Project: Time Series Forecasting

20.06.2021

—

Peeyush Vardhan
Mobile: +91-9515456727
Email: peeyushvrdhn@gmail.com

Case Study: Rose Wine Prediction

Problem Statement:

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.

Data set: Rose.csv

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

Solution:

- Reading the data:

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

YearMonth	Rose
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

- Checking the timestamp:

```
1 df.index
```

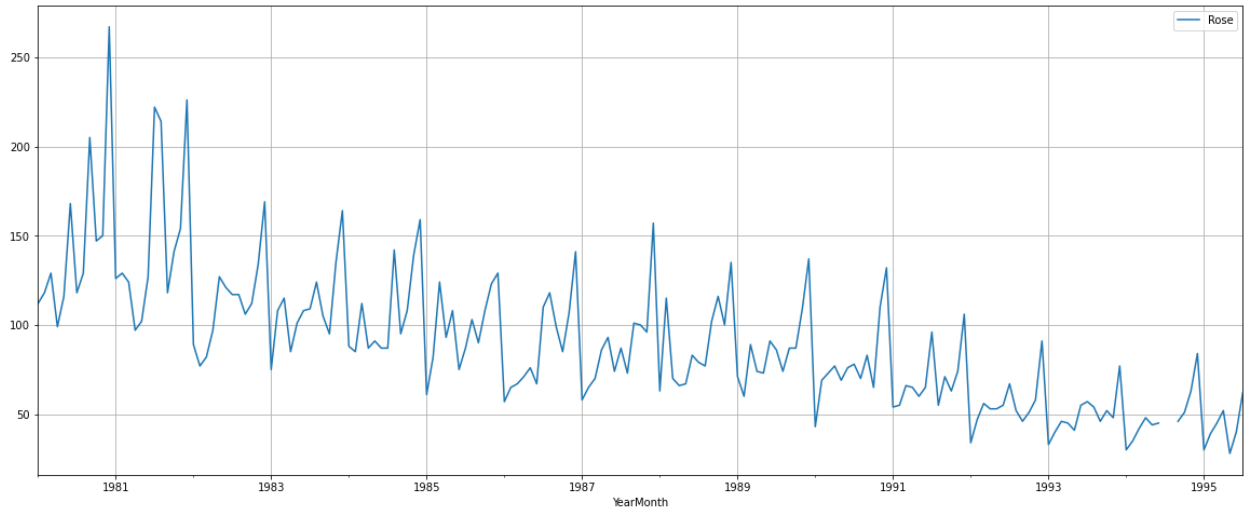
```
DatetimeIndex(['1980-01-01', '1980-02-01', '1980-03-01', '1980-04-01',
               '1980-05-01', '1980-06-01', '1980-07-01', '1980-08-01',
               '1980-09-01', '1980-10-01',
               ...
               '1994-10-01', '1994-11-01', '1994-12-01', '1995-01-01',
               '1995-02-01', '1995-03-01', '1995-04-01', '1995-05-01',
               '1995-06-01', '1995-07-01'],
              dtype='datetime64[ns]', name='YearMonth', length=187, freq=None)
```

- Checking data information:
  - *The number of rows are 187*
  - *The number of columns are 1*

```
1 df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
DatetimeIndex: 187 entries, 1980-01-01 to 1995-07-01
Data columns (total 1 columns):
 #   Column  Non-Null Count  Dtype  
---  ------  --------------  -----  
 0   Rose    185 non-null    float64
dtypes: float64(1)
memory usage: 2.9 KB
```

- Plotting the Time Series to understand the behaviour of the data:



- Insights:
  - The data seems multiplicative in nature and it has seasonality.
  - Over the years, the Rose sales has been decreasing and it was the best performing in 1981

- Between the years 1982 and 1984 the sales have seemed to be constant, however after 1985 there was decline observed.

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

Solution:

- Checking the basic statistical details:

```
1 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

- Checking missing values in the data set:
  - *There are 2 missing rows in the dataframe*

```
1 #finding number of missing values in the data set
2
3 df.isnull().sum()
```

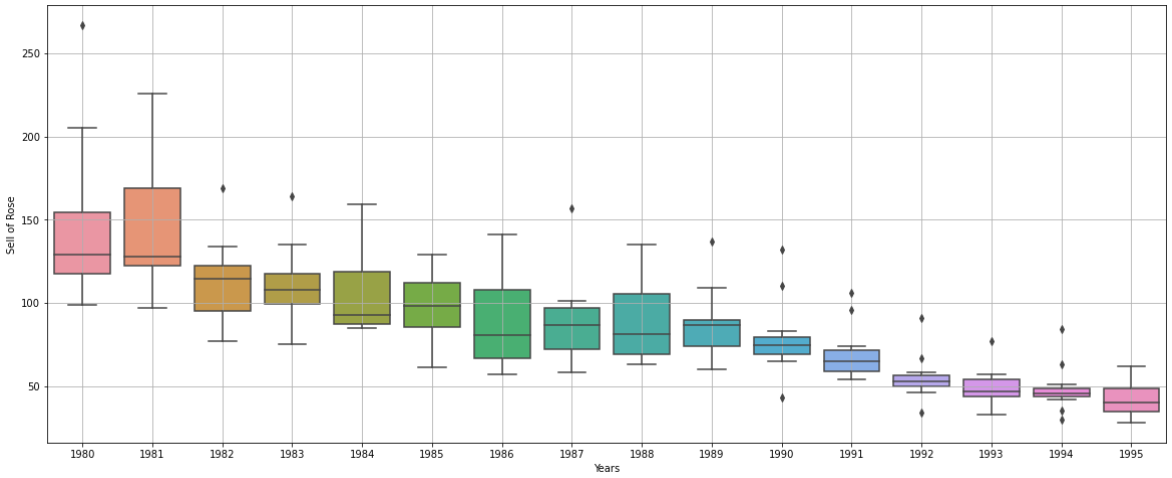
```
Rose    2
dtype: int64
```

  - Since it's a time series data - we will use the Interpolation process of finding a value between two points on a line or a curve.

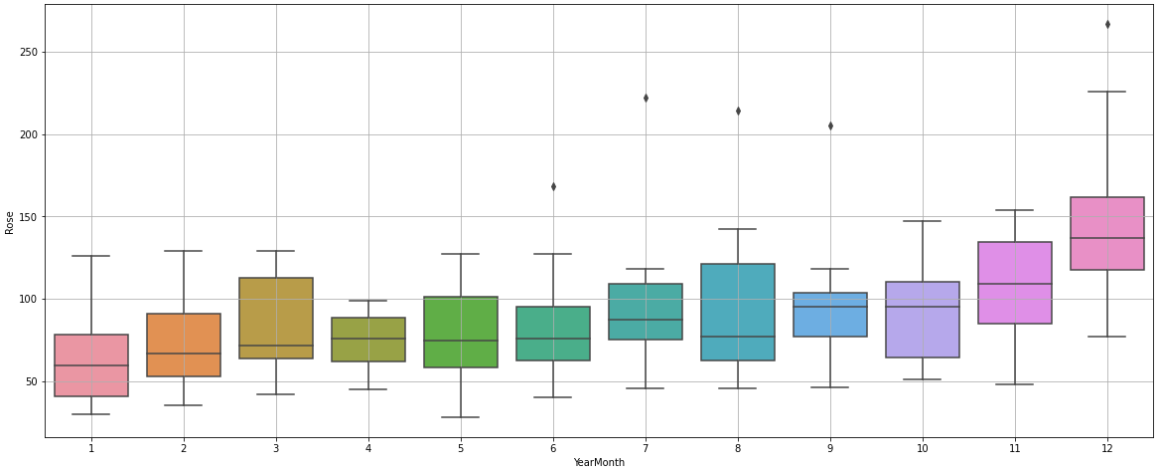
- Below is the data after the interpolation process:

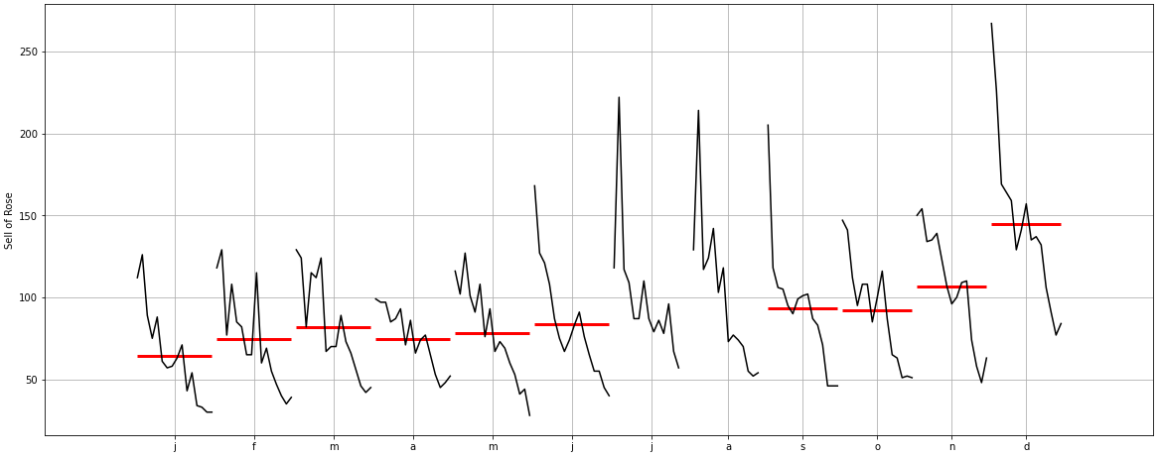
YearMonth	Rose
1994-01-01	30.000000
1994-02-01	35.000000
1994-03-01	42.000000
1994-04-01	48.000000
1994-05-01	44.000000
1994-06-01	45.000000
1994-07-01	45.333333
1994-08-01	45.666667
1994-09-01	46.000000
1994-10-01	51.000000
1994-11-01	63.000000
1994-12-01	84.000000

- Plotting a boxplot to understand the spread of accidents across different years and within different months across years:
  - Yearly Boxplot



- Monthly Boxplot

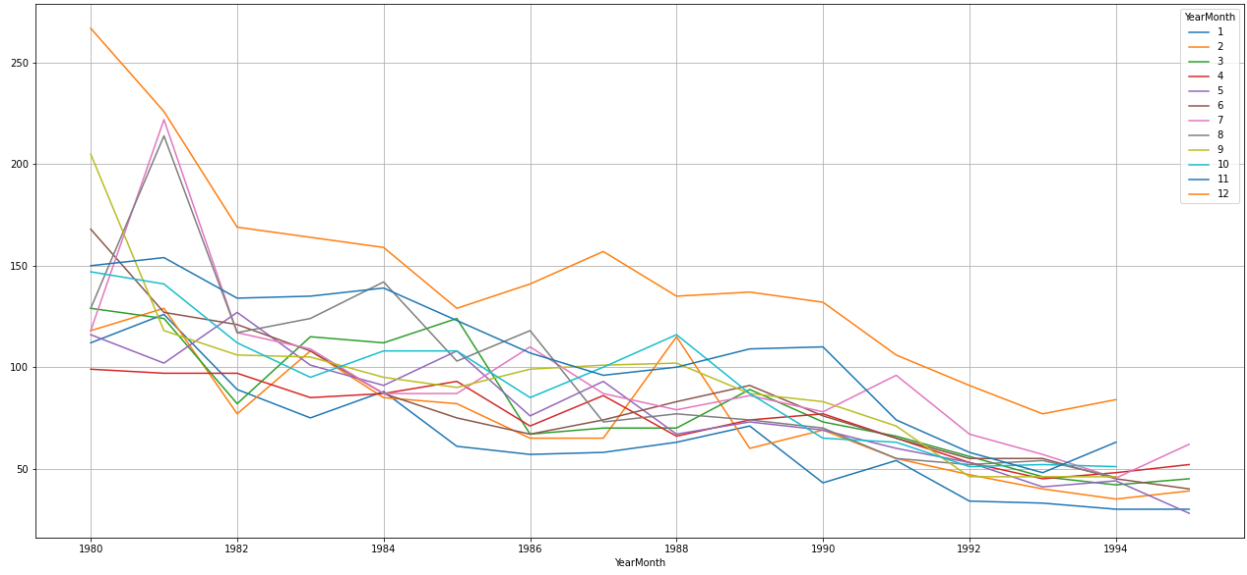




- Looking at Monthly Sale Across Different Years:

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

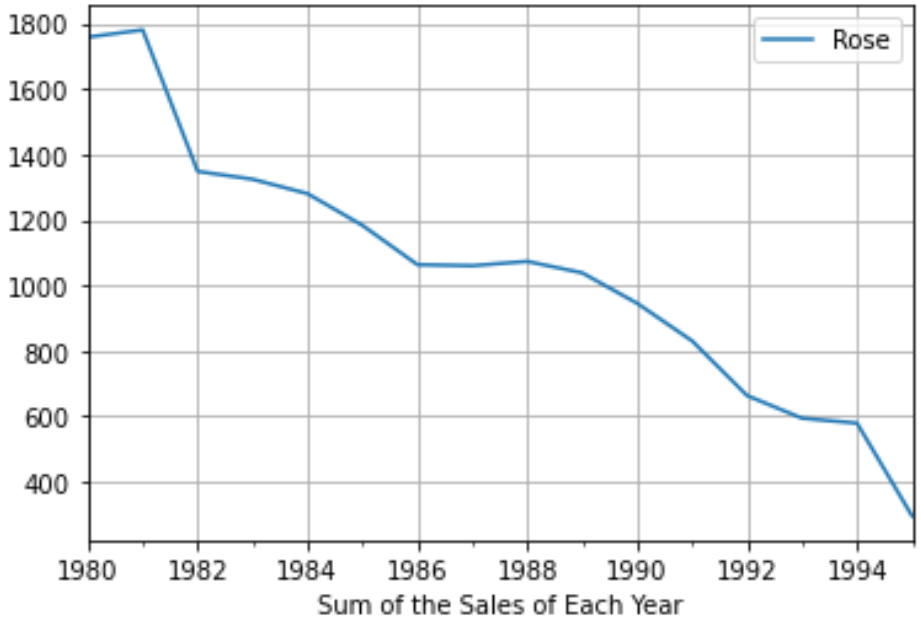
Graphical Representation:



- Details of the annual sale:

YearMonth	Rose
1980-12-31	1758.0
1981-12-31	1780.0
1982-12-31	1348.0
1983-12-31	1324.0
1984-12-31	1280.0

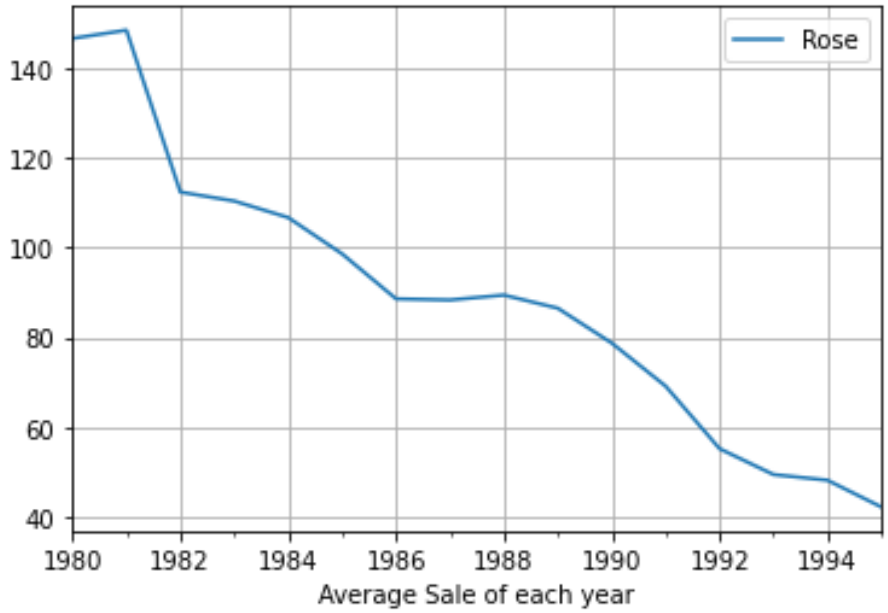
Graphical Representation:



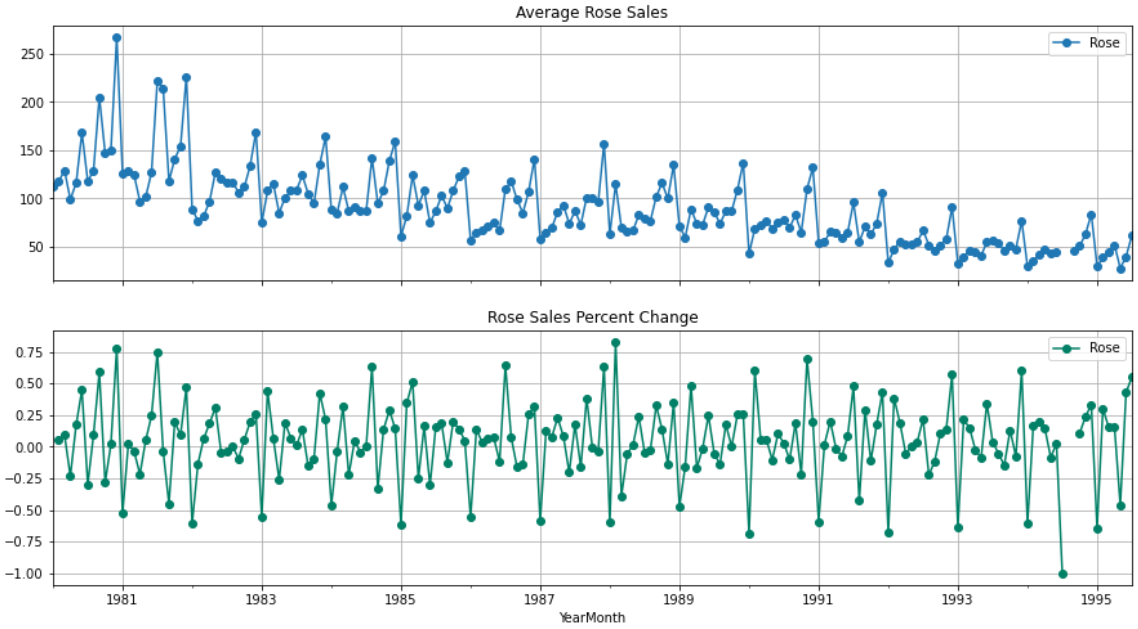
- Details of average the annual sale:

YearMonth	Rose
1980-12-31	146.500000
1981-12-31	148.333333
1982-12-31	112.333333
1983-12-31	110.333333
1984-12-31	106.666667

Graphical Representation:

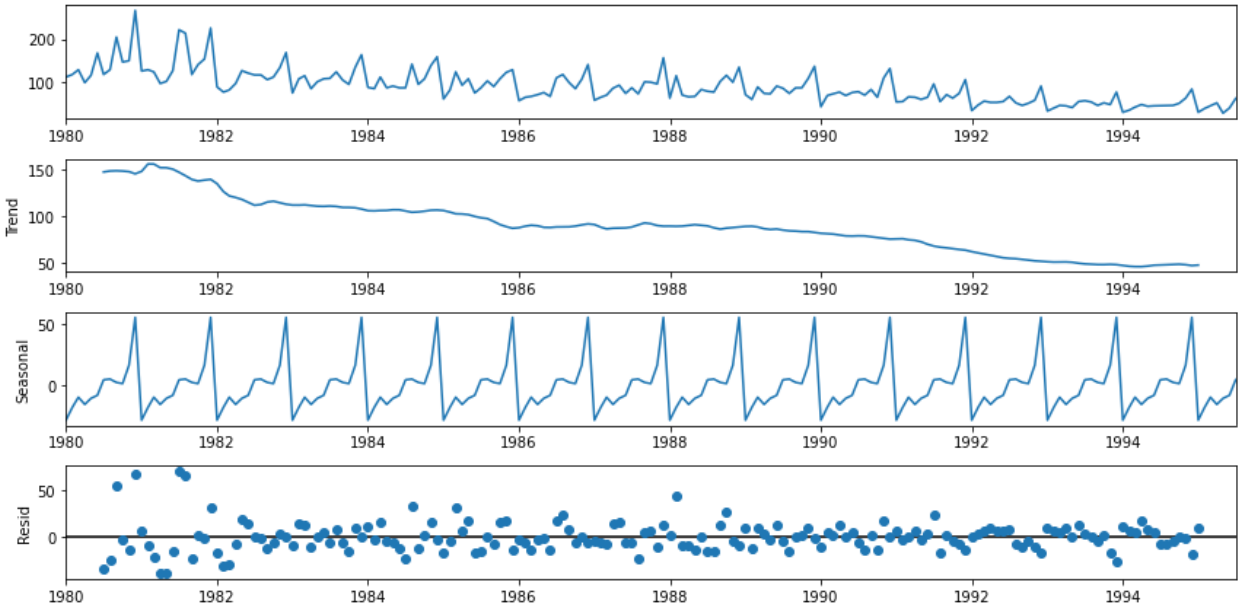


- Average Sparkling Sales per month and the month on month percentage change of Sparkling Sales:

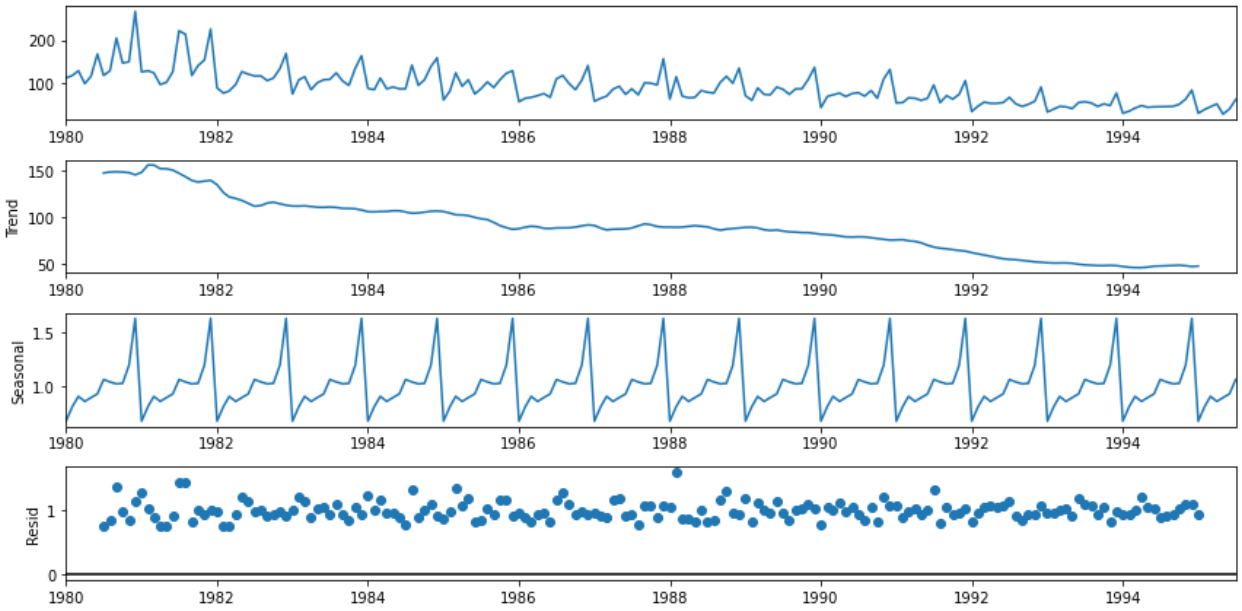


- Representation of decomposition of data for Time series analysis. This will be represented in 2 types:
  - Additive decomposition
  - Multiplicative decomposition

Additive decomposition:



Multiplicative decomposition:



- **Insights from decomposition of data:**
  - It is observed in additive decomposition that seasonality is present on a yearly basis.
  - There is a huge effect being shown in error terms as well.
  - On the other hand, in multiplicative decomposition the effect of error is reduced.

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

Solution:

- Splitting the data into train and test shape:

```
train=df2[df2.index.year < 1991]
test=df2[df2.index.year >= 1991]
print("Shape of Training Data is", train.shape)
print("Shape of Test Data is", test.shape)
```

```
Shape of Training Data is (132, 1)
Shape of Test Data is (55, 1)
```

- Bottom data from training set:

```
display(train.tail())
```

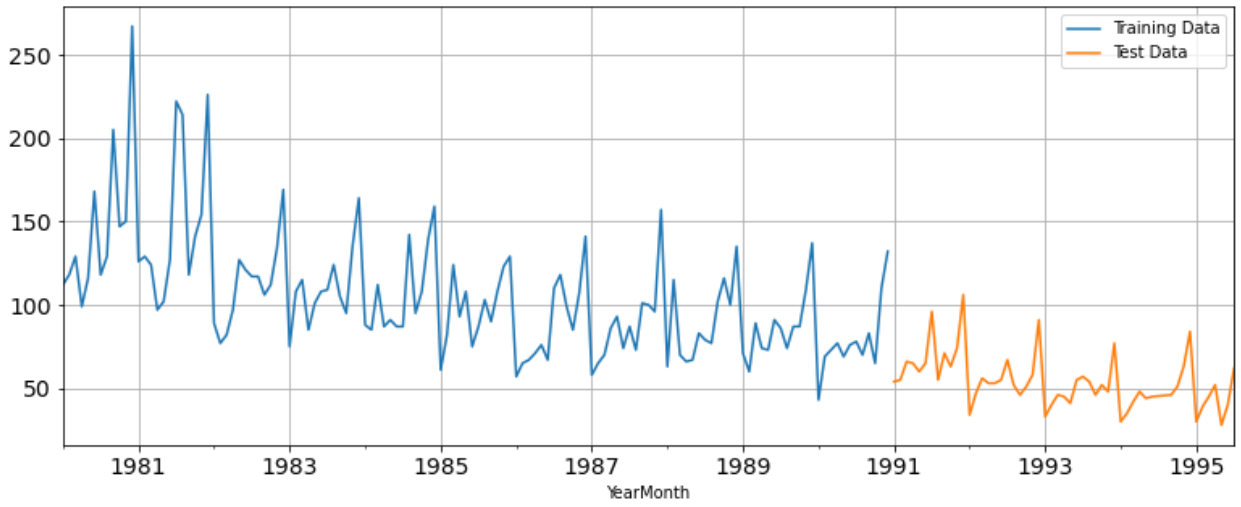
YearMonth	Rose
1990-08-01	70.0
1990-09-01	83.0
1990-10-01	65.0
1990-11-01	110.0
1990-12-01	132.0

- Top data from test set:

```
display(test.head())
```

YearMonth	Rose
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

- Plotting the training and test data:



Question 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 and simple average models. Should also be built on the training data and check the performance on the test data using RMSE.

Solution:

Please refer to the Jupyter Notebook submitted to look into the code.

4.1. Building Regression Model:

- Training Time instance

```
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]
```

- Training data head:

YearMonth	Rose	time
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

- Training data tail:

YearMonth	Rose	time
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

- Test data head:

YearMonth	Rose	time
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

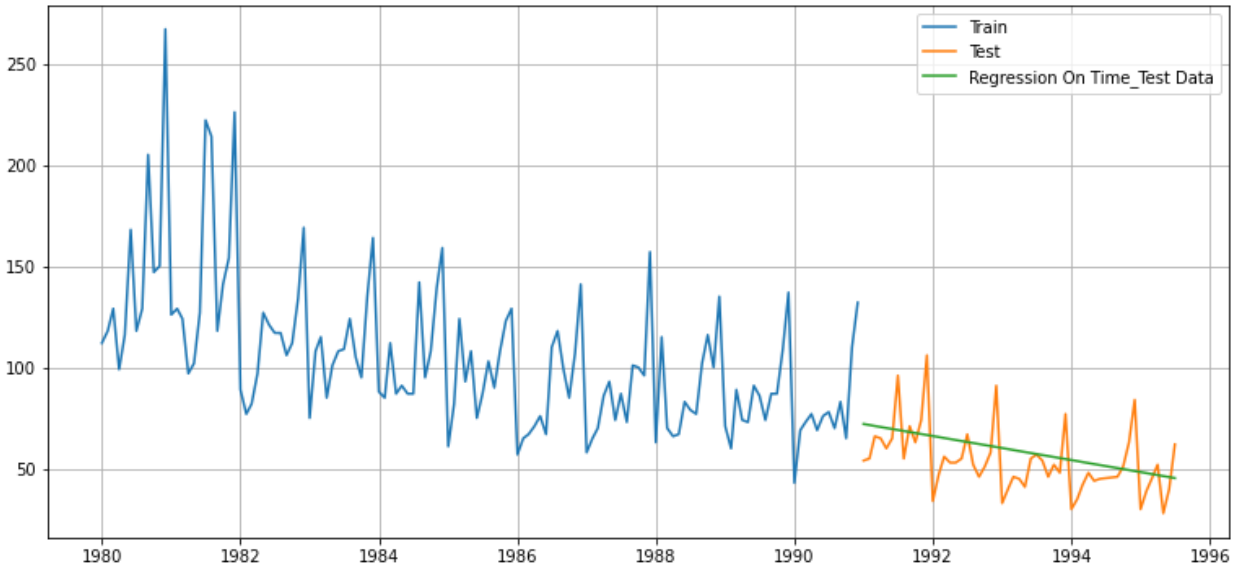
- Test data tail:

YearMonth	Rose	time
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

- Plotting Linear Regression Model:

```
import sklearn
print(sklearn.__version__)
from sklearn import linear_model
from sklearn.linear_model import LinearRegression
lr = LinearRegression()
```

```
0.24.1
```



- RMSE model evaluation:

```
from sklearn import metrics
```

```
#LR MODEL EVALUATION

rmse_lr = metrics.mean_squared_error(test['Rose'], lr_predict, squared=False)
print("RMSE of LR is %3.3f" %(rmse_lr))
```

```
RMSE of LR is 15.269
```

```
resultsDf = pd.DataFrame({'Test RMSE': [rmse_lr]},index=['RegressionOnTime'])
resultsDf
```

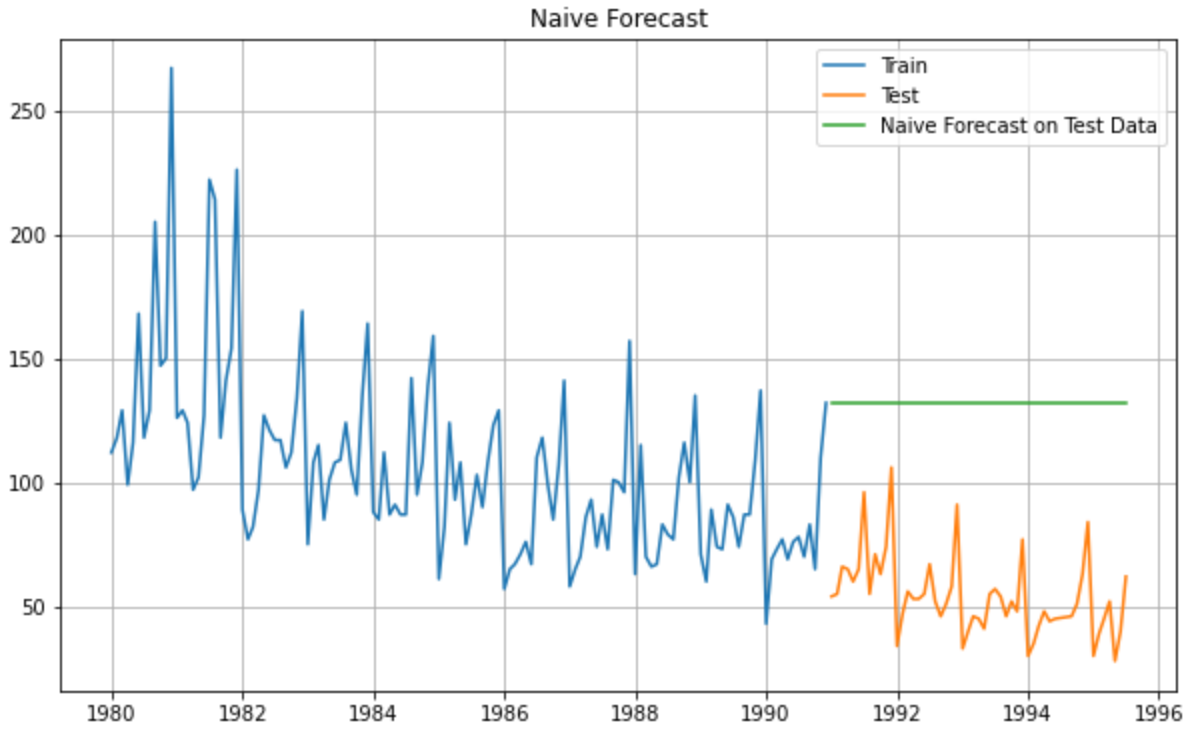
	Test RMSE
RegressionOnTime	15.268955

4.2. Building Naïve Forecast Model:

- Test data head:

```
YearMonth
1991-01-01    132.0
1991-02-01    132.0
1991-03-01    132.0
1991-04-01    132.0
1991-05-01    132.0
Name: naive, dtype: float64
```

- Train test data plot:



- Naïve Forecast Model Evaluation:
  - RMSE of NF is 79.719

```
#NF MODEL EVALUATION

rmse_nf = metrics.mean_squared_error(test['Rose'],NM_test['naive'],squared=False)
print("RMSE of NF is %3.3f" %(rmse_nf))
```

```
RMSE of NF is 79.719
```

```
resultsDf_NF = pd.DataFrame({'Test RMSE': [rmse_nf]},index=['NaiveModel'])

resultsDf = pd.concat([resultsDf, resultsDf_NF])
resultsDf
```

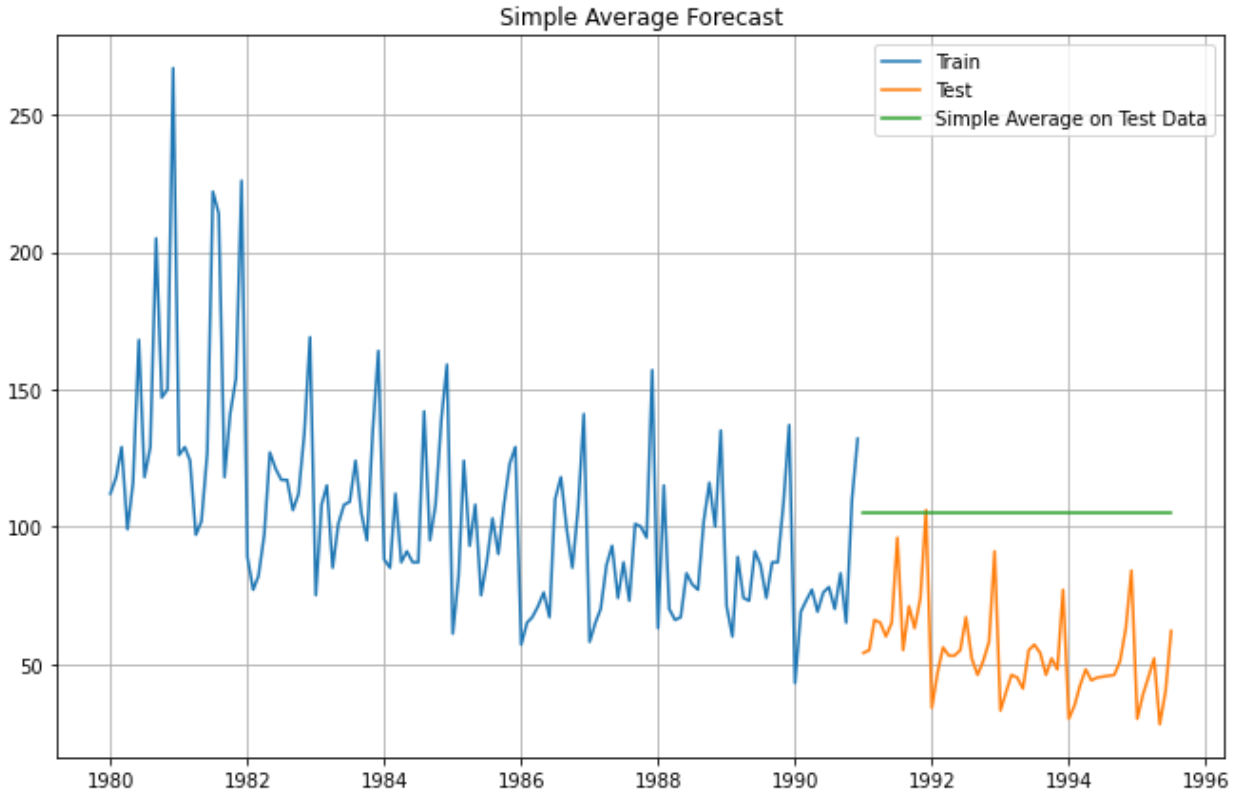
	Test RMSE
RegressionOnTime	15.268955
NaiveModel	79.718773

4.3. Building Simple Average Model:

- Test data head:

YearMonth	Rose	mean_forecast
1991-01-01	54.0	104.939394
1991-02-01	55.0	104.939394
1991-03-01	66.0	104.939394
1991-04-01	65.0	104.939394
1991-05-01	60.0	104.939394

- Train test data plot:



- Simple Average Model Evaluation:
    - RMSE of Simple Average Model is 53.461

```
#SA MODEL EVALUATION

rmse_sa = metrics.mean_squared_error(test['Rose'],SA_test['mean_forecast'],squared=False)
print("RMSE of SA isis %3.3f" %(rmse_sa))
```

```
RMSE of SA isis 53.461
```

```
resultsDf_SA = pd.DataFrame({'Test RMSE': [rmse_sa]},index=['SimpleAverageModel'])

resultsDf = pd.concat([resultsDf, resultsDf_SA])
resultsDf
```

	Test RMSE
RegressionOnTime	15.268955
NaiveModel	79.718773
SimpleAverageModel	53.460570

4.4. Single Exponential Smoothing Model (SES):

```
SES_test = test.copy()
SES_train = train.copy()
```

```
model_SES = SimpleExpSmoothing(SES_train['Rose'])
model_SES_autofit = model_SES.fit(optimized=True)
model_SES_autofit.params
```

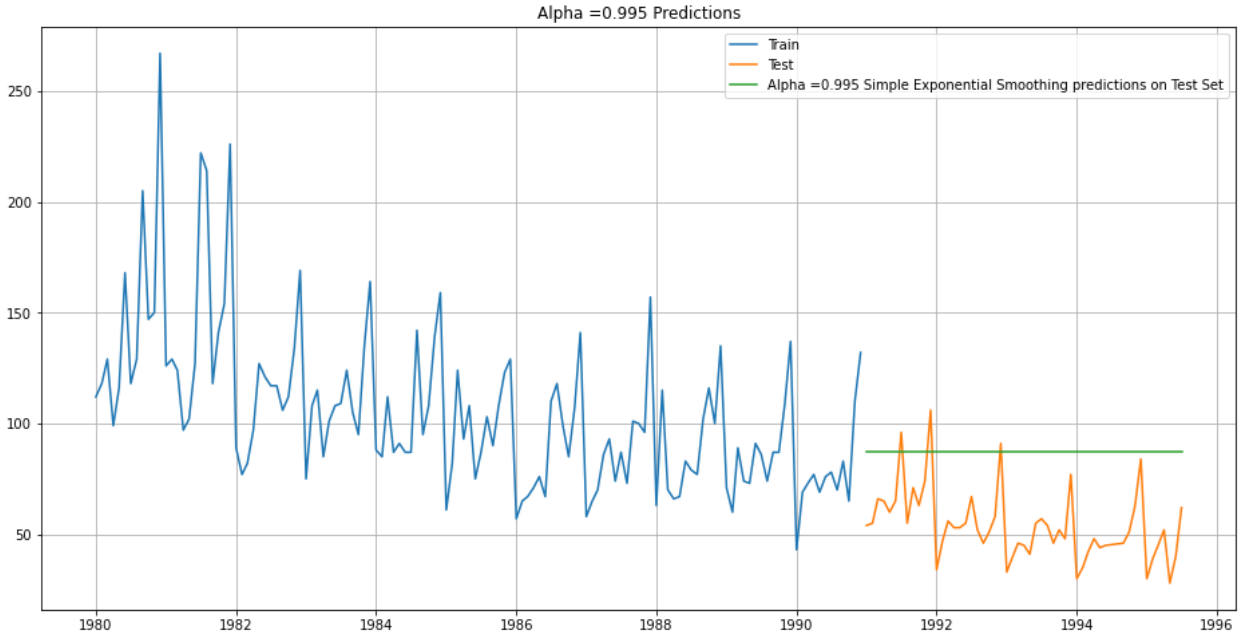
```
C:\Users\hp\anaconda3\lib\site-packages\statsmodels\tsa\base\tsa_model.py:524: ValueWarning: No frequency information was provi
ded, so inferred frequency MS will be used.
  warnings.warn('No frequency information was'
C:\Users\hp\anaconda3\lib\site-packages\statsmodels\tsa\holtwinters\model.py:427: FutureWarning: After 0.13 initialization must
be handled at model creation
  warnings.warn(
```

```
{'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}
```

- Test data head:

YearMonth	Rose	predict
1991-01-01	54.0	87.104983
1991-02-01	55.0	87.104983
1991-03-01	66.0	87.104983
1991-04-01	65.0	87.104983
1991-05-01	60.0	87.104983

- Plotting on both the Training and Test data:



- Single Exponential Smoothing Model Evaluation:

```
rmse_ses = metrics.mean_squared_error(SES_test['Rose'],SES_test['predict'],squared=False)
print("For Alpha =0.995, RMSE of SES is %3.3f" %(rmse_ses))
```

```
For Alpha =0.995, RMSE of SES is 36.796
```

```
resultsDf_SES = pd.DataFrame({'Test RMSE': [rmse_ses]},index=['Alpha=0.995,SimpleExponentialSmoothing'])
resultsDf = pd.concat([resultsDf, resultsDf_SES])
resultsDf
```

	Test RMSE
RegressionOnTime	15.268955
NaiveModel	79.718773
SimpleAverageModel	53.460570
Alpha=0.995,SimpleExponentialSmoothing	36.796227

4.5. Building Double Exponential Smoothing - Holt's Model

```
DES_test = test.copy()
DES_train = train.copy()
model_DES = Holt(DES_train['Rose'])
```

```
C:\Users\hp\anaconda3\lib\site-packages\statsmodels\tsa\base\tsa_model.py:524: ValueWarning: No frequency inf
ded, so inferred frequency MS will be used.
  warnings.warn('No frequency information was'
C:\Users\hp\anaconda3\lib\site-packages\statsmodels\tsa\holtwinters\model.py:427: FutureWarning: After 0.13 i
be handled at model creation
  warnings.warn(
```

```
resultsDf_DES = pd.DataFrame({'Alpha Values':[],'Beta Values':[],'Train RMSE':[],'Test RMSE': []})
resultsDf_DES
```

Alpha Values	Beta Values	Train RMSE	Test RMSE

- Consolidation of Double Exponential Smoothing:

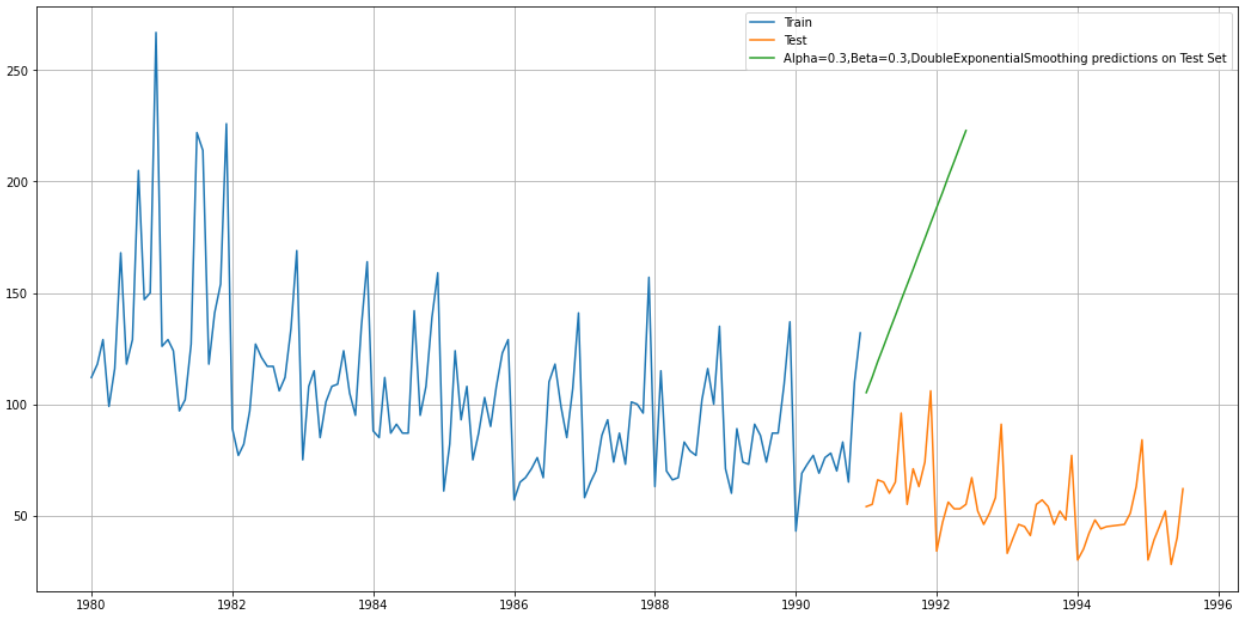
	Alpha Values	Beta Values	Train RMSE	Test RMSE
0	0.3	0.3	1292.041812	1292.041812
1	0.3	0.4	1398.254302	1398.254302
2	0.3	0.5	1512.301839	1512.301839
3	0.3	0.6	1622.532350	1622.532350
4	0.3	0.7	1714.251300	1714.251300
...	...	...	...	...
59	1.0	0.6	2686.515763	2686.515763
60	1.0	0.7	2969.927275	2969.927275
61	1.0	0.8	3290.844073	3290.844073
62	1.0	0.9	3657.142037	3657.142037
63	1.0	1.0	4079.818182	4079.818182

64 rows × 4 columns

- Double Exponential Smoothing test set top data:

	Alpha Values	Beta Values	Train RMSE	Test RMSE
0	0.3	0.3	1292.041812	1292.041812
8	0.4	0.3	1350.498017	1350.498017
1	0.3	0.4	1398.254302	1398.254302
16	0.5	0.3	1401.253029	1401.253029
9	0.4	0.4	1444.351306	1444.351306

- Plotting on both the Training and Test data (Alpha = 0.3, Beta = 0.3)



- Results of Double Exponential Smoothing:

	Test RMSE
RegressionOnTime	15.268955
NaiveModel	79.718773
SimpleAverageModel	53.460570
Alpha=0.995,SimpleExponentialSmoothing	36.796227
Alpha=0.3,Beta=0.3,DoubleExponentialSmoothing	1292.041812

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

```
TES_test = test.copy()
TES_train = train.copy()
```

```
model_TES = ExponentialSmoothing(TES_train['Rose'],trend='additive',seasonal='multiplicative',freq='MS')
#MS- Month Start
```

```
C:\Users\hp\anaconda3\lib\site-packages\statsmodels\tsa\holtwinters\model.py:427: FutureWarning: After 0.13 initiali
be handled at model creation
  warnings.warn(
```

```
model_TES_autofit = model_TES.fit()
```

- Model Parameters:

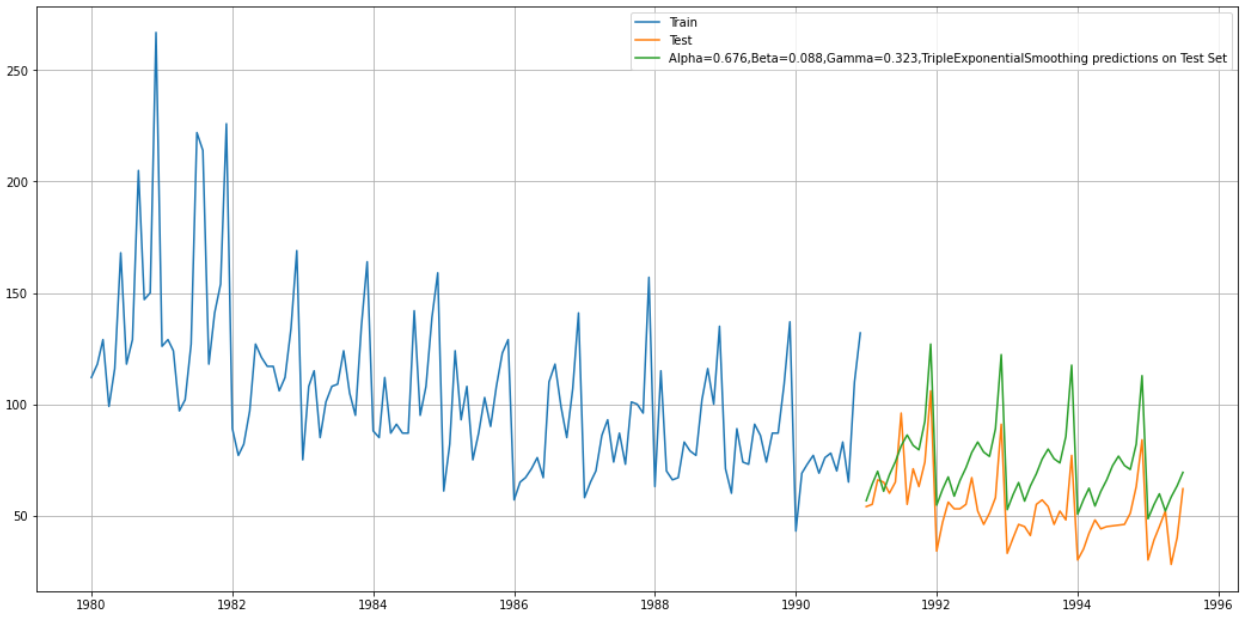
```
1 model_TES_autofit.params
```

```
{'smoothing_level': 0.06569374607191865,
 'smoothing_trend': 0.05192938504457338,
 'smoothing_seasonal': 3.879136202038614e-06,
 'damping_trend': nan,
 'initial_level': 54.10985491750761,
 'initial_trend': -0.33471965714896845,
 'initial_seasons': array([2.08282313, 2.36326666, 2.58210206, 2.25702695, 2.53757493,
       2.76639991, 3.04101803, 3.23434567, 3.06747277, 3.00164124,
       3.49893806, 4.82552476]),
 'use_boxcox': False,
 'lamda': None,
 'remove_bias': False}
```

- Triple Exponential Smoothing test set top data:

YearMonth	Rose	auto_predict
1991-01-01	54.0	56.689174
1991-02-01	55.0	64.129166
1991-03-01	66.0	69.856436
1991-04-01	65.0	60.877474
1991-05-01	60.0	68.237072

- Plotting on both the Training and Test data (Alpha = 0.676, Beta = 0.088)



- Triple Exponential Smoothing (Holt - Winter's Model) Model Evaluation:
    - For Alpha=0.676,Beta=0.088,Gamma=0.323, Triple Exponential Smoothing Model forecast on the Test Data, RMSE is 21.020

```
# Triple Exponential Smoothing (Holt - Winter's Model) Model Evaluation

rmse_TES = metrics.mean_squared_error(TES_test['Rose'],TES_test['auto_predict'],squared=False)
print("For Alpha=0.676,Beta=0.088,Gamma=0.323, Triple Exponential Smoothing Model forecast on the Test Data,  RMSE is
```

```
For Alpha=0.676,Beta=0.088,Gamma=0.323, Triple Exponential Smoothing Model forecast on the Test Data,  RMSE is 21.020
```

```
resultsDf_SES = pd.DataFrame({'Test RMSE': [rmse_ses]}
                           ,index=['Alpha=0.676,Beta=0.088,Gamma=0.323,TripleExponentialSmoothing'])

resultsDf = pd.concat([resultsDf, resultsDf_SES])
resultsDf
```

	Test RMSE
RegressionOnTime	15.268955
NaiveModel	79.718773
SimpleAverageModel	53.460570
Alpha=0.995,SimpleExponentialSmoothing	36.796227
Alpha=0.3,Beta=0.3,DoubleExponentialSmoothing	1292.041812
Alpha=0.676,Beta=0.088,Gamma=0.323,TripleExponentialSmoothing	36.796227
Alpha=0.676,Beta=0.088,Gamma=0.323,TripleExponentialSmoothing	36.796227
Alpha=0.676,Beta=0.088,Gamma=0.323,TripleExponentialSmoothing	36.796227

Question 5: Check for the stationarity of the data on which the model is being built 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.

Solution:

Please refer to the Jupyter Notebook submitted to look into the code.

- Steps Involved:
    - *Determining rolling statistics*
    - *Plot rolling statistics*
    - *Perform Dickey-Fuller test*

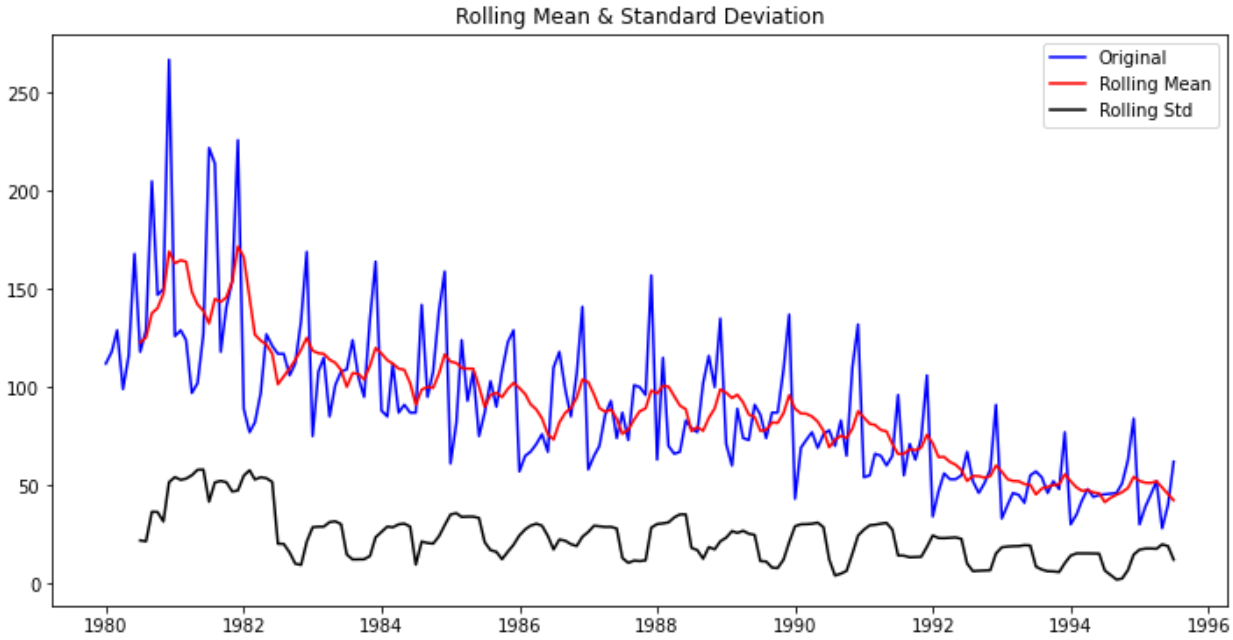
```python
def test_stationarity(timeseries):

    #Determing rolling statistics
    rolmean = timeseries.rolling(window=7).mean() #determining the rolling mean
    rolstd = timeseries.rolling(window=7).std()   #determining the rolling standard deviation

    #Plot rolling statistics:
    orig = plt.plot(timeseries, color='blue',label='Original')
    mean = plt.plot(rolmean, color='red', label='Rolling Mean')
    std = plt.plot(rolstd, color='black', label = 'Rolling Std')
    plt.legend(loc='best')
    plt.title('Rolling Mean & Standard Deviation')
    plt.show(block=False)

    #Perform Dickey-Fuller test:
    print ('Results of Dickey-Fuller Test:')
    dftest = adfuller(timeseries, autolag='AIC')
    dfoutput = pd.Series(dftest[0:4], index=['Test Statistic','p-value','#Lags Used','Number of Observations Used'])
    for key,value in dftest[4].items():
        dfoutput['Critical Value (%s)'%key] = value
    print (dfoutput,'\n')
```

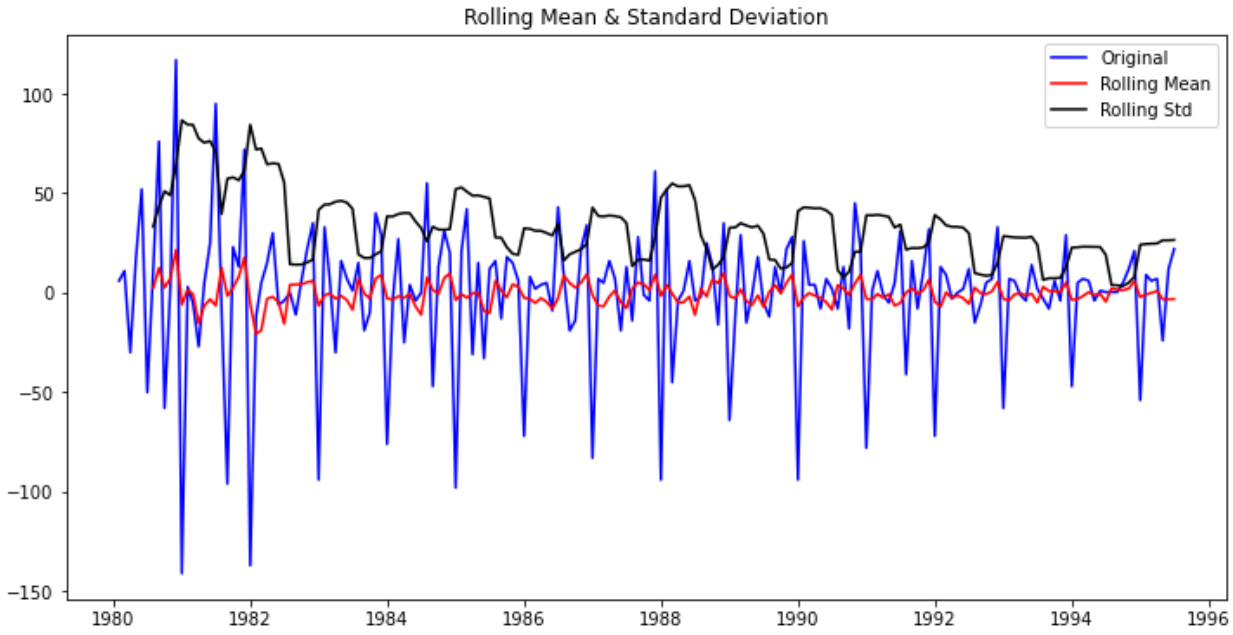
- Plot between Rolling Standard, Rolling mean and Original:



- Results of Dickey-Fuller Test:

```
Results of Dickey-Fuller Test:
Test Statistic                  -1.876699
p-value                          0.343101
#Lags Used                      13.000000
Number of Observations Used    173.000000
Critical Value (1%)             -3.468726
Critical Value (5%)             -2.878396
Critical Value (10%)            -2.575756
dtype: float64
```

- Standards:
    - p-value 0.343101, Thus the series is non-stationary.
    - Difference of order 1, to check the time series again
- Plot between Rolling Standard, Rolling mean and Original:



- Results of Dickey-Fuller Test:

```
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
```

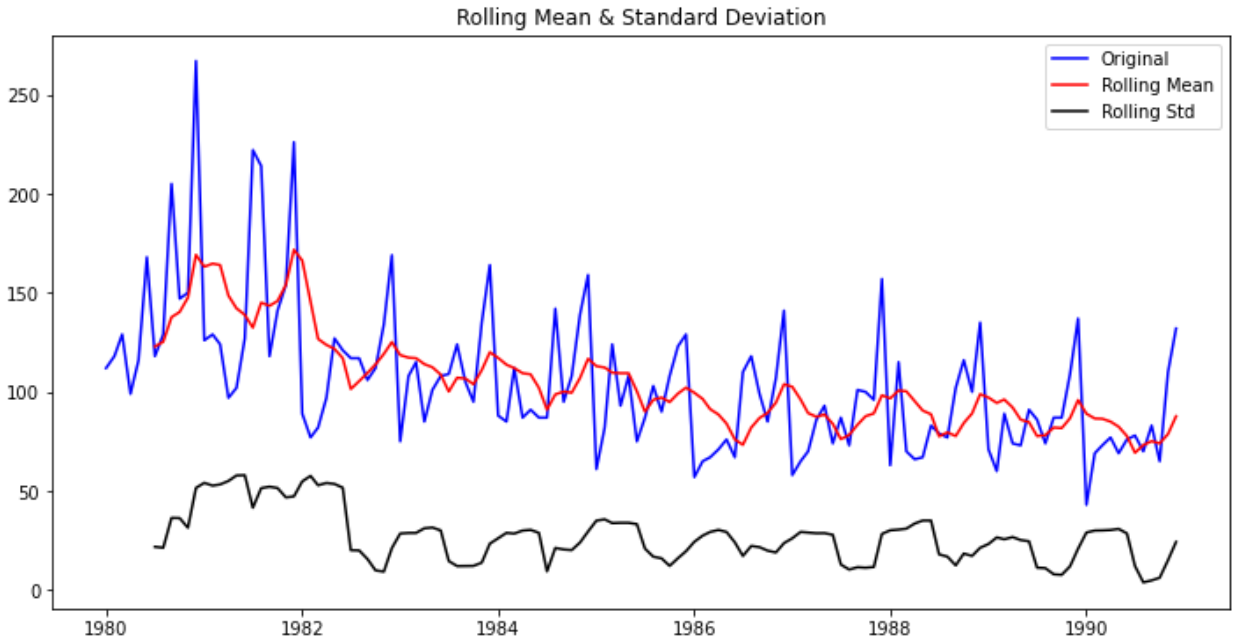
- **Insight:**
    - Now, series is stationary at alpha =0.05

Question 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.

Solution:

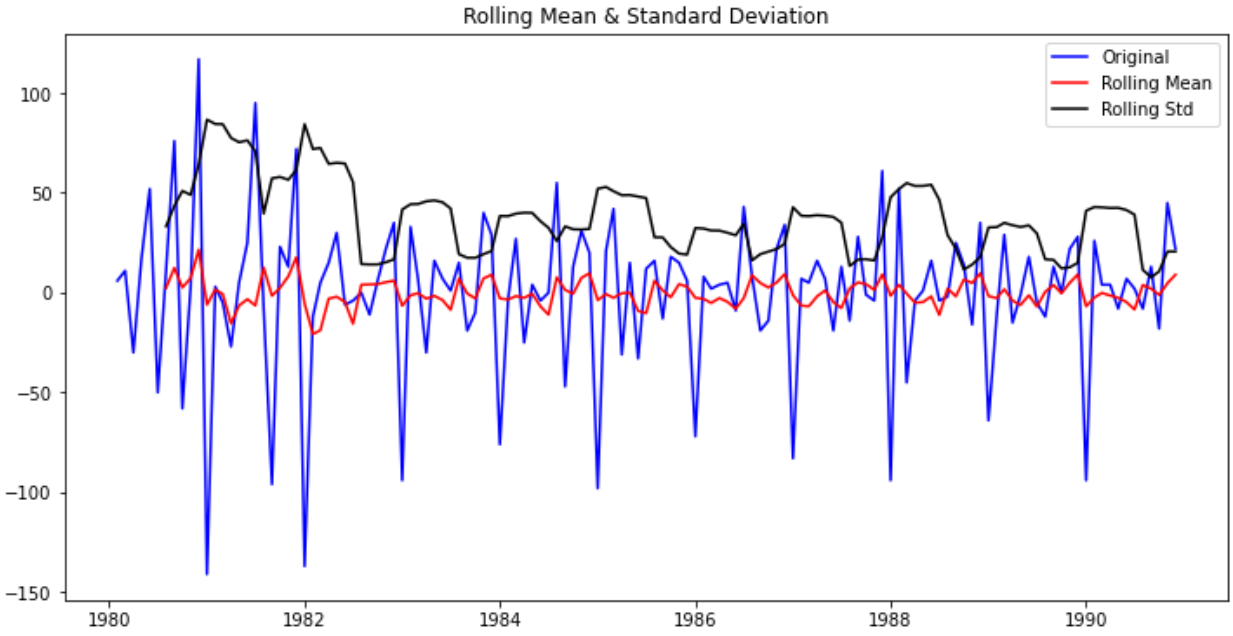
Please refer to the Jupyter Notebook submitted to look into the code.

- Test stationarity Graph:




```
Results of Dickey-Fuller Test:
Test Statistic                  -2.164250
p-value                          0.219476
#Lags Used                      13.000000
Number of Observations Used    118.000000
Critical Value (1%)             -3.487022
Critical Value (5%)             -2.886363
Critical Value (10%)            -2.580009
dtype: float64
```

- Test stationarity Graph by dropping un-necessary values:




```
Results of Dickey-Fuller Test:
Test Statistic                -6.592372e+00
p-value                        7.061944e-09
#Lags Used                     1.200000e+01
Number of Observations Used    1.180000e+02
Critical Value (1%)           -3.487022e+00
Critical Value (5%)           -2.886363e+00
Critical Value (10%)          -2.580009e+00
dtype: float64
```

- Building the parameter combinations for the Model:

```
Some parameter combinations for the Model...
Model: (0, 1, 1)
Model: (0, 1, 2)
Model: (0, 1, 3)
Model: (1, 1, 0)
Model: (1, 1, 1)
Model: (1, 1, 2)
Model: (1, 1, 3)
Model: (2, 1, 0)
Model: (2, 1, 1)
Model: (2, 1, 2)
Model: (2, 1, 3)
Model: (3, 1, 0)
Model: (3, 1, 1)
Model: (3, 1, 2)
Model: (3, 1, 3)
```

- The ARIMA Model:

```
                             ARIMA Model Results
==============================================================================
Dep. Variable:                 D.Rose   No. Observations:                  131
Model:                ARIMA(3, 1, 3)   Log Likelihood                -628.597
Method:                       css-mle   S.D. of innovations             28.356
Date:                Sun, 20 Jun 2021   AIC                           1273.194
Time:                        03:29:10   BIC                           1296.196
Sample:                    02-01-1980   HQIC                          1282.541
                         - 12-01-1990
==============================================================================
                 coef    std err          z      P>|z|      [0.025      0.975]
------------------------------------------------------------------------------
const          -0.4906      0.088     -5.548      0.000      -0.664      -0.317
ar.L1.D.Rose   -0.7243      0.086     -8.411      0.000      -0.893      -0.556
ar.L2.D.Rose   -0.7218      0.087     -8.342      0.000      -0.891      -0.552
ar.L3.D.Rose    0.2763      0.085      3.234      0.001       0.109       0.444
ma.L1.D.Rose   -0.0151      0.045     -0.339      0.735      -0.102       0.072
ma.L2.D.Rose    0.0151      0.044      0.340      0.734      -0.072       0.102
ma.L3.D.Rose   -1.0000      0.046    -21.901      0.000      -1.089      -0.911
                                    Roots
=============================================================================
                  Real          Imaginary           Modulus         Frequency
-----------------------------------------------------------------------------
AR.1           -0.5011           -0.8661j            1.0006           -0.3335
AR.2           -0.5011           +0.8661j            1.0006            0.3335
AR.3            3.6142           -0.0000j            3.6142           -0.0000
MA.1            1.0000           -0.0000j            1.0000           -0.0000
MA.2           -0.4925           -0.8703j            1.0000           -0.3320
MA.3           -0.4925           +0.8703j            1.0000            0.3320
-----------------------------------------------------------------------------
```

- Prediction of ARIMA Model:

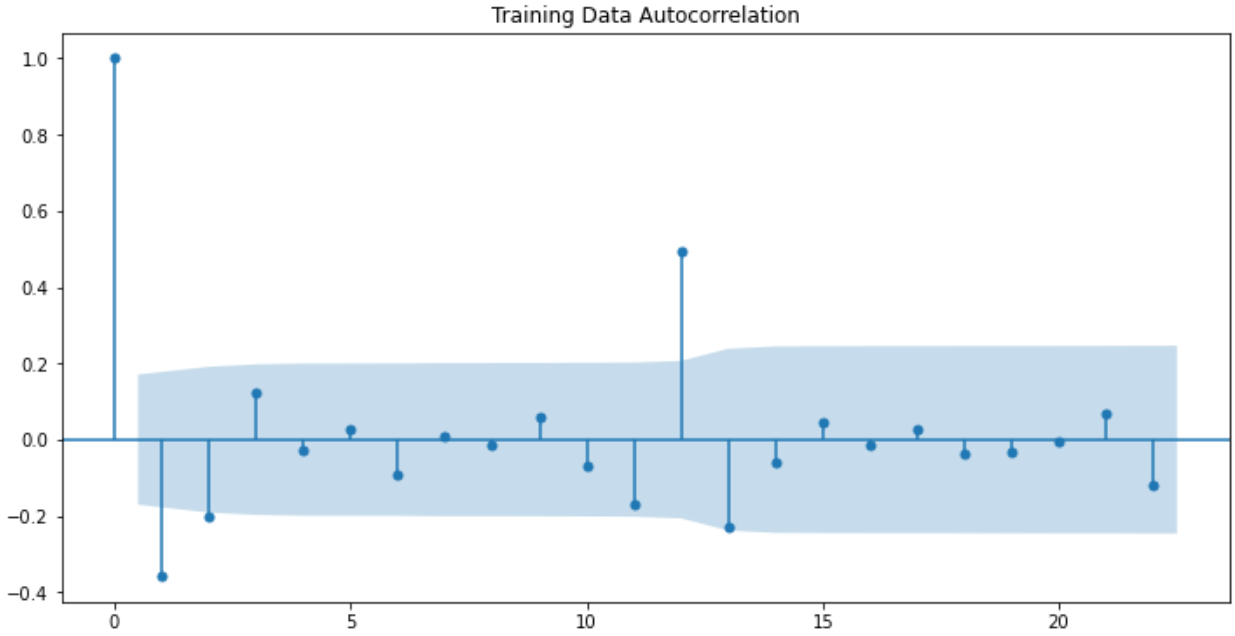
- RMSE = 15.986441154283346

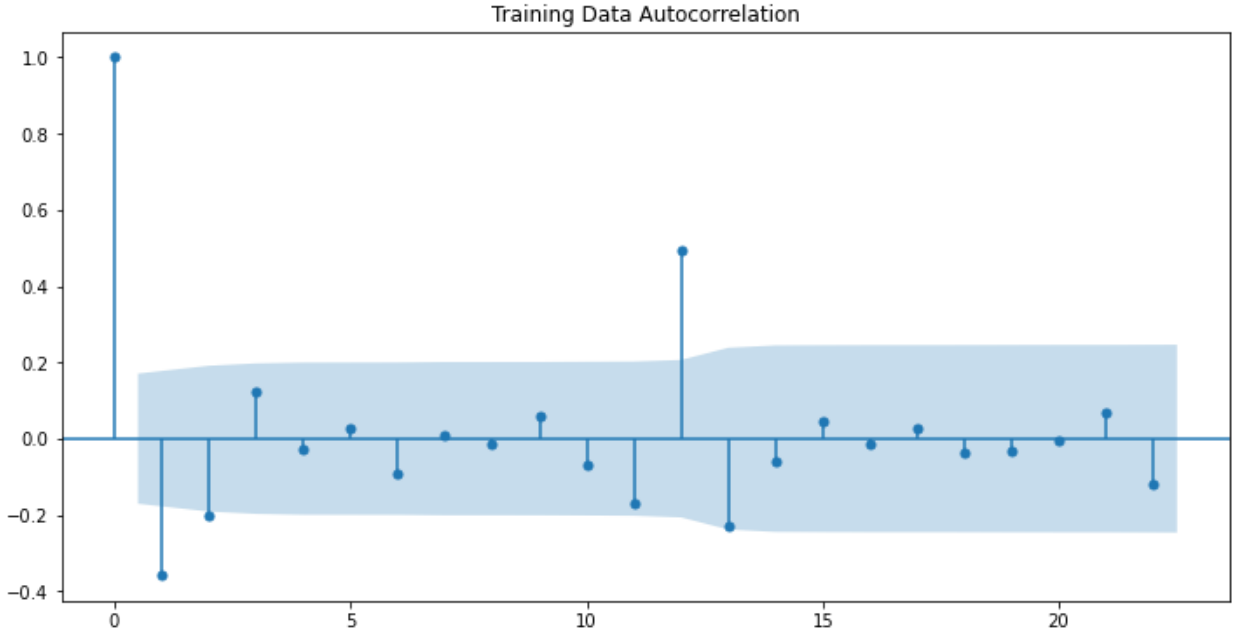
```
resultsDf = pd.DataFrame({'RMSE': [rmse]}
                         ,index=['ARIMA(3,1,3)(AIC)'])

resultsDf
```

	RMSE
ARIMA(3,1,3)(AIC)	15.986441

- Building an SARIMA model using AIC:





- Examples of the parameter combinations for the Model:

```
Examples of the parameter combinations for the Model are
Model: (0, 1, 1)(0, 0, 1, 12)
Model: (0, 1, 2)(0, 0, 2, 12)
Model: (0, 1, 3)(0, 0, 3, 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: (1, 1, 3)(1, 0, 3, 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)
Model: (2, 1, 3)(2, 0, 3, 12)
Model: (3, 1, 0)(3, 0, 0, 12)
Model: (3, 1, 1)(3, 0, 1, 12)
Model: (3, 1, 2)(3, 0, 2, 12)
Model: (3, 1, 3)(3, 0, 3, 12)
```

- The SARIMA Model:

```
                                     SARIMAX Results
==========================================================================================
Dep. Variable:                                Rose   No. Observations:                  132
Model:             SARIMAX(2, 1, 3)x(0, 0, 3, 12)   Log Likelihood                0.000
Date:                            Sun, 20 Jun 2021   AIC                           18.000
Time:                                    03:53:11   BIC                           40.598
Sample:                                01-01-1980   HQIC                          27.117
                                     - 12-01-1990
Covariance Type:                              opg
==============================================================================
                 coef    std err          z      P>|z|      [0.025      0.975]
------------------------------------------------------------------------------
ar.L1          1.3303     21.012      0.063      0.950     -39.853      42.513
ar.L2          2.7021         -0       -inf      0.000       2.702       2.702
ma.L1         -0.1861         -0        inf      0.000      -0.186      -0.186
ma.L2         -4.3572      0.463     -9.405      0.000      -5.265      -3.449
ma.L3         -0.1160         -0        inf      0.000      -0.116      -0.116
ma.S.L12   -3.244e+13         -0        inf      0.000   -3.24e+13   -3.24e+13
ma.S.L24   -7.053e+13         -0        inf      0.000   -7.05e+13   -7.05e+13
ma.S.L36    1.981e+14         -0       -inf      0.000    1.98e+14    1.98e+14
sigma2       547.8359         -0       -inf      0.000     547.836     547.836
==============================================================================
Ljung-Box (L1) (Q):                   nan   Jarque-Bera (JB):                34.12
Prob(Q):                              nan   Prob(JB):                         0.00
Heteroskedasticity (H):               nan   Skew:                            0.00
Prob(H) (two-sided):                  nan   Kurtosis:                        0.00
==========================================================================================
```

- The Auto ARIMA Summary:

Rose	mean	mean_se	mean_ci_lower	mean_ci_upper
1991-01-01	-3.321542e+46	NaN	NaN	NaN
1991-02-01	-8.099408e+46	NaN	NaN	NaN
1991-03-01	-1.974999e+47	NaN	NaN	NaN
1991-04-01	-4.815932e+47	NaN	NaN	NaN
1991-05-01	-1.174340e+48	NaN	NaN	NaN

- RMSE Output: RMSE: 3.937978160647845e+66

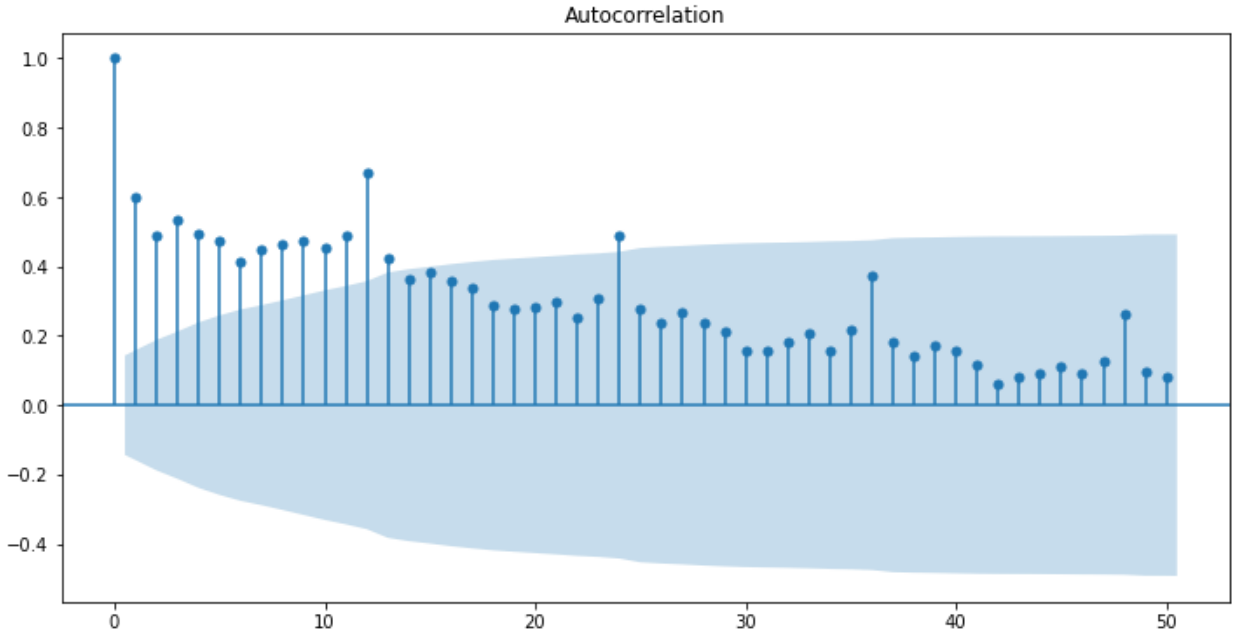
	RMSE
ARIMA(3,1,3)(AIC)	1.598644e+01
ARIMA(2,1,2)(ACF &PACF)	1.535487e+01
SARIMA(2,1,3)(0,0,3,12)(SARIMA AIC)	3.937978e+66

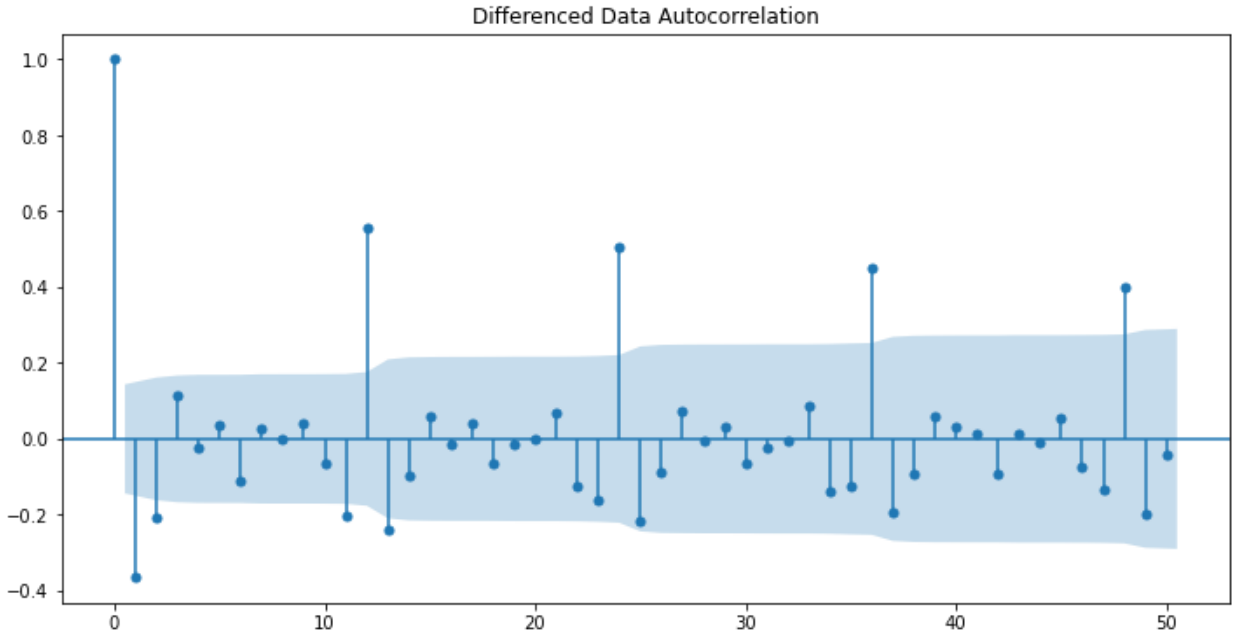
Question 7: Build ARIMA/SARIMA models based on the cut-off points of ACF and PACF on the training data and evaluate this model on the test data using RMSE.

Solution:

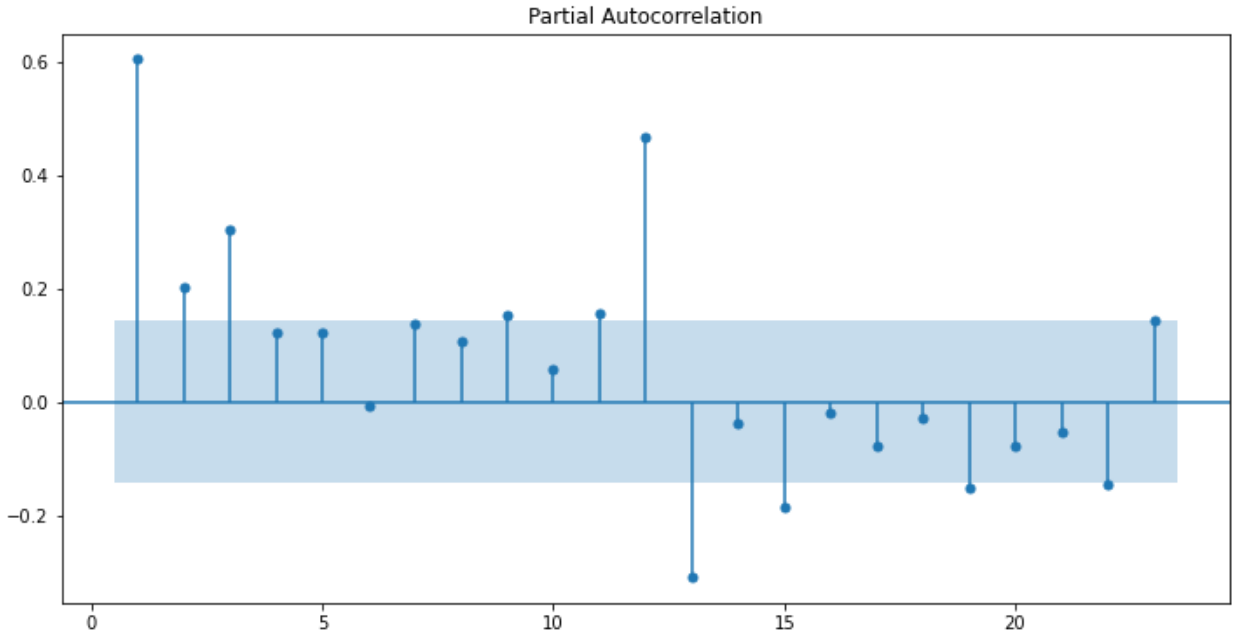
Please refer to the Jupyter Notebook submitted to look into the code.

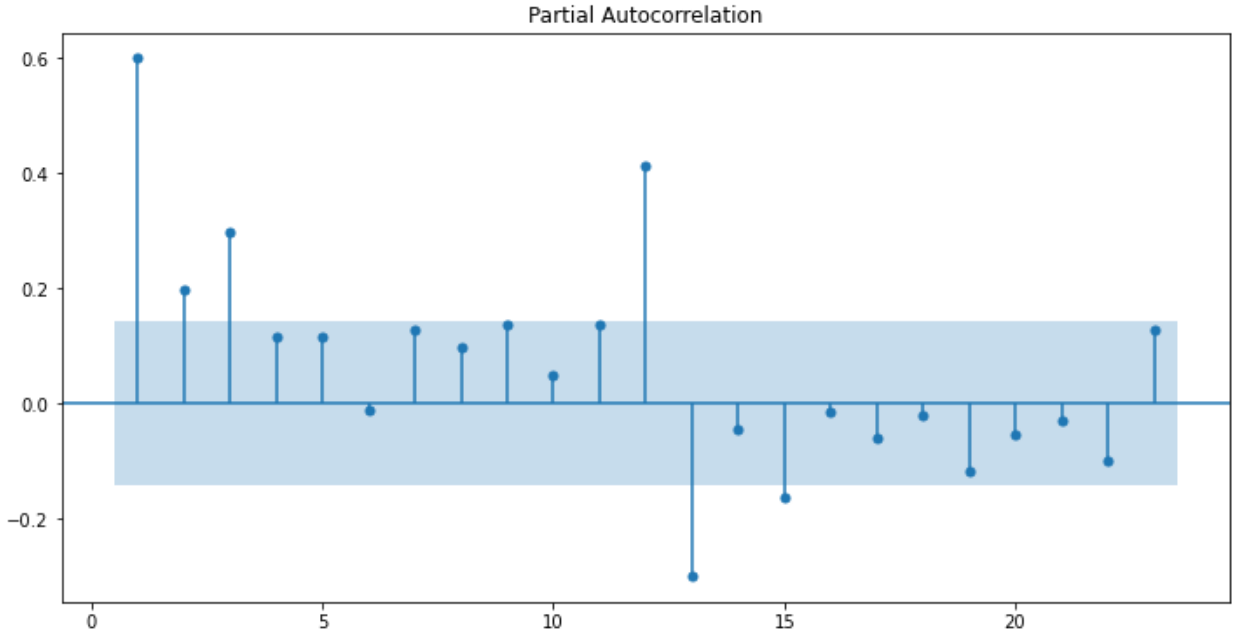
- Plotting the Autocorrelation graph:





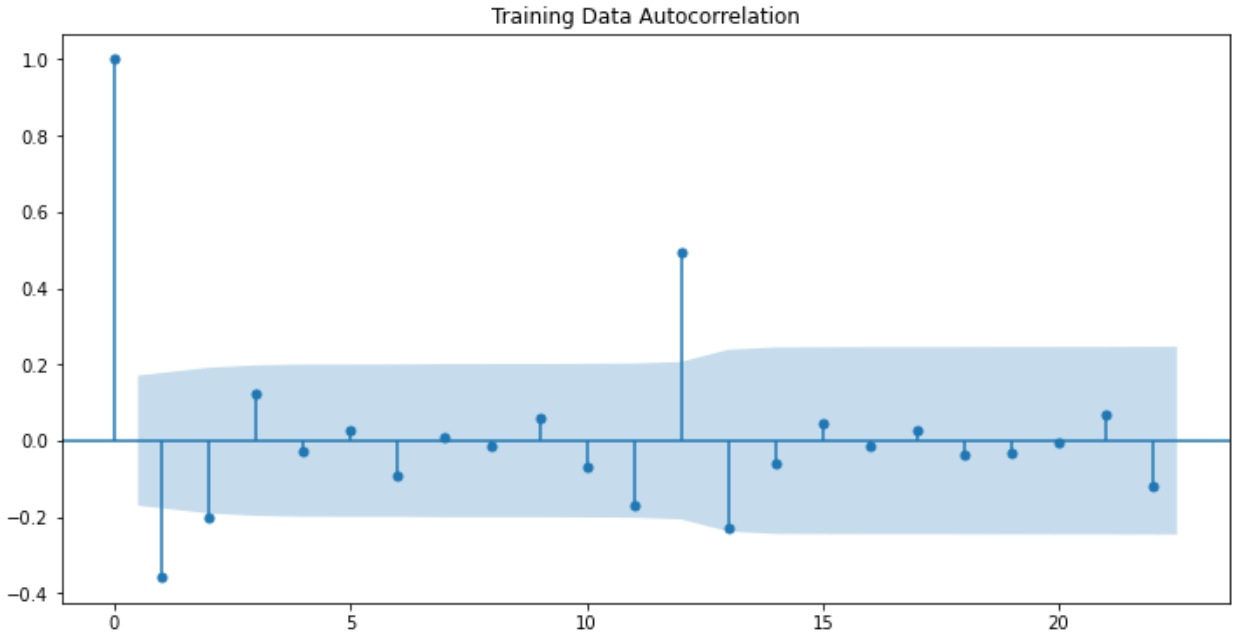
- Plotting the Partial Autocorrelation graph:



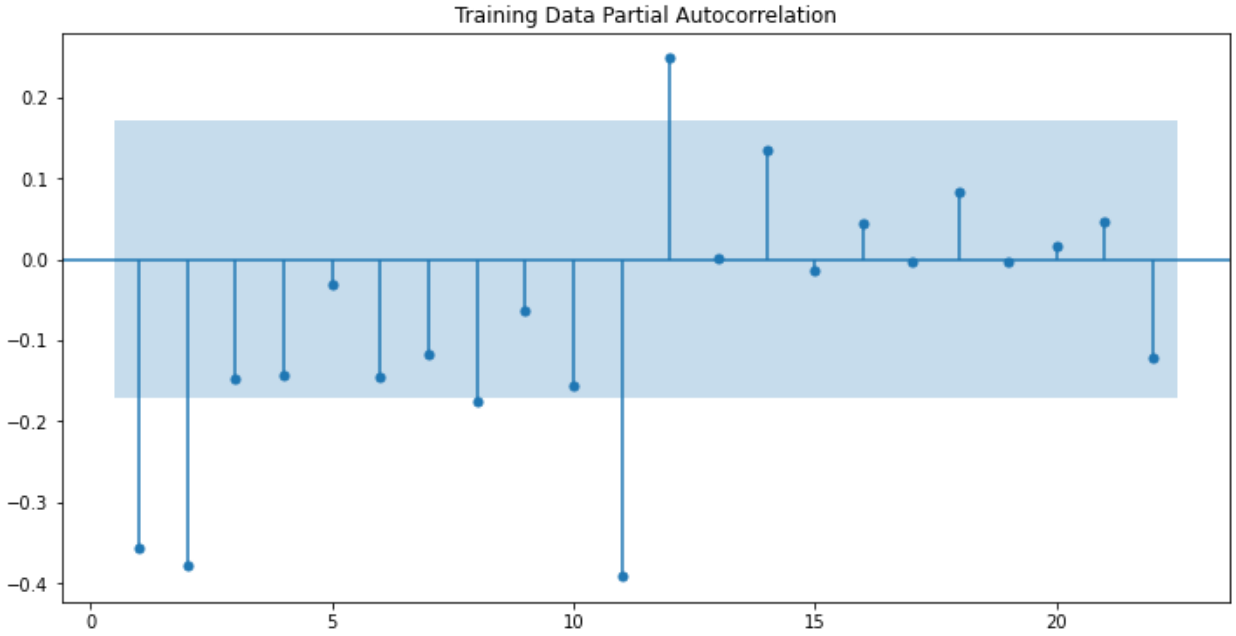


- Finding auto-correlation function and partial auto-correlation function on training data only:

- Training data autocorrelation:



- Training data partial autocorrelation:



- Insight:
  - Values of Q and P respectively form Autocorrelation and Partial Autocorrelation are Q=2, P=2

- The ARIMA Model:

ARIMA Model Results

Dep. Variable:	D.Rose	No. Observations:	131
Model:	ARIMA(2, 1, 2)	Log Likelihood	-633.649
Method:	css-mle	S.D. of innovations	29.975
Date:	Sun, 20 Jun 2021	AIC	1279.299
Time:	04:04:04	BIC	1296.550
Sample:	02-01-1980 - 12-01-1990	HQIC	1286.309

	coef	std err	z	P>|z|	[0.025	0.975]
const	-0.4911	0.081	-6.076	0.000	-0.649	-0.333
ar.L1.D.Rose	-0.4383	0.218	-2.015	0.044	-0.865	-0.012
ar.L2.D.Rose	0.0269	0.109	0.246	0.806	-0.188	0.241
ma.L1.D.Rose	-0.3316	0.203	-1.633	0.102	-0.729	0.066
ma.L2.D.Rose	-0.6684	0.201	-3.332	0.001	-1.062	-0.275

Roots

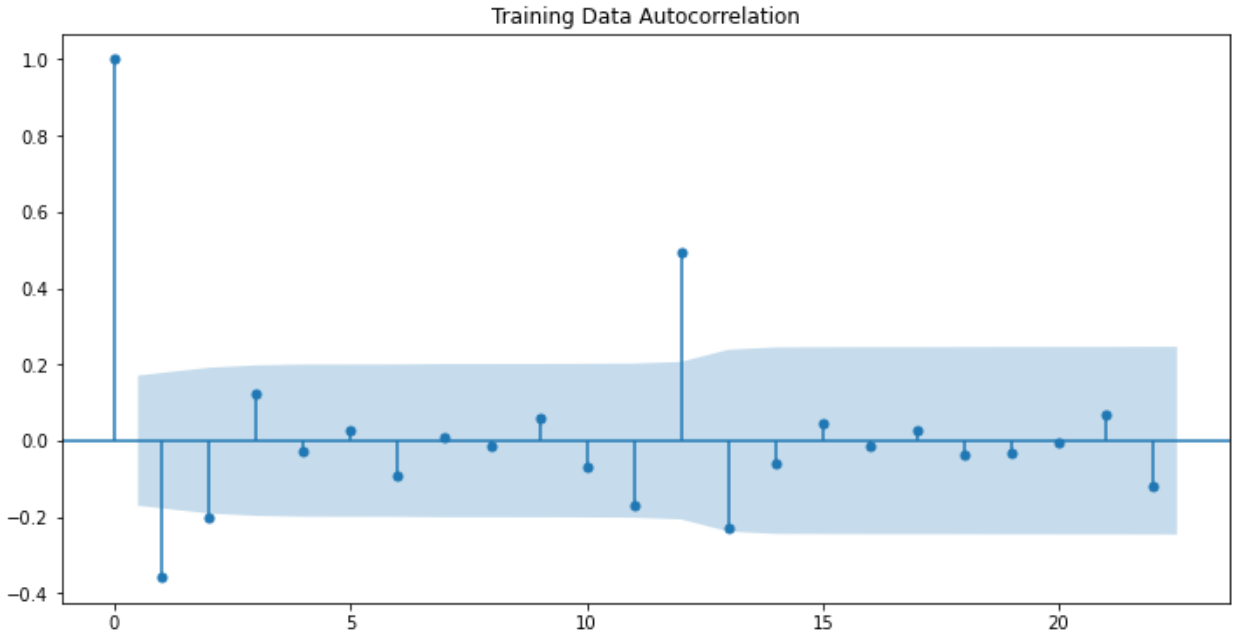
	Real	Imaginary	Modulus	Frequency
AR.1	-2.0290	+0.0000j	2.0290	0.5000
AR.2	18.3389	+0.0000j	18.3389	0.0000
MA.1	1.0000	+0.0000j	1.0000	0.0000
MA.2	-1.4961	+0.0000j	1.4961	0.5000

- RMSE Correlation is 15.354873412846443
- Model results:

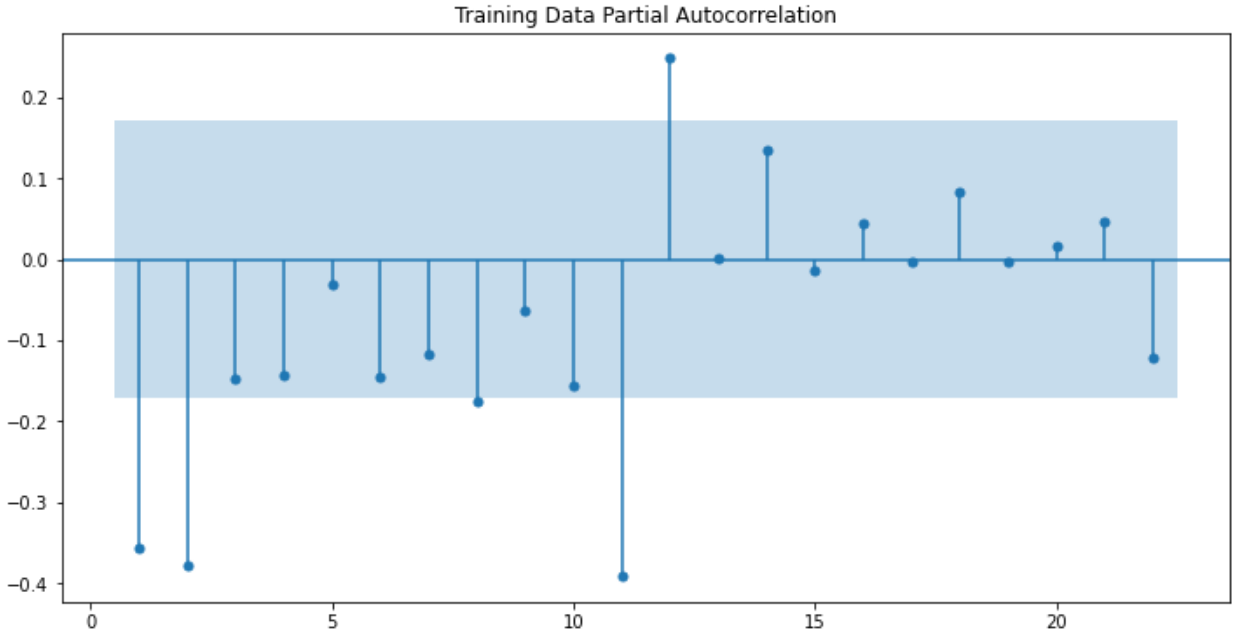
	RMSE
ARIMA(3,1,3)(AIC)	1.598644e+01
ARIMA(2,1,2)(ACF &PACF)	1.535487e+01
SARIMA(2,1,3)(0,0,3,12)(SARIMA AIC)	3.937978e+66
ARIMA(2,1,2)(ACF &PACF)	1.535487e+01

SARIMA using auto-correlation function and partial auto-correlation function:

- Training data autocorrelation:



- Training data partial autocorrelation:



- Insights:
  - Looking in seasonality, so value of Q =12, no seasonality (pattern) for P in Training Data Partial Autocorrelation so P=0, No Further Differentiation so D =0

- The SARIMA Model

```
                                SARIMAX Results
==========================================================================================
Dep. Variable:                                Rose   No. Observations:                  132
Model:             SARIMAX(3, 1, 3)x(0, 0, 3, 12)   Log Likelihood                -383.493
Date:                            Sun, 20 Jun 2021   AIC                             786.986
Time:                                    04:12:16   BIC                             812.095
Sample:                                01-01-1980   HQIC                            797.116
                                     - 12-01-1990
Covariance Type:                              opg
==============================================================================
                 coef    std err          z      P>|z|      [0.025      0.975]
------------------------------------------------------------------------------
ar.L1         -1.4106      0.026    -55.192      0.000      -1.461      -1.361
ar.L2         -0.0776      0.035     -2.242      0.025      -0.146      -0.010
ar.L3          0.4917      0.058      8.424      0.000       0.377       0.606
ma.L1         -1.4278   1.31e-06  -1.09e+06      0.000      -1.428      -1.428
ma.L2         -1.1898      0.001  -1486.665      0.000      -1.191      -1.188
ma.L3         -1.9251      0.000  -1.13e+04      0.000      -1.925      -1.925
ma.S.L12   -3.244e+13   2.42e-16  -1.34e+29      0.000   -3.24e+13   -3.24e+13
ma.S.L24   -7.053e+13    1.3e-16  -5.41e+29      0.000   -7.05e+13   -7.05e+13
ma.S.L36    1.981e+14    6.7e-17   2.96e+30      0.000    1.98e+14    1.98e+14
sigma2       804.7937   7.47e-09   1.08e+11      0.000     804.794     804.794
==========================================================================================
Ljung-Box (L1) (Q):                   0.03   Jarque-Bera (JB):             25551.37
Prob(Q):                              0.87   Prob(JB):                         0.00
Heteroskedasticity (H):             512.84   Skew:                            -8.93
Prob(H) (two-sided):                  0.00   Kurtosis:                        83.12
==========================================================================================
```

- **Results for SARIMA model:**
  - RMSE: 5000631069257765.0

Question 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.

Solution:

```
temp_resultsDf_SA_AIC = pd.DataFrame({'RMSE': [rmse]}
                           ,index=['SARIMA(3,1,3)(0,0,3,12)(SARIMA AIC & PACF)'])


resultsDf = pd.concat([resultsDf,temp_resultsDf_SA_AIC])

resultsDf
```

	RMSE
ARIMA(3,1,3)(AIC)	1.598644e+01
ARIMA(2,1,2)(ACF &PACF)	1.535487e+01
SARIMA(2,1,3)(0,0,3,12)(SARIMA AIC)	3.937978e+66
ARIMA(2,1,2)(ACF &PACF)	1.535487e+01
SARIMA(3,1,3)(0,0,3,12)(SARIMA AIC & PACF)	5.000631e+15

Question 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.

Solution:

Please refer to the Jupyter Notebook submitted to look into the code.

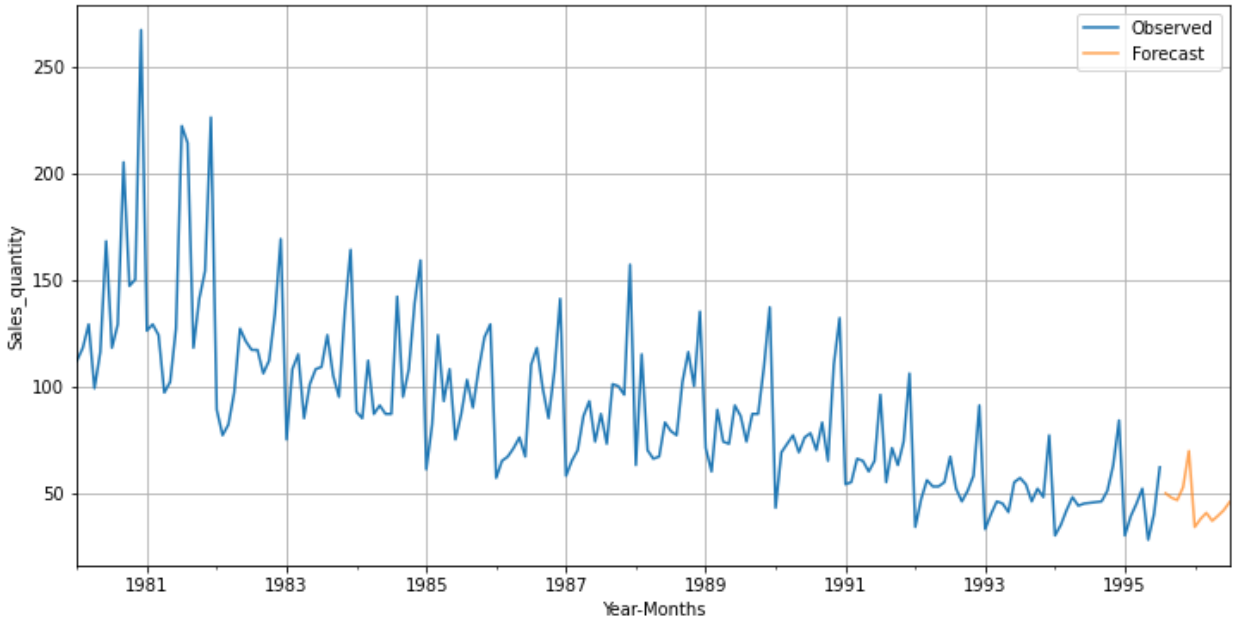
```
                                SARIMAX Results
==========================================================================================
Dep. Variable:                                Rose   No. Observations:                  187
Model:             SARIMAX(0, 1, 1)x(1, 0, 1, 12)   Log Likelihood                -741.076
Date:                            Sun, 20 Jun 2021   AIC                           1490.152
Time:                                    04:20:29   BIC                           1502.742
Sample:                                01-01-1980   HQIC                          1495.260
                                     - 07-01-1995
Covariance Type:                              opg
==============================================================================
                 coef    std err          z      P>|z|      [0.025      0.975]
------------------------------------------------------------------------------
ma.L1         -1.1119      0.052    -21.305      0.000      -1.214      -1.010
ar.S.L12       0.9117      0.018     50.228      0.000       0.876       0.947
ma.S.L12      -1.1022      0.187     -5.905      0.000      -1.468      -0.736
sigma2       190.4572     39.679      4.800      0.000     112.687     268.227
===================================================================================
Ljung-Box (L1) (Q):                   1.87   Jarque-Bera (JB):               140.09
Prob(Q):                              0.17   Prob(JB):                         0.00
Heteroskedasticity (H):               0.15   Skew:                             0.51
Prob(H) (two-sided):                  0.00   Kurtosis:                         7.30
===================================================================================

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

- **Forecasting for upcoming one year:**

Rose	mean	mean_se	mean_ci_lower	mean_ci_upper
1995-08-01	49.780482	17.000156	16.460788	83.100175
1995-09-01	47.749843	17.085973	14.261952	81.237735
1995-10-01	46.616105	17.171360	12.960858	80.271353
1995-11-01	52.640036	17.256325	18.818260	86.461812
1995-12-01	69.670551	17.340875	35.683062	103.658041

- Upcoming year forecasting graph:



- **Insights:**
  - The forecasting prediction is that there will be an increase in demand in 1996 than 1995 in Rose Wine.
  - There will be a reduction in seasonality.

Question 10: Comment on the model thus built and report your findings and suggest the measures that the company should be taking for future sales. Please explain and summarise the various steps performed in this project. There should be proper business interpretation and actionable insights present.

Solution:

This Rose data contains the 15 years of data of sales. Two data were missing from the data set, those were imputed using interpolation of Rose data. Once I dug the data it seems to have a strong seasonality in the last year every year. It shows in festive times the demand for wine increases all around. The trend of data shows a continuous decrease in sales than the seasonal increase in demand.

- Once predicting the future demand for the next 12 months, it is observed that demand will show very slight improvement and the seasonality factor is also reducing for the next 12 months.

- Once the sales data decomposed shows the trend of demand as decreasing over the years.

Various exponential smoothing models development of this project contains the deep analysis of data:

- First data is converted into Time series index, and then EDA has provided the complete insight of data along with Decomposition of data on both additive and multiplicative basis. Additive analysis shows some trends in error terms so multiplicative decomposition is required here.
- Then different techniques of various exponential smoothing models are done training data and its effect is observed on the test data. The RMSE value of the different models is observed, and each model's RMSE value is enclosed here for better understanding. Double Exponential Smoothing follows the test data most accurately comparing the other models.

Test	RMSE
Regression On Time	15.268955
Naive Model	79.718773
Simple Average Model	53.460570
Alpha = 0.995, SimpleExponentialSmoothing	36.796227
Alpha=0.3, Beta=0.3, Double Exponential Smoothing	1292.041812
Alpha = 0.3, Beta = 0.3, Double Exponential Smoothing	1292.041812
Alpha = 0.676, Beta = 0.088, Gamma = 0.323, Triple Exponential Smoothing	36.796227

- Stationarity shows the statistical properties of a time series that do not change over a period of time. Here data was not stationary and the first order of differentiation was required to make it stationary. The augmented Dickey-Fuller test at alpha level =0.05 is used to develop the above model.

- ARIMA and SARIMA models are then developed to generate the future prediction for the next 12 months. ARIMA and SARIMA both are developed using AIC first then using ACF and PACF.

The RMSE value of different models is inclosed here

Test	RMSE
ARIMA(3,1,3) (AIC)	1.598644e+01
ARIMA(2,1,2) (ACF & PACF)	1.535487e+01
SARIMA (2,1,3) (0,0,3,12) (SARIMA AIC)	3.937978e+66
ARIMA (2,1,2) (ACF & PACF)	1.535487e+01
SARIMA (3,1,3) (0,0,3,12) (SARIMA AIC & PACF)	5.000631e+15

The Overall RMSE of model when implemented on whole data (Training+Test) = RMSE of the Full Model 28.321066260560638

Action Insights for the company:

- Supply Chain Enablement:
    - Looking at the result and future forecasting, we've seen that there's an increase in demand - hence, to match with this increase in demand the company has to focus on enabling the supply chain to fulfill the consumer demand.
- Brand Marketing and Promotions:
    - This is the exact time when the company should work on increasing the promotional budget and build a branding team. The team should work with broadcasting partners as TV, Radio and other sporting tournament sponsors to promote the brand.
- With the identified new customer base, if there would be right and seamless supply, the Rose wine can keep up the potential growth in the market.