# Hybrid forecasting algorithm for pollution index

## Using ETS ARIMA model

**Your name**

**12/04/2023**

# Table of Content

# 1. Introduction

Developing a hybrid forecasting model for time series prediction is a complex task which involves combining different techniques like Pre-processing (Pre-ETS), ARIMA (Autoregressive Integrated Moving Average) So, Let's break down the project into individual tasks:

1. Pre-processing (Pre-ETS):

Pre-processing techniques involve data cleaning, handling missing values, outliers, and possibly applying ETS (Error-Trend-Seasonality) models to capture the inherent patterns before feeding it to the forecasting models. ETS models help capture seasonality, trend, and noise components in the data.

2. ARIMA (Autoregressive Integrated Moving Average):

ARIMA models are powerful time series models that can capture linear relationships in the data, accounting for autocorrelation and seasonality after differencing the data. These models work well when there's a stationary component in the time series data. A traditional Arima model is denoted as ARIMA(p,d,q).

- p (AR - Autoregressive): Represents the number of lag observations included in the model. It captures the effect of past values on the current value. A higher 'p' value implies a more complex relationship with past observations.
- d (I - Integrated): Denotes the degree of differencing needed to make the time series stationary.
- q (MA - Moving Average): Represents the order of the moving average. It captures the relationship between the current observation and the residual errors from a moving average model applied to lagged observations.

## 2. Environment setup

Create a python virtual environment and install the following packages using pip:

I. NumPy
II. Pandas
III. Matplotlib
IV. Scikit-learn
V. Statsmodel
VI. TensorFlow

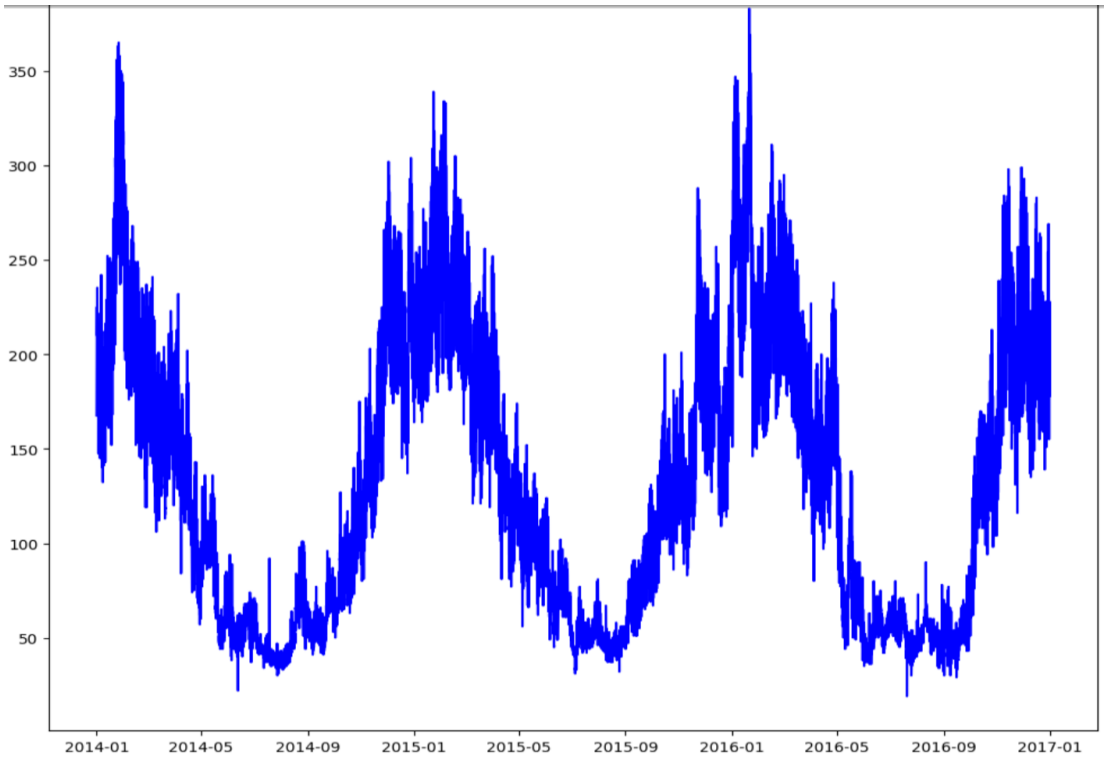
## 3. Development Procedure

A. Data Preprocessing

- The dataset is loaded with pandas read_csv method
- Missing or empty values are removed from the dataset
- Index values are set to Datetime index so that we can use time periods for index slicing.

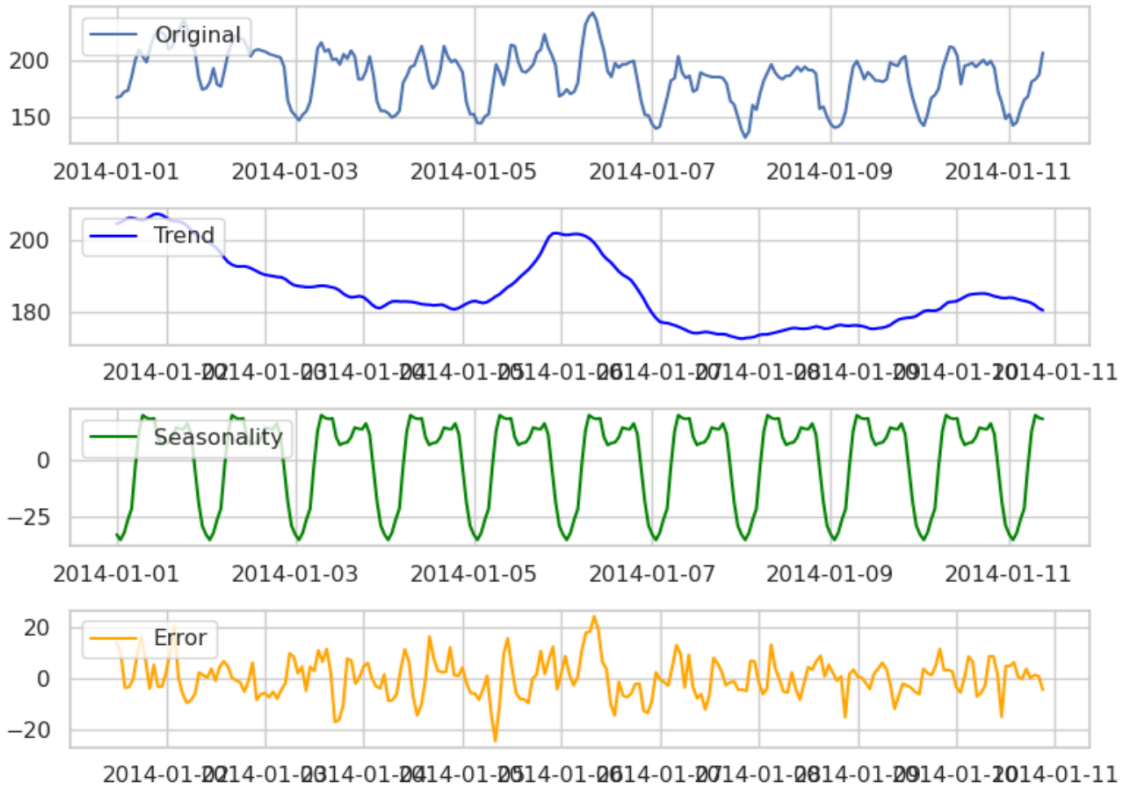
B. Initial Exploratory Data Analysis (EDA)

In this step we will use Initial Exploratory Data Analysis (EDA) to understand its structure, patterns, and potential insights through summary statistics and visualizations.




C. ETS seasonal decomposition of time-series data

In this step, we will use statsmodel seasonal_decompose method to break down our original time series data into 3 components. Error, Trend and Seasonality.




D. AD Fuller Test for stationarity check

To train the arima model on the seasonal data, first we need to make sure that the data is stationary. Stationary means when the mean and variance of the dataset isn't changing w.r.t time. In our test we observed that the dataset is indeed stationary.

```
ADF Statistic: -16.33832022915397
p-value: 3.0132415712678645e-29
Critical Values: {'1%': -3.466398230774071, '5%': -2.8773796387256514, '10%': -2.575213838610586}
```

E. Splitting dataset into training and test set

In this step we will split the dataset in the ratio of 80/20 with 20% of the dataset allocated to test set. To do this, we will use train_test_split method from scikit-learn library.

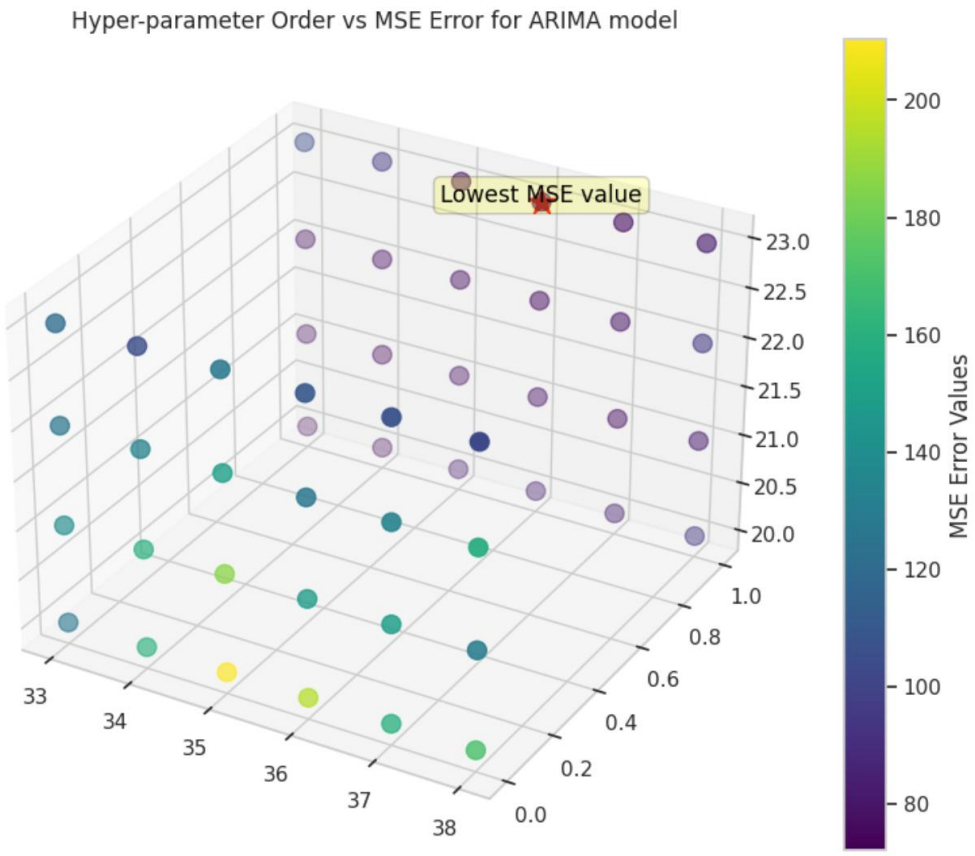
F. Model Training with hyper parameter tuning

We will train 2 models, both using hyper param tuning to find best parameter set to achieve lowest mean squared error(MSE) values.

1. ARIMA model
2. Pre-ETS-ARIMA model

**ARIMA model**

Arima model contains 3 parameters p, d, q. We need to find a parameter set which will minimize the mse error. We will be needing the model on the training set. To do this, we will use a custom function which will go through several values of p, d and q and find the best set. The code for training the model is given in the provided Jupyter notebook. After training the model, we can visualize the different parameter set and the respective mse values and locate the set which gave lowest mse value.

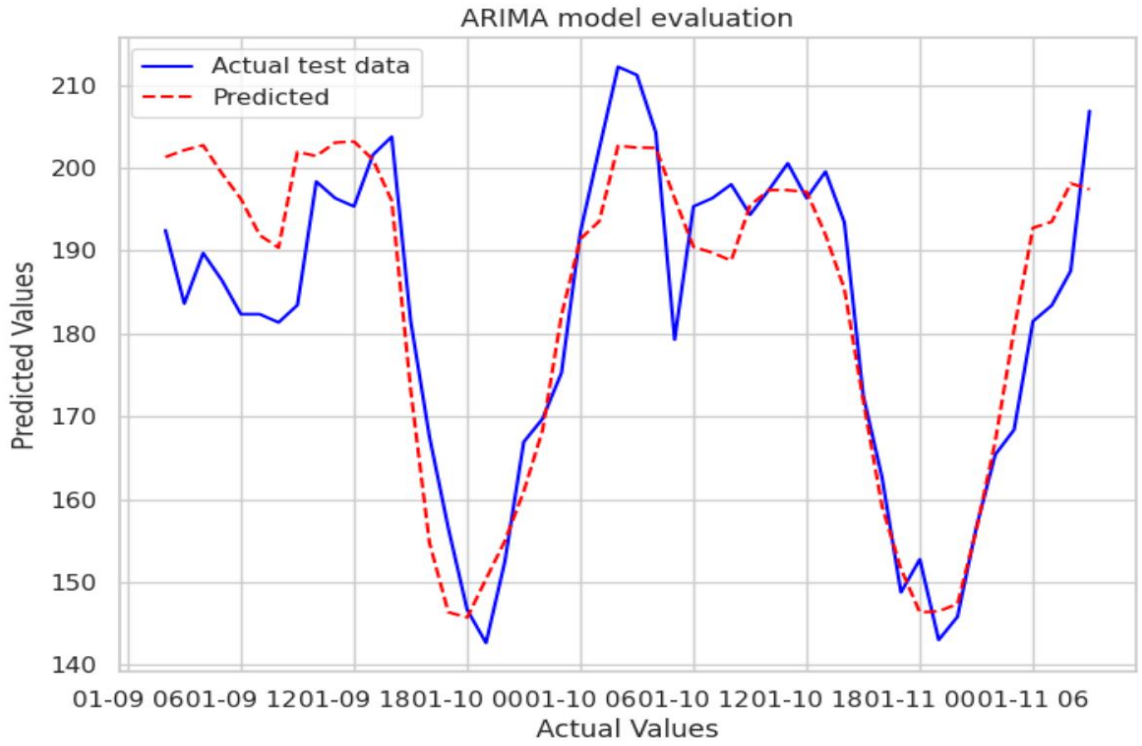



**ARIMA model Evaluation**

We can use the test set to determine how good the model is performing on unseen data. To do this, we will use R-squared and RMSE metrics. The results are:

```
R-squared: 0.804462132517455
RMSE: 8.494552603630757
```

## Visualization of predictions vs test set

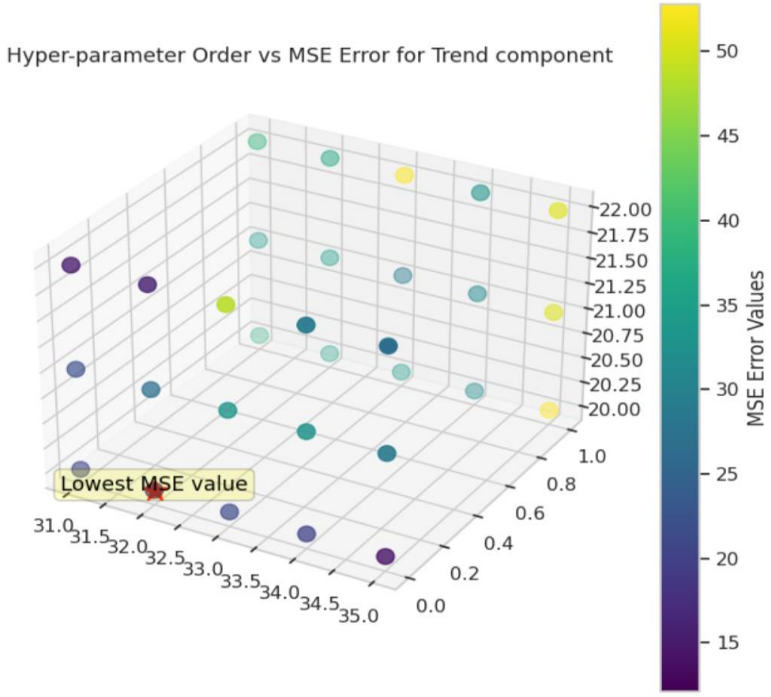



## Pre-ETS-ARIMA model

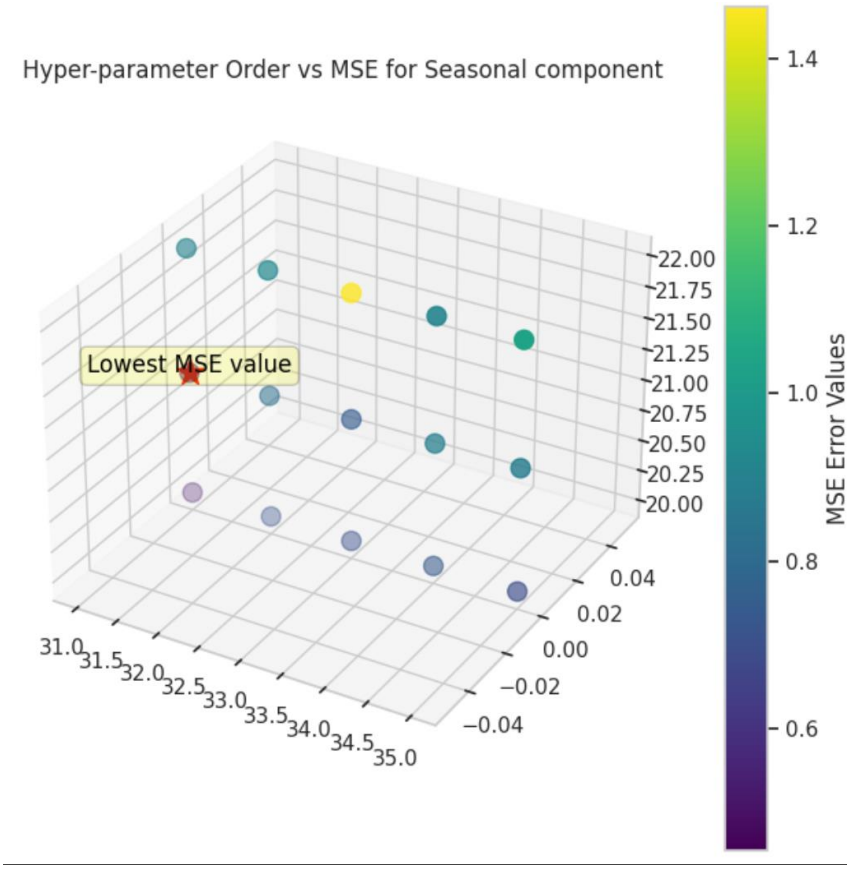
Using a hybrid model can allow us to achieve better results. Decision to build a hybrid model is taken when a baseline model performance is not sufficient to justify its use in real life situations. Since in this project we are dealing with a time series dataset which exhibits seasonal trends, we can use ETS decomposing to break our original time series data into 3 components namely Error (E), Trend (T) and Seasonal(S). And then, we will train Arima model separately on each component with hyper param tuning. After that, we will make predictions on each component and then combine the 3 predictions to reconstruct our predicted time series. The code to perform seasonal decomposition and training arima model in

given in the provided Jupyter notebook. After training the model, we can visualize the different parameter set and the respective mse values and locate the set which gave lowest mse value.

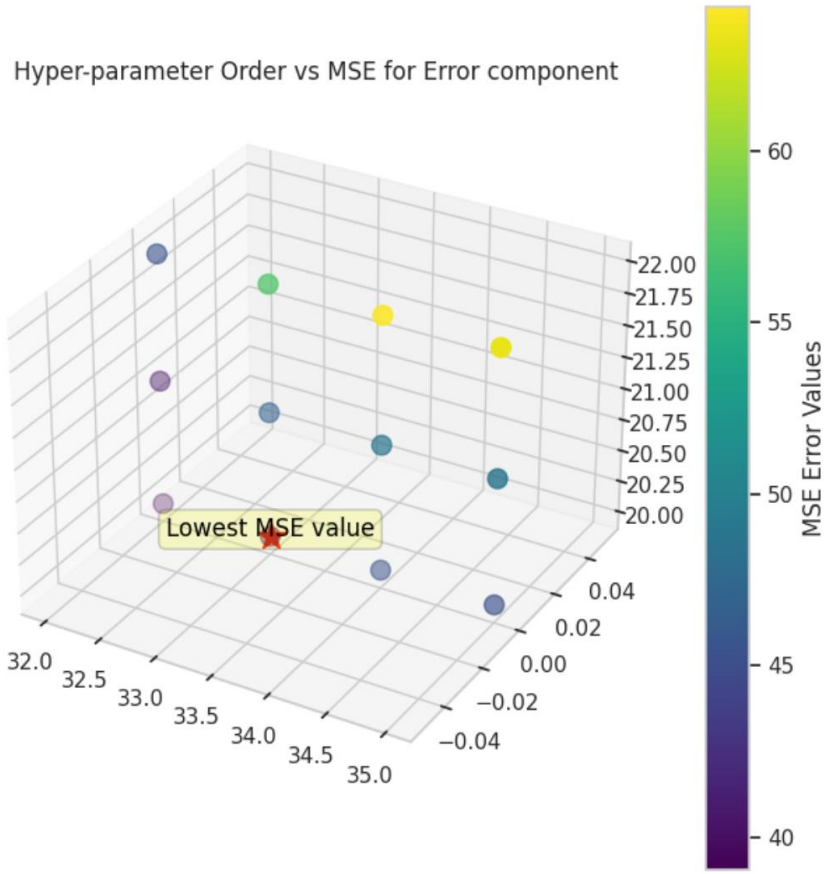
- Visualizing different parameter set for ‘trend component’ with their mse values



- Visualizing different parameter set for ‘seasonal component’ with their mse values



- Visualizing different parameter set for ‘error component’ with their mse values



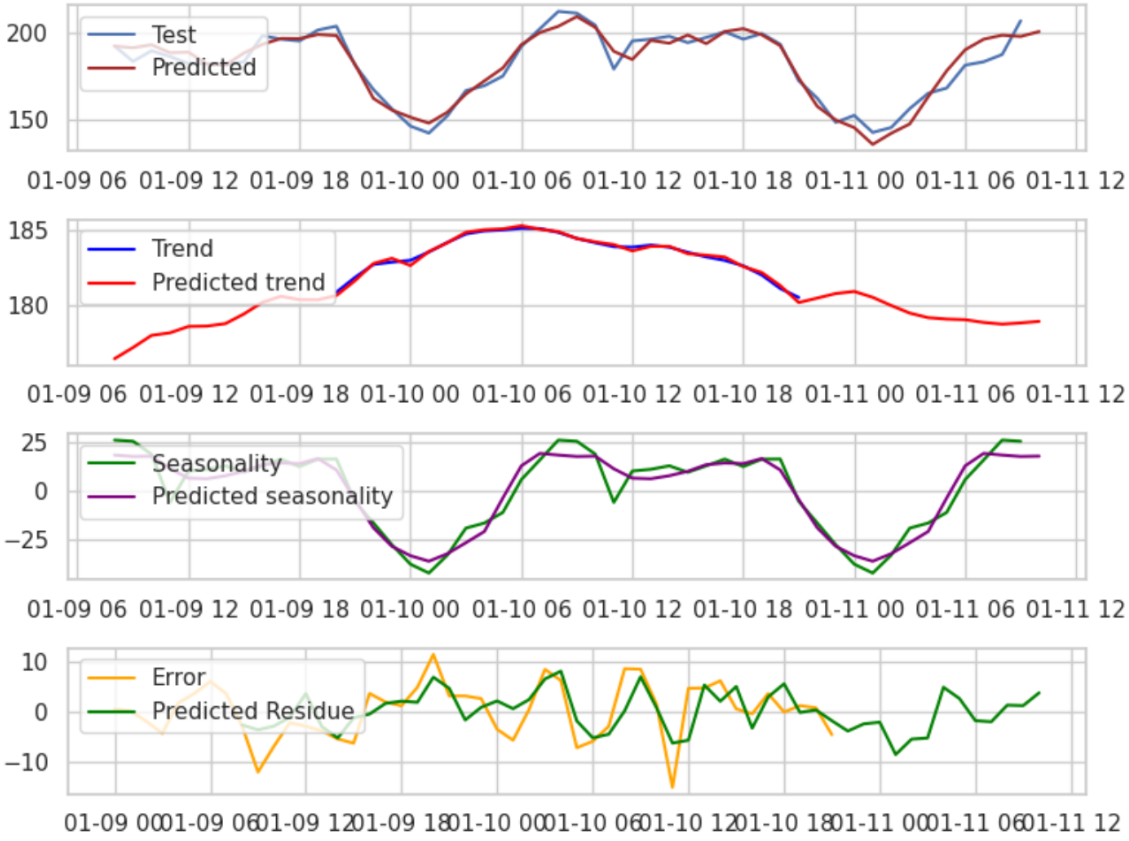

**Pre ETS ARIMA model Evaluation**

We can use the test set to determine how good the model is performing on unseen data. To do this, we will use R-squared and RMSE metrics. The results are:

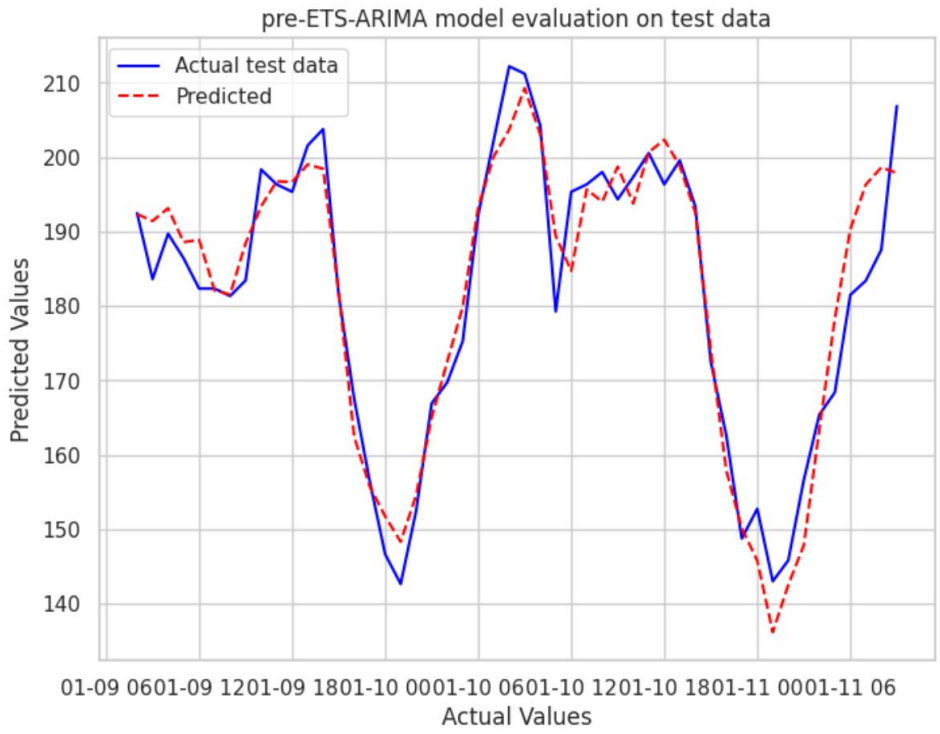
```
R-squared: 0.9203529730662968
RMSE: 5.421380101369927
```

G. Visualizing Reconstructed time series from step 3.c

In this step, first we will compare the seasonal decomposition of test set and the predicted E,T and S components of test set.



## Visualization of predictions vs test set




### H. Comparison between ARIMA and ETS-ARIMA model

To understand which model performed better, we will use RMSE metric on the set to evaluate Arima model and pre-ETS-Arima model.

- RMSE for ARIMA model is **8.4945**
- RMSE for ARIMA model is **5.4213**

So, the hybrid ets-arima model is 36.19% better than the arima model.