2020

Mr.Jaydeep Bhaskar Ashtekar
ashtekarjaydeep@yahoo.in
PGPDSBA:Group 9

[TIME SERIES FORECASTING]

This report gives report of case study based on Sales of Rose and Sparkling. Report includes the number of models and best model is selected for the purpose of the forecasting.

A. Data of Rose

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

Ans:

a.Liabraries are imported:

```
import pandas as pd
import numpy as np
import seaborn as sns
from matplotlib import pyplot as plt
```

b.Data is read by parsing method as follows:

```
df =  pd.read_csv('Rose.csv',  index_col = 0, parse_dates = True, squeeze = True)
df.head()
```

```
YearMonth
1980-01-01    112.0
1980-02-01    118.0
1980-03-01    129.0
1980-04-01     99.0
1980-05-01    116.0
Name: Rose, dtype: float64
```

Data is of time timeseries with Rose sale as float type.

df.shape: `(187, 1)`

c.Found NaN values

df.isna().sum()

2

d.Replacing NaN by interpolating

df=df.interpolate(method='linear')

d. df.describe()

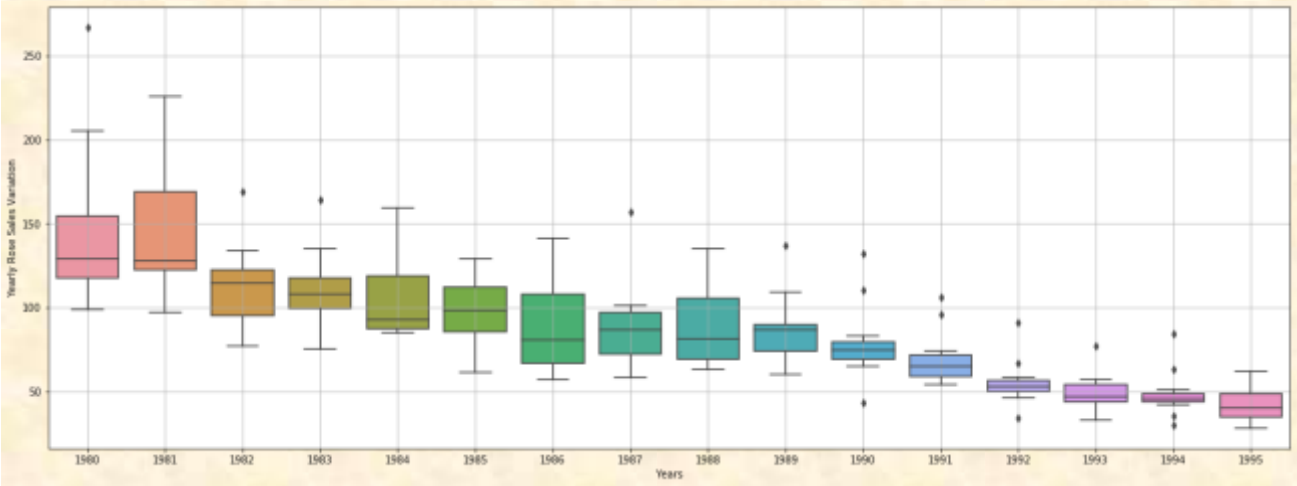
```
count    187.000000
mean      89.914439
std       39.238325
min       28.000000
25%       62.500000
50%       85.000000
75%      111.000000
max      267.000000
```

```
Name: Rose, dtype: float64
```

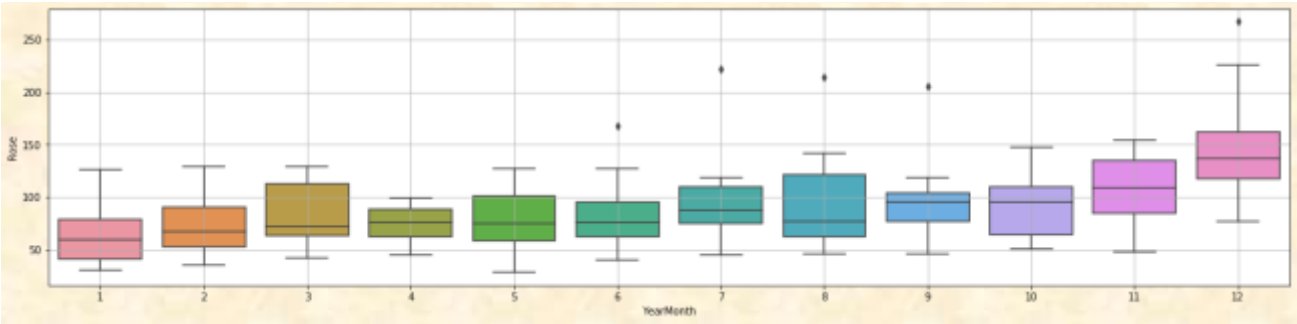
2. Perform appropriate Exploratory Data Analysis to understand the data and also perform decomposition.

Ans:

a.Boxplot:



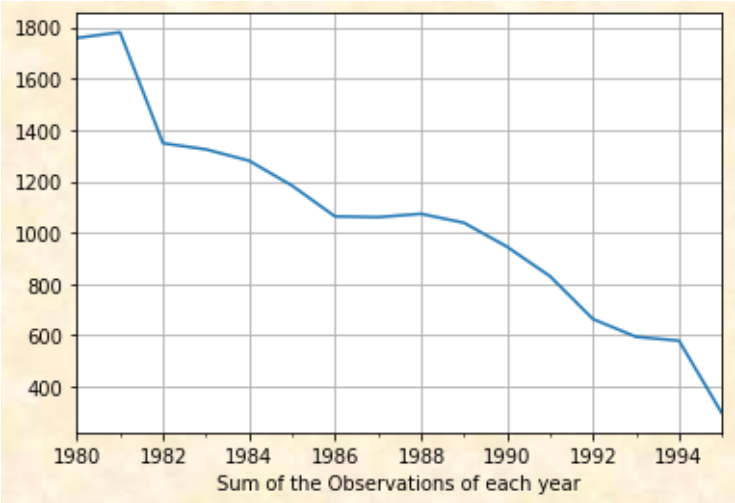
It straight forward shows decreasing trend of sale as time forwards.



Maximum sales is in month of December.

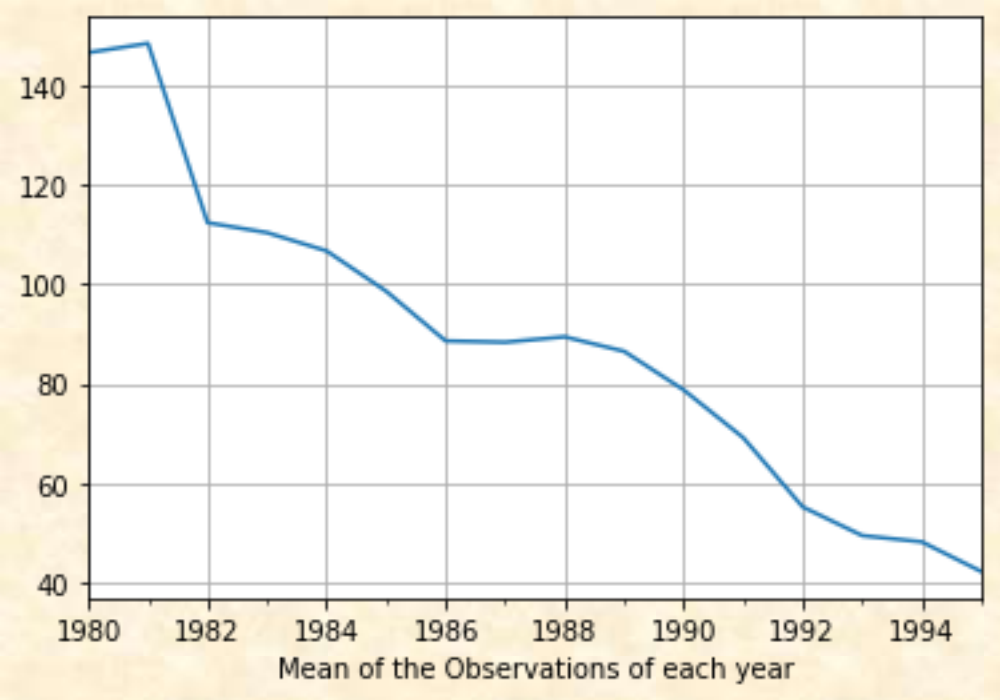
b.sales vs year

Sales sum of each year



Sales goes on decreasing each year.Maximum sale is in 1981.

Mean Sales sum of each year



Mean Sales goes on decreasing each year.Maximum mean sale is in 1981.

c. Time Series:

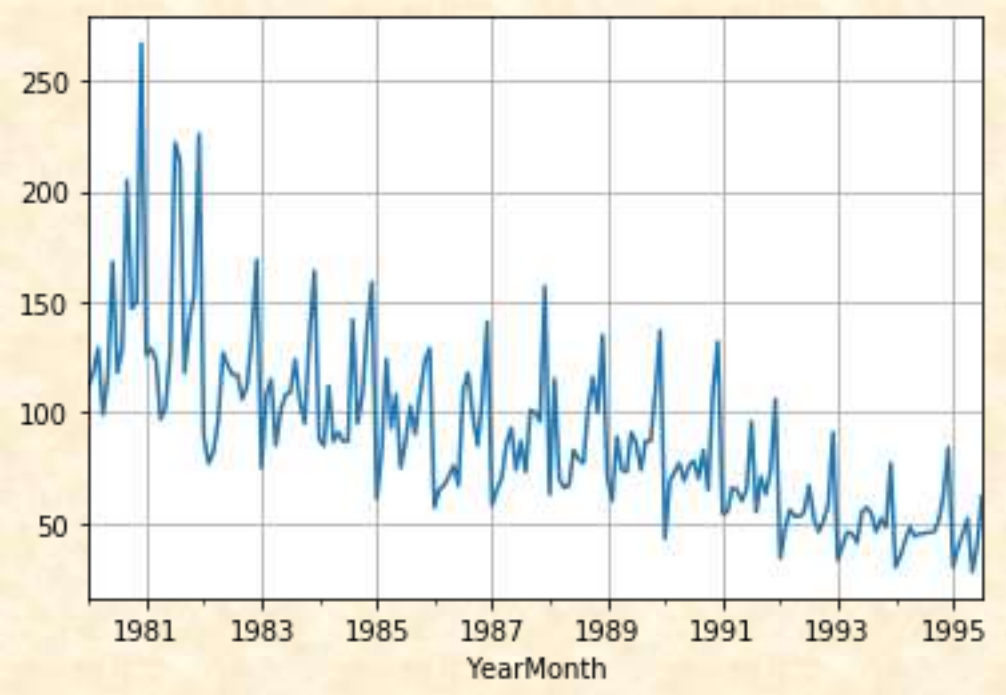
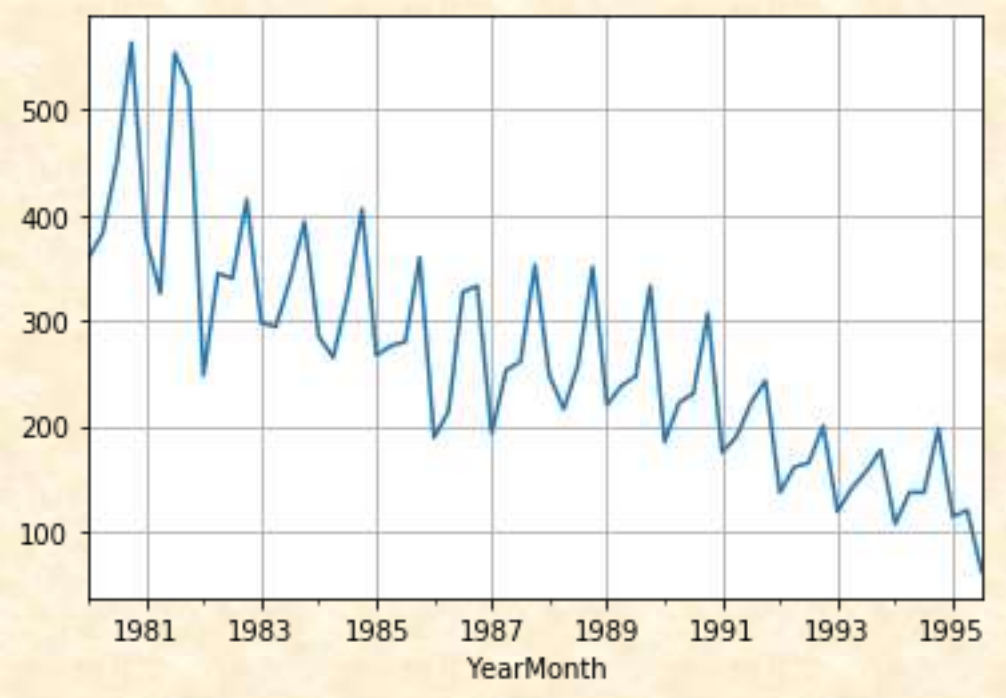
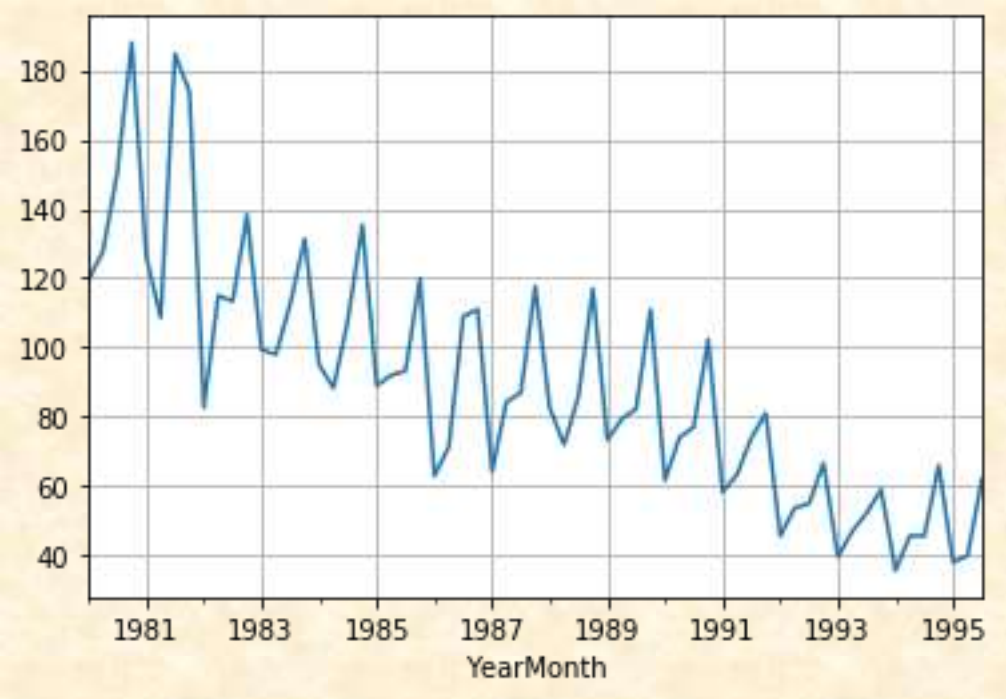


Figure shows the sales decreasing trend and peaks show seasonality in each year.

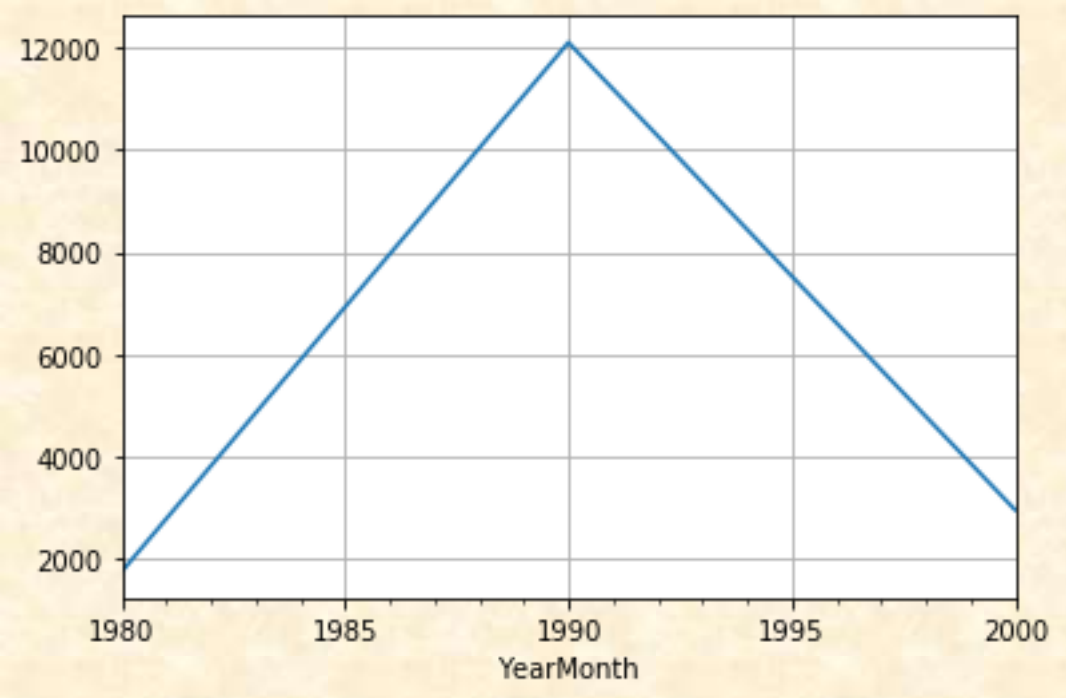
d. **Quarterly_sum:**



Maximum quarterly sales was in 1981 with 500 plus sales.
e.quarterly_mean

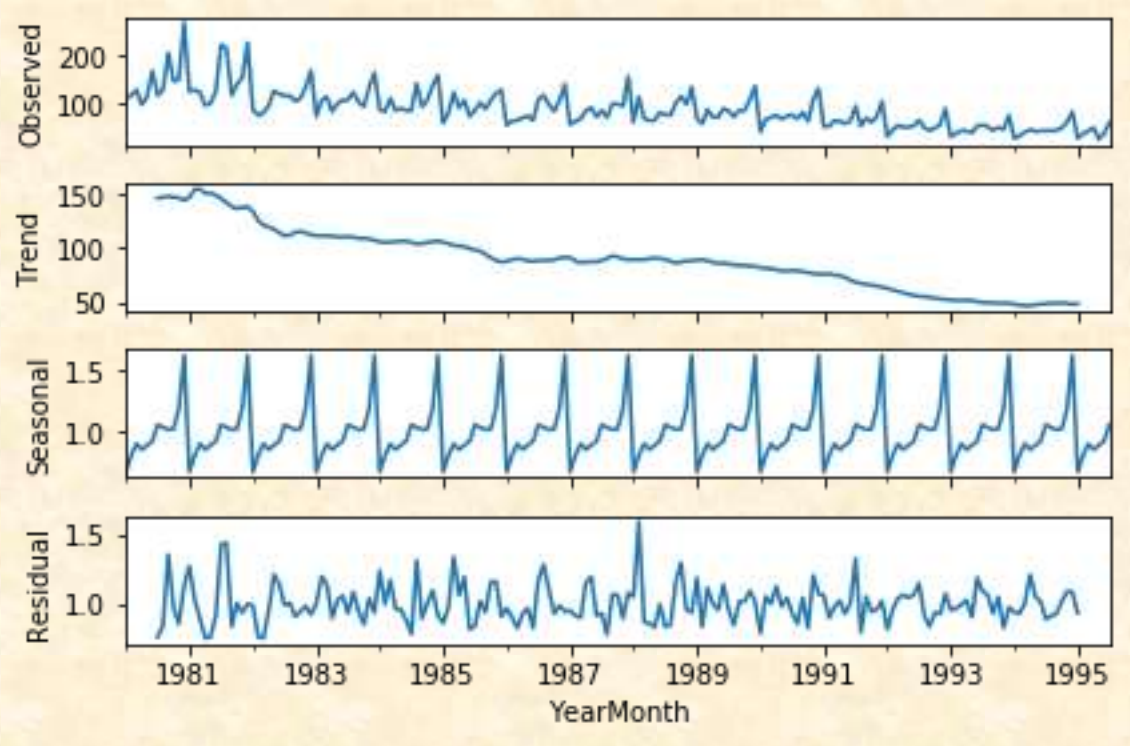


Maximum quarterly mean is in 1981 then it started decreasing.
f.decade_sum



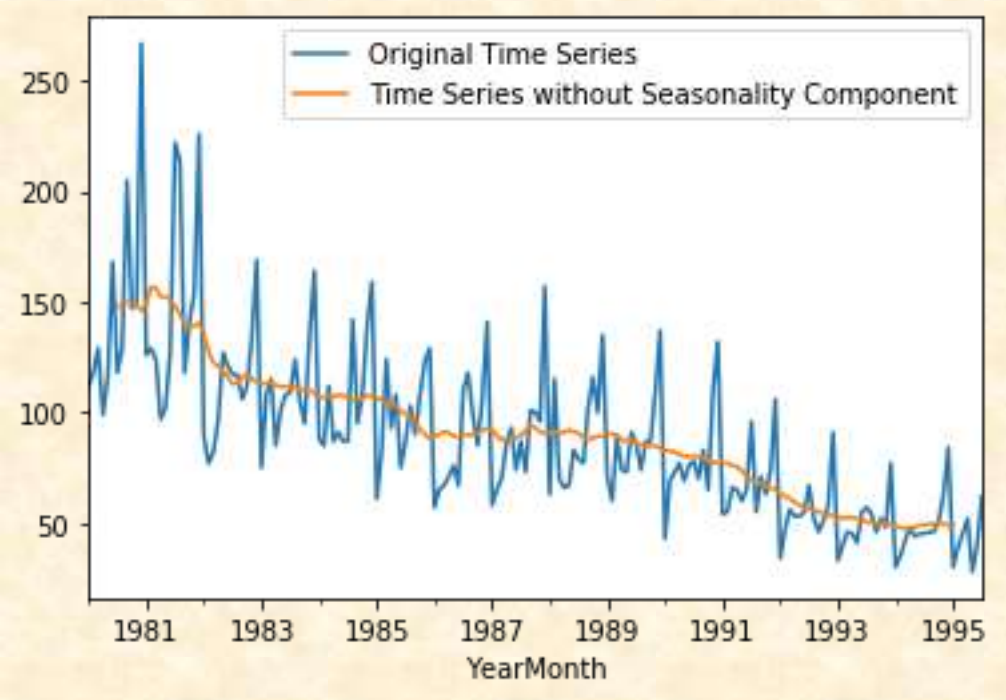
Maximum sales is at 1980-1990 of 12000 plus.Then it decreased.

g.Decomposition of series:



Trend is decreasing.
There is small seasonality maximum of 1.6
Residual is random and has noise.Multiplicative model is used.

h.deseasonal time searies

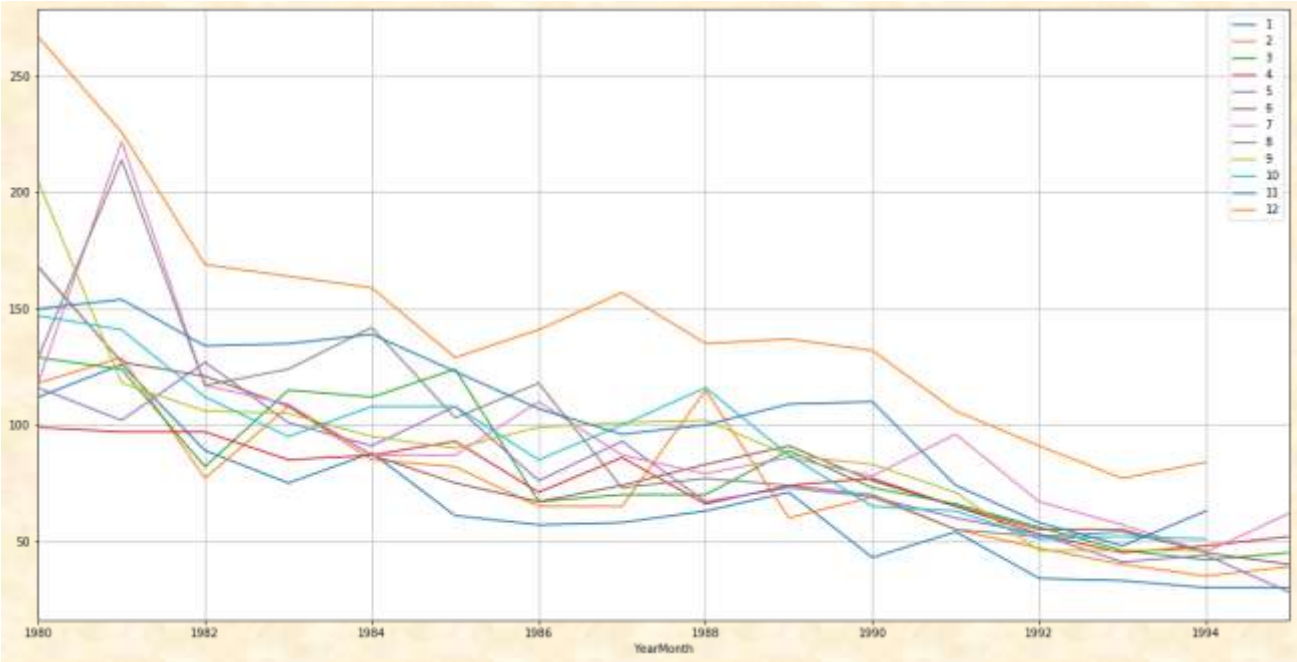


i,Pivot table:

YearMonth	1	2	3	4	5	6	7	8	9	10	11	12
YearMonth												
1980	112.0	118.0	129.0	99.0	116.0	168.0	118.000000	129.000000	205.0	147.0	150.0	267.0
1981	126.0	129.0	124.0	97.0	102.0	127.0	222.000000	214.000000	118.0	141.0	154.0	226.0
1982	89.0	77.0	82.0	97.0	127.0	121.0	117.000000	117.000000	106.0	112.0	134.0	169.0
1983	75.0	108.0	115.0	85.0	101.0	108.0	109.000000	124.000000	105.0	95.0	135.0	164.0
1984	88.0	85.0	112.0	87.0	91.0	87.0	87.000000	142.000000	95.0	108.0	139.0	159.0
1985	61.0	82.0	124.0	93.0	108.0	75.0	87.000000	103.000000	90.0	108.0	123.0	129.0
1986	57.0	65.0	67.0	71.0	76.0	67.0	110.000000	118.000000	99.0	85.0	107.0	141.0
1987	58.0	65.0	70.0	86.0	93.0	74.0	87.000000	73.000000	101.0	100.0	96.0	157.0
1988	63.0	115.0	70.0	66.0	67.0	83.0	79.000000	77.000000	102.0	116.0	100.0	135.0
1989	71.0	60.0	89.0	74.0	73.0	91.0	86.000000	74.000000	87.0	87.0	109.0	137.0
1990	43.0	69.0	73.0	77.0	69.0	76.0	78.000000	70.000000	83.0	65.0	110.0	132.0
1991	54.0	55.0	66.0	65.0	60.0	65.0	96.000000	55.000000	71.0	63.0	74.0	106.0
1992	34.0	47.0	56.0	53.0	53.0	55.0	67.000000	52.000000	46.0	51.0	58.0	91.0
1993	33.0	40.0	46.0	45.0	41.0	55.0	57.000000	54.000000	46.0	52.0	48.0	77.0
1994	30.0	35.0	42.0	48.0	44.0	45.0	45.333333	45.666667	46.0	51.0	63.0	84.0
1995	30.0	39.0	45.0	52.0	28.0	40.0	62.000000	NaN	NaN	NaN	NaN	NaN

j. monthly_sales_across_years

In all years monthly sale of December is highest which decreases with years.



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.tail(5))

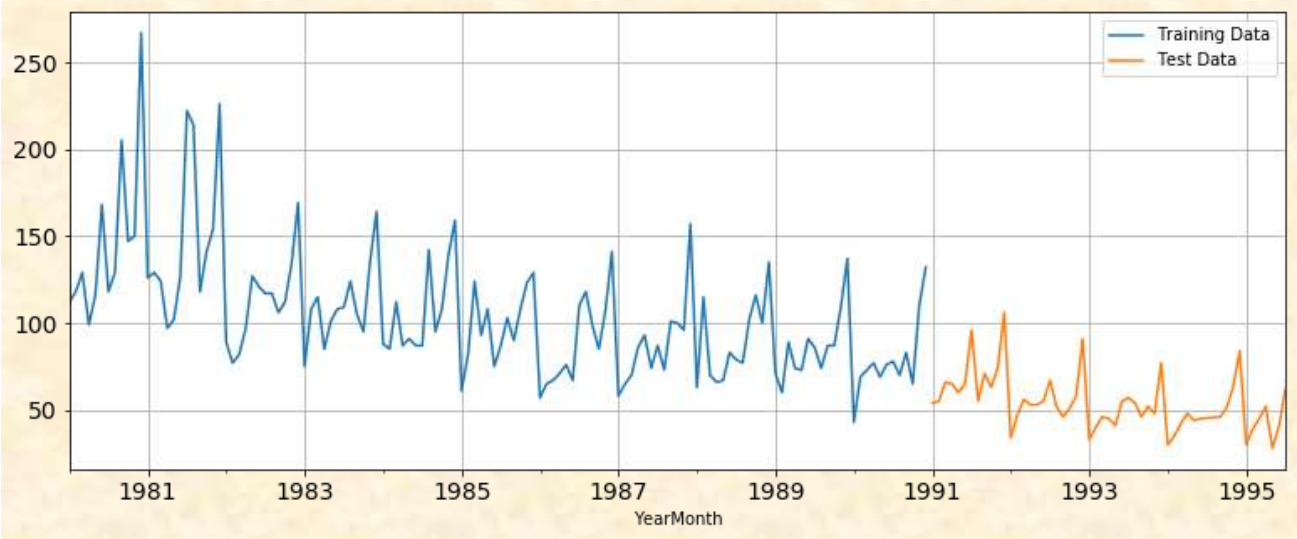
print(test.head(5))

```
             Rose
YearMonth
1990-08-01   70.0
1990-09-01   83.0
1990-10-01   65.0
1990-11-01  110.0
1990-12-01  132.0
            Rose
YearMonth
1991-01-01  54.0
1991-02-01  55.0
1991-03-01  66.0
1991-04-01  65.0
1991-05-01  60.0
```

print(train.shape): `(132, 1)`
print(test.shape): `(55, 1)`

```
First few rows of Training Data
              Rose
YearMonth
1980-01-01  112.0
1980-02-01  118.0
1980-03-01  129.0
1980-04-01   99.0
1980-05-01  116.0

Last few rows of Training Data
              Rose
YearMonth
1990-08-01   70.0
1990-09-01   83.0
1990-10-01   65.0
1990-11-01  110.0
1990-12-01  132.0

First few rows of Test Data
             Rose
YearMonth
1991-01-01  54.0
1991-02-01  55.0
1991-03-01  66.0
1991-04-01  65.0
1991-05-01  60.0

Last few rows of Test Data
             Rose
YearMonth
1995-03-01  45.0
1995-04-01  52.0
1995-05-01  28.0
1995-06-01  40.0
1995-07-01  62.0
```

Representation of train and test data:



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. - Please do try to build as many models as possible and as many iterations of models as possible with different parameters.

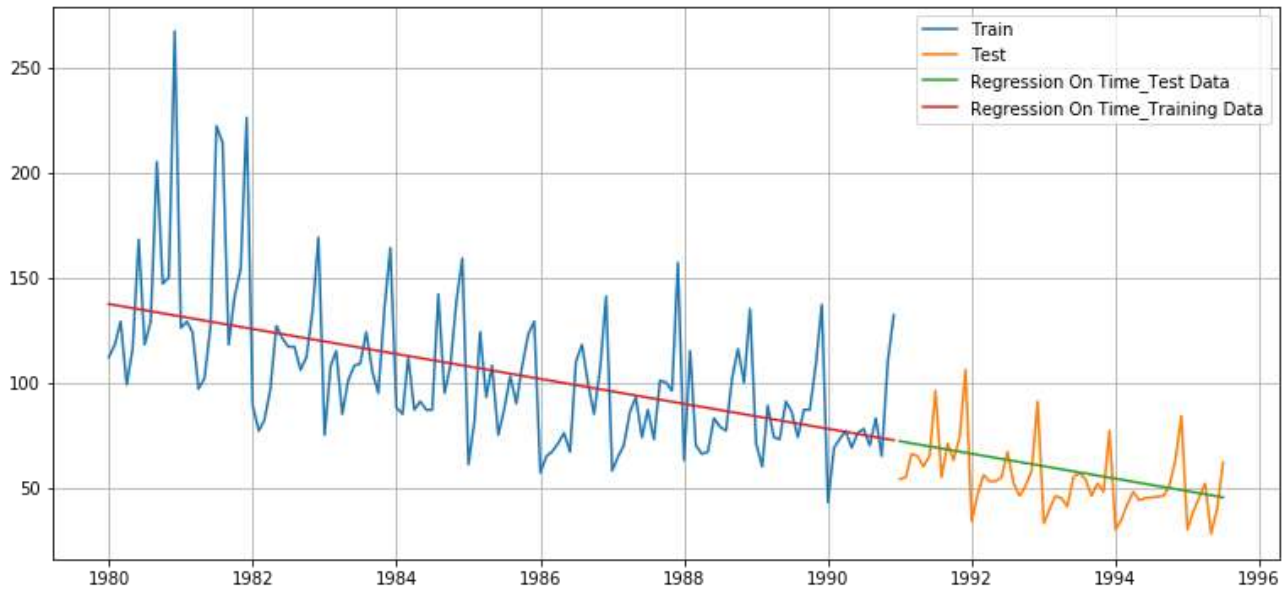
Ans:

Model 1: Linear Regression

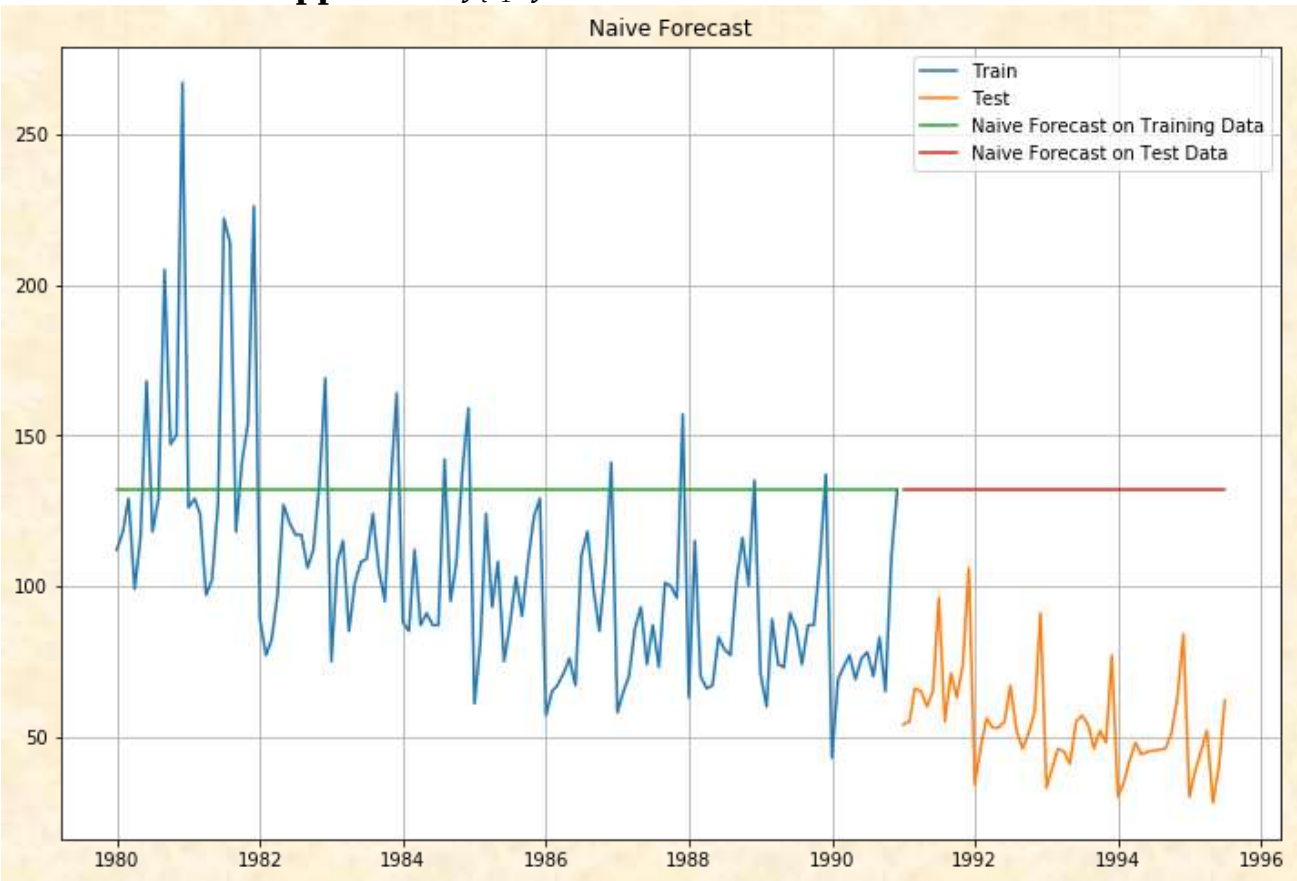
In model linear regression is done by importing regression liabrary.

Following figure shows the decreasing slope of line.

```
First few rows of Training Data
              Rose  time
YearMonth
1980-01-01  112.0     1
1980-02-01  118.0     2
1980-03-01  129.0     3
1980-04-01   99.0     4
1980-05-01  116.0     5

Last few rows of Training Data
              Rose  time
YearMonth
1990-08-01   70.0   128
1990-09-01   83.0   129
1990-10-01   65.0   130
1990-11-01  110.0   131
1990-12-01  132.0   132

First few rows of Test Data
             Rose  time
YearMonth
1991-01-01  54.0   133
1991-02-01  55.0   134
1991-03-01  66.0   135
1991-04-01  65.0   136
1991-05-01  60.0   137

Last few rows of Test Data
             Rose  time
YearMonth
1995-03-01  45.0   183
1995-04-01  52.0   184
1995-05-01  28.0   185
1995-06-01  40.0   186
1995-07-01  62.0   187


For RegressionOnTime forecast on the Training Data,  RMSE is 943.604 MAPE is 21.22
For RegressionOnTime forecast on the Test Data,  RMSE is 233.141 MAPE is 22.82
```



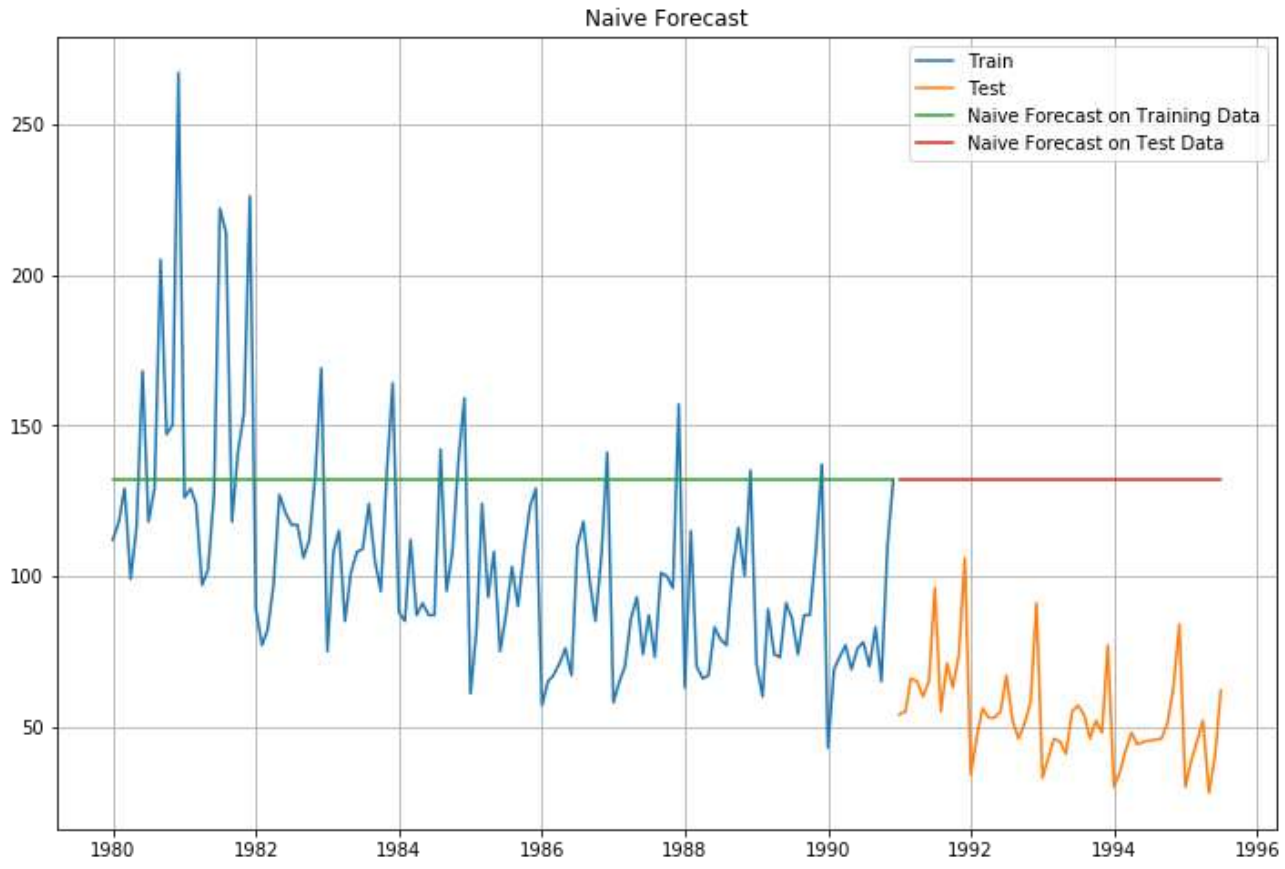
Model 2: Naive Approach: $y_{t+1}=y_t$




```
NaiveModel_train['naive'] = np.asarray(train['Rose'])[len(np.asarray(train['Rose']))-1]
NaiveModel_train['naive'].head()
YearMonth
1980-01-01    132.0
1980-02-01    132.0
1980-03-01    132.0
```

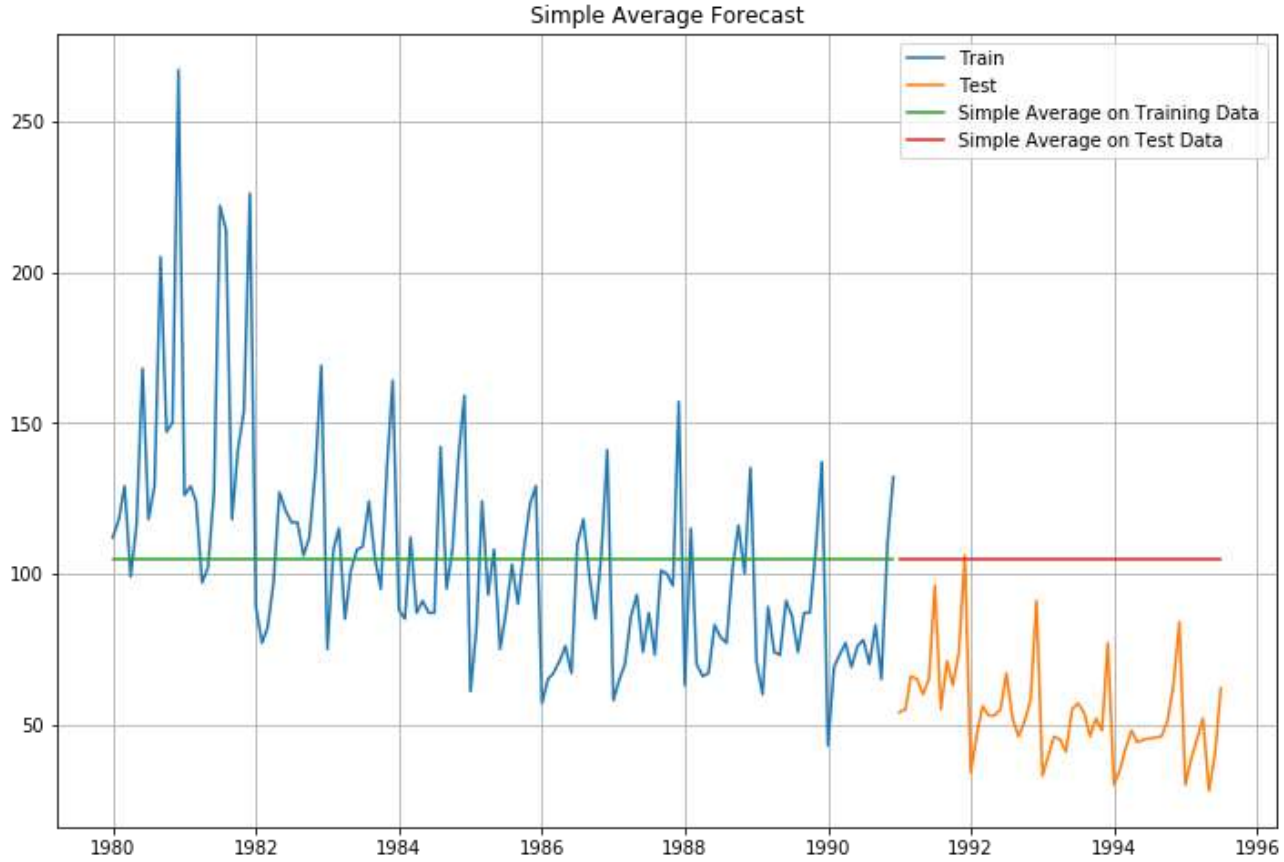
```
1980-04-01    132.0
1980-05-01    132.0
Name: naive, dtype: float64
```

NaiveModel_test['naive'] = np.asarray(train['Rose'])[len(np.asarray(train['Rose']))-1]
NaiveModel_test['naive'].head()




```
For Naive Model forecast on the Training Data,  RMSE is 2030.742 MAPE is 36.38
For RegressionOnTime forecast on the Test Data,  RMSE is 6355.083 MAPE is 145.10
```

Model 3: Simple Average



SimpleAverage_train['mean_forecast'] = train['Rose'].mean()
SimpleAverage_train.head()

	Rose	mean_forecast
YearMonth		
01/01/1980	112	104.93939
01/02/1980	118	104.93939
01/03/1980	129	104.93939
01/04/1980	99	104.93939
01/05/1980	116	104.93939

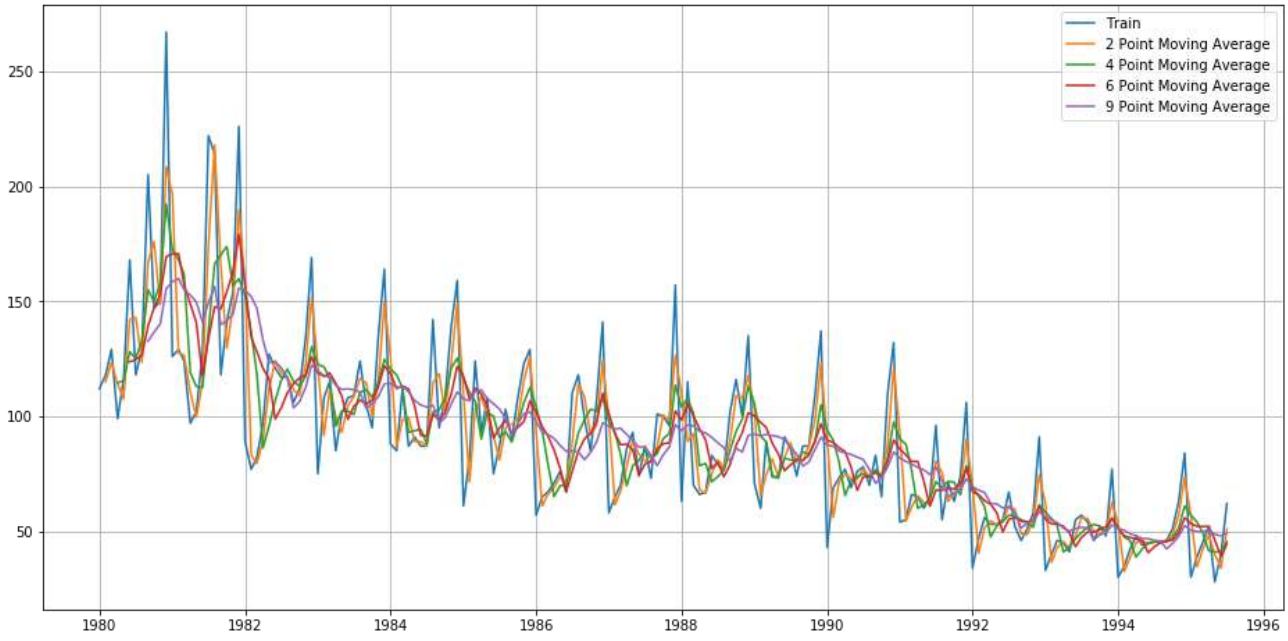
SimpleAverage_test.head()

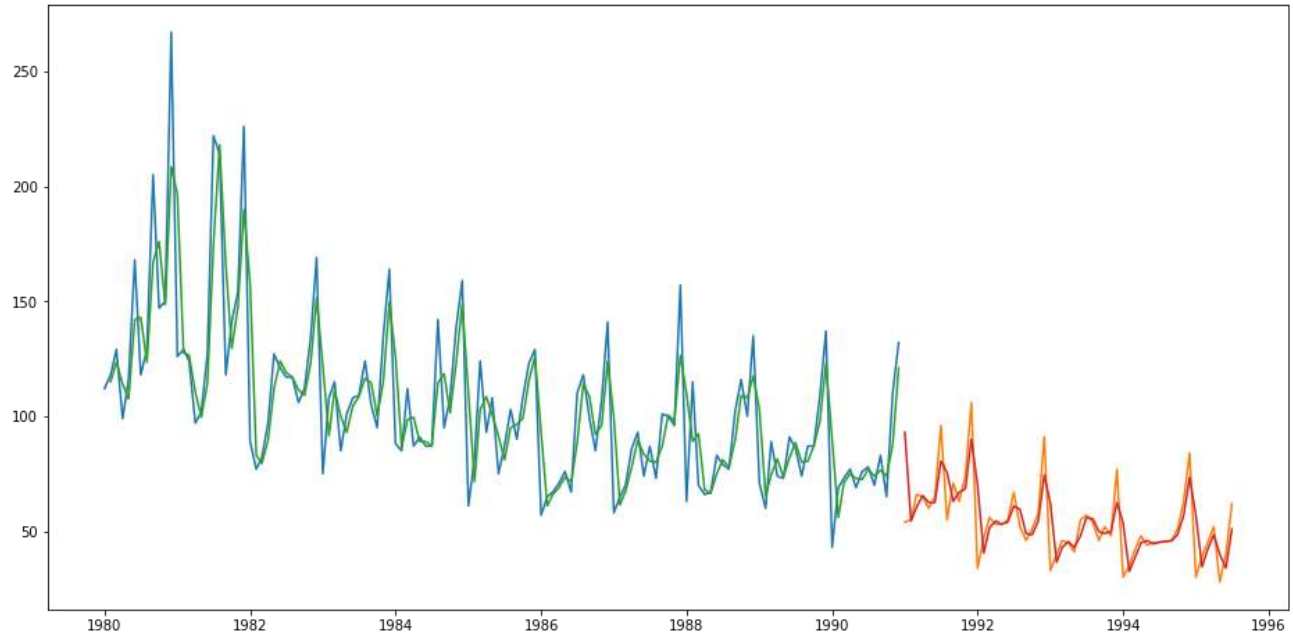
	Rose	mean_forecast
YearMonth		
01/01/1991	54	104.93939
01/02/1991	55	104.93939
01/03/1991	66	104.93939
01/04/1991	65	104.93939
01/05/1991	60	104.93939

```
For Simple Average Model forecast on the Training Data,  RMSE is 1298.466 MAPE is 25.39
For Simple Average forecast on the Test Data,  RMSE is 2858.033 MAPE is 94.93
```

Model 4: Moving Average(MA)

	Rose	Trailing_1	Trailing_2	Trailing_4	Trailing_6	Trailing_9
YearMonth						
01/01/1980	112	112	NaN	NaN	NaN	NaN
01/02/1980	118	118	115	NaN	NaN	NaN
01/03/1980	129	129	123.5	NaN	NaN	NaN
01/04/1980	99	99	114	114.5	NaN	NaN
01/05/1980	116	116	107.5	115.5	NaN	NaN
01/06/1980	168	168	142	128	123.6667	NaN
01/07/1980	118	118	143	125.25	124.6667	NaN
01/08/1980	129	129	123.5	132.75	126.5	NaN
01/09/1980	205	205	167	155	139.1667	132.6667
01/10/1980	147	147	176	149.75	147.1667	136.5556





Out of all two point MA traces the series properly.Others show lag.

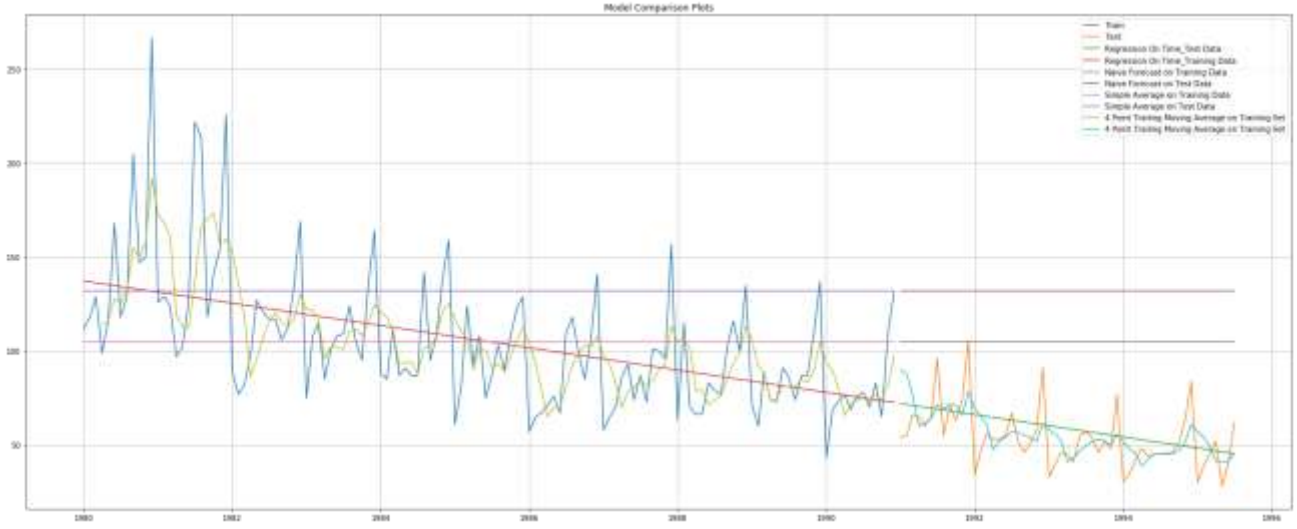
```
For 2 point Moving Average Model forecast on the Testing Data,  RMSE is 132.924 MAPE is
13.54
For 4 point Moving Average Model forecast on the Testing Data,  RMSE is 208.843 MAPE is
19.49
For 6 point Moving Average Model forecast on the Testing Data,  RMSE is 212.178 MAPE is
20.82
For 9 point Moving Average Model forecast on the Testing Data,  RMSE is 216.903 MAPE is
21.01
```

Performance of all above models on test data:

	Test RMSE	Test MAPE
RegressionOnTime	233.140993	22.82
NaiveModel	6355.08283	145.1
SimpleAverageModel	2858.03251	94.93
2pointTrailingMovingAverage	132.924242	13.54
4pointTrailingMovingAverage	208.843056	19.49
6pointTrailingMovingAverage	212.17789	20.82
9pointTrailingMovingAverage	216.90308	21.01

Out of all above models 2 point MA gives best result in terms of RMSE as well as MAPE.

Before building exponential model let's see previous models performance.



Only moving averages are following the series but other models are linear.So lets explore exponential models.

Model 5: Simple Exponential Smoothing

For SES following optimized parameters using
model_SES_autofit.params

```
{'smoothing_level': 0.09874980344929209,
 'smoothing_slope': nan,
 'smoothing_seasonal': nan,
 'damping_slope': nan,
 'initial_level': 134.3872817416308,
 'initial_slope': nan,
 'initial_seasons': array([], dtype=float64),
 'use_boxcox': False,
 'lamda': None,
 'remove_bias': False}
```

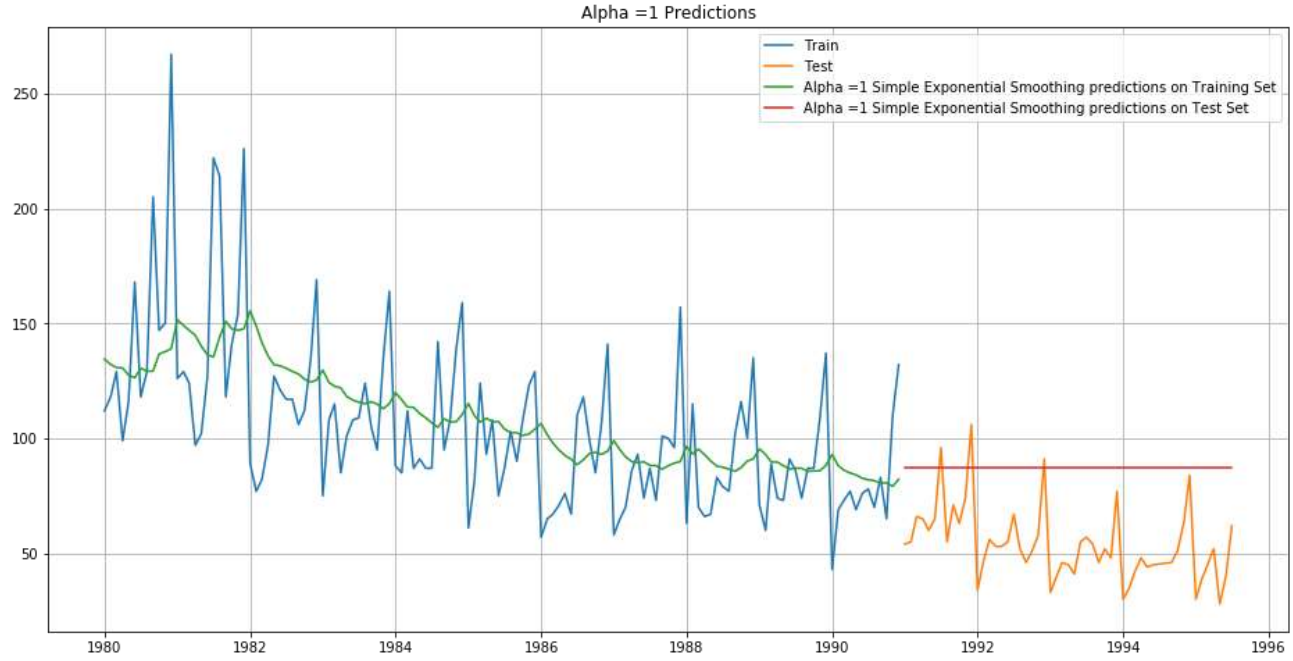
In [88]:

After fitting above model train values are predicted as
SES_train['predict'] = model_SES_autofit.fittedvalues
SES_train.head()

	Rose	predict
YearMonth		
01/01/1980	112	134.3873
01/02/1980	118	132.1765
01/03/1980	129	130.7766
01/04/1980	99	130.6012
01/05/1980	116	127.4806

```
SES_test['predict'] = model_SES_autofit.forecast(steps=55)
SES_test
```

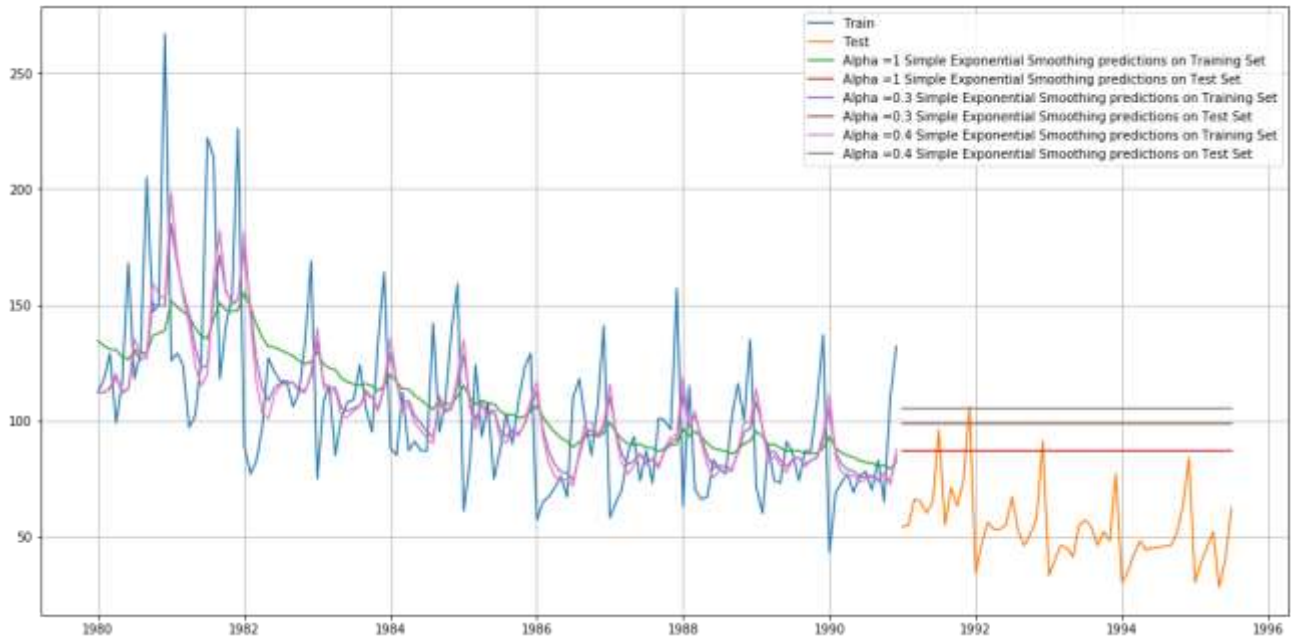
	Rose	predict
YearMonth		
01/01/1991	54	87.104996
01/02/1991	55	87.104996
01/03/1991	66	87.104996
01/04/1991	65	87.104996
01/05/1991	60	87.104996
01/06/1991	65	87.104996




```
For Alpha =1 Simple Exponential Smoothing Model forecast on the Training Data,  RMSE is
992.305 MAPE is 22.73
For Alpha =1 Simple Exponential Smoothing Model forecast on the Training Data,  RMSE is
1353.963 MAPE is 63.88
```

Setting different alpha values. Remember, the higher the alpha value more weightage is given to the more recent observation. That means, what happened recently will happen again. We will run a loop with different alpha values to understand which particular value works best for alpha on the test set. Setting values from 0.3 to 1 with interval of 0.1.

Alpha Values	Train RMSE	Train MAPE	Test RMSE	Test MAPE
0.3	1054.312	22.31	2256.708	83.71
0.4	1091.32	22.5	2890.934	95.5
0.5	1134.534	22.68	3557.143	106.81
0.6	1186.194	22.88	4221.268	117.04
0.7	1247.733	23.08	4857.834	126.07
0.8	1320.203	23.41	5442.602	133.83
0.9	1404.959	23.93	5950.468	140.22



for alpha=0.3 gives least RMSE and MAPE values.

Model 6: Double Exponential Smoothing (Holt's Model)

Two parameters α and β are estimated in this model. Level and Trend are accounted for in this model.

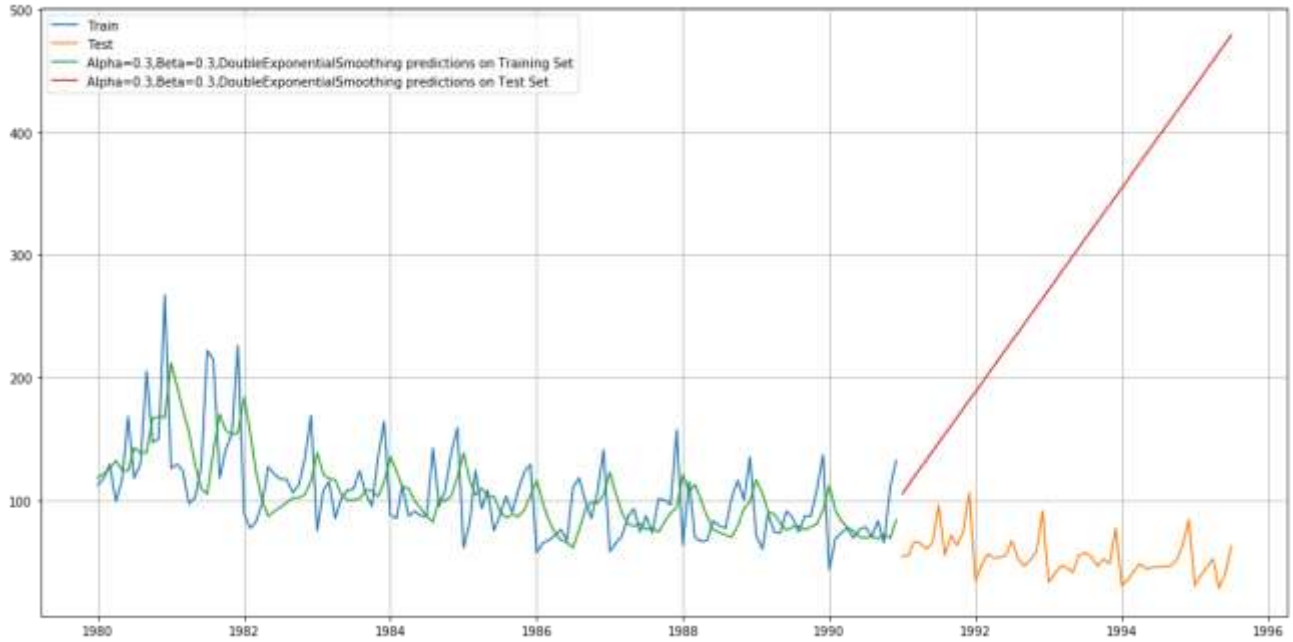
resultsDf_7.sort_values(by=['Test MAPE']).head() gives the values of first five values of MAPE

	Alpha Values	Beta Values	Train RMSE	Train MAPE	Test RMSE	Test MAPE
0	0.3	0.3	1292.042	25.48	70526.15	442.5
8	0.4	0.3	1350.498	25.55	115128.9	565.42
1	0.3	0.4	1398.254	26.53	128702.2	593.91
16	0.5	0.3	1401.253	25.46	155450.9	657.17
9	0.4	0.4	1444.351	26.4	195333.6	732.26

resultsDf_7.sort_values(by=['Test RMSE']).head()

	Alpha Values	Beta Values	Train RMSE	Train MAPE	Test RMSE	Test MAPE
0	0.3	0.3	1292.042	25.48	70526.15	442.5
8	0.4	0.3	1350.498	25.55	115128.9	565.42
1	0.3	0.4	1398.254	26.53	128702.2	593.91
16	0.5	0.3	1401.253	25.46	155450.9	657.17
24	0.6	0.3	1470.645	25.6	192981	732.29

So combination of alpha=0.3 and beta=0.3 gives least values of RMSE and MAPE for test data



So it shows that alpha=0.3 and beta=0.3 gives increasing values of test data so has large error.

Model 7: Triple Exponential Smoothing (Holt - Winter's Model)

Three parameters α , β and γ are estimated in this model. Level, Trend and Seasonality are accounted for in this model.

Considering multiplicative trend and additive seasonality:
model_TES =
ExponentialSmoothing(TES_train['Rose'].astype('double'),trend='additive',seasonal='multiplicative')

best α , β and γ are

```
{'smoothing_level': 0.10609623406683481,
 'smoothing_slope': 0.04843856869902737,
 'smoothing_seasonal': 0.0,
 'damping_slope': nan,
 'initial_level': 76.65565231019384,
 'initial_slope': 0.0,
 'initial_seasons': array([1.47550289, 1.65927166, 1.80572656, 1.58888846, 1.77822723,
```

```
       1.92604397, 2.1164949 , 2.25135226, 2.1169061 , 2.08112849,
       2.40927316, 3.30448171]),
 'use_boxcox': False,
 'lamda': None,
 'remove_bias': False}
```

fitting above values to train and test data gives following results:

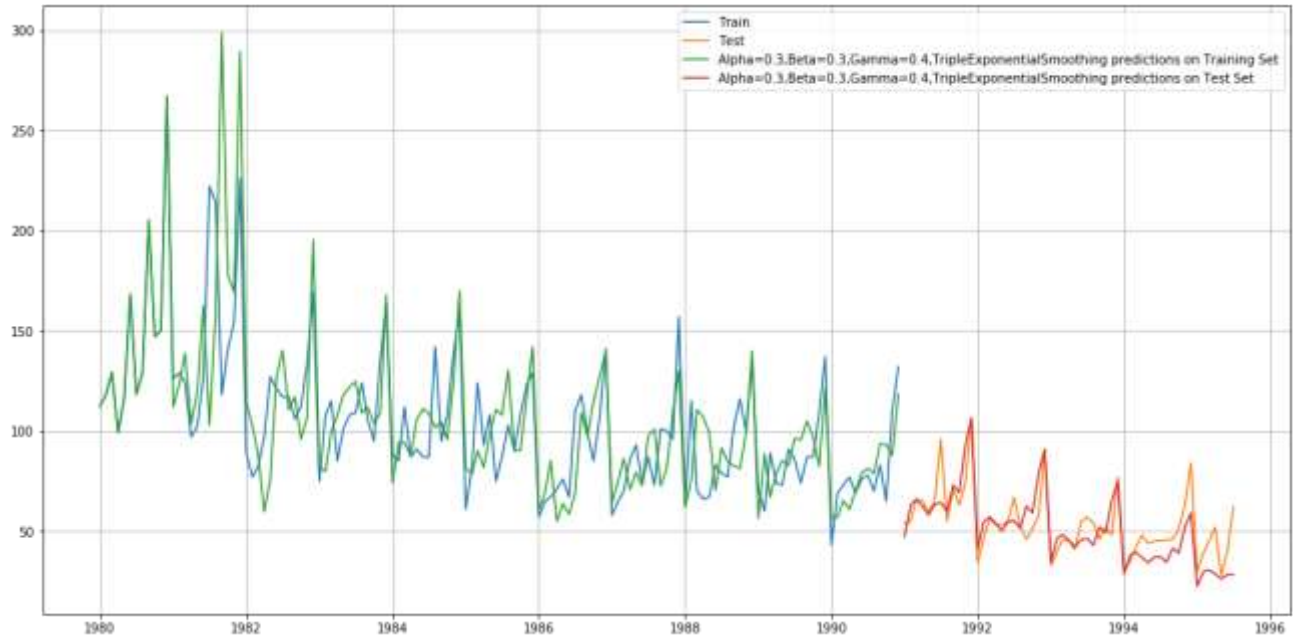
	Rose	auto_predict
YearMonth		
01/01/1980	112	113.1056
01/02/1980	118	127.0542
01/03/1980	129	137.1656
01/04/1980	99	119.8444
01/05/1980	116	131.4326

```
For Alpha= 0.1061,Beta=0.0484,Gamma=0.0, Triple Exponential Smoothing Model forecast on
the Training Data,  RMSE is 345.174 MAPE is 13.21
```

TES_test['auto_predict'] = model_TES_autofit.forecast(steps=55)
TES_test.head()

	Rose	auto_predict
YearMonth		
01/01/1991	54	56.67434
01/02/1991	55	63.47128
01/03/1991	66	68.78879
01/04/1991	65	60.27783
01/05/1991	60	67.18038

```
For Alpha= 0.1061,Beta=0.0484,Gamma=0.0, Triple Exponential Smoothing Model forecast on
the Test Data,  RMSE is 301.699 MAPE is 28.88
```



Now taking combinations of the α , β and γ as (0.3,1.1,0.1)
resultsDf_8_2.sort_values(by=['Test MAPE']).head() gives
first five increasing 'Test MAPE'

	Alpha Values	Beta Values	Gamma Values	Train RMSE	Train MAPE	Test RMSE	Test MAPE
1	0.3	0.3	0.4	750.7104	16.83	125.4766	15.02
8	0.3	0.4	0.3	790.278145	17.08	119.8025	15.08
69	0.4	0.3	0.8	1062.85719	19.3	159.1536	15.72
16	0.3	0.5	0.3	846.083842	17.5	207.7808	21.25
131	0.5	0.3	0.6	1033.28642	18.75	279.5825	24.57

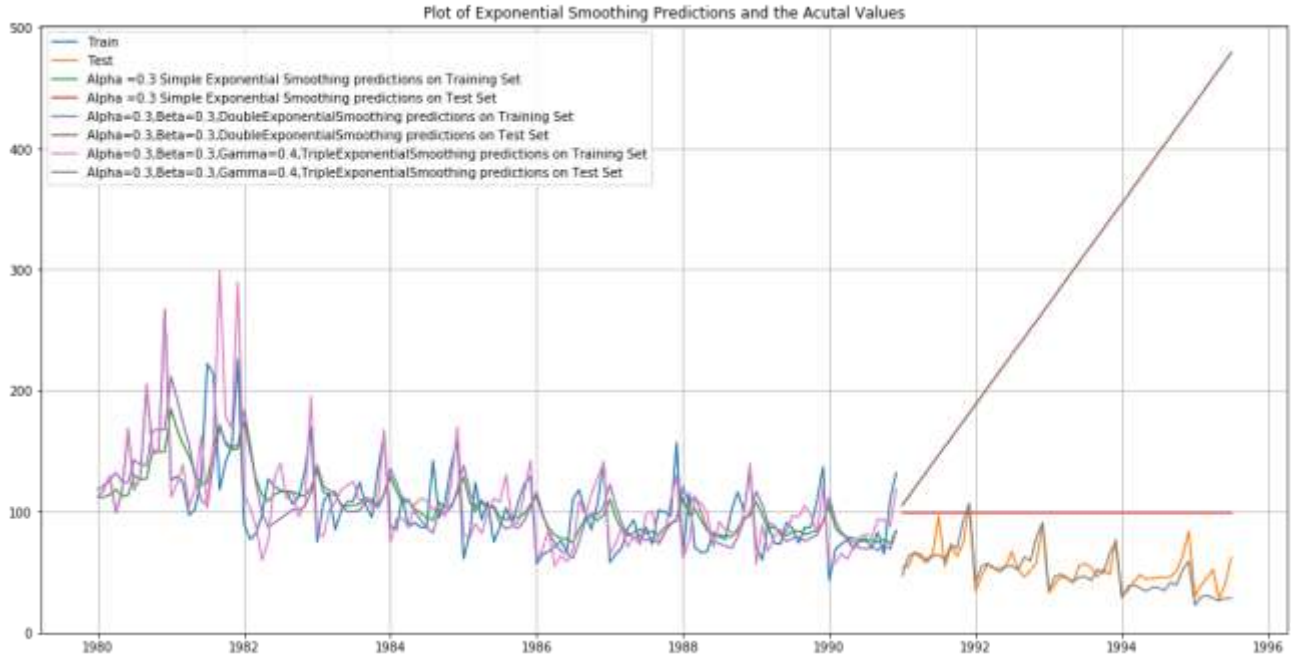
resultsDf_8_2.sort_values(by=['Test RMSE']).head() gives
first five increasing 'Test RMSE'

	Alpha Values	Beta Values	Gamma Values	Train RMSE	Train MAPE	Test RMSE	Test MAPE
8	0.3	0.4	0.3	790.278145	17.08	119.8025	15.08
1	0.3	0.3	0.4	750.7104	16.83	125.4766	15.02
69	0.4	0.3	0.8	1062.85719	19.3	159.1536	15.72
16	0.3	0.5	0.3	846.083842	17.5	207.7808	21.25
131	0.5	0.3	0.6	1033.28642	18.75	279.5825	24.57

Comparison of all models on the basis of Test RMSE and Test MAPE

	Test RMSE	Test MAPE
RegressionOnTime	233.140993	22.82
NaiveModel	6355.08283	145.1
SimpleAverageModel	2858.03251	94.93
2pointTrailingMovingAverage	132.924242	13.54
4pointTrailingMovingAverage	208.843056	19.49
6pointTrailingMovingAverage	212.17789	20.82
9pointTrailingMovingAverage	216.90308	21.01
Alpha=1,SimpleExponentialSmoothing	1353.96326	63.88
Alpha=0.3,SimpleExponentialSmoothing	2256.70802	83.71
Alpha=0.4,SimpleExponentialSmoothing	2890.93393	95.5
Alpha=0.3,Beta=0.3,DoubleExponentialSmoothing	70526.147	442.5
For Alpha= 0.1061,Beta=0.0484,Gamma=0.0,TripleExponentialSmoothing	301.699164	28.88
Alpha=0.3,Beta=0.3,Gamma=0.4,TripleExponentialSmoothing	125.476581	15.02

Alpha=0.3,Beta=0.3,Gamma=0.4,TripleExponentialSmoothing has least value of RMSE of 125.4766
On the basis of MAPE 2 point trailing model is best.



We see that the best model is the Triple Exponential Smoothing with multiplicative seasonality with the parameters α = 0.3, β = 0.3 and γ = 0.4.

For this particular mentored learning session, we will go ahead and build only the top 2 models which gaves us the best accuracy (least RMSE and MAPE).

The two models to be built on the whole data are the following:

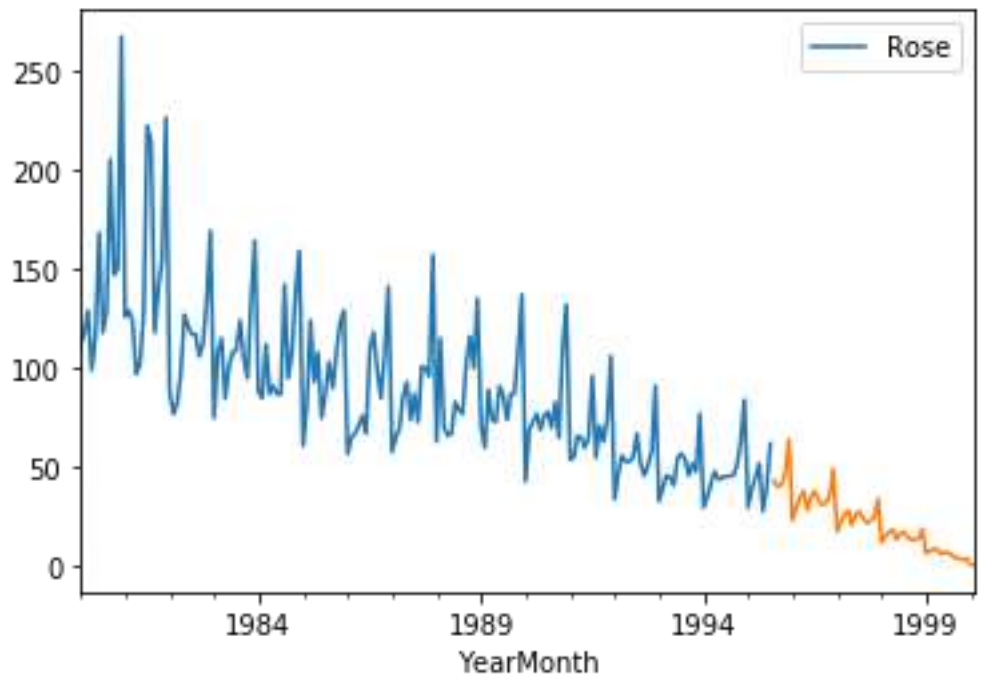
Alpha=0.3,Beta=0.3,Gamma=0.4,TripleExponentialSmoothing
2pointTrailingMovingAverage

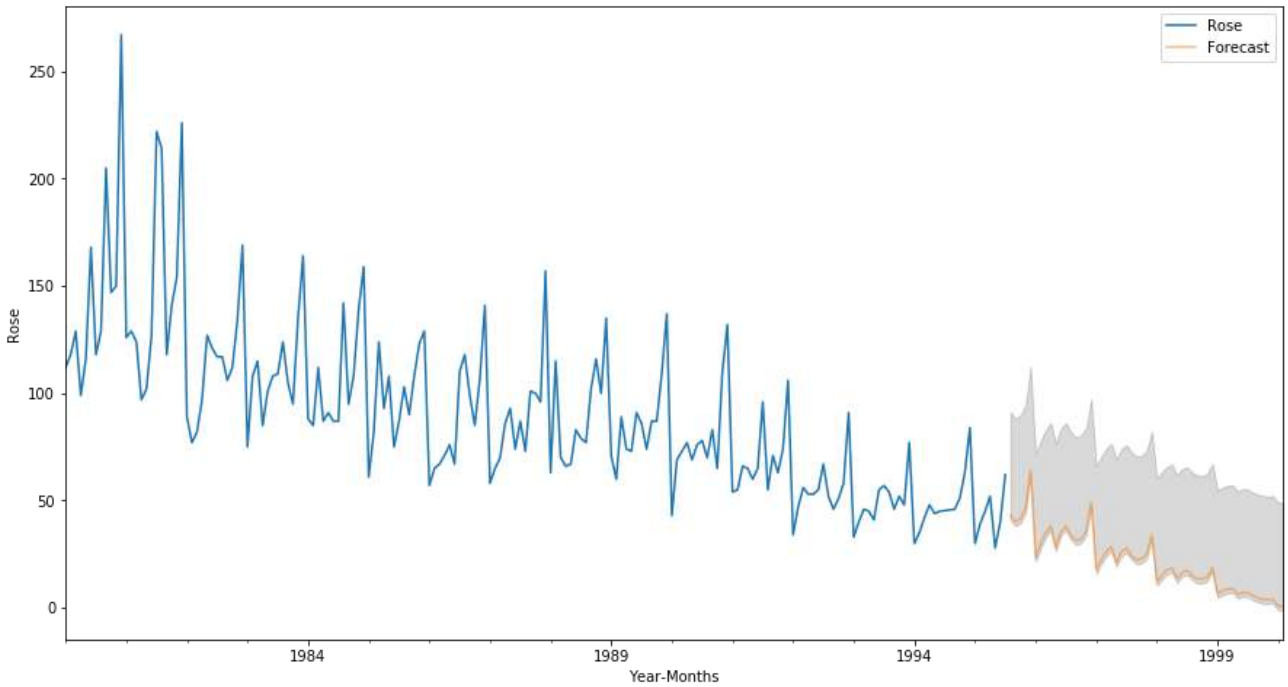
Using full model as follows:

fullmodel1 = ExponentialSmoothing(df.astype('double'),trend='additive',seasonal='multiplicative').fit(smoothing_level=0.3, smoothing_slope=0.3, smoothing_seasonal=0.4)

fitting above model with Rose dataframe

```
RMSE: 441.5553934841176
MAPE: 16.21
by forecasting the data
```



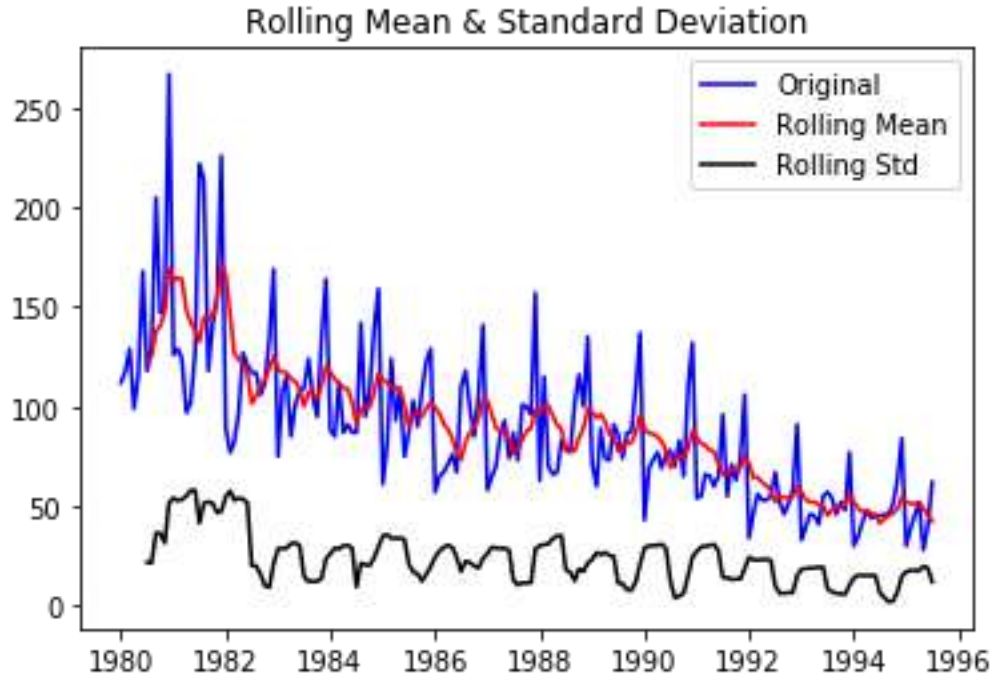


When we look at the predicted values with the respective confidence intervals, we see that even for a 95% confidence interval, the shaded area in the above graph (confidence interval) is quite high).

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.

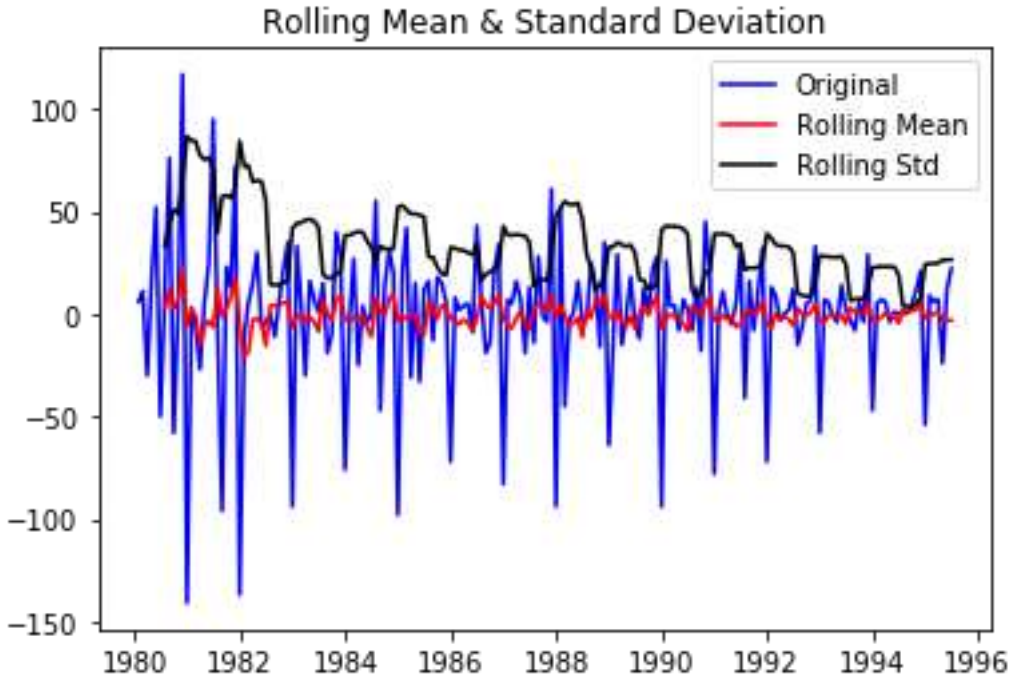
Ans:

from statsmodels.tsa.stattools import adfuller




```
Results of Dickey-Fuller Test:
Test Statistic                -8.044392e+00
p-value                        1.810895e-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
```

We see that at 5% significant level the Time Series is non-stationary.
Let us take a difference of order 1 and check whether the Time Series is stationary or not.




```
Results of Dickey-Fuller Test:
Test Statistic                -8.044392e+00
p-value                        1.810895e-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
```

We see that at alpha = 0.05 the Time Series is indeed 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.

Ans:

selecting following `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)
```

```
for above values of (p,q,d) and importing
from statsmodels.tsa.arima_model import ARIMA
```

```
ARIMA(0, 1, 0) - AIC:1335.1526583086775
ARIMA(0, 1, 1) - AIC:1280.7261830464033
ARIMA(0, 1, 2) - AIC:1276.835381295044
ARIMA(1, 1, 0) - AIC:1319.3483105801806
ARIMA(1, 1, 1) - AIC:1277.775750920457
ARIMA(1, 1, 2) - AIC:1277.3592235347464
ARIMA(2, 1, 0) - AIC:1300.6092611744373
ARIMA(2, 1, 1) - AIC:1279.0456894093154
ARIMA(2, 1, 2) - AIC:1279.2986939364948
```

above AIC values are obtained.

	param	AIC
2	(0, 1, 2)	1276.8354
5	(1, 1, 2)	1277.3592
4	(1, 1, 1)	1277.7758
7	(2, 1, 1)	1279.0457
8	(2, 1, 2)	1279.2987
1	(0, 1, 1)	1280.7262
6	(2, 1, 0)	1300.6093
3	(1, 1, 0)	1319.3483
0	(0, 1, 0)	1335.1527

Lowest AIC is applied to model with (0, 1, 2)
Model summary is obtained as follows:

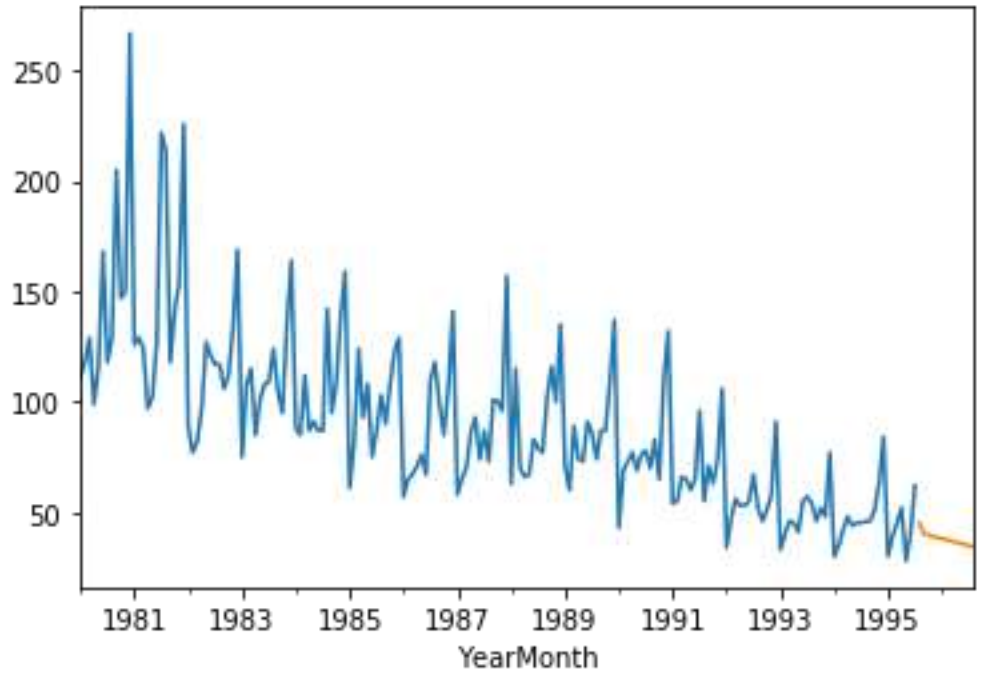
results_Arima.summary()

```
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             30.168
Date:                Fri, 31 Jul 2020   AIC                           1276.835
Time:                        18:46:49   BIC                           1288.336
Sample:                    02-01-1980   HQIC                          1281.509
                         - 12-01-1990
==============================================================================
                 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   -0.7600      0.101     -7.500      0.000      -0.959      -0.561
ma.L2.D.Rose   -0.2398      0.095     -2.518      0.013      -0.427      -0.053
                                    Roots
=============================================================================
                  Real          Imaginary           Modulus         Frequency
-----------------------------------------------------------------------------
MA.1            1.0001           +0.0000j            1.0001            0.0000
MA.2           -4.1695           +0.0000j            4.1695            0.5000
-----------------------------------------------------------------------------
```

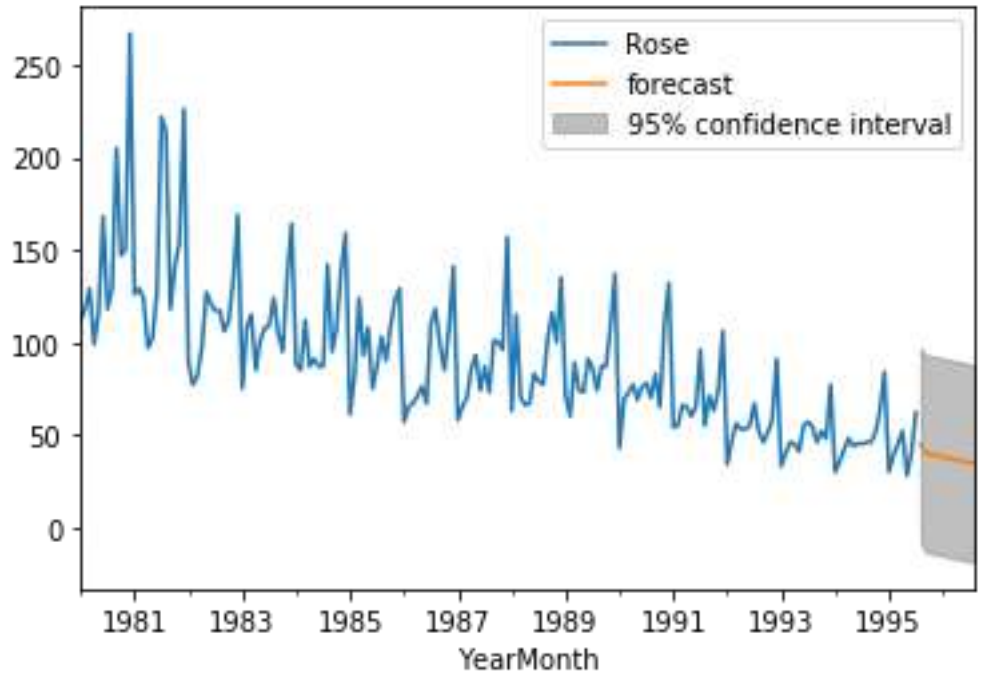
For Test data RMSE=243.9712170851629

Prediction for above test data is as follows:
predicted = results_Arima.predict(start='1995-08-01',end='1996-08-01',typ='levels')

```
1995-08-01    44.883243
1995-09-01    40.103168
1995-10-01    39.580268
1995-11-01    39.057368
1995-12-01    38.534467
1996-01-01    38.011567
1996-02-01    37.488667
1996-03-01    36.965766
1996-04-01    36.442866
1996-05-01    35.919966
1996-06-01    35.397066
1996-07-01    34.874165
1996-08-01    34.351265
Freq: MS, dtype: float64
```



Yellow one shows the predicted data.

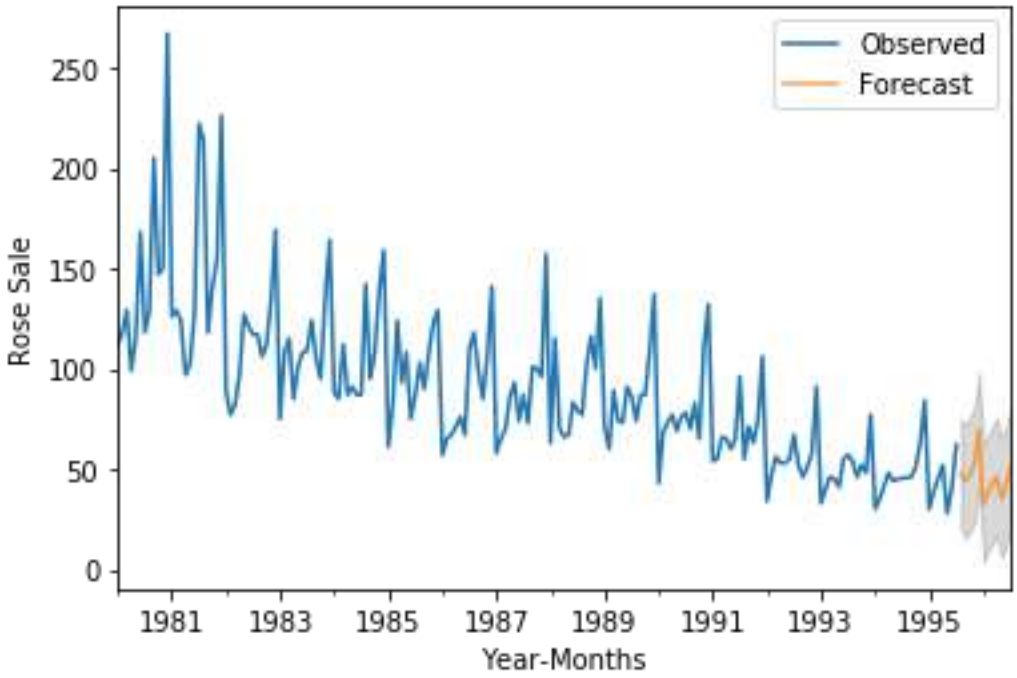


SARIMA MODEL:

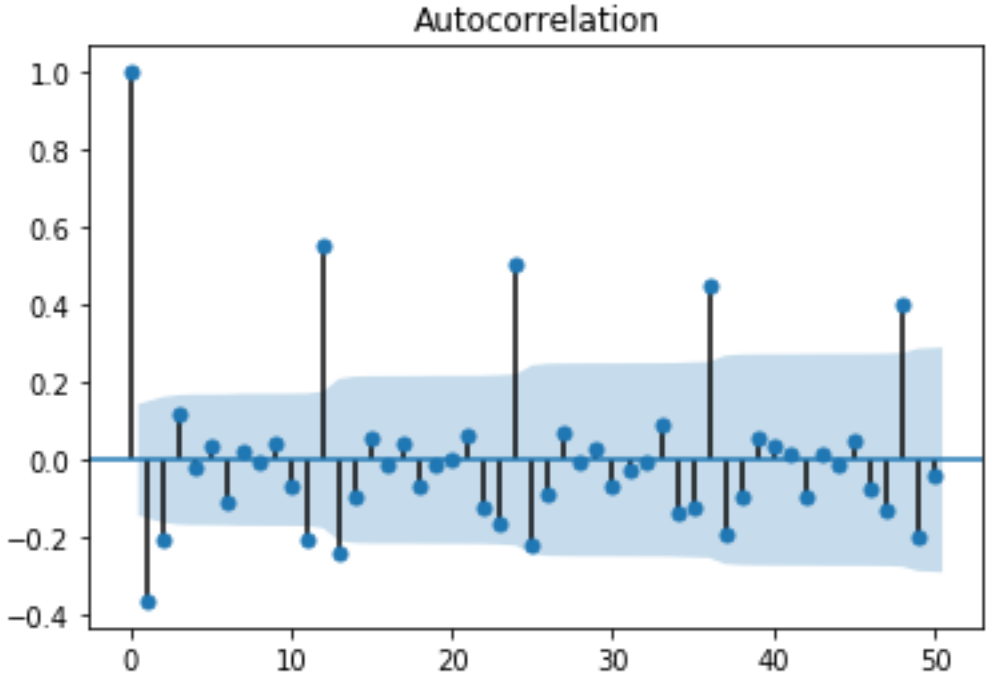
```
Selecting parameter combinations for Model...
Model: (0, 1, 1)(0, 0, 1, 12)
Model: (0, 1, 2)(0, 0, 2, 12)
Model: (1, 1, 0)(1, 0, 0, 12)
Model: (1, 1, 1)(1, 0, 1, 12)
Model: (1, 1, 2)(1, 0, 2, 12)
Model: (2, 1, 0)(2, 0, 0, 12)
Model: (2, 1, 1)(2, 0, 1, 12)
Model: (2, 1, 2)(2, 0, 2, 12)
```

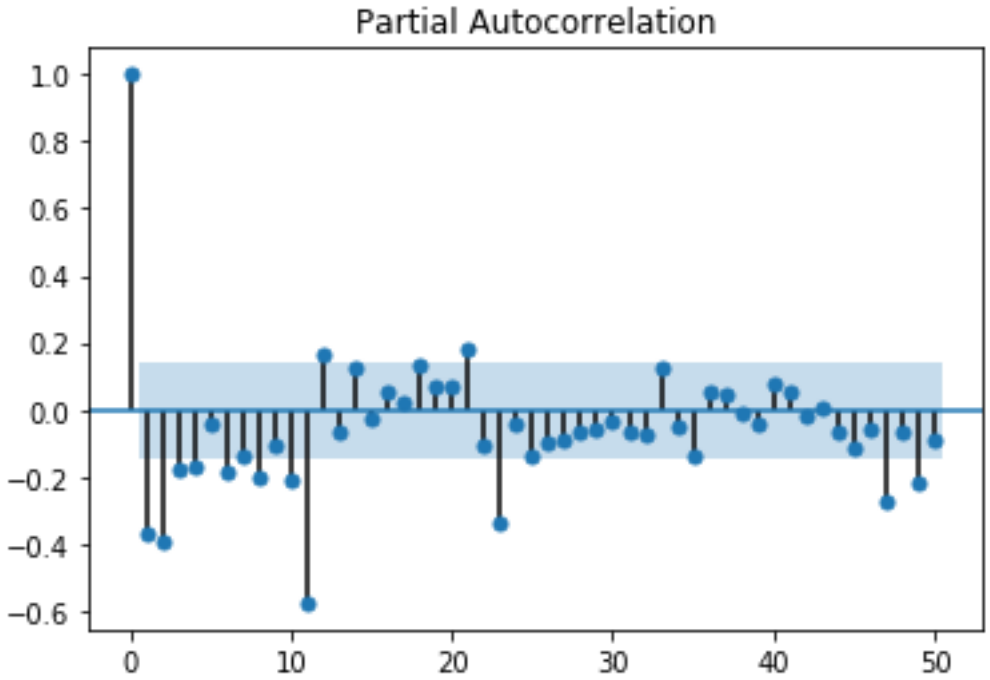
	param	seasonal	AIC
26	(0, 1, 2)	(2, 0, 2, 12)	887.9375
53	(1, 1, 2)	(2, 0, 2, 12)	889.9017
80	(2, 1, 2)	(2, 0, 2, 12)	890.6688
69	(2, 1, 1)	(2, 0, 0, 12)	896.5182
78	(2, 1, 2)	(2, 0, 0, 12)	897.3464

Prediction of SARIMA
pred = results_SARIMA.get_forecast(steps=len(test))
SARIMA(0, 1, 2)(2, 0, 2, 12) combination gives RMSE of 243.971217



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.





For ACF and PACF same models can be build.

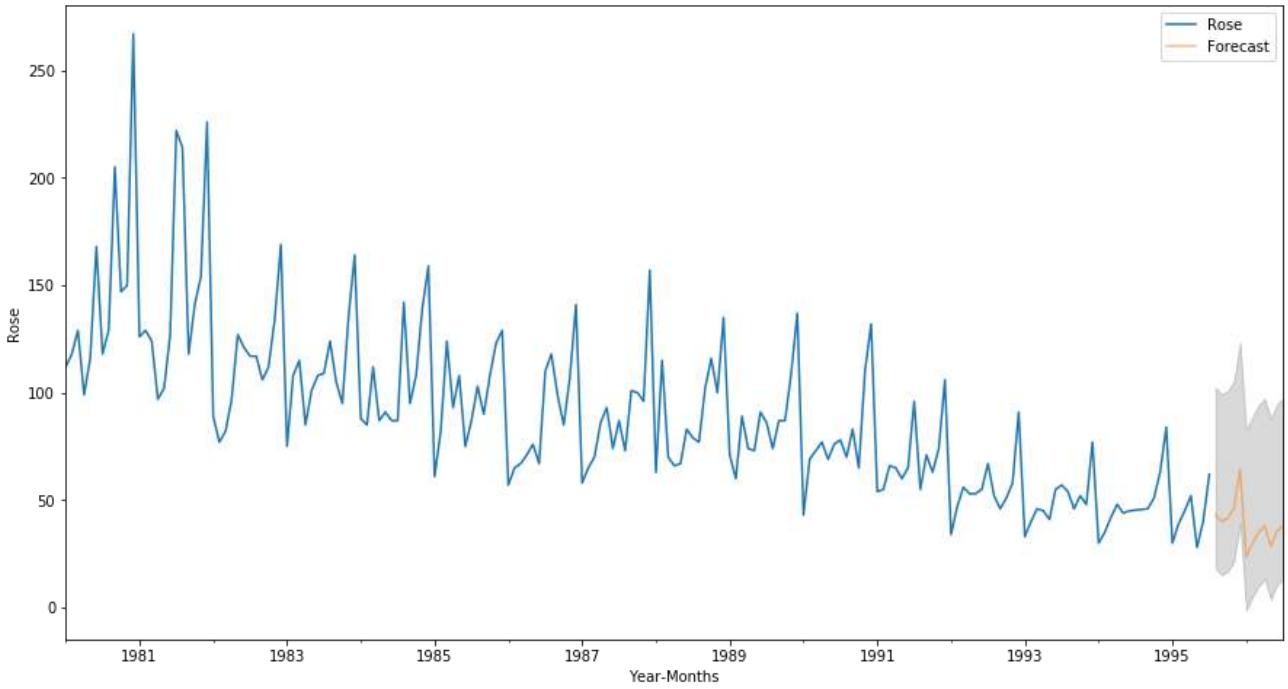
8. Build a table (create a data frame) with all the models built along with their corresponding parameters and the respective RMSE values on the test data.

	Test RMSE	Test MAPE
RegressionOnTime	233.140993	22.82
NaiveModel	6355.08283	145.1
SimpleAverageModel	2858.03251	94.93
2pointTrailingMovingAverage	132.924242	13.54
4pointTrailingMovingAverage	208.843056	19.49
6pointTrailingMovingAverage	212.17789	20.82
9pointTrailingMovingAverage	216.90308	21.01
Alpha=1,SimpleExponentialSmoothing	1353.96326	63.88
Alpha=0.3,SimpleExponentialSmoothing	2256.70802	83.71
Alpha=0.4,SimpleExponentialSmoothing	2890.93393	95.5
Alpha=0.3,Beta=0.3,DoubleExponentialSmoothing	70526.147	442.5
For Alpha= 0.1061,Beta=0.0484,Gamma=0.0,TripleExponentialSmoothing	301.699164	28.88
Alpha=0.3,Beta=0.3,Gamma=0.4,TripleExponentialSmoothing	125.476581	15.02
ARIMA(0,1,2)	243.971217	
SARIMA(0, 1, 2)(2, 0, 2, 12)243.971217	243.971217	

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.

So selecting **Alpha=0.3,Beta=0.3,Gamma=0.4,TripleExponentialSmoothing model**

```
1995-08-01    43.170615
1995-09-01    40.142453
1995-10-01    41.562403
1995-11-01    46.295150
1995-12-01    64.191818
1996-01-01    23.552225
1996-02-01    29.634652
1996-03-01    34.887809
1996-04-01    38.053789
1996-05-01    28.513430
1996-06-01    35.434576
1996-07-01    38.112183
```



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

Ans:

From above exercise the conclusion can be drawn as follows:

1.Sale of Rose is decreasing continuously.In future it will decrease upto minimum zero or maximum upto 140.

B. Data of Sparkling

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

Ans:

a.Liabraries are imported:

```
import pandas as pd
import numpy as np
import seaborn as sns
from matplotlib import pyplot as plt
```

b.Data is read by parsing method as follows:

df = pd.read_csv('Sparkling.csv', index_col = 0, parse_dates = True, squeeze = True)

df.head

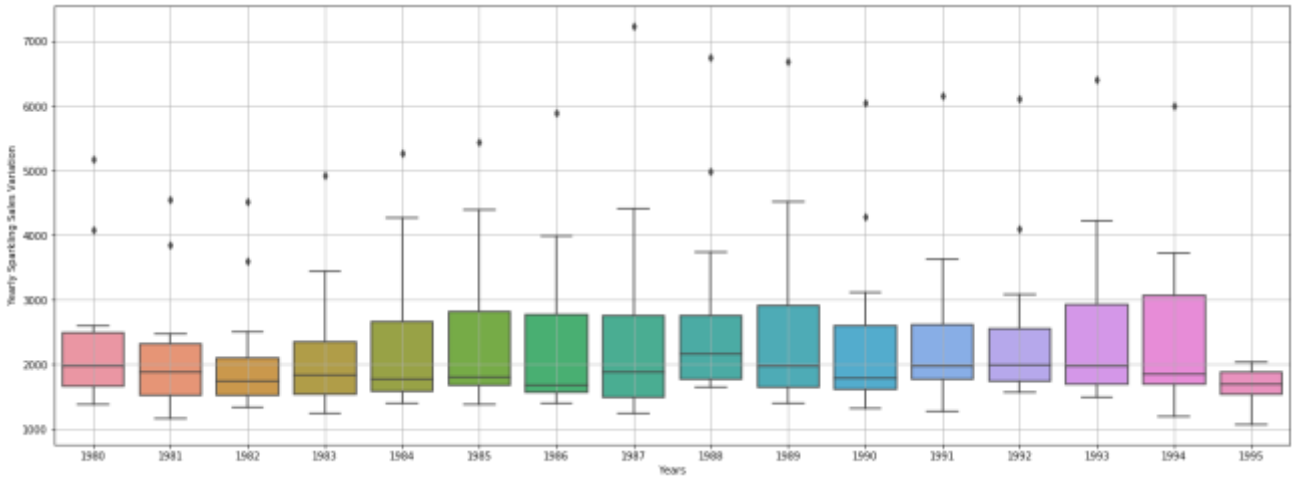
```
YearMonth
1980-01-01    1686
1980-02-01    1591
1980-03-01    2304
1980-04-01    1712
1980-05-01    1471
c.No null values in dataset
d. df.describe()
count     187.000000
mean     2402.417112
std      1295.111540
min      1070.000000
25%      1605.000000
50%      1874.000000
75%      2549.000000
max      7242.000000
Name: Sparkling, dtype: float64
```

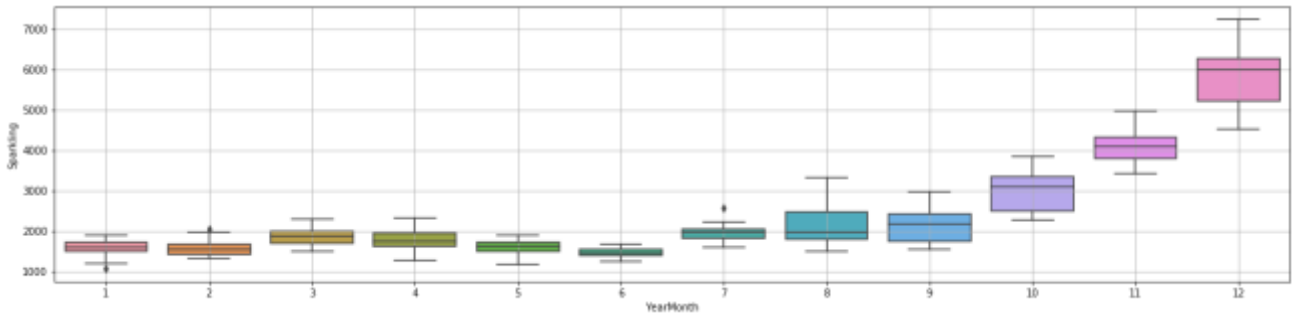
2.Perform appropriate Exploratory Data Analysis to understand the data and also perform decomposition.

Ans:

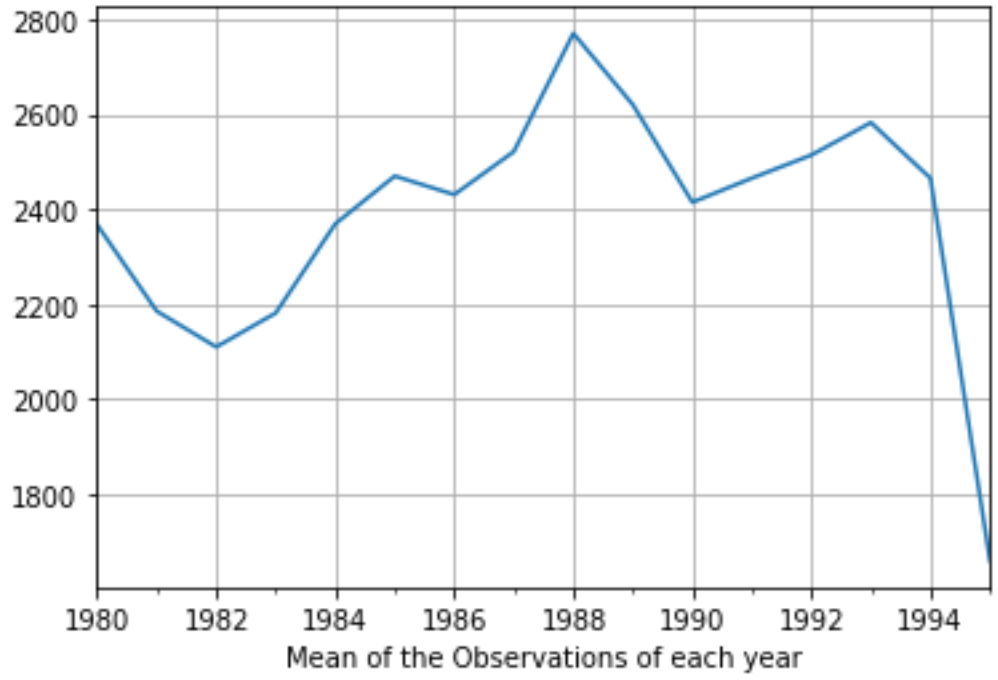
a. EDA

This box plot shows the Sale of sparkling which has increased in 1987.

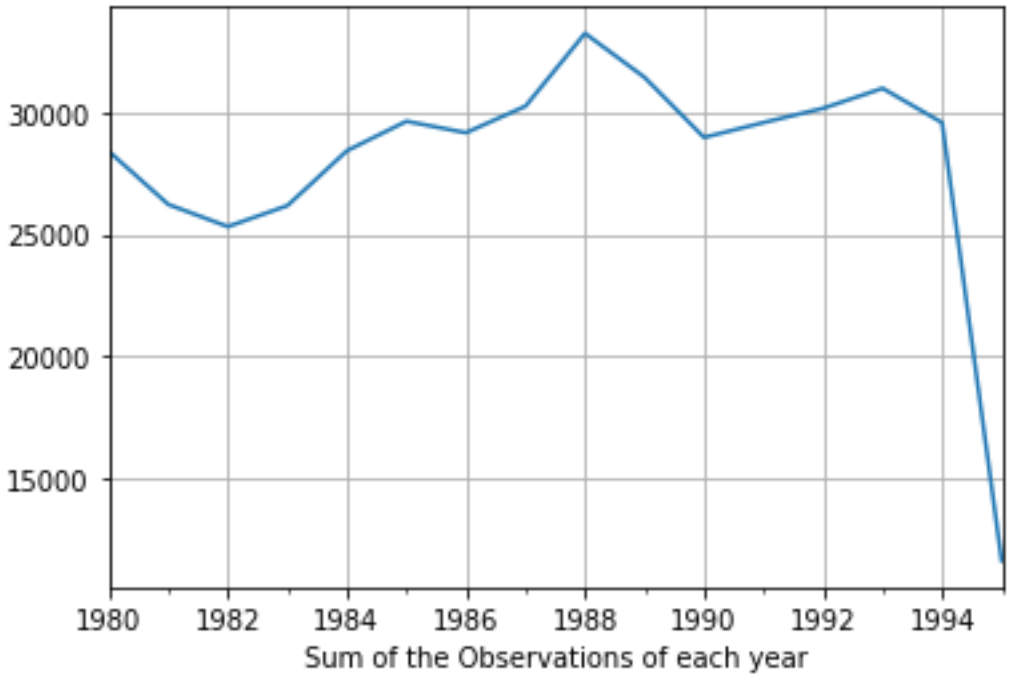




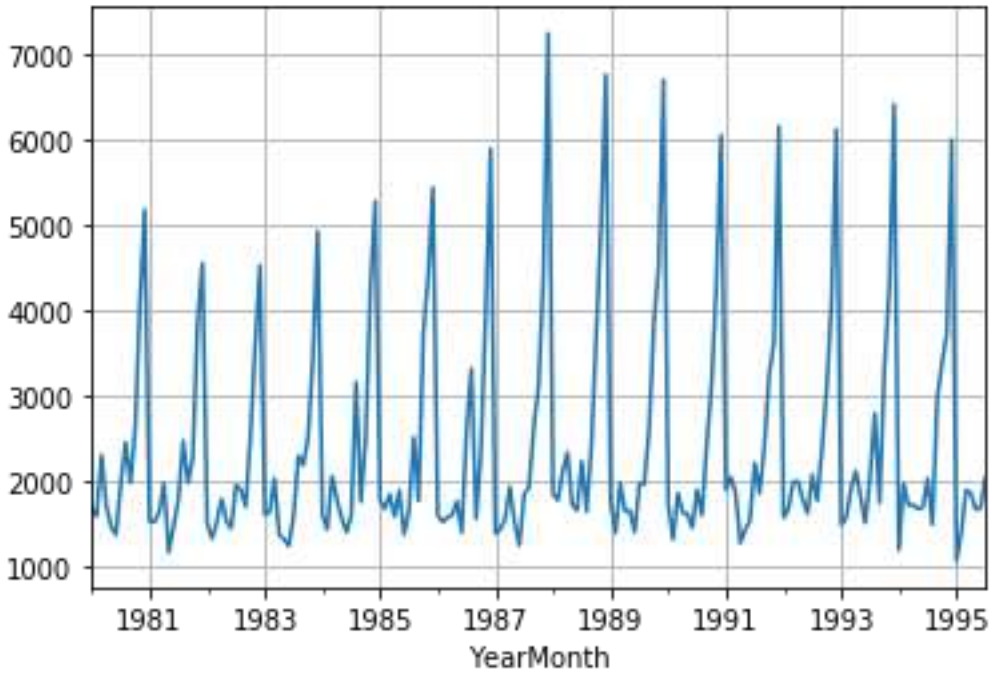
In the month of January sale is maximum.

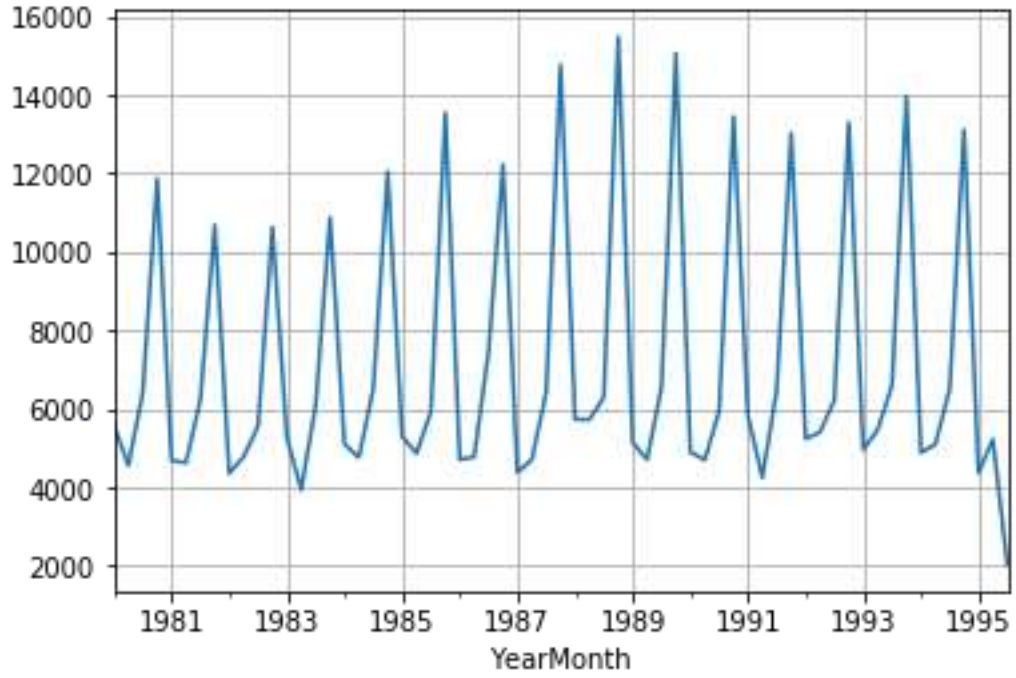


In year 1988 mean sale is maximum near to 2800.

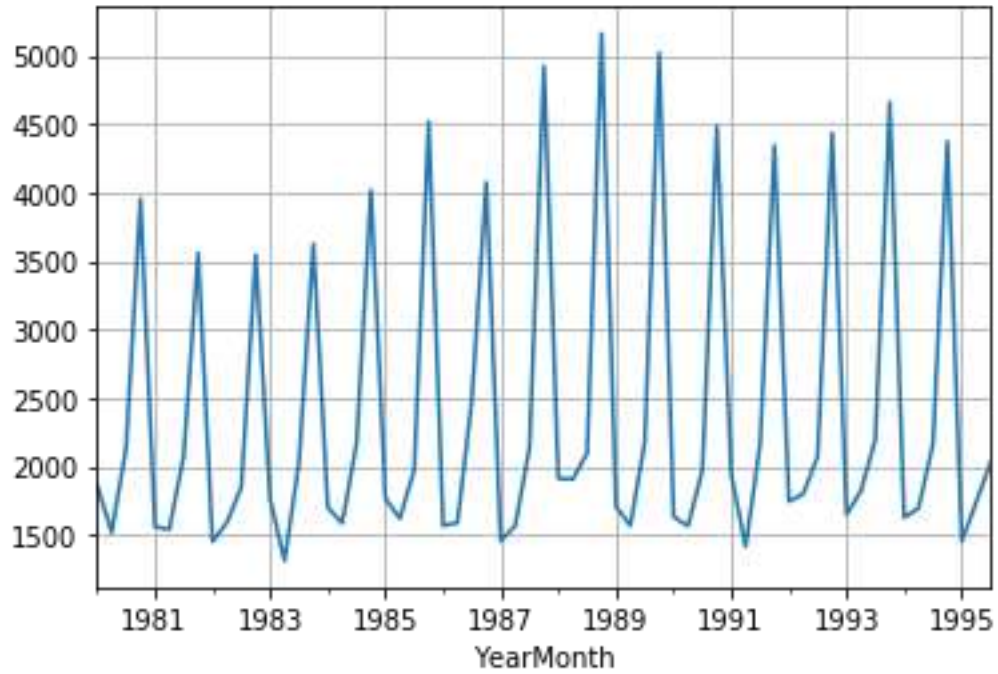


Sum of the sparkling is maximum in 1988 upto 35000

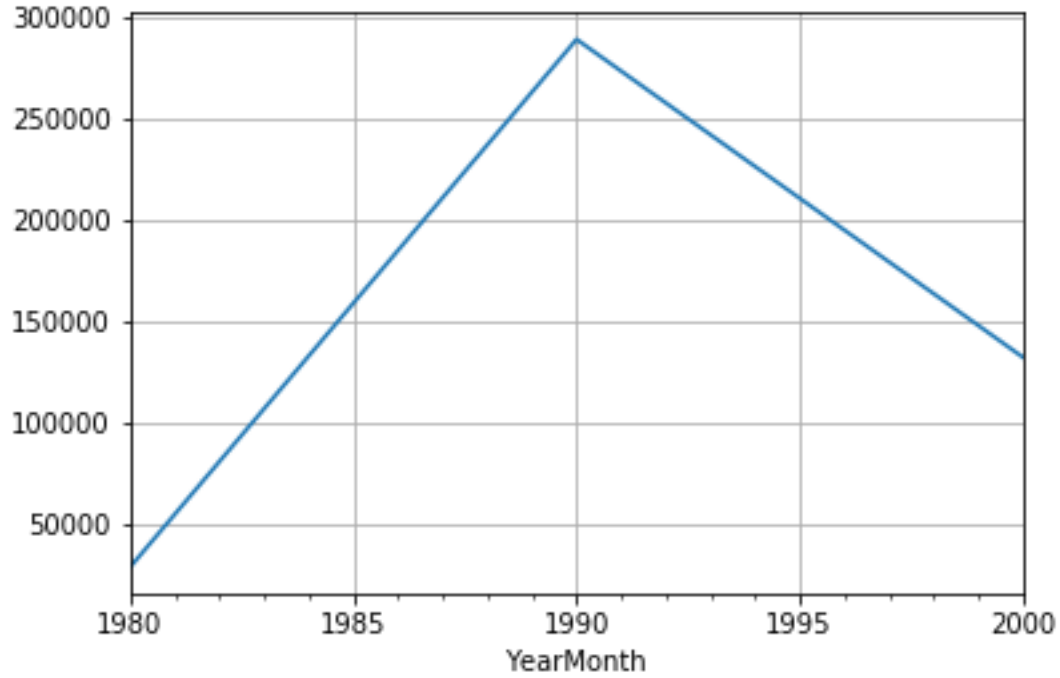




Quartarly sum is maximum in last quarter of 15000.

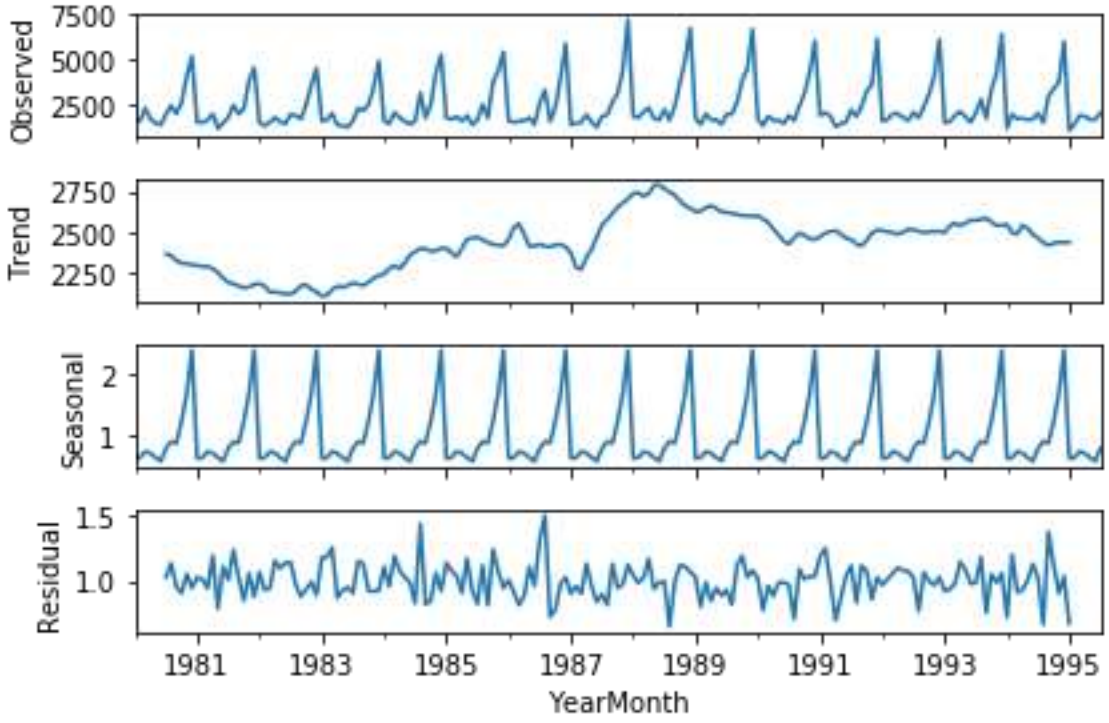


Quarterly mean is maximum is maximum in last quarter of 1988.

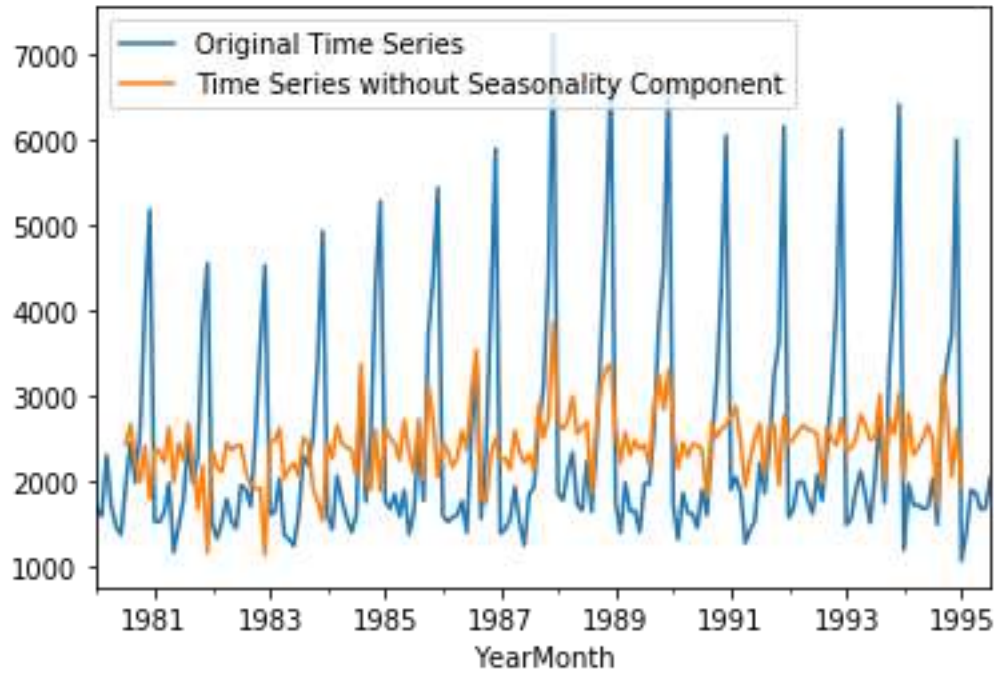


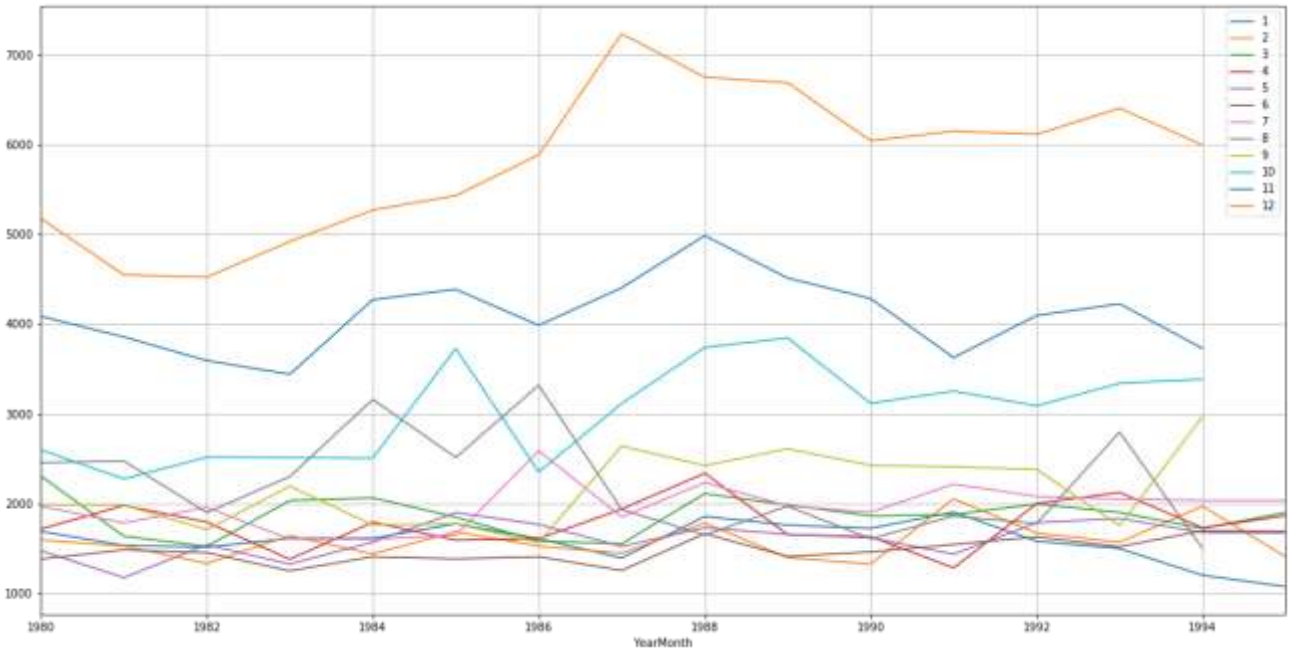
Sale is Maximum in the first decade of 1980-1990 upto 280000.

b.Series decomposition.



Series has seasonality in every December of year with period of 12 th month.





12 month sales is maximum

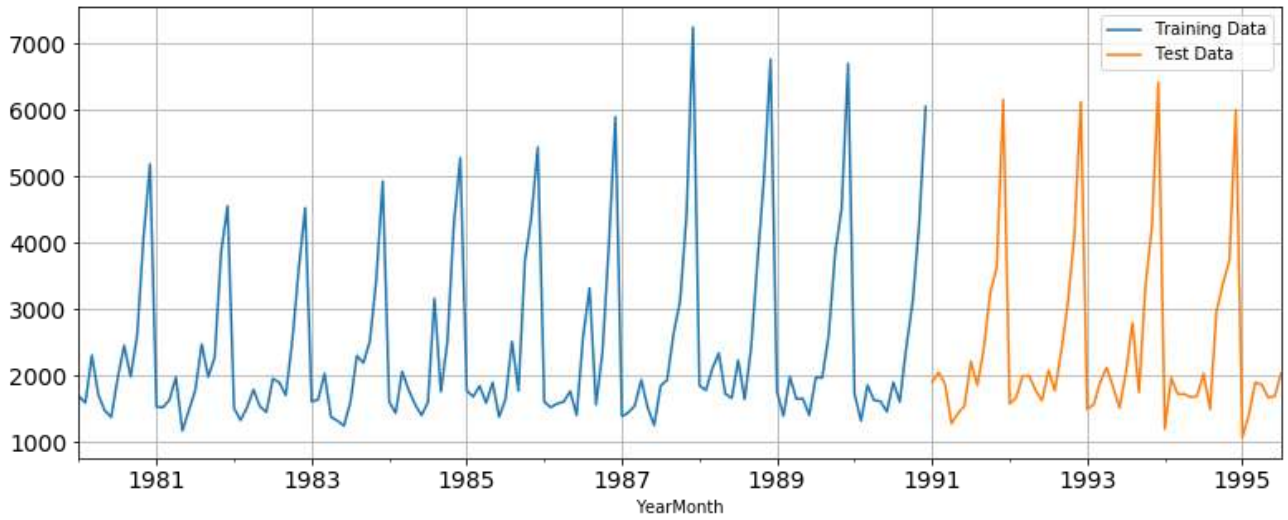
3.Split the data into training and test. The test data should start in 1991.

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.tail(5))
```

print(test.head(5))

```
            Sparkling
YearMonth
1990-08-01       1605
1990-09-01       2424
1990-10-01       3116
1990-11-01       4286
1990-12-01       6047
            Sparkling
YearMonth
1991-01-01       1902
1991-02-01       2049
1991-03-01       1874
1991-04-01       1279
1991-05-01       1432
print(train.shape): (132, 1)


print(test.shape): (55, 1)
First few rows of Training Data
             Sparkling
YearMonth
1980-01-01       1686
1980-02-01       1591
1980-03-01       2304
1980-04-01       1712
1980-05-01       1471

Last few rows of Training Data
             Sparkling
YearMonth
1990-08-01       1605
1990-09-01       2424
1990-10-01       3116
1990-11-01       4286
1990-12-01       6047

First few rows of Test Data
             Sparkling
YearMonth
1991-01-01       1902
1991-02-01       2049
1991-03-01       1874
1991-04-01       1279
1991-05-01       1432

Last few rows of Test Data
             Sparkling
YearMonth
1995-03-01       1897
1995-04-01       1862
1995-05-01       1670
1995-06-01       1688
1995-07-01       2031
```

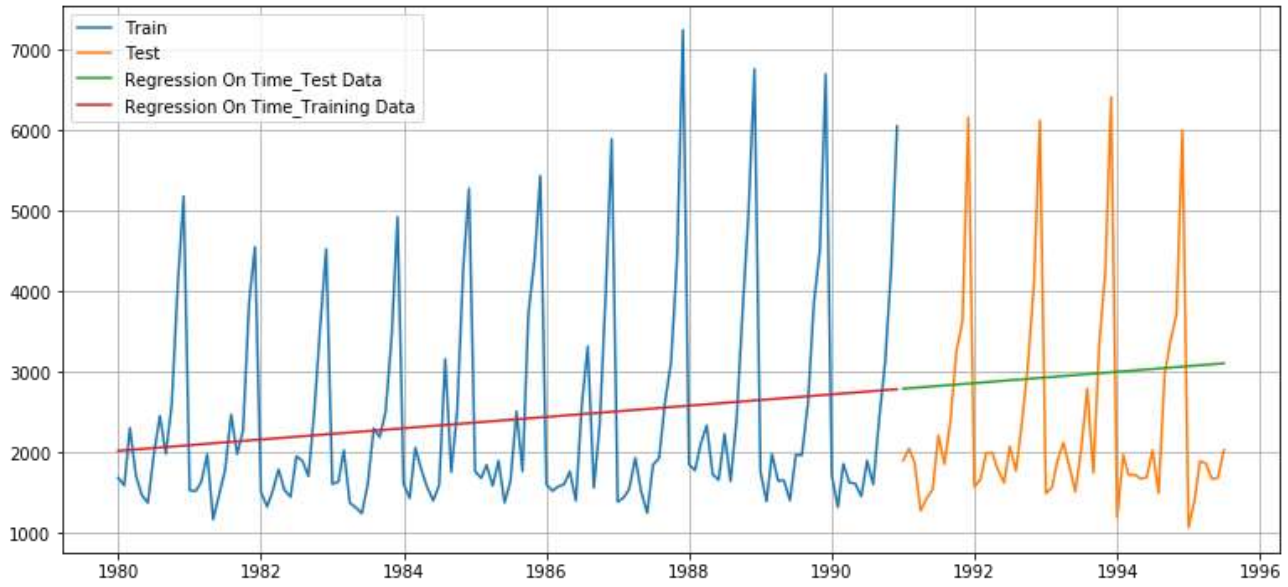


4.Build various exponential smoothing models on the training data and evaluate the model using RMSE on the test data. Other models such as regression,naïve forecast models, simple average models etc. should also be built on the training data and check the performance on the test data using RMSE. - Please do try to build as many models as possible and as many iterations of models as possible with different parameters.

Ans:Same approach is used as above data set of Rose.

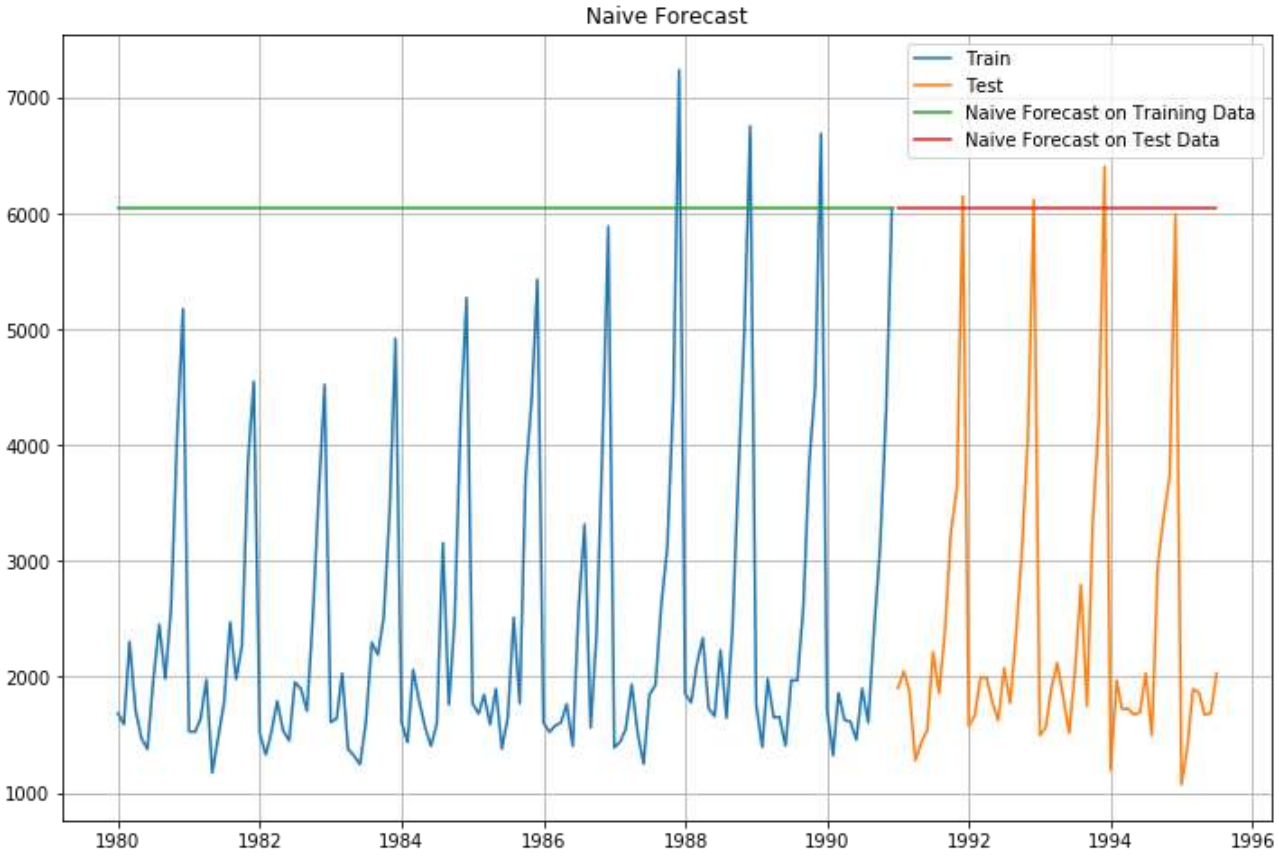
Model 1: Linear Regression

```
For RegressionOnTime forecast on the Training Data,  RMSE is 1636665.665 MAPE
is 40.05
For RegressionOnTime forecast on the Test Data,  RMSE is 1929696.534 MAPE is
50.15
```



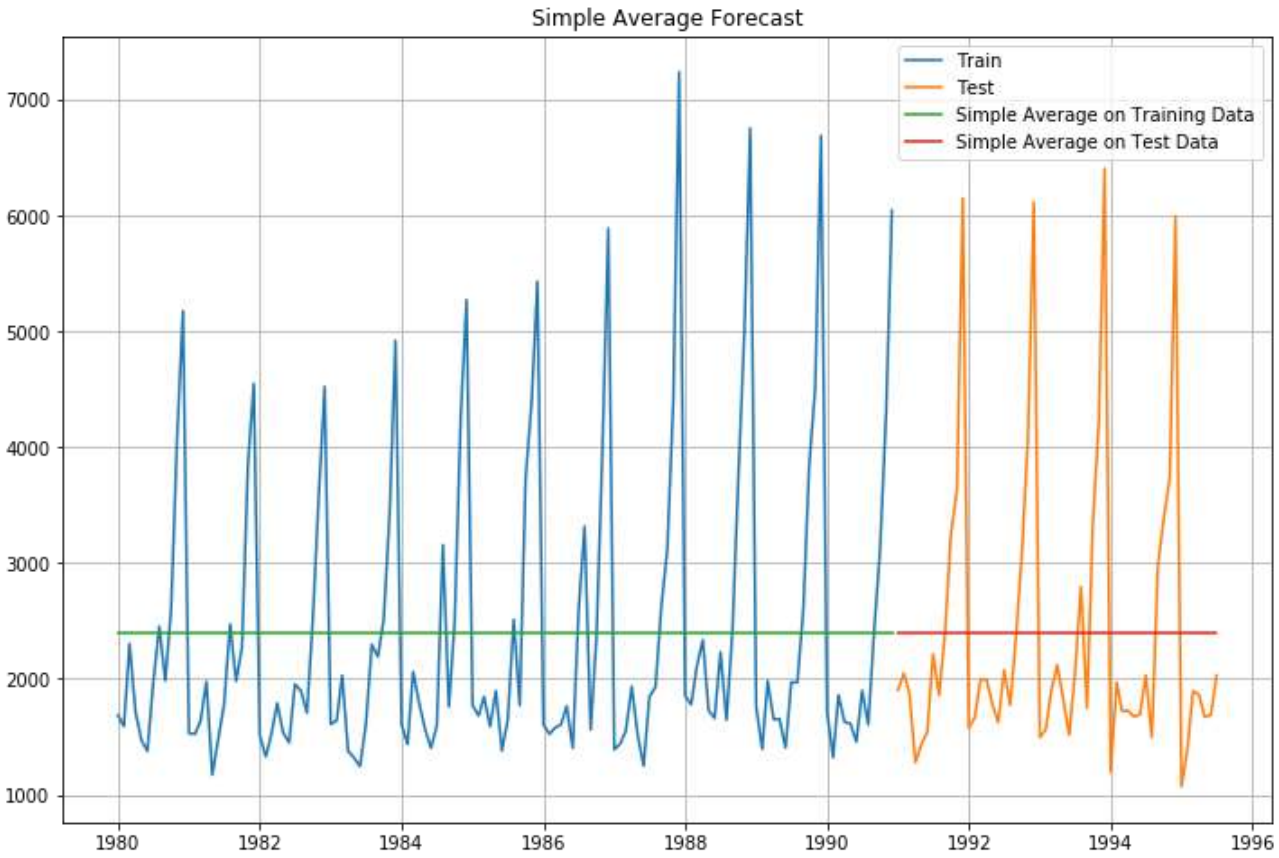
Model 2: Naive Approach: $\hat{y}_{t+1}=y_t$

For Naive Model forecast on the Training Data, RMSE is 14959109.492 MAPE is 153.17
For Naive Model forecast on the Test Data, RMSE is 14932654.909 MAPE is 152.87



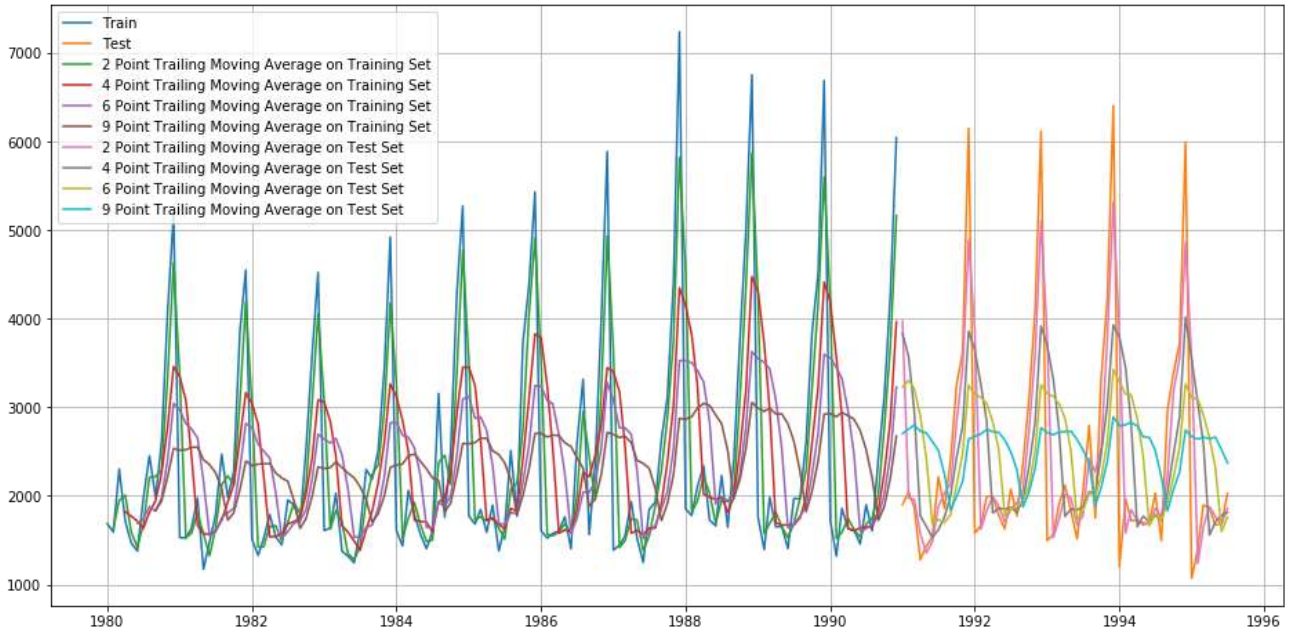
Model 3: Simple Average

For Simple Average Model forecast on the Training Data, RMSE is 1686059.732 MAPE is 40.36
For Simple Average forecast on the Test Data, RMSE is 1625833.606 MAPE is 38.90

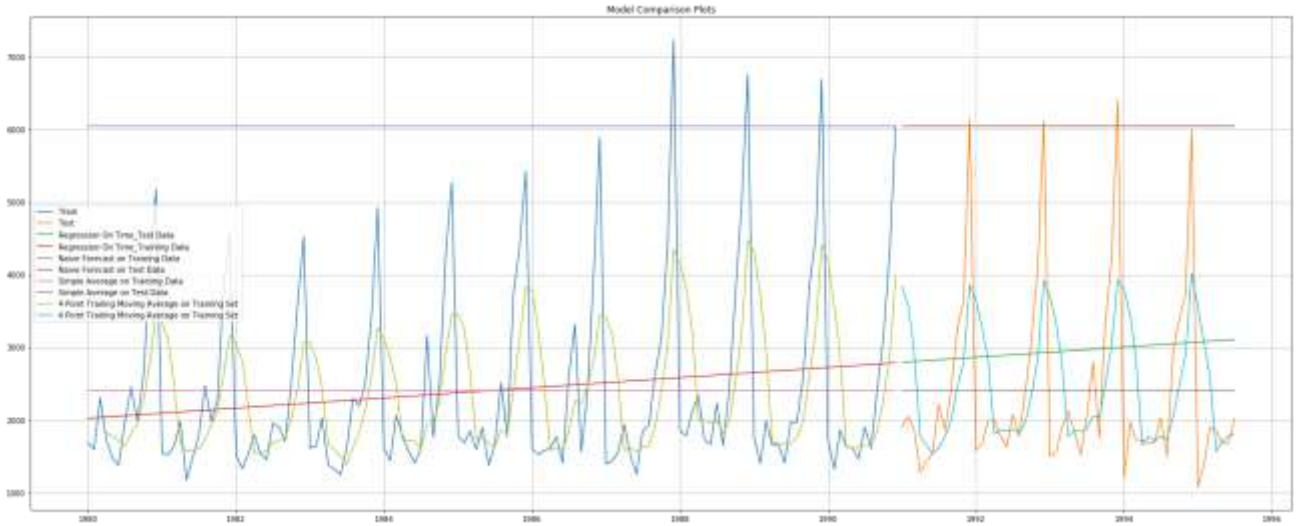


Model 4: Moving Average(MA)

	Test RMSE	Test MAPE
2pointTrailingMovingAverage	6.62E+05	19.7
4pointTrailingMovingAverage	1.34E+06	35.96
6pointTrailingMovingAverage	1.65E+06	43.86
9pointTrailingMovingAverage	1.81E+06	46.86



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



	Test RMSE	Test MAPE
RegressionOnTime	1.93E+06	50.15
NaiveModel	1.49E+07	152.87
SimpleAverageModel	1.63E+06	38.9
2pointTrailingMovingAverage	6.62E+05	19.7
4pointTrailingMovingAverage	1.34E+06	35.96
6pointTrailingMovingAverage	1.65E+06	43.86
9pointTrailingMovingAverage	1.81E+06	46.86

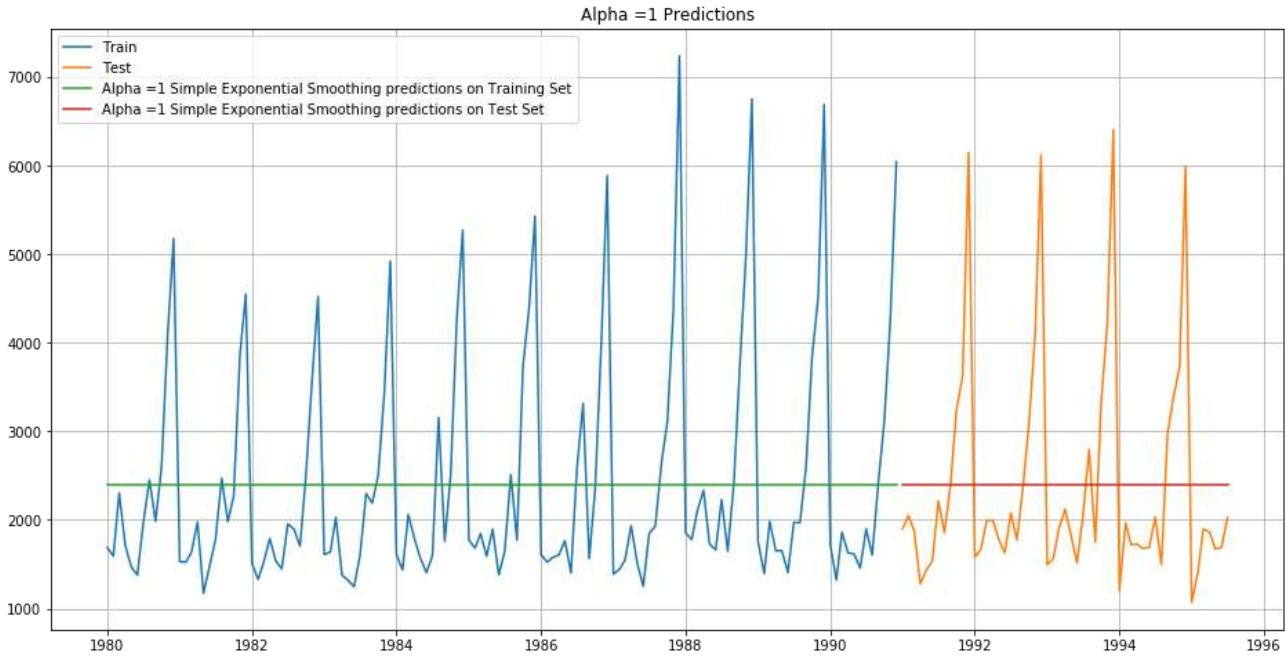
Model 5: Simple Exponential Smoothing

Best parameters as follws:

```
model_SES_autofit.params
{'smoothing_level': 0.0,
 'smoothing_slope': nan,
 'smoothing_seasonal': nan,
 'damping_slope': nan,
 'initial_level': 2403.783249368794,
 'initial_slope': nan,
 'initial_seasons': array([], dtype=float64),
 'use_boxcox': False,
 'lamda': None,
 'remove_bias': False}
```

After applying above model prediction is as follows for Train data.

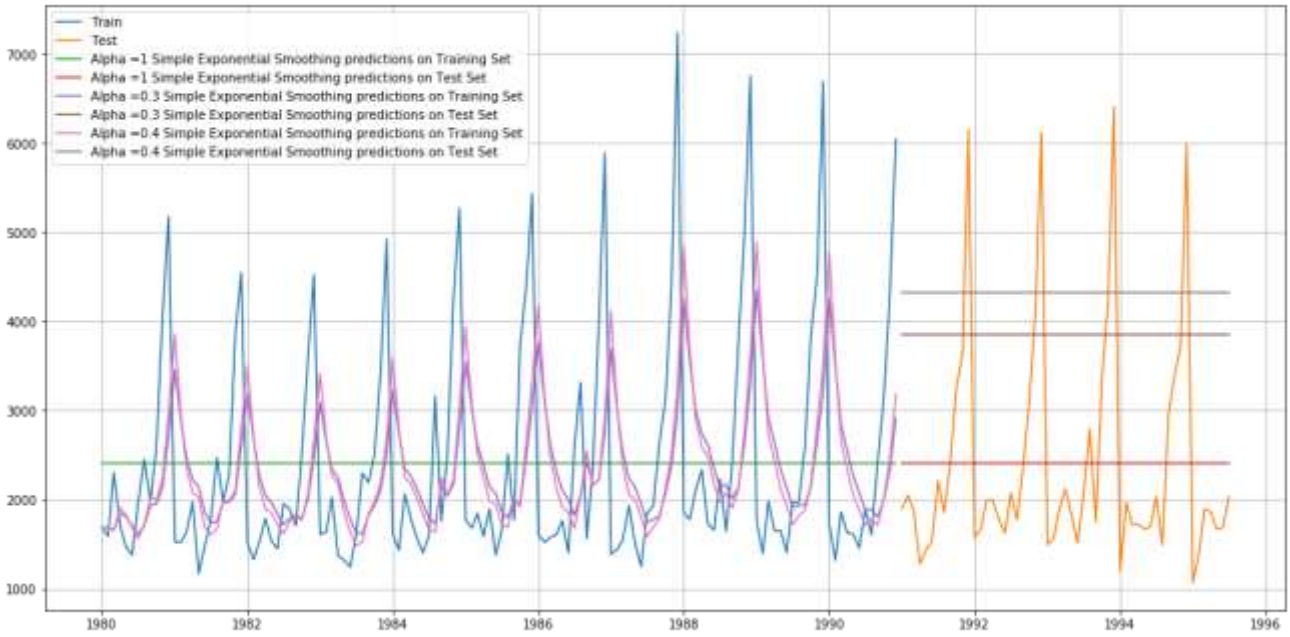
YearMonth	Sparkling	predict
1980-01-01	1686	2403.783249
1980-02-01	1591	2403.783249
1980-03-01	2304	2403.783249
1980-04-01	1712	2403.783249
1980-05-01	1471	2403.783249




```
For Alpha =1 Simple Exponential Smoothing Model forecast on the Training
Data,  RMSE is 1625833.633 MAPE is 38.90
For Alpha =1 Simple Exponential Smoothing Model forecast on the Training
Data,  RMSE is 1686059.732 MAPE is 40.36
```

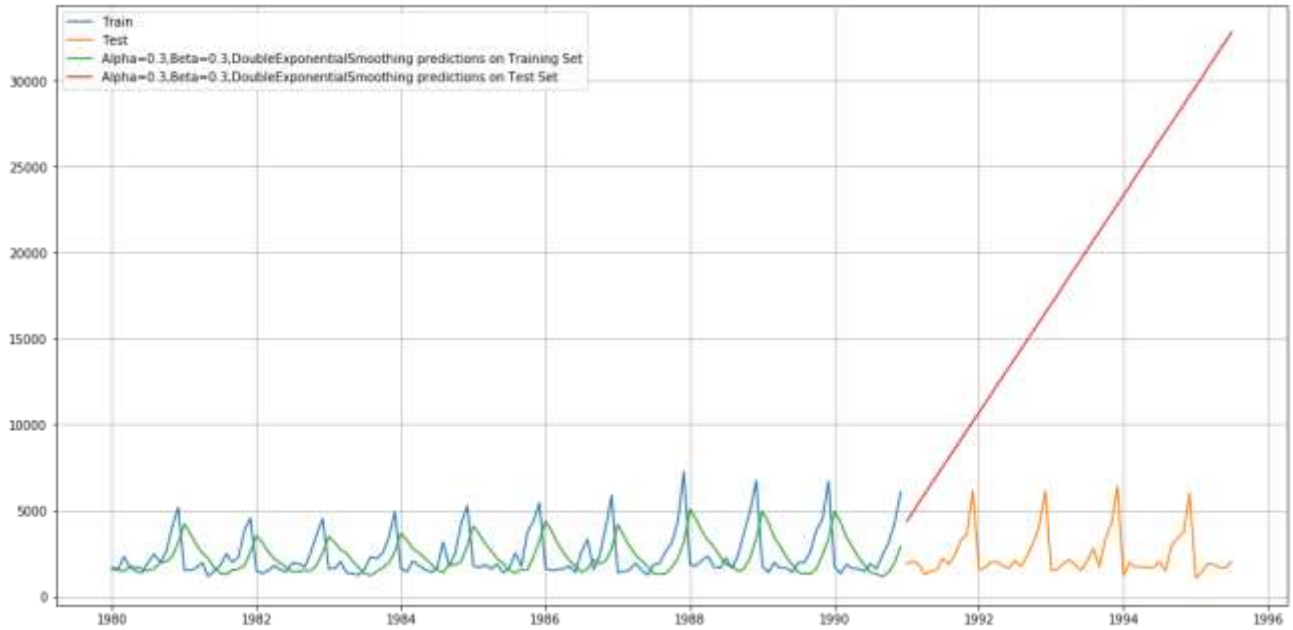
Setting different alpha values.

	Alpha Values	Train RMSE	Train MAPE	Test RMSE	Test MAPE
0	0.3	1.85E+06	43.73	3.75E+06	75.66
1	0.4	1.83E+06	42.75	5.34E+06	91.55
2	0.5	1.81E+06	41.16	7.11E+06	106.27
3	0.6	1.79E+06	39.8	8.88E+06	118.77
4	0.7	1.79E+06	38.55	1.06E+07	129.34
5	0.8	1.81E+06	37.6	1.21E+07	138.34
6	0.9	1.84E+06	36.79	1.36E+07	146.08



Model 6: Double Exponential Smoothing (Holt's Model)

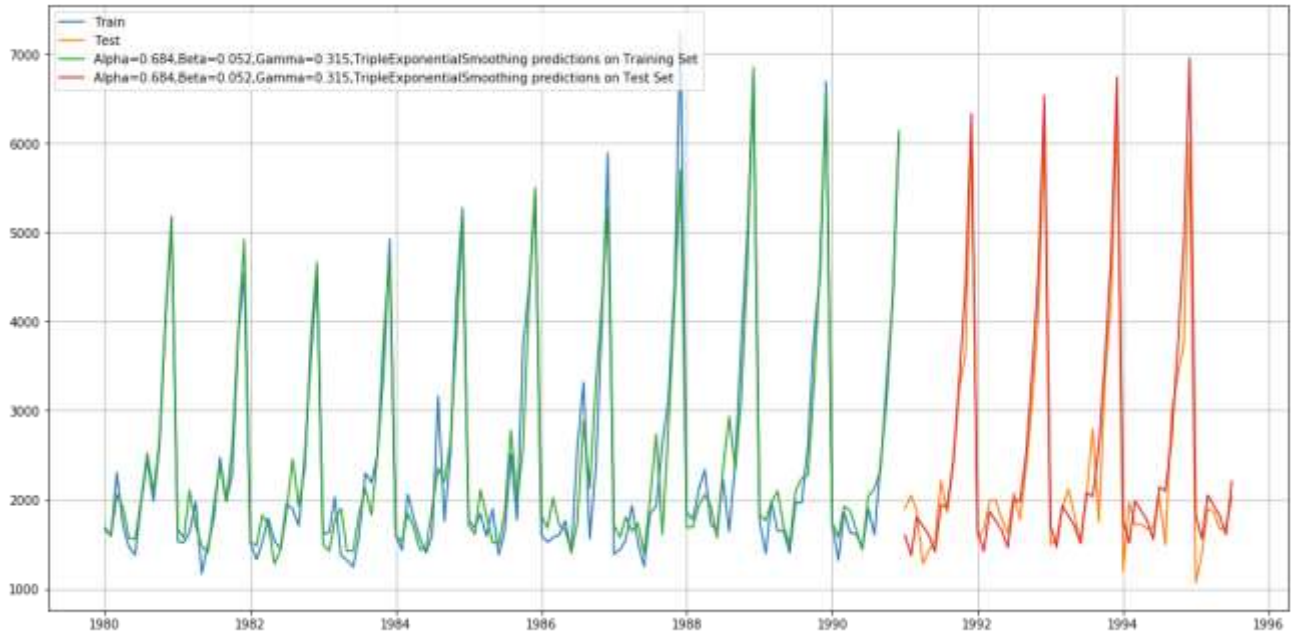
	Alpha Values	Beta Values	Train RMSE	Train MAPE	Test RMSE	Test MAPE
0	0.3	0.3	2.54E+06	53.75	3.33E+08	675.28
8	0.4	0.3	2.46E+06	50.06	5.70E+08	886
1	0.3	0.4	2.83E+06	57.13	6.80E+08	960.18
16	0.5	0.3	2.34E+06	45.92	7.34E+08	1007.39
24	0.6	0.3	2.27E+06	42.79	8.45E+08	1082.18



Method 7: Triple Exponential Smoothing (Holt - Winter's Model)

Three parameters α , β and γ are estimated in this model. Level, Trend and Seasonality are accounted for in this model.

```
model_TES =
ExponentialSmoothing(TES_train['Sparkling'].astype('double'),trend='additive'
,seasonal='multiplicative',freq='MS')
model_TES_autofit = model_TES.fit()
Best parameters of models are
model_TES_autofit.params
{'smoothing_level': 0.15441886446831232,
 'smoothing_slope': 1.853254099719361e-28,
 'smoothing_seasonal': 0.37117496389879046,
 'damping_slope': nan,
 'initial_level': 1639.9993450103527,
 'initial_slope': 4.883165537787007,
 'initial_seasons': array([1.00843108, 0.96900676, 1.24176108, 1.13205788,
0.93987123,
        0.93813452, 1.22455213, 1.54419491, 1.27333552, 1.63190518,
        2.48262582, 3.11822969]),
 'use_boxcox': False,
 'lamda': None,
 'remove_bias': False}

For Alpha: 0.15441886446831232,Beta: 1.853254099719361e-28, and Gamma:
0.37117496389879046, Triple Exponential Smoothing Model forecast on the
Training Data,  RMSE is 124876.800 MAPE is 10.18

For Alpha: 0.15441886446831232,Beta: 1.853254099719361e-28, and Gamma:
0.37117496389879046, Triple Exponential Smoothing Model forecast on the
Training Data,  RMSE is 147578.274 MAPE is 11.94
```



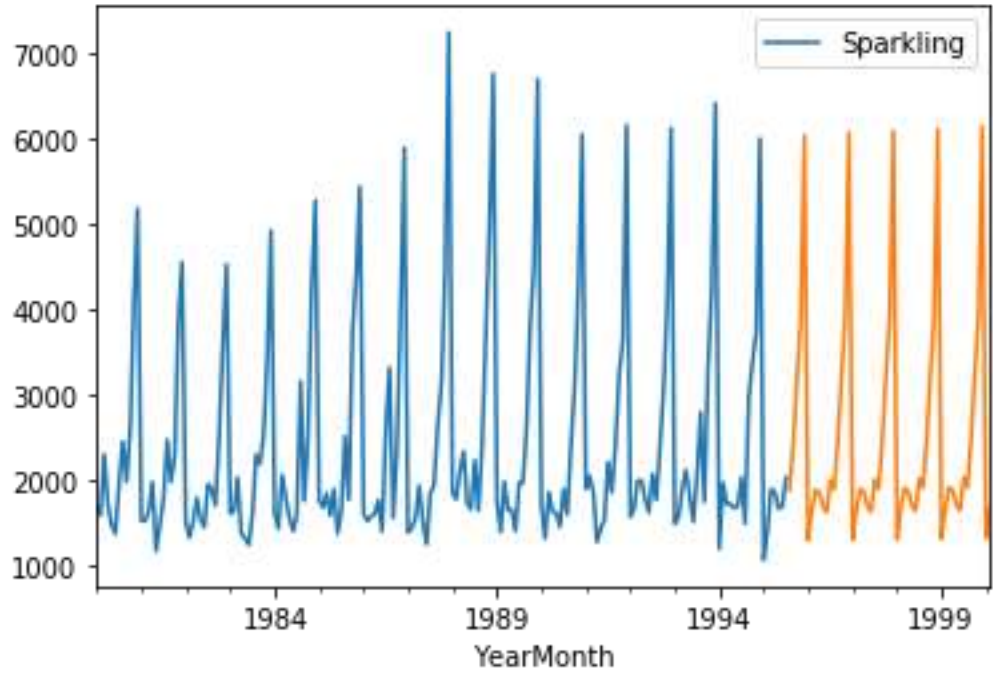
Now trying different values of alpha,beta and gama

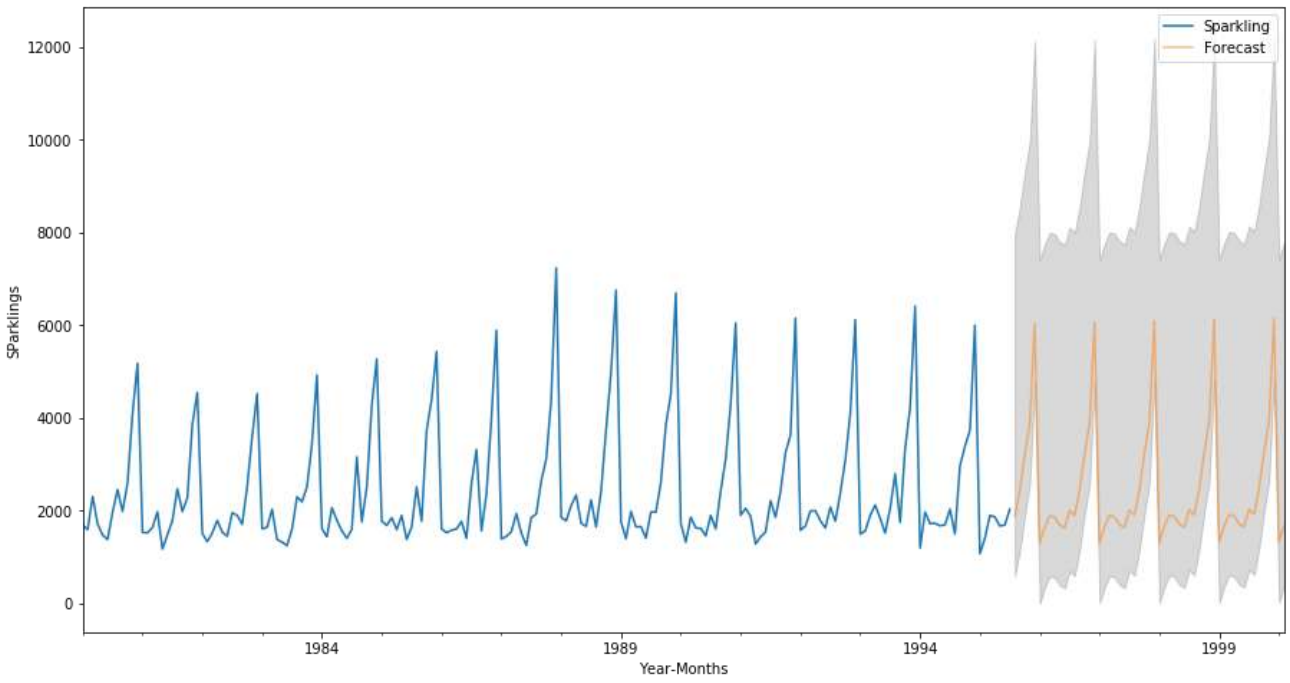
	Alpha Values	Beta Values	Gamma Values	Train RMSE	Train MAPE	Test RMSE	Test MAPE
0	0.3	0.3	0.3	163631	11.21	154281	13.51
8	0.3	0.4	0.3	180478.9	11.78	168801.5	12.9
65	0.4	0.3	0.4	189706.9	11.76	177585.7	14.84
296	0.7	0.8	0.3	490445	18.33	268519.6	18.86
130	0.5	0.3	0.5	248243	13.87	293954.3	18.23

	Test RMSE	Test MAPE
RegressionOnTime	1.93E+06	50.15
NaiveModel	1.49E+07	152.9
SimpleAverageModel	1.63E+06	38.9
2pointTrailingMovingAverage	6.62E+05	19.7
4pointTrailingMovingAverage	1.34E+06	35.96
6pointTrailingMovingAverage	1.65E+06	43.86
9pointTrailingMovingAverage	1.81E+06	46.86
Alpha=1,SimpleExponentialSmoothing	1.63E+06	38.9
Alpha=0.3,SimpleExponentialSmoothing	3.75E+06	75.66
Alpha=0.4,SimpleExponentialSmoothing	5.34E+06	91.55
Alpha=0.3,Beta=0.3,DoubleExponentialSmoothing	3.33E+08	675.3
Alpha=0.154,Beta=1.85e-28,Gamma=0.37117,TripleExponentialSmoothing	1.48E+05	11.94
Alpha=0.3,Beta=0.4,Gamma=0.3,TripleExponentialSmoothing	8.58E+07	302.8

We see that the best model is the Triple Exponential Smoothing with multiplicative seasonality with the parameters Alpha=0.154,Beta=1.85e28,Gamma=0.37117,TripleExponentialSmoothing

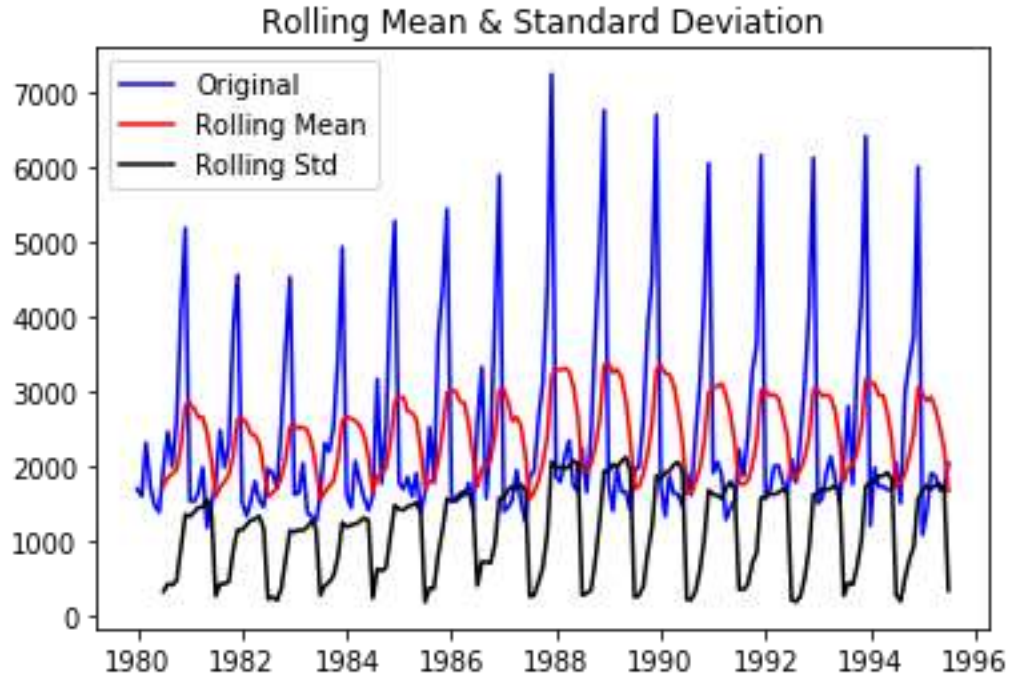
RMSE: 125237.54411586224
MAPE: 10.22





6.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.



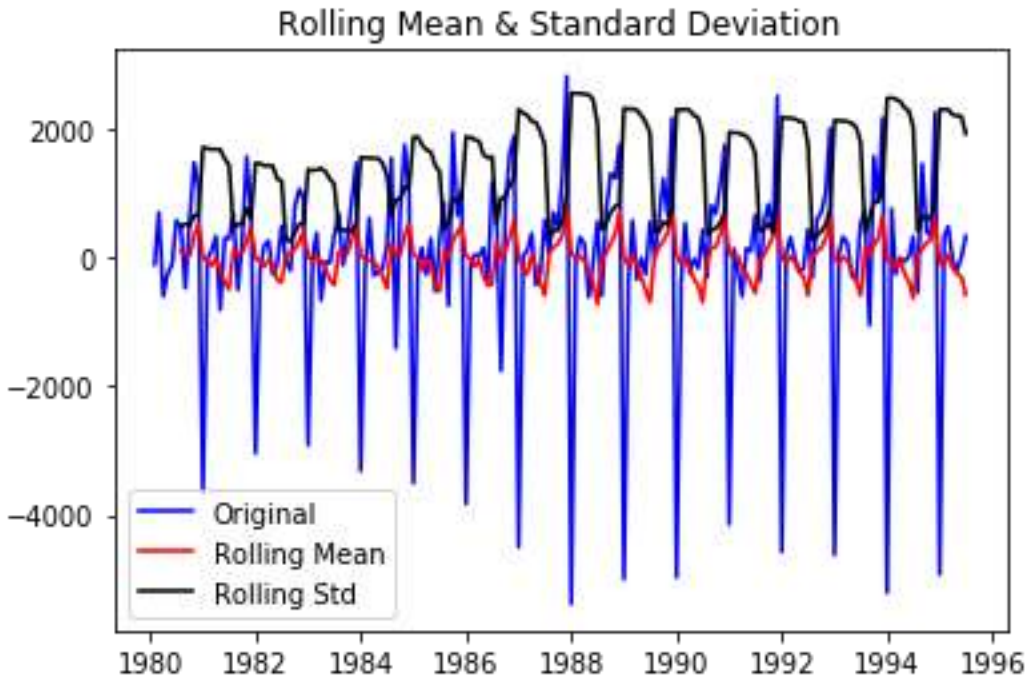

```
Results of Dickey-Fuller Test:
Test Statistic                  -1.360497
p-value                          0.601061
#Lags Used                      11.000000
Number of Observations Used    175.000000
Critical Value (1%)             -3.468280
Critical Value (5%)             -2.878202
Critical Value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

We see that at 5% significant level the Time Series is non-stationary.

Let us take a difference of order 1 and check whether the Time Series is stationary or not.

```
 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
```

We see that at α = 0.05 the Time Series is indeed 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.

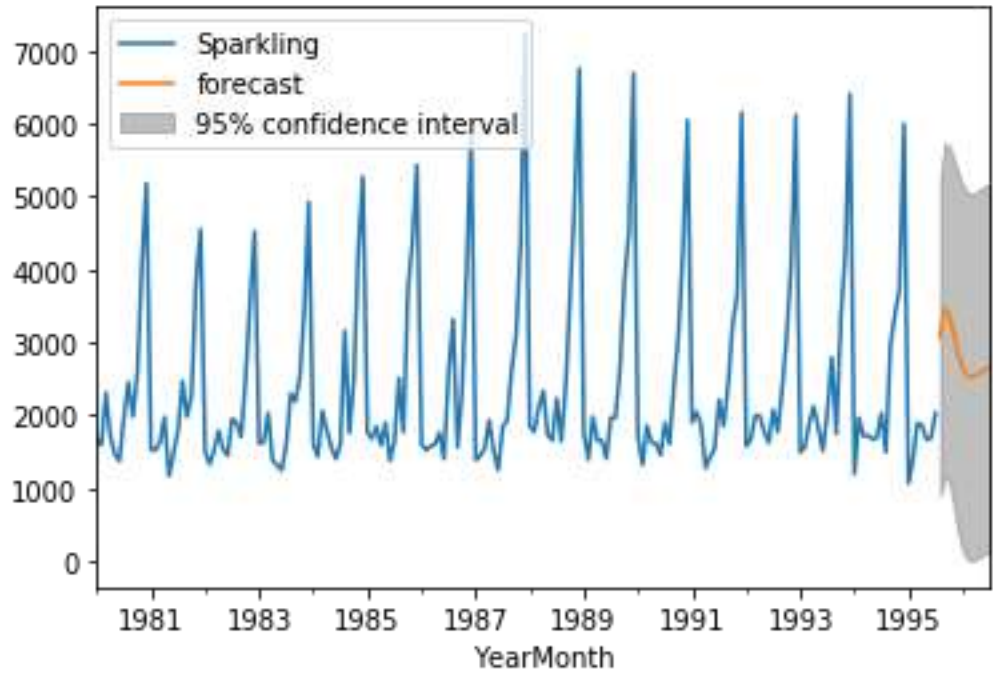
Ans:

1.ARIMA model

By trying different combinatiosn of pqd

	param	AIC
8	(2, 1, 2)	2210.616
7	(2, 1, 1)	2232.36
2	(0, 1, 2)	2232.783
5	(1, 1, 2)	2233.598
4	(1, 1, 1)	2235.014
6	(2, 1, 0)	2262.036
1	(0, 1, 1)	2264.906
3	(1, 1, 0)	2268.528
0	(0, 1, 0)	2269.583

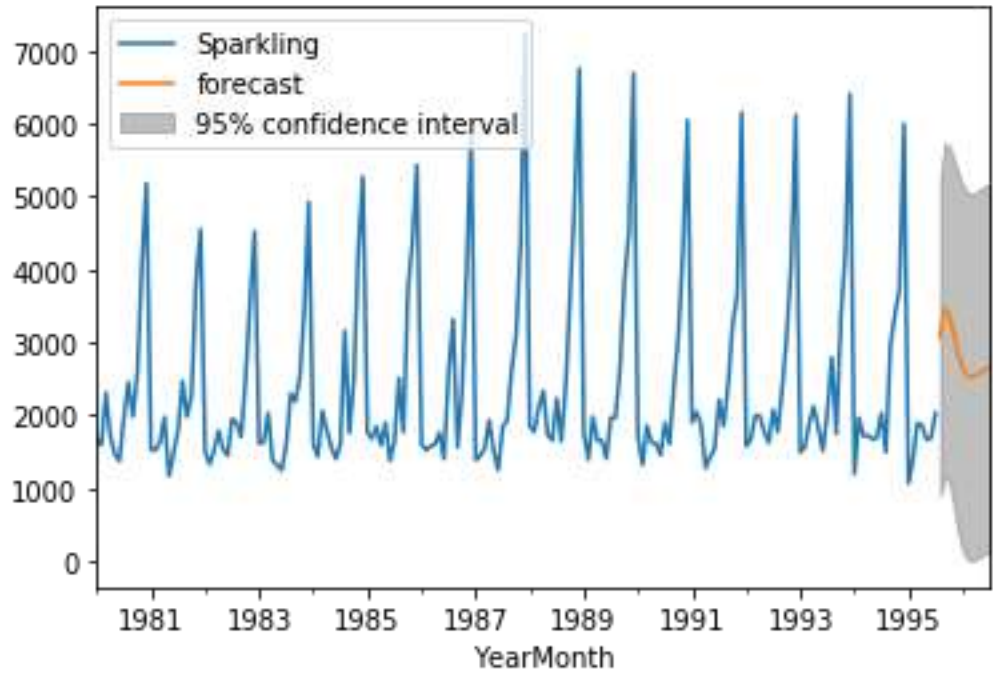
ARIMA(2,1,2) RMSE is 1.891234e+06



2.SARIMA Model

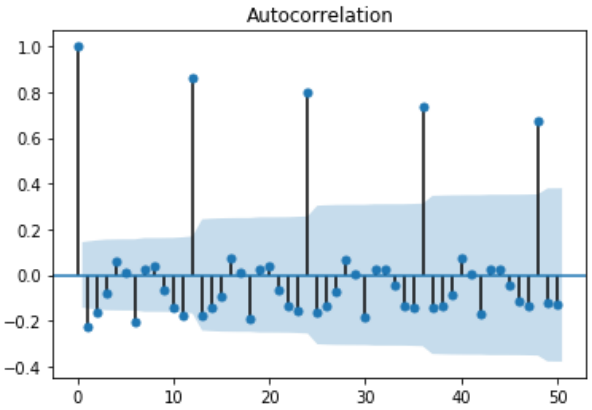
	param	seasonal	AIC
50	(1, 1, 2)	(1, 0, 2, 12)	1555.58
53	(1, 1, 2)	(2, 0, 2, 12)	1555.93
26	(0, 1, 2)	(2, 0, 2, 12)	1557.12
23	(0, 1, 2)	(1, 0, 2, 12)	1557.16
77	(2, 1, 2)	(1, 0, 2, 12)	1557.34

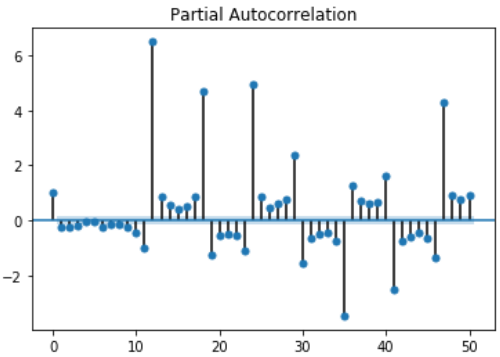
SARIMA(0, 1, 2)(2, 0, 2, 12) RMSE is 3.427605e+05



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.

Ans:





For above ACF and PACF same model of ARIMA and SARIMA can be used.

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

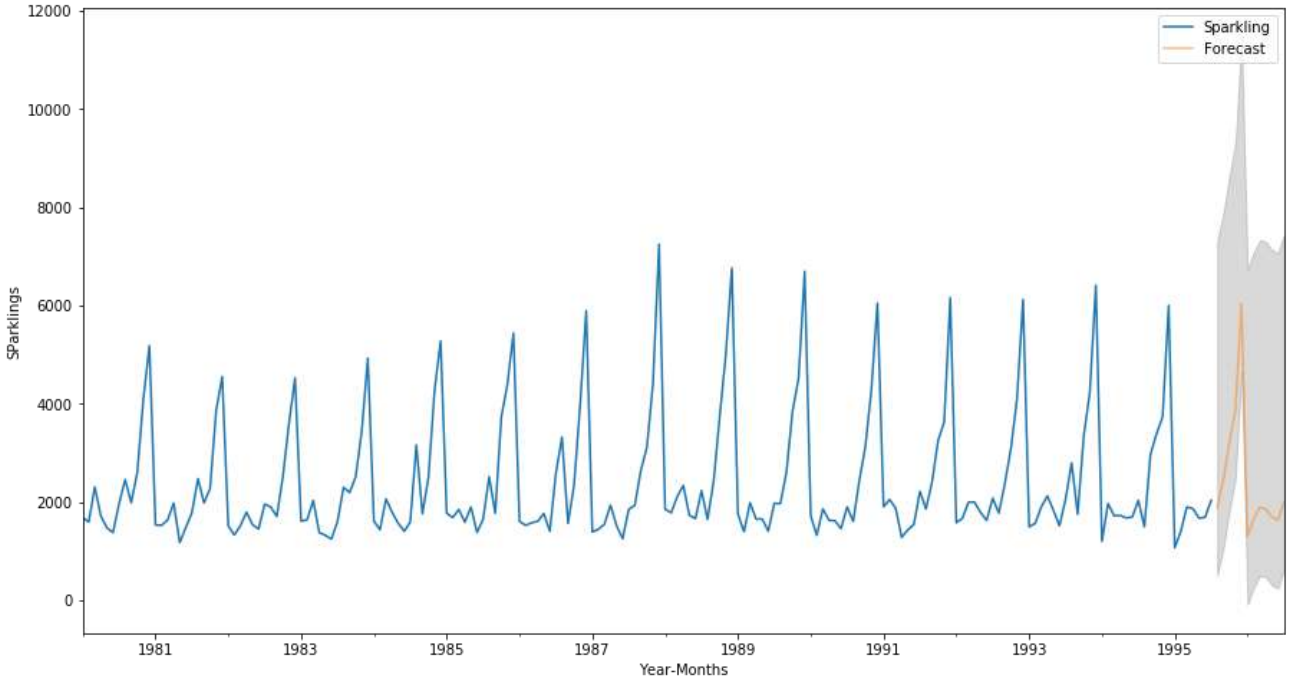
Ans:

	Test RMSE	Test MAPE
RegressionOnTime	1.93E+06	50.15
NaiveModel	1.49E+07	152.9
SimpleAverageModel	1.63E+06	38.9
2pointTrailingMovingAverage	6.62E+05	19.7
4pointTrailingMovingAverage	1.34E+06	35.96
6pointTrailingMovingAverage	1.65E+06	43.86
9pointTrailingMovingAverage	1.81E+06	46.86
Alpha=1,SimpleExponentialSmoothing	1.63E+06	38.9
Alpha=0.3,SimpleExponentialSmoothing	3.75E+06	75.66
Alpha=0.4,SimpleExponentialSmoothing	5.34E+06	91.55
Alpha=0.3,Beta=0.3,DoubleExponentialSmoothing	3.33E+08	675.3
Alpha=0.154,Beta=1.85e-28,Gamma=0.37117,TripleExponentialSmoothing	1.48E+05	11.94
Alpha=0.3,Beta=0.4,Gamma=0.3,TripleExponentialSmoothing	8.58E+07	302.8
ARIMA(2,1,2)	1.89E+06	
SARIMA(0, 1, 2)(2, 0, 2, 12)	3.43E+05	

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

Ans: So best model is

Alpha=0.154,Beta=1.85e-28,Gamma=0.37117,TripleExponentialSmoothing with RMSE 1.48E+05



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

Ans:Model used is triple exponential smoothing.

From above graph it can be noted that maximum sale can go near to 12000 and minimum can go upto 100 with 95% confidence interval