Time Series Forecasting GRADED PROJECT

PREPARED FOR

Time Series Forecasting **MENTOR**

PREPARED BY

NITIN KUMAR SINGH

INDEX

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

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

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

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

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.

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.

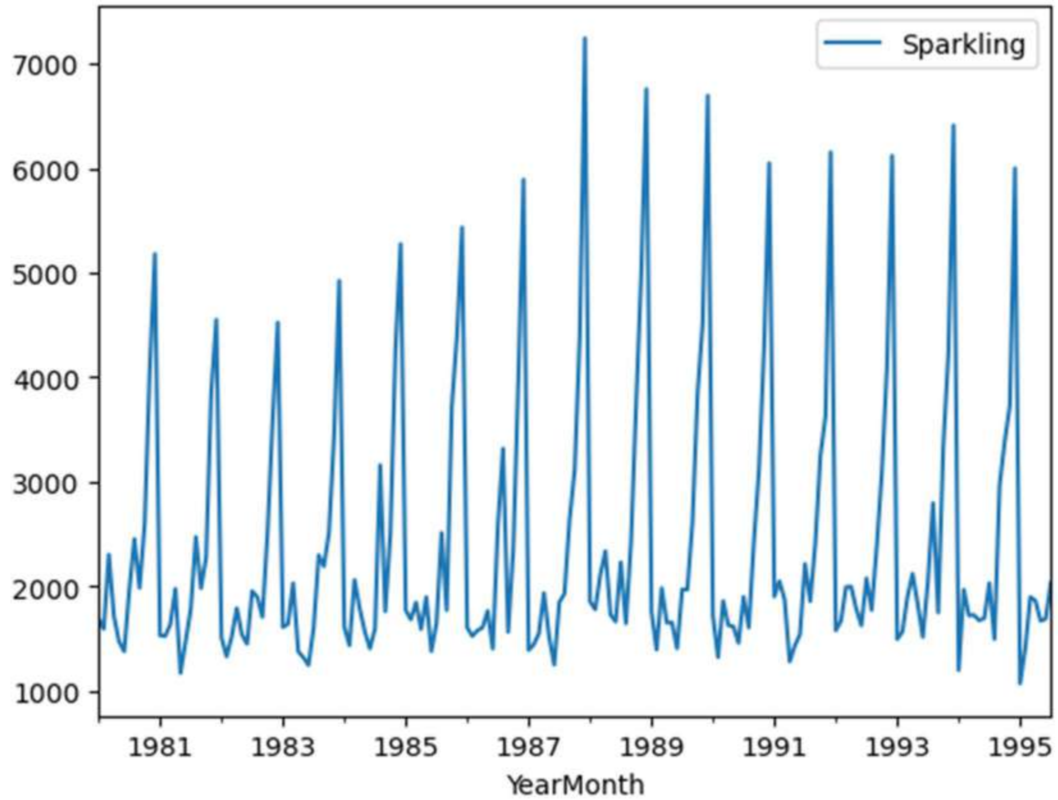
7. Build a table (create a data frame) with all the models built along with their corresponding parameters and the respective RMSE values on the test data.

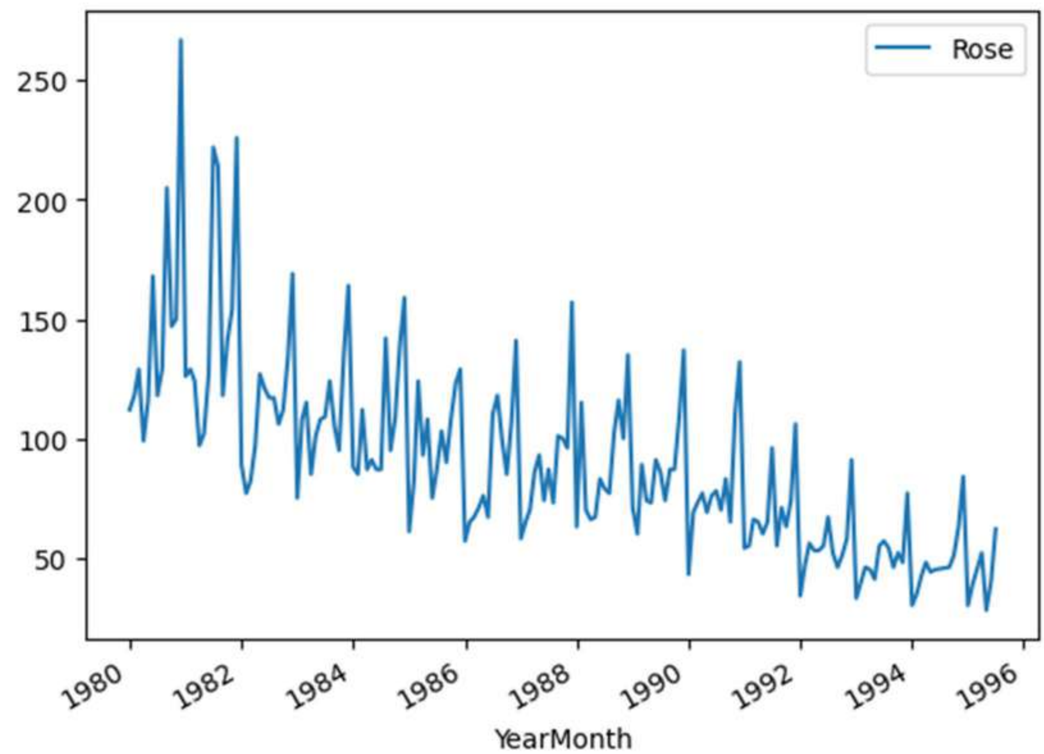
8. Based on the model-building exercise, build the most optimum model(s) on the complete data and predict 12 months into the future with appropriate confidence intervals/bands.

9. 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.

Plot The Data





Exploratory Data Analysis

```
df_spark.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 187 entries, 0 to 186
Data columns (total 2 columns):
 #   Column     Non-Null Count  Dtype 
---  ------     --------------  ----- 
 0   YearMonth  187 non-null    object
 1   Sparkling  187 non-null    int64 
dtypes: int64(1), object(1)
memory usage: 3.0+ KB
```

```
df_spark.isna().sum()
```

```
YearMonth    0
Sparkling    0
dtype: int64
```

```
df_spark['YearMonth'] = pd.to_datetime(df_spark['YearMonth'])
```

```
df_spark.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 187 entries, 0 to 186
Data columns (total 2 columns):
 #   Column     Non-Null Count  Dtype         
---  ------     --------------  -----         
 0   YearMonth  187 non-null    datetime64[ns]
 1   Sparkling  187 non-null    int64         
dtypes: datetime64[ns](1), int64(1)
memory usage: 3.0 KB
```

There are 2 columns in dataset(year month and sparkling) and no nul value present in sparkling dataset

```
df_rose.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 187 entries, 0 to 186
Data columns (total 2 columns):
 #   Column     Non-Null Count  Dtype  
---  ------     --------------  -----  
 0   YearMonth  187 non-null    object 
 1   Rose       185 non-null    float64
dtypes: float64(1), object(1)
memory usage: 3.0+ KB
```

```
df_rose.isna().sum()
```

```
YearMonth    0
Rose         2
dtype: int64
```

```
df_rose.dropna(inplace=True)
```

```
df_rose.isna().sum()
```

```
YearMonth    0
Rose         0
dtype: int64
```

There are 2 columns in dataset(year month and rose) and null value present in after remove the null value in rose dataset because null value is bad impact in our model

Split the data into training and test.

```python
from datetime import datetime,timedelta
```

```python
train_dataset_end=datetime(1990,12,1)
test_dataset_end=datetime(1995,3,1)
```

Sparkling dataset

```python
train_data
```

YearMonth	Sparkling	spark first difference
1980-01-01	1686	NaN
1980-02-01	1591	-95.0
1980-03-01	2304	713.0
1980-04-01	1712	-592.0
1980-05-01	1471	-241.0
...	...	...
1990-08-01	1605	-294.0
1990-09-01	2424	819.0
1990-10-01	3116	692.0
1990-11-01	4286	1170.0
1990-12-01	6047	1761.0

```python
test_data
```

YearMonth	Sparkling	spark first difference
1991-01-01	1902	-4145.0
1991-02-01	2049	147.0
1991-03-01	1874	-175.0
1991-04-01	1279	-595.0
1991-05-01	1432	153.0
1991-06-01	1540	108.0
1991-07-01	2214	674.0
1991-08-01	1857	-357.0
1991-09-01	2408	551.0
1991-10-01	3252	844.0
1991-11-01	3627	375.0
1991-12-01	6153	2526.0
1992-01-01	1577	-4576.0
1992-02-01	1667	90.0
1992-03-01	1993	326.0
1992-04-01	1997	4.0
1992-05-01	1783	-214.0
1992-06-01	1625	-158.0
1992-07-01	2076	451.0
1992-08-01	1773	-303.0

```python
from datetime import datetime,timedelta
```

```python
train_dataset_end=datetime(1990,12,1)
test_dataset_end=datetime(1995,3,1)
```

rose dataset

```python
train_data
```

YearMonth	Rose	rose difference
1980-01-01	112.0	NaN
1980-02-01	118.0	6.0
1980-03-01	129.0	11.0
1980-04-01	99.0	-30.0
1980-05-01	116.0	17.0
...	...	...
1990-08-01	70.0	-8.0
1990-09-01	83.0	13.0
1990-10-01	65.0	-18.0
1990-11-01	110.0	45.0
1990-12-01	132.0	22.0

```python
test_data
```

YearMonth	Rose	rose difference	Predicted_ARIMA	Predicted_SARIMA
1991-01-01	54.0	-78.0	-36.893584	-91.183796
1991-02-01	55.0	1.0	30.540893	7.118906
[illegible]	[illegible]	[illegible]	27.306362	9.312434
1991-04-01	65.0	-1.0	8.745835	-2.437362
1991-05-01	60.0	-5.0	-11.783770	-0.140594
1991-06-01	65.0	5.0	6.843150	8.809749
1991-07-01	96.0	31.0	-6.950591	7.972127
1991-08-01	55.0	-41.0	-12.935630	-16.941520
1991-09-01	71.0	16.0	-0.231140	15.178974
1991-10-01	63.0	-8.0	12.068924	7.229183
1991-11-01	74.0	11.0	-4.459419	28.893361
1991-12-01	106.0	32.0	0.556585	49.915998
1992-01-01	34.0	-72.0	-27.001106	-70.008452
1992-02-01	47.0	13.0	26.079392	18.028012
1992-03-01	56.0	9.0	17.047336	6.629856
1992-04-01	53.0	-3.0	2.780997	-3.293289
1992-05-01	53.0	0.0	-10.978266	-3.647945
1992-06-01	55.0	2.0	4.952986	-0.184594

to scroll output; double click to hide

RMSE

Sparkling dataset

```
# Compute the root mean square error
test_data[['spark first difference','Predicted_SARIMA','Predicted_ARIMA']].mean()
```

```
spark first difference    -81.372549
Predicted_SARIMA          -81.098649
Predicted_ARIMA            19.933200
dtype: float64
```

```
from sklearn.metrics import mean_squared_error
from math import sqrt
```

```
rmse = sqrt(mean_squared_error(pred,test_data['spark first difference']))
print(rmse)
```

```
1504.5172197495635
```

```
rmse = sqrt(mean_squared_error(pred,test_data['Predicted_SARIMA']))
print(rmse)
```

```
1521.4052082864753
```

```
rmse = sqrt(mean_squared_error(pred,test_data['Predicted_ARIMA']))
print(rmse)
```

```
0.0
```

Rose dataset

In [79]:
```
from sklearn.metrics import mean_squared_error
from math import sqrt
```

In [80]:
```
rmse = sqrt(mean_squared_error(pred,test_data['rose difference']))
print(rmse)
```

```
18.432381053752028
```

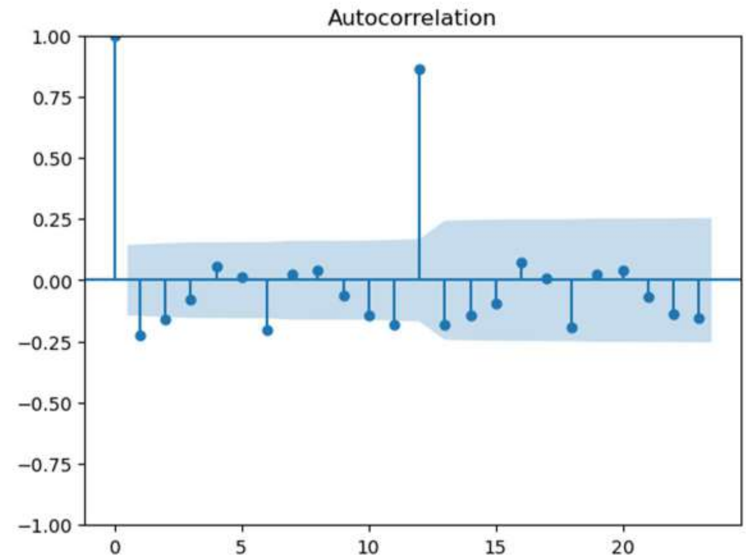
In [81]:
```
rmse = sqrt(mean_squared_error(pred,test_data['Predicted_SARIMA']))
print(rmse)
```

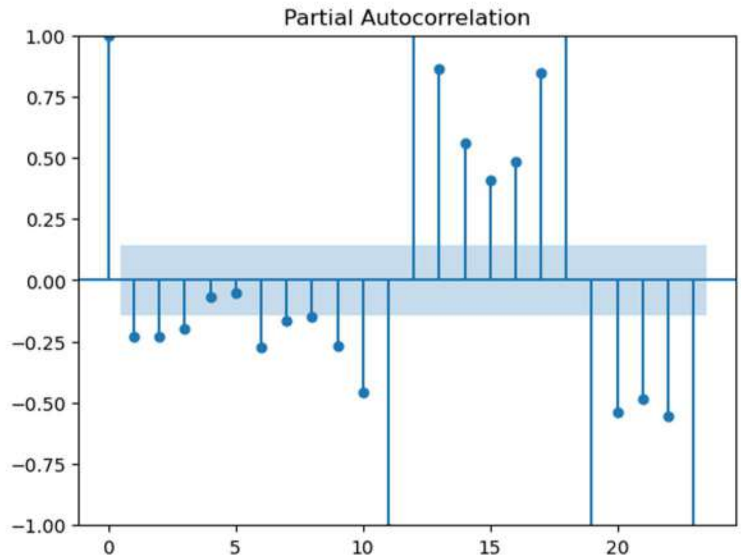
```
21.330329127087015
```

In [82]:
```
rmse = sqrt(mean_squared_error(pred,test_data['Predicted_ARIMA']))
print(rmse)
```

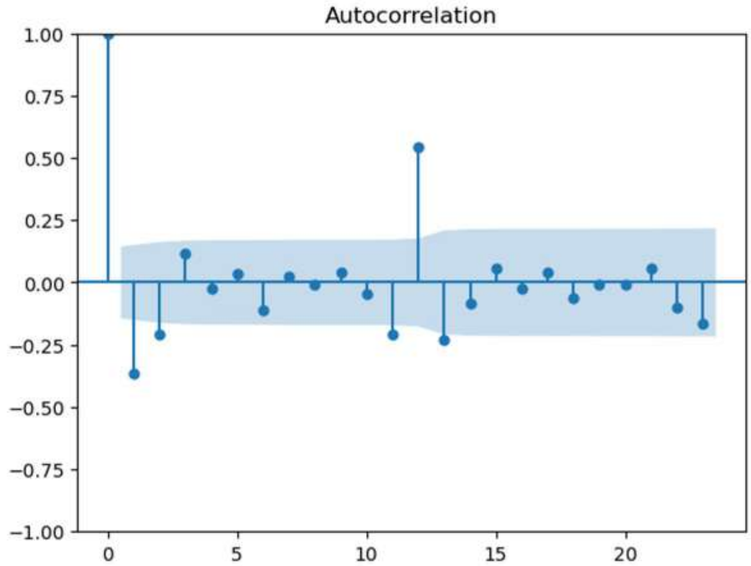
```
0.0
```

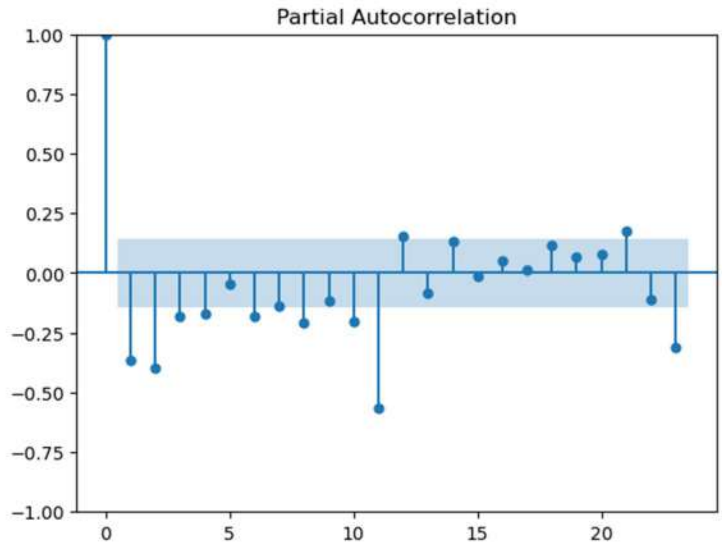
Sparkling Dataset





Rose Dataset





Use **dickey-fuller test** to check data is stationary or not first check data is not stationary but after shifting and check data is stationary

In **dickey-fuller** test check data set with p-value less than 5% data is stationary and value more than 5% data is not stationary

Sparkling dataset

```
from statsmodels.tsa.stattools import adfuller
```

```
def test_adfuller(sales):
    result  = adfuller(sales)
    print('ADF statistics: {}'.format(result[0]))
    print('p-value: {}'.format(result[1]))
    if result[1]<=0.05:
        print('data is stationary')
    else:
        print('data is not stationary')
```

```
test_adfuller(df_spark)
```

```
ADF statistics: -1.3604974548123325
p-value: 0.6010608871634875
data is not stationary
```

```
df_spark
```

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

```
df_spark['spark first difference'] = df_spark['Sparkling']-df_spark['Sparkling'].shift(1)
```

```
df_spark
```

YearMonth	Sparkling	spark first difference
1980-01-01	1686	NaN
1980-02-01	1591	-95.0
1980-03-01	2304	713.0
1980-04-01	1712	-592.0
1980-05-01	1471	-241.0
...	...	...
1995-03-01	1897	495.0
1995-04-01	1862	-35.0
1995-05-01	1670	-192.0
1995-06-01	1688	18.0

```
test_adfuller(df_spark['spark first difference'].dropna())
```

```
ADF statistics: -45.05030093619527
p-value: 0.0
data is stationary
```

Rose dataset

```
from statsmodels.tsa.stattools import adfuller
```

```
def test_adfuller(sales):
    result  = adfuller(sales)
    print('ADF statistics: {}'.format(result[0]))
    print('p-value: {}'.format(result[1]))
    if result[1]<=0.05:
        print('data is stationary')
    else:
        print('data is not stationary')
```

```
test_adfuller(df_rose)
```

```
ADF statistics: -1.8380327966021965
p-value: 0.3617495457657554
data is not stationary
```

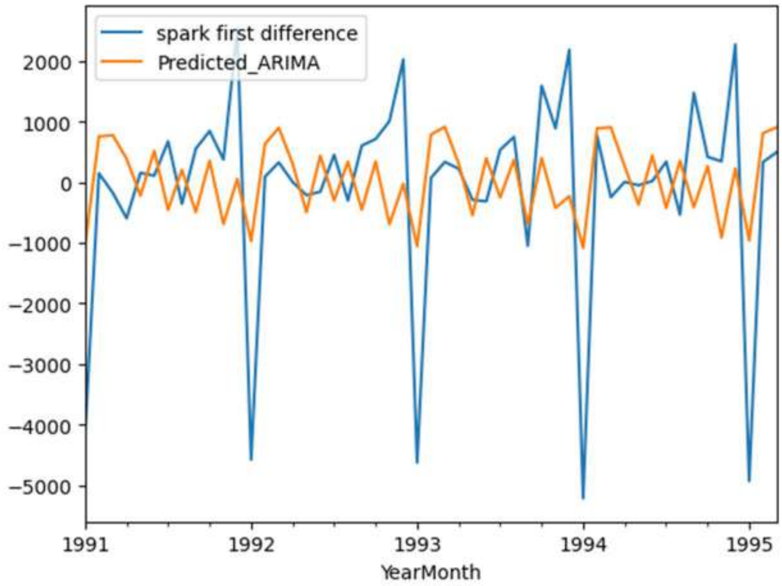
```
df_rose['rose difference'] = df_rose['Rose']-df_rose['Rose'].shift(1)
```

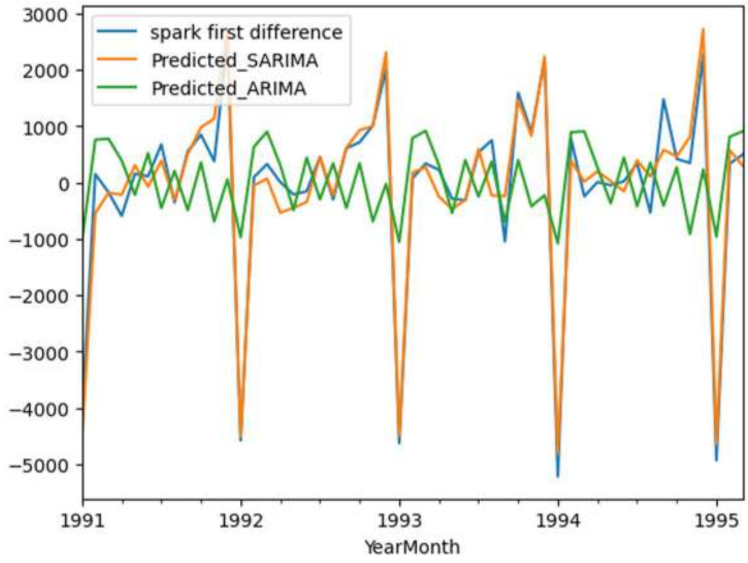
```
test_adfuller(df_rose['rose difference'].dropna())
```

```
ADF statistics: -8.167161332563701
p-value: 8.819857658212933e-13
data is stationary
```

ARIMA/SARIMA

Sparkling dataset





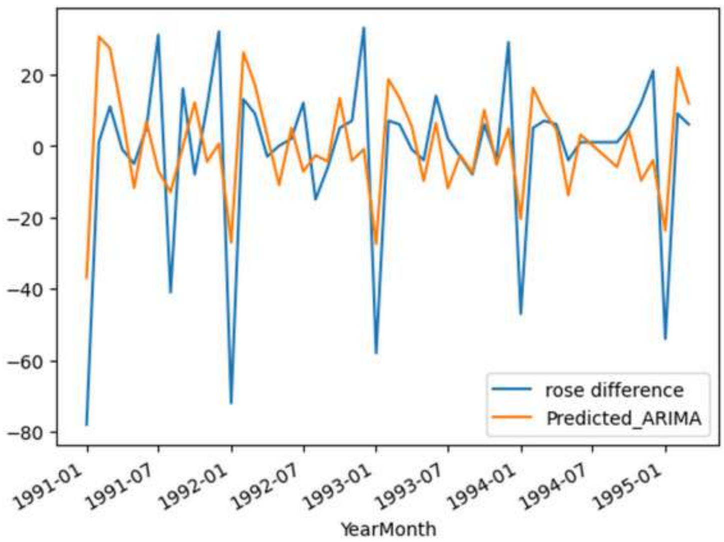

```
rmse = sqrt(mean_squared_error(pred,test_data['Predicted_SARIMA']))
print(rmse)
```

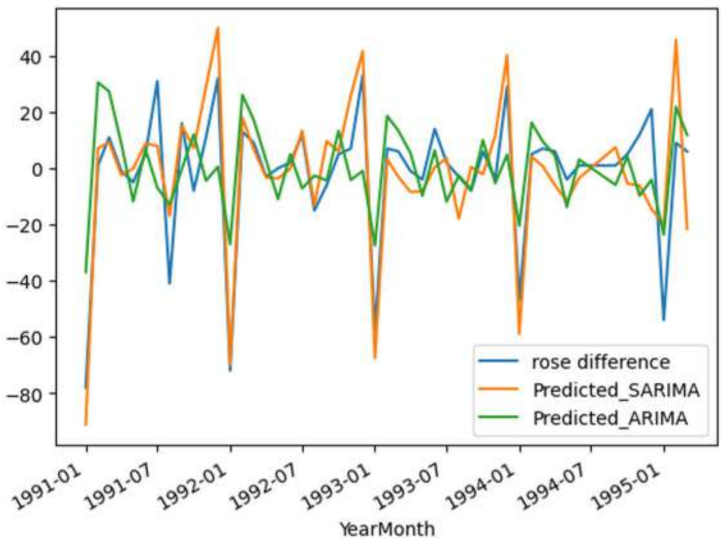
1521.4052082864753

```
rmse = sqrt(mean_squared_error(pred,test_data['Predicted_ARIMA']))
print(rmse)
```

0.0

Rose dataset






```
In [81]: rmse = sqrt(mean_squared_error(pred,test_data['Predicted_SARIMA']))
         print(rmse)
```

21.330329127087015

```
In [82]: rmse = sqrt(mean_squared_error(pred,test_data['Predicted_ARIMA']))
         print(rmse)
```

0.0

Bussiness Insights

First read the getter set, set its info, set its index, plotted the data, checked whether the data is stationary from BP blood test, first the data was not stationary, then it was stationary again

Then plotted ACF and PACF, split the data into train and train test data, then shot the model, ARIMA and SARIMAX

Sales go down in the last of the year and in the starting 1 month of starting year

Where the company should either give more discount to increase the sale or do more marketing so that the sale increases, the company has to give less to the friend.