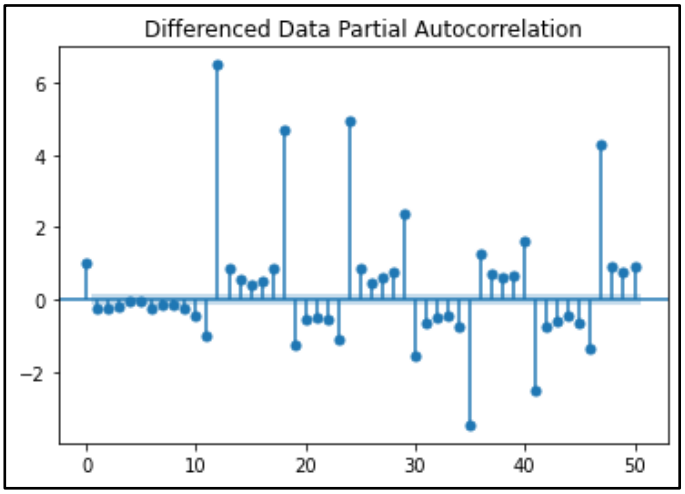
y	mean	mean_se	mean_ci_lower	mean_ci_upper
0	1327.416698	388.338122	566.287965	2088.545430
1	1315.162146	402.001930	527.252842	2103.071450
2	1621.635228	401.995527	833.738474	2409.531983
3	1598.910009	407.232465	800.749045	2397.070973
4	1392.734434	407.962339	593.142942	2192.325926

The accuracy is given by RMSE, **RMSE= 528.566**

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.

- **ARIMA based on ACF and PACF cut-off points**





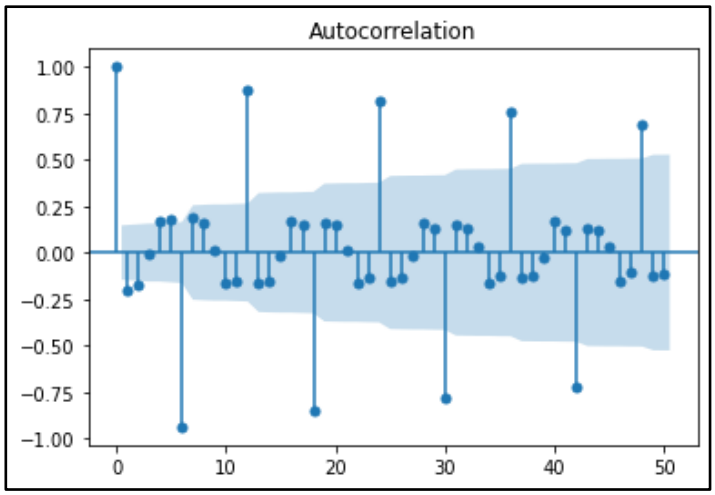
The Moving Average i.e q comes from lag before ACF cut-off i.e 0 and Auto-Regressive parameter i.e p is also 2 in our case.

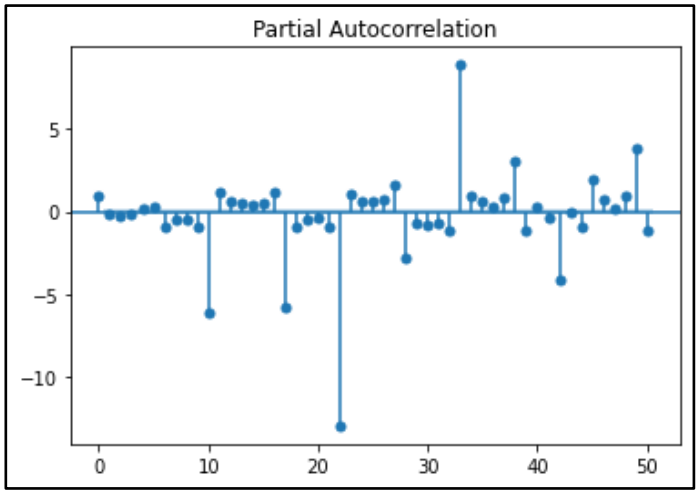
We build model on (2,1,0)

```
                             ARIMA Model Results
==============================================================================
Dep. Variable:            D.Sparkling   No. Observations:                  131
Model:                 ARIMA(2, 1, 0)   Log Likelihood               -1127.018
Method:                       css-mle   S.D. of innovations           1317.723
Date:                Sat, 24 Apr 2021   AIC                           2262.036
Time:                        17:11:49   BIC                           2273.536
Sample:                    02-29-1980   HQIC                          2266.709
                         - 12-31-1990
==========================================================================================
                             coef    std err          z      P>|z|      [0.025      0.975]
------------------------------------------------------------------------------------------
const                   27.0918     80.017      0.339      0.735    -129.738     183.922
ar.L1.D.Sparkling       -0.1929      0.085     -2.271      0.023      -0.360      -0.026
ar.L2.D.Sparkling       -0.2514      0.085     -2.966      0.003      -0.418      -0.085
                                    Roots
=============================================================================
                  Real          Imaginary           Modulus         Frequency
-----------------------------------------------------------------------------
AR.1           -0.3838           -1.9572j            1.9945           -0.2808
AR.2           -0.3838           +1.9572j            1.9945            0.2808
-----------------------------------------------------------------------------
```

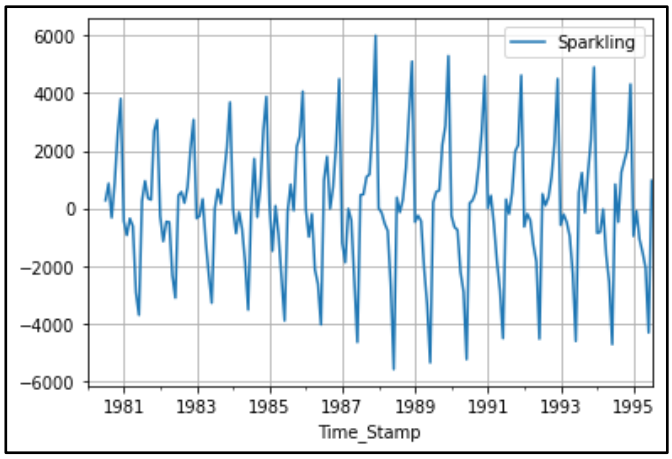
The accuracy is given by RMSE, **RMSE=3908.035**

- **SARIMA based on ACF and PACF cut-off points**
  - **6-month seasonality-**



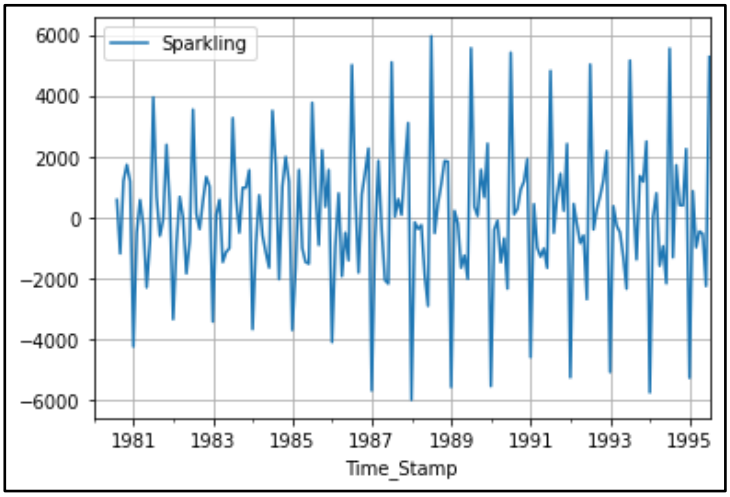


Now we will make a plot for data set considering seasonal difference of 6. Graph is represented as below-

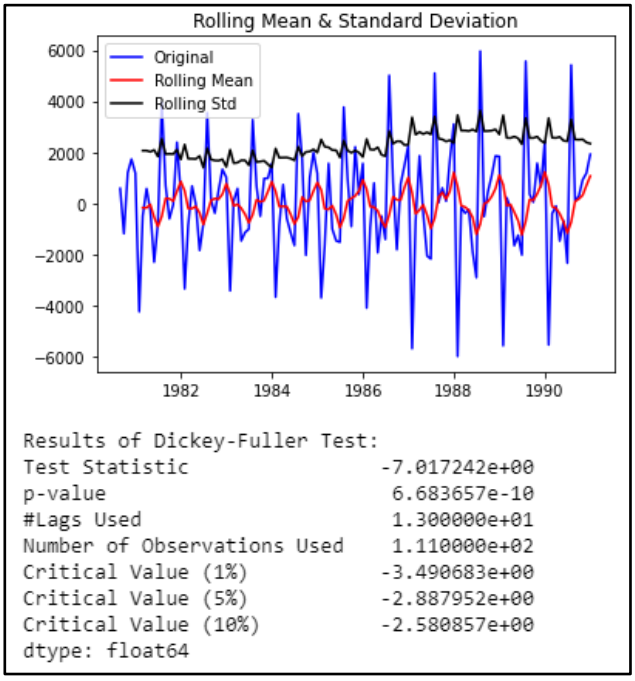


The lag we have used in our model is 50.

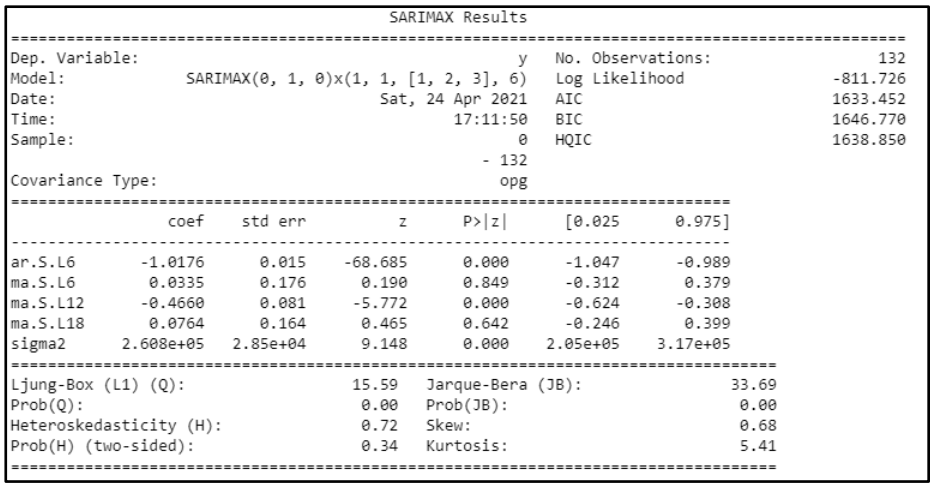
We see that there might be a slight trend which can be noticed in the data. So we take a differencing of first order on the seasonally differenced series. Graph is represented as below-



We check the stationarity of data-



The data is stationary as p-value is less than 0.05 , We take p,d,q as 0,1,0 and build the model.



SARIMAX Results

Dep. Variable:	y	No. Observations:	132
Model:	SARIMAX(0, 1, 0)x(1, 1, [1, 2, 3], 6)	Log Likelihood	-811.726
Date:	Sat, 24 Apr 2021	AIC	1633.452
Time:	17:11:50	BIC	1646.770
Sample:	0	HQIC	1638.850
	- 132		
Covariance Type:	opg		

	coef	std err	z	P>|z|	[0.025	0.975]
ar.S.L6	-1.0176	0.015	-68.685	0.000	-1.047	-0.989
ma.S.L6	0.0335	0.176	0.190	0.849	-0.312	0.379
ma.S.L12	-0.4660	0.081	-5.772	0.000	-0.624	-0.308
ma.S.L18	0.0764	0.164	0.465	0.642	-0.246	0.399
sigma2	2.608e+05	2.85e+04	9.148	0.000	2.05e+05	3.17e+05

Ljung-Box (L1) (Q):	15.59	Jarque-Bera (JB):	33.69
Prob(Q):	0.00	Prob(JB):	0.00
Heteroskedasticity (H):	0.72	Skew:	0.68
Prob(H) (two-sided):	0.34	Kurtosis:	5.41

We compute the mean and confidence intervals-

y	mean	mean_se	mean_ci_lower	mean_ci_upper
0	907.314903	510.711730	-93.661694	1908.291499
1	529.980659	722.254743	-885.612624	1945.573943
2	1125.426914	884.577501	-608.313131	2859.166958
3	933.719009	1021.421949	-1068.231224	2935.669243
4	743.534275	1141.984344	-1494.713910	2981.782459

The accuracy is given by **RMSE=1914.883**

So, all the ARIMA and SARIMA models i.e based on AIC values and ACF/PACF cut-off points with its accuracy are given as –

	RMSE
ARIMA_AIC(0,1,2)	15.640546
SARIMA_AIC(1,1,2)(2,0,2,6)	26.209519
SARIMA_AIC(0,1,2)(2,0,2,12)	26.992038
ARIMA_ACF/PACF(1,1,1)	15.756589
SARIMA_ACF/PACF(2,1,2)(1,1,3,6)	20.350502
SARIMA_ACF/PACF(3,1,2)(1,1,3,12)	20.470001

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

	RMSE
LinearRegression	1389.135175
NaiveModel	3864.279352
SimpleAverageModel	1275.081804
2pointTrailingMovingAverage	813.400684
5pointTrailingMovingAverage	1234.045344
8pointTrailingMovingAverage	1342.567772
SimpleExponentialSmoothing @Alpha=0.995	1316.035487
SimpleExponentialSmoothing @Alpha=0.1	1375.393398
DoubleExponentialSmoothing @Alpha=0.1,Beta=0.1	1778.564670
TripleExponentialSmoothing @Alpha=0.111,Beta=0.061,Gamma=0.395	469.767970
TripleExponentialSmoothing @Alpha=0.4,Beta=0.1,Gamma=0.2	336.715250
ARIMA_AIC(2,1,2)	1374.484105
SARIMA_AIC(1,1,2)(2,0,2,6)	626.931138
SARIMA_AIC(1,1,2)(1,0,2,12)	528.566817
ARIMA_ACF/PACF(2,1,0)	3908.035724
SARIMA_ACF/PACF(0,1,0)(1,1,3,6)	1914.883365

When we sort the values by RMSE scores,

	RMSE
TripleExponentialSmoothing @Alpha=0.4,Beta=0.1,Gamma=0.2	336.715250
TripleExponentialSmoothing @Alpha=0.111,Beta=0.061,Gamma=0.395	469.767970
SARIMA_AIC(1,1,2)(1,0,2,12)	528.566817
SARIMA_AIC(1,1,2)(2,0,2,6)	626.931138
2pointTrailingMovingAverage	813.400684
5pointTrailingMovingAverage	1234.045344
SimpleAverageModel	1275.081804
SimpleExponentialSmoothing @Alpha=0.995	1316.035487
8pointTrailingMovingAverage	1342.567772
ARIMA_AIC(2,1,2)	1374.484105
SimpleExponentialSmoothing @Alpha=0.1	1375.393398
LinearRegression	1389.135175
DoubleExponentialSmoothing @Alpha=0.1,Beta=0.1	1778.564670
SARIMA_ACF/PACF(0,1,0)(1,1,3,6)	1914.883365
NaiveModel	3864.279352
ARIMA_ACF/PACF(2,1,0)	3908.035724

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.

Considering the best model i.e SARIMA with order (1,1,2) at 12 month seasonality, we build a model to predict the next 12 months.

```
full_data_model = sm.tsa.statespace.SARIMAX(df,
                                order=(1,1,2),
                                seasonal_order=(1, 0, 2, 12),
                                enforce_stationarity=False,
                                enforce_invertibility=False)
results_full_data_model = full_data_model.fit(maxiter=1000)
print(results_full_data_model.summary())
```

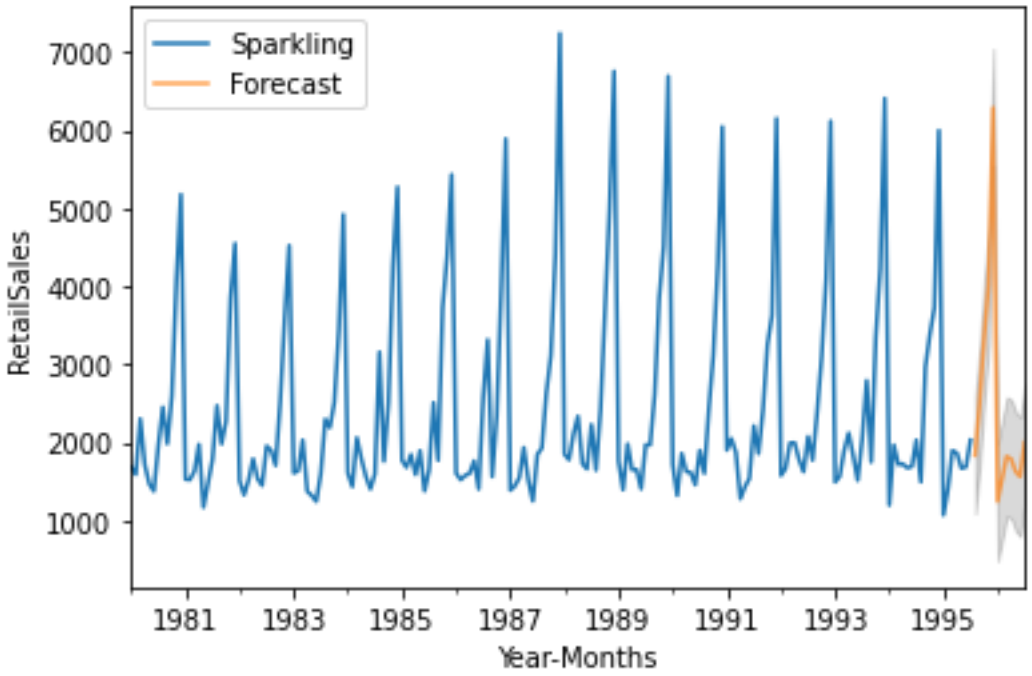
```
                                 SARIMAX Results
==========================================================================================
Dep. Variable:                           Sparkling   No. Observations:                  187
Model:             SARIMAX(1, 1, 2)x(1, 0, 2, 12)   Log Likelihood               -1173.413
Date:                            Sat, 24 Apr 2021   AIC                           2360.827
Time:                                    17:11:52   BIC                           2382.309
Sample:                                01-31-1980   HQIC                          2369.551
                                     - 07-31-1995
Covariance Type:                              opg
==============================================================================
                 coef    std err          z      P>|z|      [0.025      0.975]
------------------------------------------------------------------------------
ar.L1         -0.6609      0.242     -2.734      0.006      -1.135      -0.187
ma.L1         -0.2739      0.200     -1.368      0.171      -0.666       0.118
ma.L2         -0.8112      0.227     -3.577      0.000      -1.256      -0.367
ar.S.L12       1.0157      0.012     84.460      0.000       0.992       1.039
ma.S.L12      -1.3873      0.338     -4.102      0.000      -2.050      -0.724
ma.S.L24      -0.1461      0.146     -1.001      0.317      -0.432       0.140
sigma2      5.948e+04   1.84e+04      3.232      0.001    2.34e+04    9.56e+04
===================================================================================
Ljung-Box (L1) (Q):                   0.00   Jarque-Bera (JB):                27.47
Prob(Q):                              0.96   Prob(JB):                         0.00
Heteroskedasticity (H):               1.03   Skew:                             0.52
Prob(H) (two-sided):                  0.93   Kurtosis:                         4.76
===================================================================================
```

Now , forecasting the values with confidence intervals-

Sparkling	mean	mean_se	mean_ci_lower	mean_ci_upper
1995-08-31	1836.362650	379.707390	1092.149841	2580.575459
1995-09-30	2489.610462	384.471543	1736.060085	3243.160839
1995-10-31	3324.596841	384.577445	2570.838900	4078.354782
1995-11-30	4020.242509	386.335376	3263.039085	4777.445932
1995-12-31	6290.019811	386.389881	5532.709560	7047.330063

RMSE of the full model - **RMSE= 539.98**

The graph showing the predictions for 12 months period from start='1995-08-31',end='1996-7-31' shows that predicted numbers will follow the past trend years . highest sales will be ~6000 units and lowest sales will be ~1200 units over a period of 12 months.



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

The model that we have built is based on a SARIMA model that has repeated seasonality every 6 months, and the p,d,q as 1,1,2 respectively. Based on the model forecasting , we can observe that the sales range between 1200 to 6000 units being sold, which is similar compared to what the company trend, but as we have seen the pattern/ trend decreasing over years, this doesn't look like an anomaly.

What we suggest is that the company should encourage sales in the summer and other seasons by putting some discounts and other offers which encourages more customers to buy mainly during the first mid-year i.e May- August.