GEORG-SIMON-OHM
HOCHSCHULE NÜRNBERG

Project work:

Time Series Forecasting with ARIMA and SARIMA.

A machine learning based prediction.

Submitted by:

Rohit Upreti

Dept of Applied Physics and Mathematics.

Matrikelnummer

355627

Submitted to:

Prof. Dr. Elke Wilkczok

Table of Contents

1 Motivation

It will not be a world of man versus machine. It will be a world of man plus machines.

Ginni Rometty.

Former IBM CEO Ginni Rometty's prediction for the future is becoming a reality. Ever since 1960s, for the first-time the term Machine Learning was used, machines are gradually becoming an inevitable part for everyday life. Either heavy industrial machines or small computers and mobile phone, our life is directly affected by its presence.

Machine Learning, as the name suggests, is a guided process, where the Machines learns itself from the given input data, sets of commands and can give future predictions as well. Machine Learning itself is a very broad topic. Artificial Intelligence, Deep Learning, Neural network, Time series analysis are some of the applications of Machine learning.

In this project, a time series analysis of the sales of the UPS (Uninterruptable power supply) from ABB Company in Helsinki, Finland is done. Here ARIMA and SARIMA modules from python are used to make the future sales prediction. Time series and its statistical methods are first discussed and a program in JUPYTER notebook is then written. All the codes are then presented in Code section.

2 Series and Analysis

2.1 Time series and it's components:

Data taken or observation made over a different or regular interval of time are called Time series. The frequency of the data taken can be monthly, daily, hourly etc. Depending upon the frequency, duration of observed period there are some important characteristics of Time series, which needs to be considered before making any further analysis [1]. These characteristics are:

- Trend: Trend in time series indicates the flow of the direction of the data. It is a visual representation, if the mean of the data in the series is increasing or decreasing for the given duration of time.

- Seasonality: Seasonality exhibits a trend that repeats with time, magnitude, and direction. Increase in sales of cooler in summer and increase in number of heating apparatus in winter can be taken as seasonality, as this increase is observed for a particular season.

- Residual: The difference between the trend and the seasonality component is called Residual. It shows any if there are any irregularities or anomalies in the data.

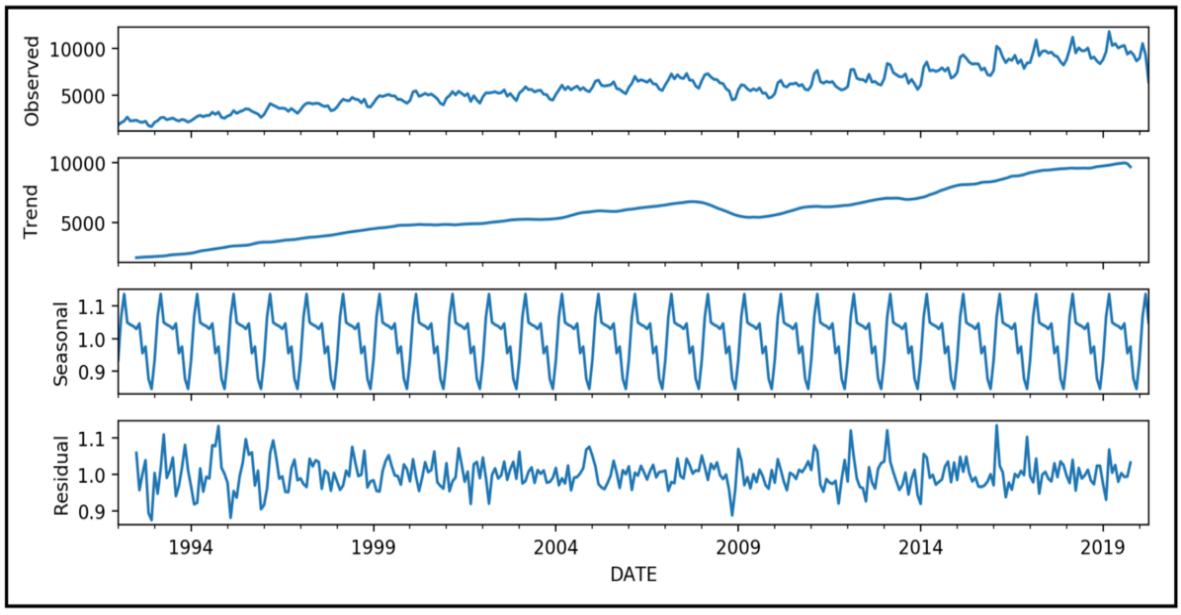

Figure 1 Decomposition into Trend, Seasonality and Residual from observed data.[2]

2.2 Stationarity and Stationarity test:

After decomposing the time series and analyzing the trend and the seasonality, the series is made stationary. Based on trend (increasing and decreasing) and seasonality, the series might have increased and decreasing mean, or repeating mean in case of seasonality. The simplest way to make the series stationary is taking the difference (once or twice). The difference between the consecutive term is known as the first order differencing and difference of the first order is called the second order differencing. Usually, the first order differencing makes the stationary but there are other transformation and log difference, which makes the series stationary.

$$\boldsymbol{d(t) = x(t) - x(t-1)} \ \boldsymbol{.....} \ 1$$

Where, $\boldsymbol{d(t)}$ is the first order difference.

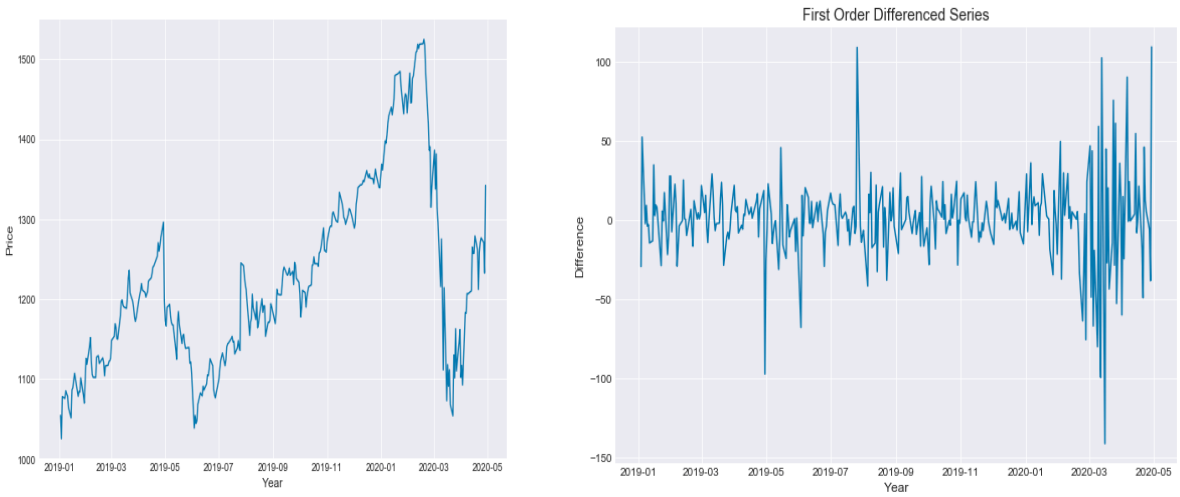

Figure 2 Transformation of non-stationary series to stationary series after differencing.[3]

Stationarity-test:

As stated above, there are few methods to make the series stationarity. For example, log transformation, transformation function, differencing etc. After differencing or taking an appropriate transformation, it is also necessary to check if the series is indeed stationary. For this step as well, there are various steps and most of them are build in python packages under ''Stats models''.

In this project, Augmented Dickey-Fuller Test is done to check the stationarity of the time series. This can be done by importing ''Statsmodel'' package and the ADF test can be done from there. If the p-value (ADF value) is less than 0.05 then the series is considered stationary, if the value is more than 0.05 then the series is not stationary and certain operations must be performed to the stationary again. This method and process of the importing and using this test is shown in code section. [4]

3 ARIMA Time series

3.1 Regression.

Having defined the time series and process to make it stationary, we can now proceed to the statistical method of the forecasting. Before discussing ARIMA and it's term, it is insightful to discuss a rather simpler form of prediction, it's limitation and it's solution.The most common ML algorithm for prediction of Linear regression.

Regression in statistical term relates the dependency between variables. Linear regression then establishes a dependency between two variable. Mean of the series remains constant. And a dependency between Y (dependent variable) and X (independent variable) is made. [5]

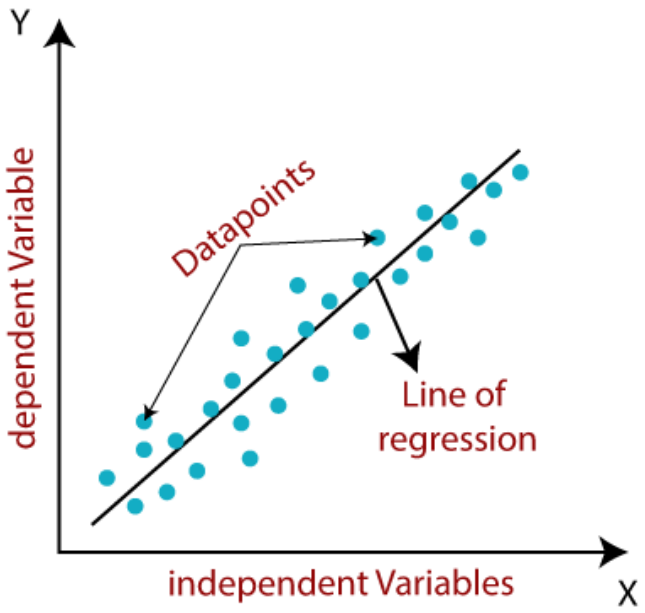

Figure 3 Linear regression between Y and X

And the mathematical equation the describes this relationship is,

$$\boldsymbol{Y} = \emptyset.\boldsymbol{X} + \boldsymbol{C} \quad \text{.....} \ 2$$ (C being the intercept, the value of Y, when X is 0)

Here, Ø is the regression coefficient.

Linear regression is one of the common and simple form of forecasting. But the problem arises when the mean and variance isn't constant, and the seasonality is introduced. To overcome this limitation the autoregressive method is used. In autoregressive model, the prediction is made using the linear combination from past values of variable. And its order depends upon the number of the past values considered for the prediction. Depending upon the number of past values taken, expression for the AR series can be written as:

$$\boldsymbol{y'(t)} = \boldsymbol{C} + \emptyset\boldsymbol{1}.\boldsymbol{y(t-1)} + \emptyset\boldsymbol{2}.\boldsymbol{y(t-2)} + \ldots\ldots + \emptyset\boldsymbol{p}.\boldsymbol{y(t-p)} + \boldsymbol{\varepsilon}$$

Where, $\boldsymbol{\varepsilon}$ is the white noise of the series, $\boldsymbol{\emptyset 1}$, $\boldsymbol{\emptyset 2}$ are the parameters of the Autoregression and the series is order P and $\boldsymbol{y'(t)}$ is the differenced time series.[6]

3.2 ARIMA Model.

ARIMA model stands for Autoregressive Integrated Moving average and is a Machine learning time series algorithm for forecasting. As the name suggestes this model comprises of three components AR(Autoregressive) , I(Integrated), MA(Movingaverage).

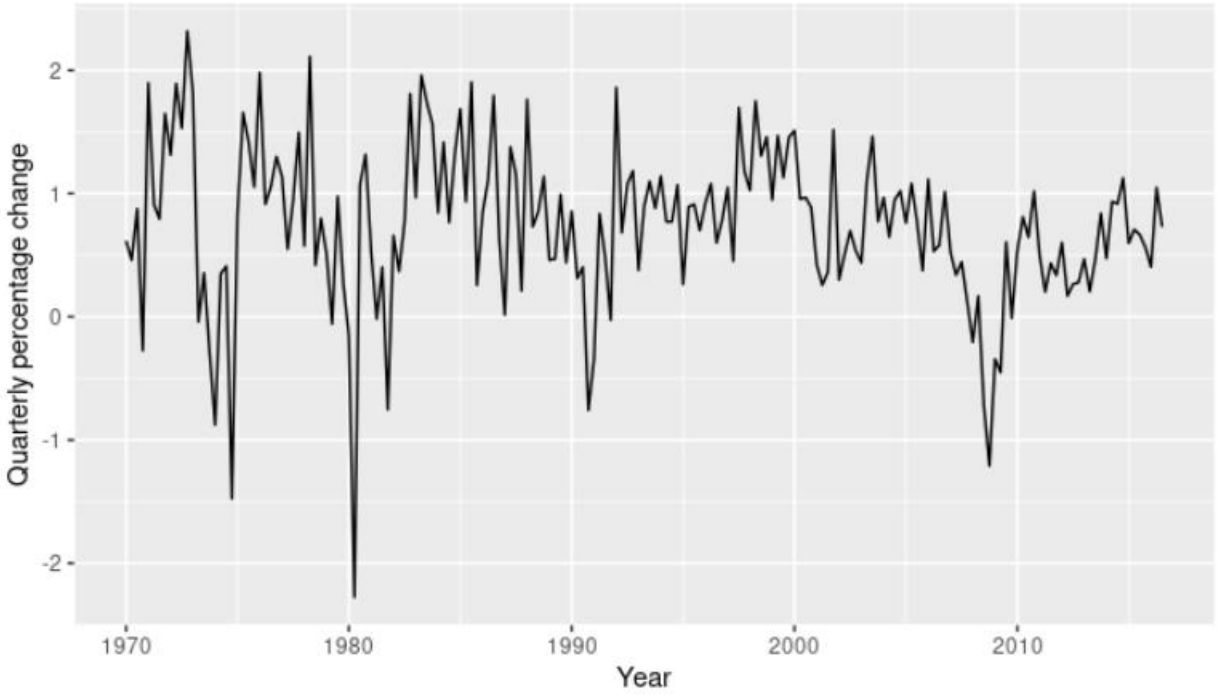

Figure 4 Time series of quarterly percentage change. [7]

The time series above can be taken as an example to break down the components of the ARIMA time model. Here, quarterly percentage change is plotted against time. The series is stationary, and the task is to forecast the change for the next five years.

ARIMA algorithm is used for such prediction, where the variable in Y-axis has varying value for a given period. The spikes are constantly changing. It is then also natural to expect the similar kind of graph in the next coming year as well. While making such prediction, couple of important thing needs to be considered. First the shape of the time series, the shape of the spikes needs to be similar and secondly, the time shift of the spikes needs to be in appropriate order as well. The components in ARIMA models are responsible for this shape and shift of the forecast.

- Autoregressive (p): It is a time series model, where the prediction is based on the past values of the variable. It is denoted by p and is the measure between the Autocorrelation between the data. This value can be obtained by plotting the ACF plot or the optimal value can be calculated by a for loop program.

$$\boldsymbol{y'(t) = C + \emptyset 1 . y(t-1) + \emptyset 2 . y(t-2) + \ldots\ldots + \emptyset p . y(t-p) + \varepsilon}$$

- Integrated (d): Integrated here represents the differencing of the series and is represented by d. Depending upon the order of differencing, d can take the values like 1,2 and 3 etc.

- Moving average(q): A moving average model is also like the AR model but unlike the AR model which uses the past value for the prediction. Moving average model uses the past forecast error in a regression like model and is denoted by q. It is the measure of the partial auto correlation function. For the optimal parameter of p, d and q, a for loop can be employed. Moving average model can be written like AR model as,

$$\boldsymbol{y'(t) = C + \theta 1.\varepsilon(t-1) + \theta 2.\varepsilon(t-2) + \ldots\ldots + \theta q.\varepsilon(t-p) + \varepsilon}$$

  Where, $\boldsymbol{\theta 1}$, $\boldsymbol{\theta 2}$ are the MA parameter and is written together with the past forecast error.

Together with AR, I and MA term ARIMA (p, d, q) time series is formed and then formed series is given by, [8]

$$\boldsymbol{y'(t) = C + \emptyset 1.y(t-1) + \emptyset 2.y(t-2) + \ldots\ldots + \emptyset p.y(t-p) + \theta 1.\varepsilon(t-1) + \theta 2.\varepsilon(t-2) + \ldots\ldots + \theta q.\varepsilon(t-q) + \varepsilon}$$

4 Forecasting

4.1 Non seasonal ARIMA

After defing the basis for ARIMA model, the next step is to make the prediction. A step wise method for this prediction is done in this section.

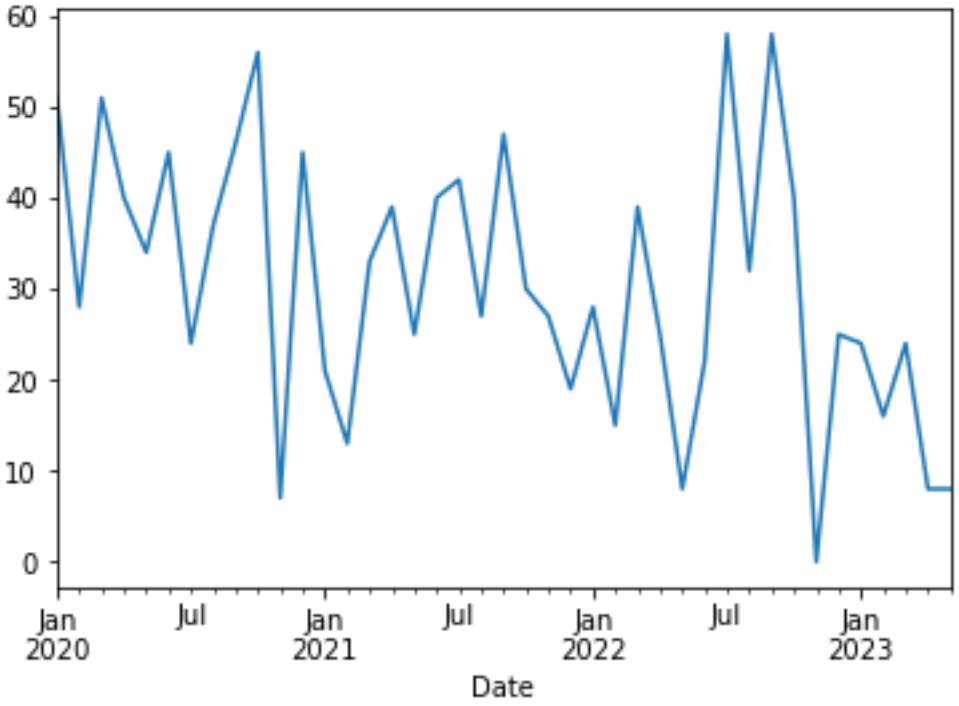

Figure 5 Sales graph in a time series.

The graph above is the sales of the company ABB Helsinki, Finland and is the sample series and the forecast needs to be made in this series. Visually it can be seen that the series is clearly not stationary. Below is discussed the necessary steps to make the series stationary, stationary test, optimal values of p, d, and q.

1. Stationarity check:

   As stated above, for the prediction to be made the series needs to be stationary. There are several methods to determine the stationarity of the series. It can be done by checking the series visually or doing the Augmented dicky fuller test. Upon decomposing the above graph into it's Trend, Seasonality and Residual, following results were obtained.

Figure 6 Decomposition of the data into Trend, Seasonality and Residual.

From above data it can be visually seen that the Trend is decreasing, and a seasonality is present in the series, which makes in not stationary.

The adfuller test value for this series was calculated to be 0.4641, which is greater than 0.05. And thus, the series is not stationary.

```
#Stationarity check and making the series stationarity, value should be less␣
 ↪than 0.05.
from statsmodels.tsa.stattools import adfuller
adftest=adfuller(ts)
print('Pvalue: ', adftest[1])
```

```
Pvalue:  0.46410399478059766
```

The next step in forecasting is to make the series stationary. To do so, differencing is done in the series. After the first order differencing the following series was obtained.

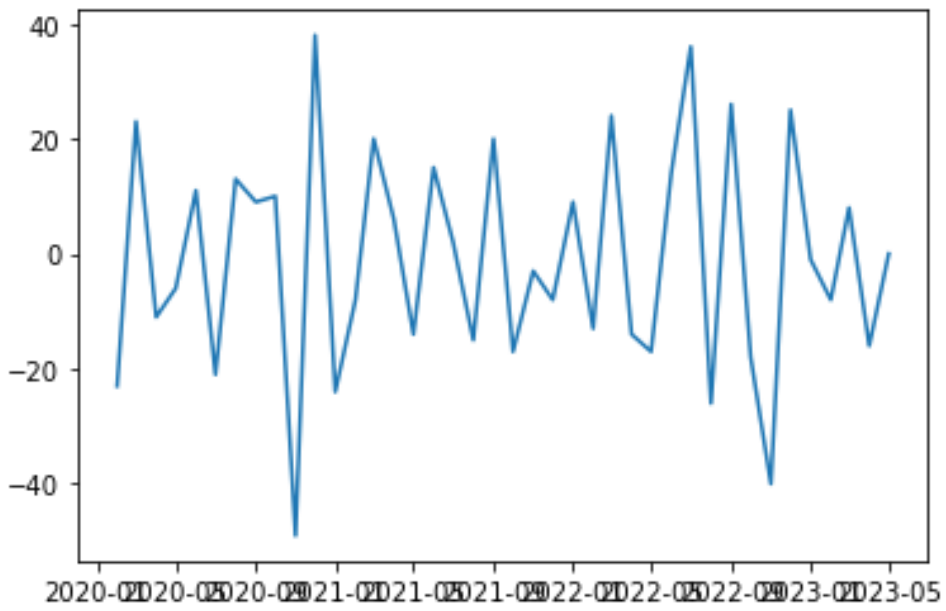

Figure 7 Stationary series after first order differencing.

The series above seems to be stationary after the first order differencing. Upon the adfuller test of the series the adftest value was found to be 0.00323, which is less than 0.05 and the series can be further taken into forecasting.

```
ts1=ts.diff().dropna()
plt.plot(ts1)
```

```
adftest=adfuller(ts1)
print('Pvalue: ',adftest[1])
```

```
Pvalue:  0.0003232378856268717
```

2. Modelling the ARIMA series:
   After the stationary check, comes the important step of the forecasting and modelling the data for the algorithm. There were altogether the data of the 40 months, starting from 2020-01 to 2023-05. For any ML based algorithm prediction, the dataset needs to be classified into training data and the testing data. The algorithm then makes the prediction on the testing data and eventually the final prediction.
   For this prediction, 50% of the data were taken as the training data and the rest 50% were taken as the testing data. This was done via simple python command.
   train = ts1[:20]
   test = ts1[20:], ts1 represents here the first order differenced time series.

After splitting the training and testing data, the ARIMA package from the "statsmodels" is imported. To understand the nature of the ARIMA series, first the basic parameters were selected.

(1,1,0) model:

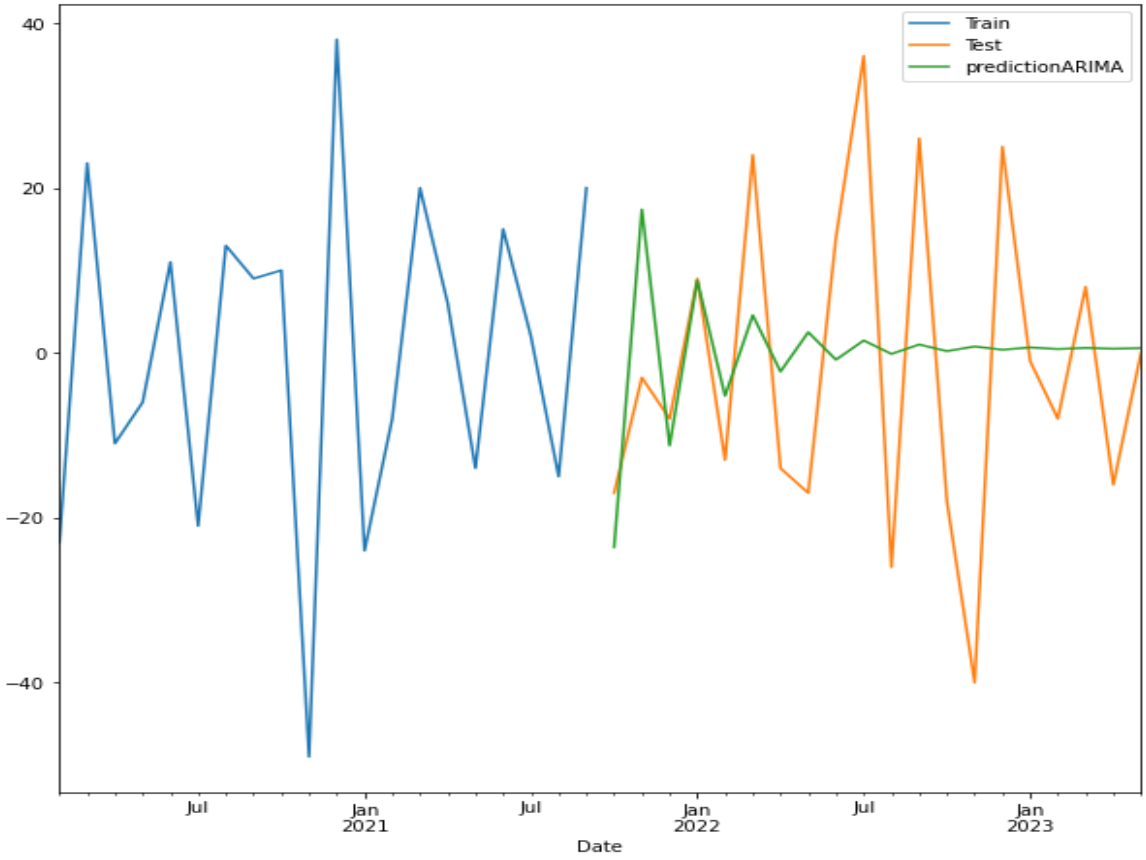

Figure 8 ARIMA prediction for (1,1,0)

(0,1,1) model:

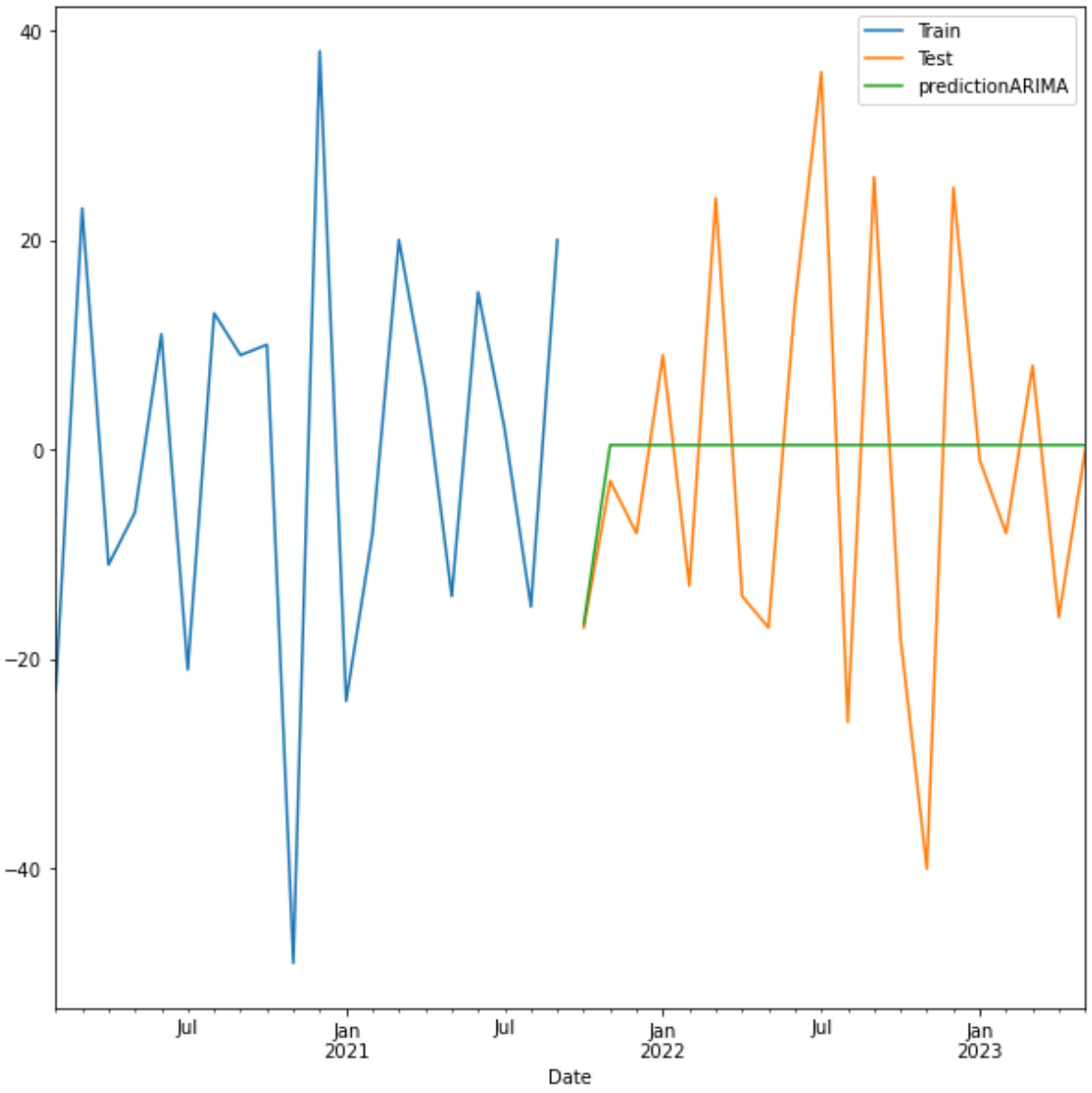

Figure 9 ARIMA prediction for (0,1,1)

Both (1,1,0) and (0,1,1) model gives a visual representation about how the prediction looks for simple basic input parameter. Both AR and MA model is responsible for the change time series while forecasting.

3. Finding the optimal values for p, d, and q:
   As seen from the (1,1,0) and (0,1,1) model, the prediction for the testing data is not quite up to the order. The prediction tends to cover the testing data, but it seems to have significant error. Therefore, it is essential to have the optimal parameter to make the proper forecast.
   Finding the optimal value of p, d, and q:

The optimal value of the p, d and q can be found by my multiple ways. ACF and PACF plot are the one of the common methods. Another method is making a loop function for the ARIMA of the order p, d and q and find all possible combination with least mean squared error. Then the parameter with least RMSE is then taken for the final ARIMA forecast. The code of the loop function is given in CODE section. From this loop combination, the optimal values were found to be (7,1,2) and the forecast for the next 20 months was:

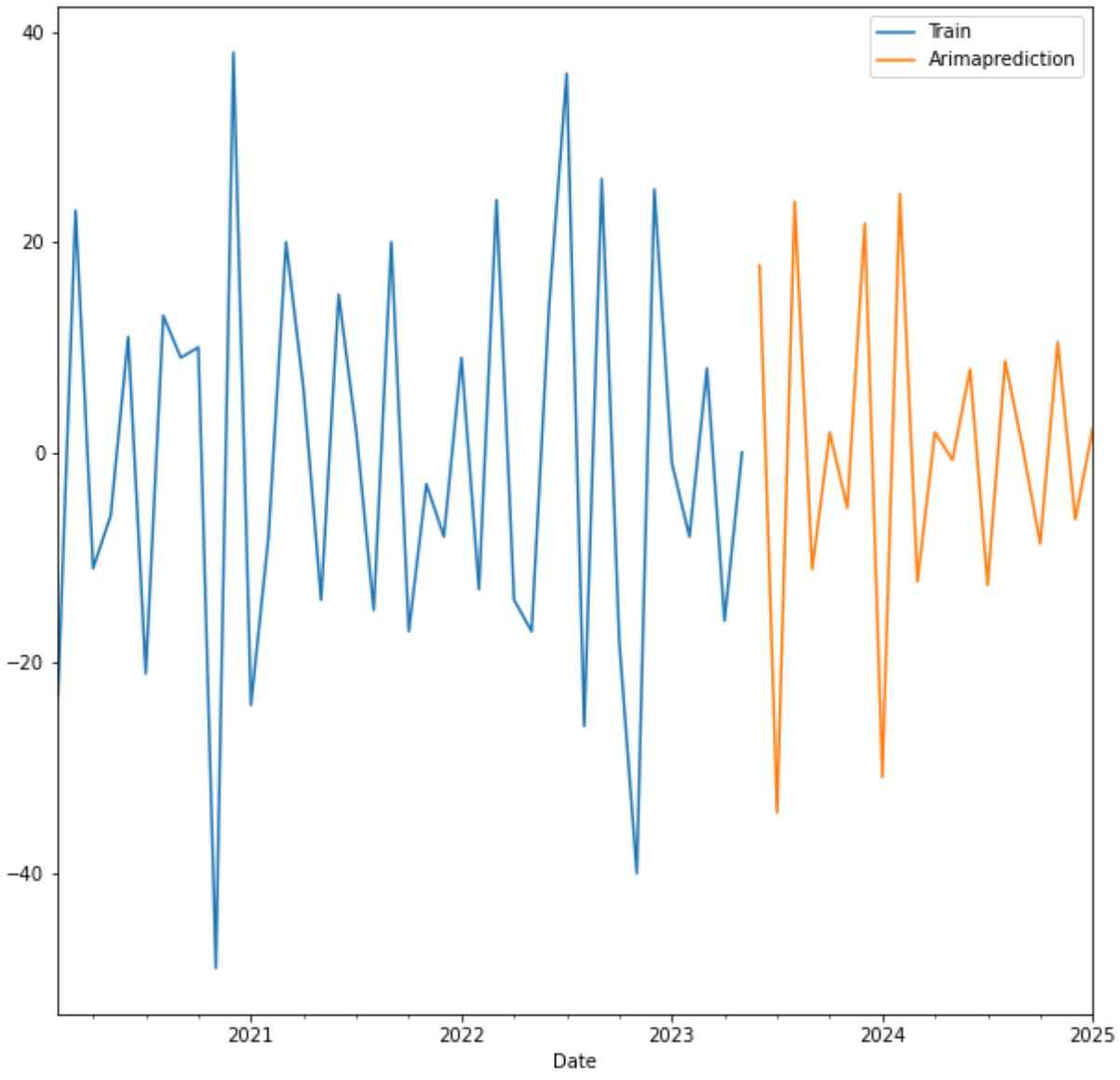

Figure 10 ARIMA (7,1,2) model.

And the root mean squared error between the test and prediction was found to be 24.37659.

4.2 Seasonal ARIMA

As stated above Trend, and Seasonality are characteristics of a time series. ARIMA model is used to make the prediction for non seasonal time series. For the seasonal time series, Seasonal ARIMA algorithm or SARIMA is used. Like the ARIMA model, this model also consists of parameters. The SARIMA parameter is given by (p, d, q) (P, D, Q, S). The term (p, d, q) is term taken from ARIMA, where there is no seasonality in series. And the term (P, D, Q, S) is introduced for the part where the seasonality and the term S is the length of seasonality and in this time series S takes the value of 12.

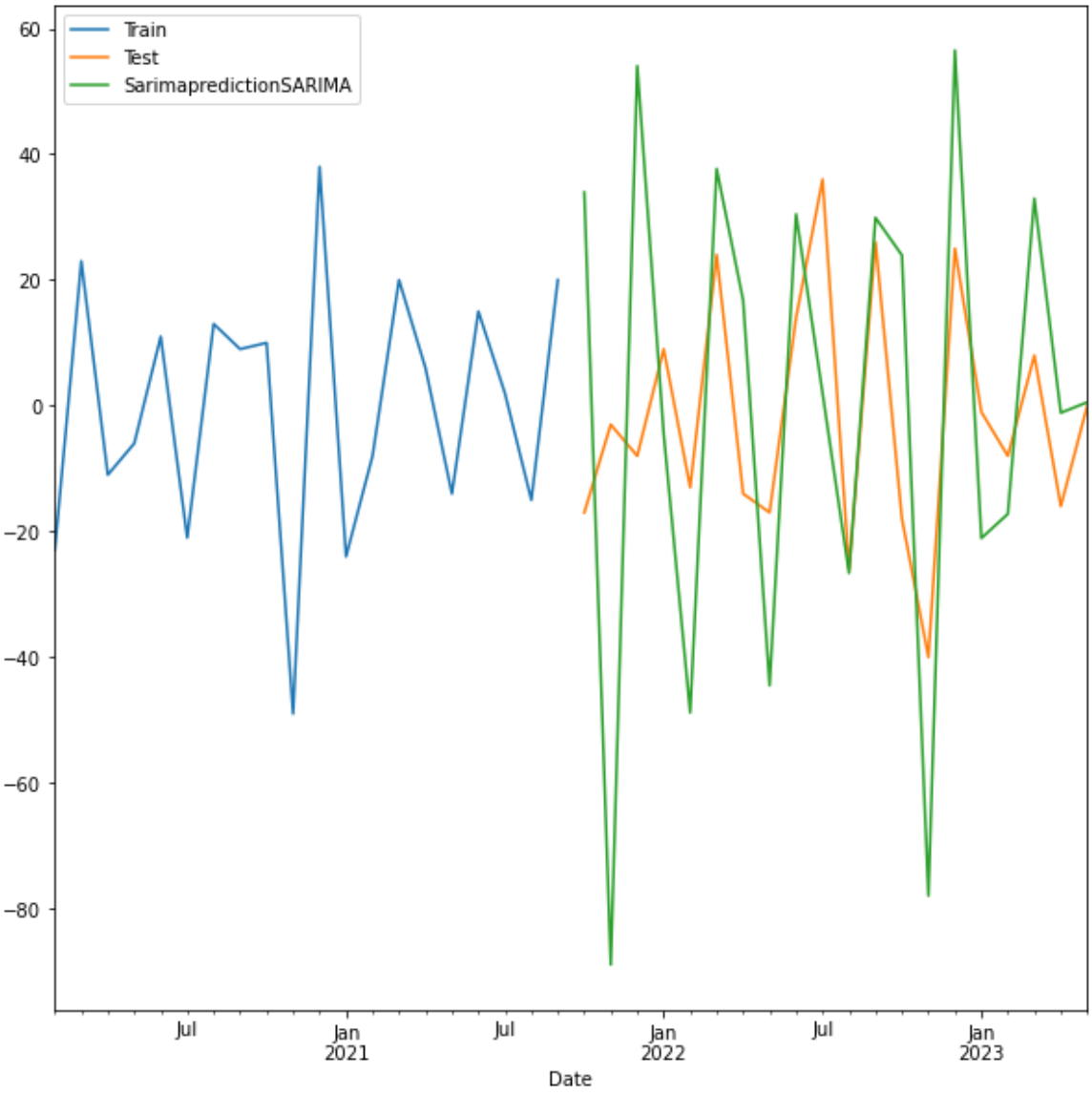

Figure 11 SARIMA model (7,1,2) (3,1,3,12)

The above SARIMA prediction is done from the random sets of parameter. It can be seen that taken parameter are not of the suitable order as the prediction doesn't quite overlap with the test series. Like ARIMA forecast, there needs to be optimal parameter for the SARIMA model as well. This optimal parameter can be found by a loop function and is in CODE section.

Then the optimal parameters were found to be (3,1,3)(3,1,3,12) and the forecast was:

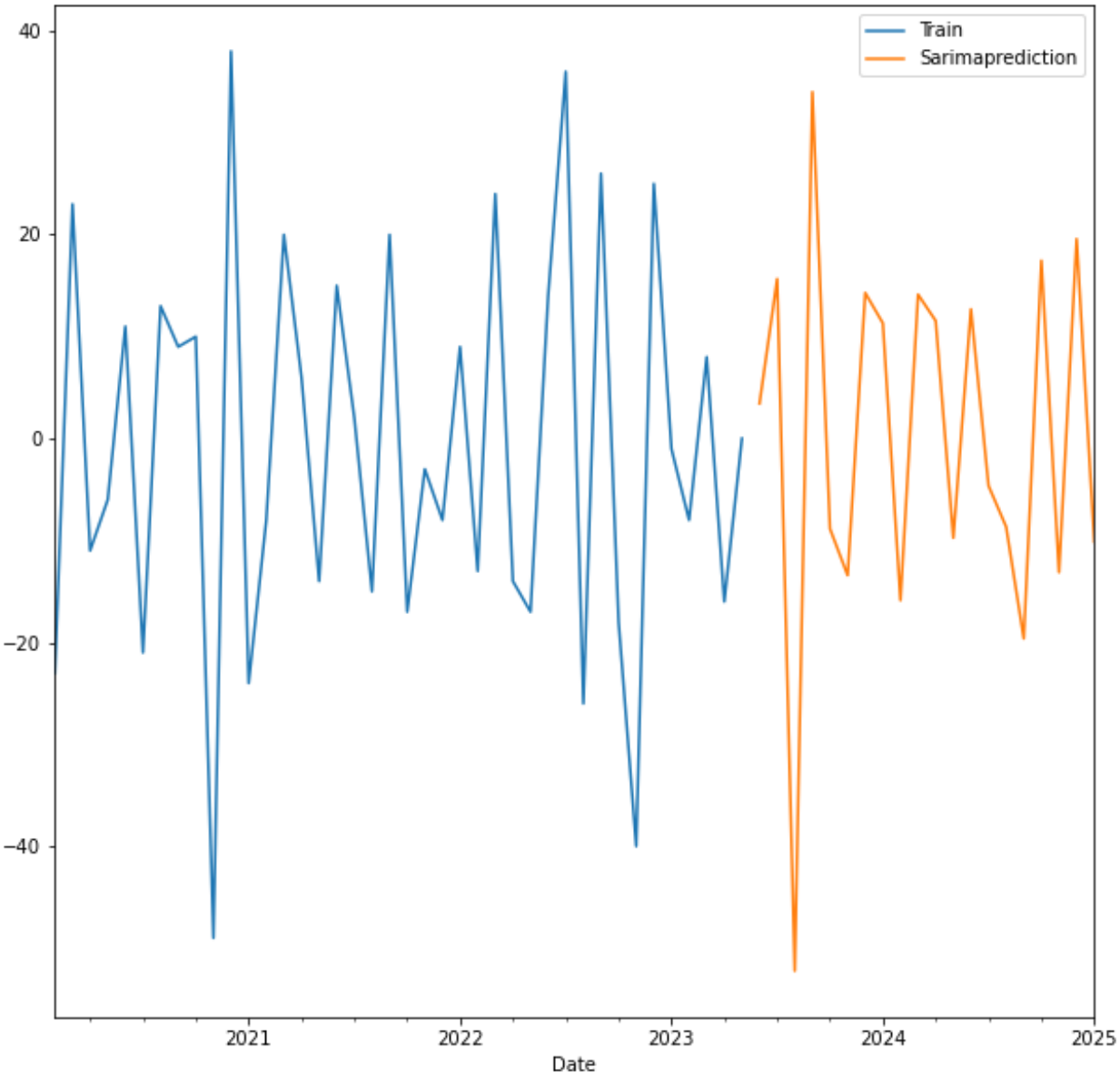

Figure 12 SARIMA forecast (3,1,3) (3,1,3,12)

With the RMSE of 28.77

ARIMA and SARIMA forececast together with the original series.

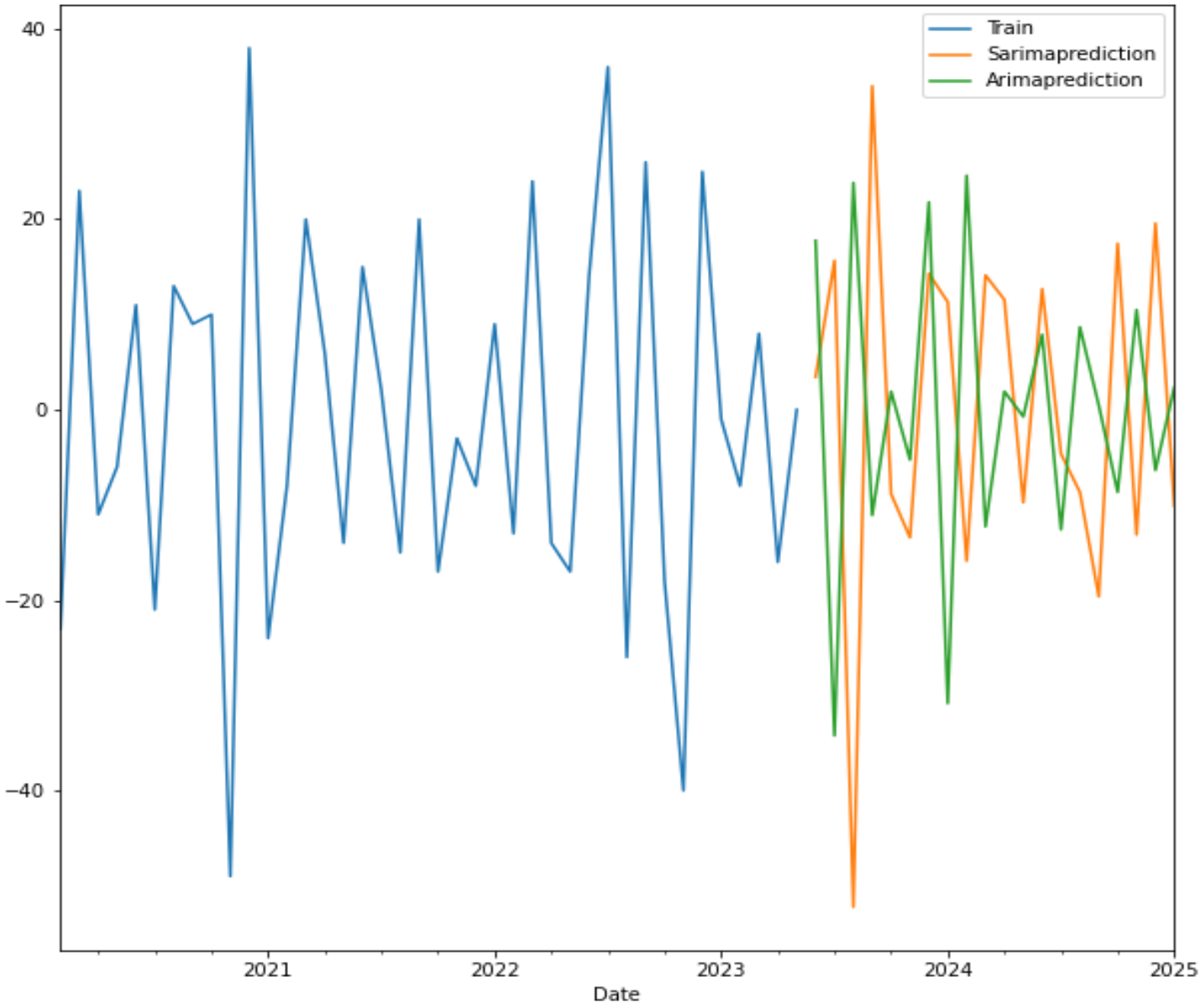

Figure 13 Side by side plot ARIMA and SARIMA forecast.

5 Summary

The task of this project was to make a time series forecast using Machine learning. Python was used as the programming language with ARIMA and SARIMA being the machine learning algorithm. To do this forecast, various packages from the python was employed.

For this time series analysis, a real time sales data from ABB company Helsinki, Finland was taken. The company makes industrial appliances and for this project, the sales of the UPS(uninterruptable power supply) was taken. The task was to make the sales forecast for the coming month. Pandas was used to import the data in the JUPYTER notebook. Numpy and matplotlib were also imported for the numerical and graphics purpose.

The first step was to decompse the series in it's components, this was done by importing statsmodel.api. The series was decomposed into Trend, Seasonality and Residual. After analyzing the decomposed series, the series was found to be not stationarity on Trend and Seasonality. The next step was to check the stationarity. To check the stationarity, statsmodels.tsa.stattstools was imported and the adftest was done. And the series was made stationary by differencing the series.

After the series was made stationary, the data split into training and testing data. 50% of the data was taken as training data and 50% of them were taken as training data. Then from statsmodel, ARIMA was imported. The series was then modeled ARIMA and from the loop function the optimal parameter for the p, d and q were found. And the forecast for the next 20 months was made. Inorder to check the error, from sklearn.metrics, mean squared error was import and the error between the test and prediction was calculated.

After the non seasonal ARIMA was predicted, seasonal ARIMA forecast was done. This module was imported from statsmodel.tsa.statespace. Like ARIMA model, SARIMA model also requires the parameter (p , d, q) (P, D, Q, S). From the loop function the parameter were calculated and the mean squared error as well.

6 Reference:

[1] https://www.kaggle.com/code/complete-guide-on-time-series-analysis-in-python

[2] https://timeseriesreasoning.com/contents/time-series-decomposition/

[3] [4] https://blog.quantinsti.com/stationarity/

[5] https://www.javatpoint.com/linear-regression-in-machine-learning

[6] [7] [8] https://otexts.com/fpp2/non-seasonal-arima.html

7 CODE

```
import os
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
```

```
df=pd.read_csv('C:/Users/upretiro81935/Documents/datafile.csv',␣
↪parse_dates=['Date'], index_col=['Date'])
df.head()
```

```
#stationary check.
import statsmodels.api as sm
decomposition = sm.tsa.seasonal_decompose(ts, model='additive')
fig = decomposition.plot()
plt.show()
```

```
ts1=ts.diff().dropna()
plt.plot(ts1)
```

```
len(ts1)
```

40

```
train=ts1[:20]
test=ts1[20:]
```

```
from statsmodels.tsa.arima_model import ARIMA
```

```
model=ARIMA(train,order=(0,1,1)).fit()
```

```
import warnings
warnings.filterwarnings('ignore')
```

```
pred=model.predict(start=len(train),end=len(ts1)-1)
```

```
from sklearn.metrics import mean_squared_error
error=np.sqrt(mean_squared_error(test,pred))
error
```

```
import itertools
p=range(0,8)
q=range(0,8)
```

```
d=range(0,2)
pdq_combination=list(itertools.product(p,d,q))
len(pdq_combination)
```

128

```
rmse=[]
order1=[]
for pdq in pdq_combination:
    try:
        model=ARIMA(train,order=pdq).fit()
        pred=model.predict(start=len(train),end=(len(ts1)-1))
        error=np.sqrt(mean_squared_error(test,pred))
        order1.append(pdq)
        rmse.append(error)
    except:
        continue
```

```
results=pd.DataFrame(index=order1,data=rmse,columns=['RMSE'])
```

```
results.to_csv('ARIMA.csv')
```

```
from statsmodels.tsa.statespace.sarimax import SARIMAX
model_1=SARIMAX(train,order=(7,1,2),seasonal_order=(3,1,3,12)).fit()
pred=model_1.predict(start=len(train),end=len(ts1)-1)
```

```
import itertools
p=range(1,4)
d=range(1,2)
q=range(1,4)
s=range(10,12)
pdq = list(itertools.product(p,d,q))
```

```
seasonal_pdq=list(itertools.product(p,d,q,s))
for param in pdq:
    for param_seasonal in seasonal_pdq:
        try:
            model=SARIMAX(train,order=param,seasonal_order=param_seasonal)
            result=model.fit()
            print('SARIMAX{}{} -AIC:{}'.format(param, param_seasonal,result.
 ↪aic))
        except:
            continue
```