Hybrid forecasting algorithm for pollution index

Pre-ETS-ARIMA hybrid model

# 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) where

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.

## Step 1: Install necessary libraries

We need several packages for this project. Here is the list:

1. NumPy
2. Pandas
3. Matplotlib
4. Scikit-learn
5. Statsmodel
6. TensorFlow

## Step 2: Load the time series data and perform data preprocessing

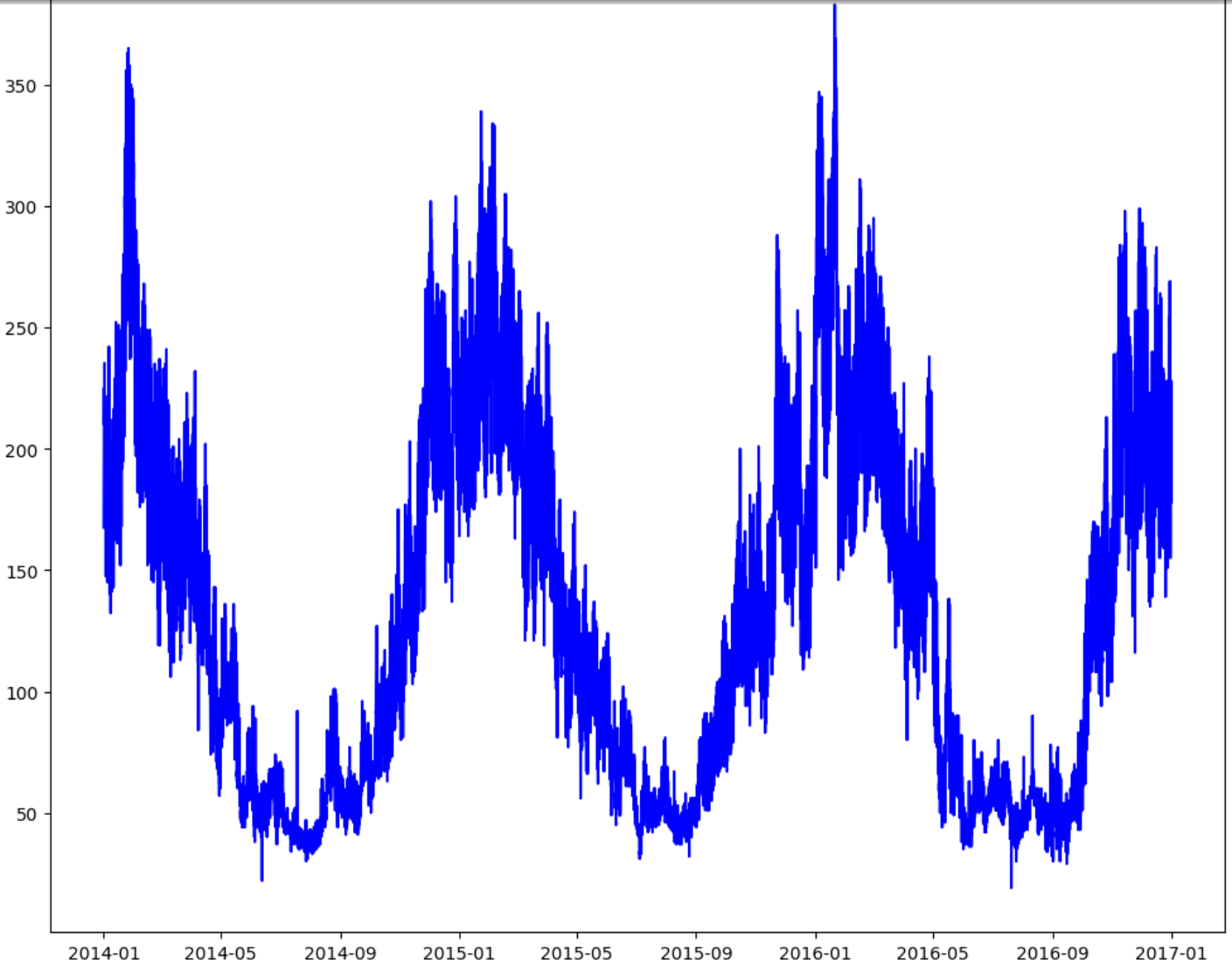
2.1. The dataset is loaded with pandas read\_csv method

2.2. Missing or empty values are removed from the dataset

2.3. Index values are set to Datetime index so that we can use time periods for index slicing.

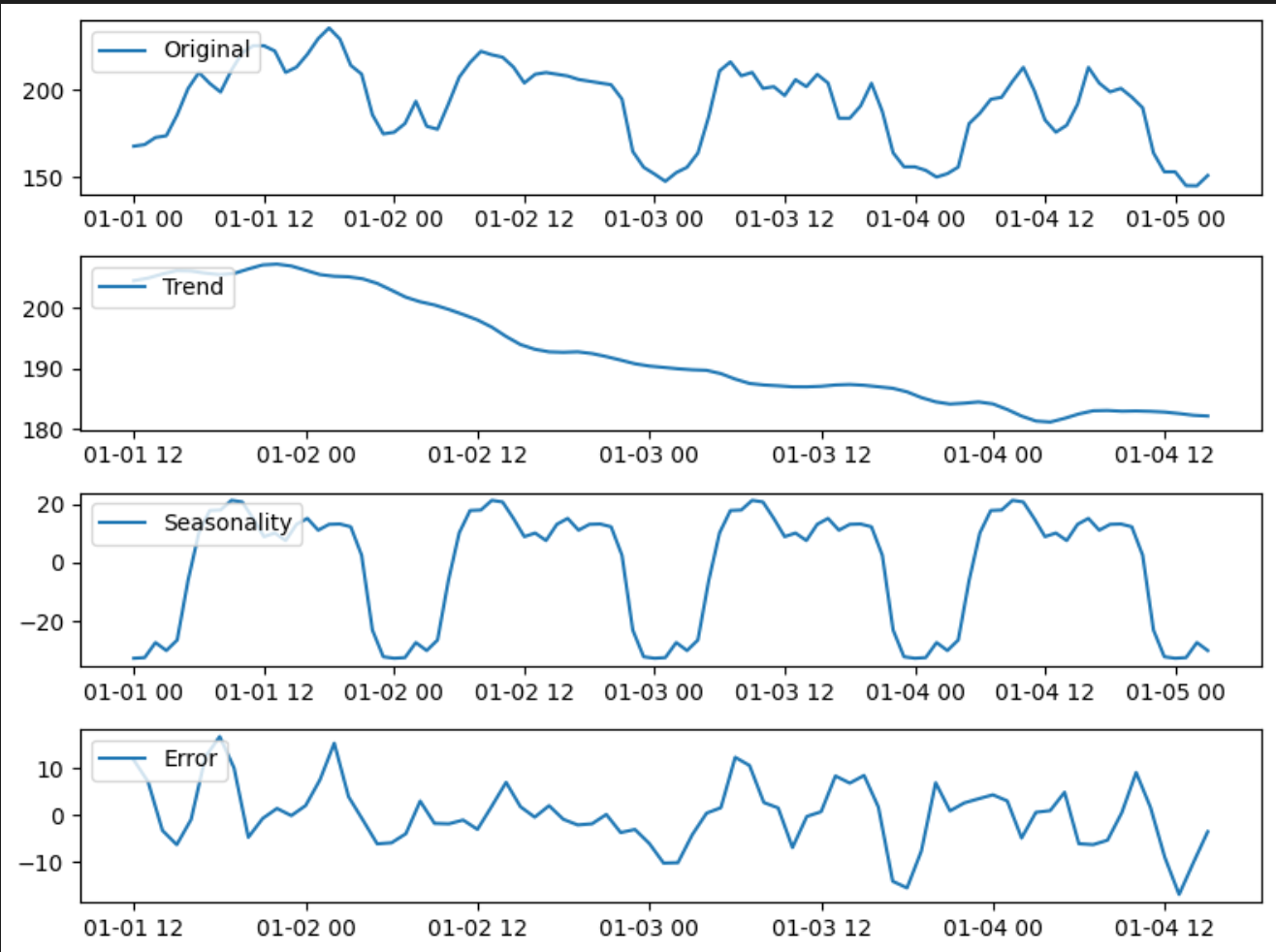
## Step 3: Initial EDA

In this step we take a look at the data



## Step 3: Pre-ETS Seasonal decomposition

In this step, we use statsmodel seasonal\_decompose method to break down our original time series data into 3 components. Error, Trend and Seasonality. Here’s how it looks:



## Step 4: Use AD Fuller test to check for stationary of the seasonal data

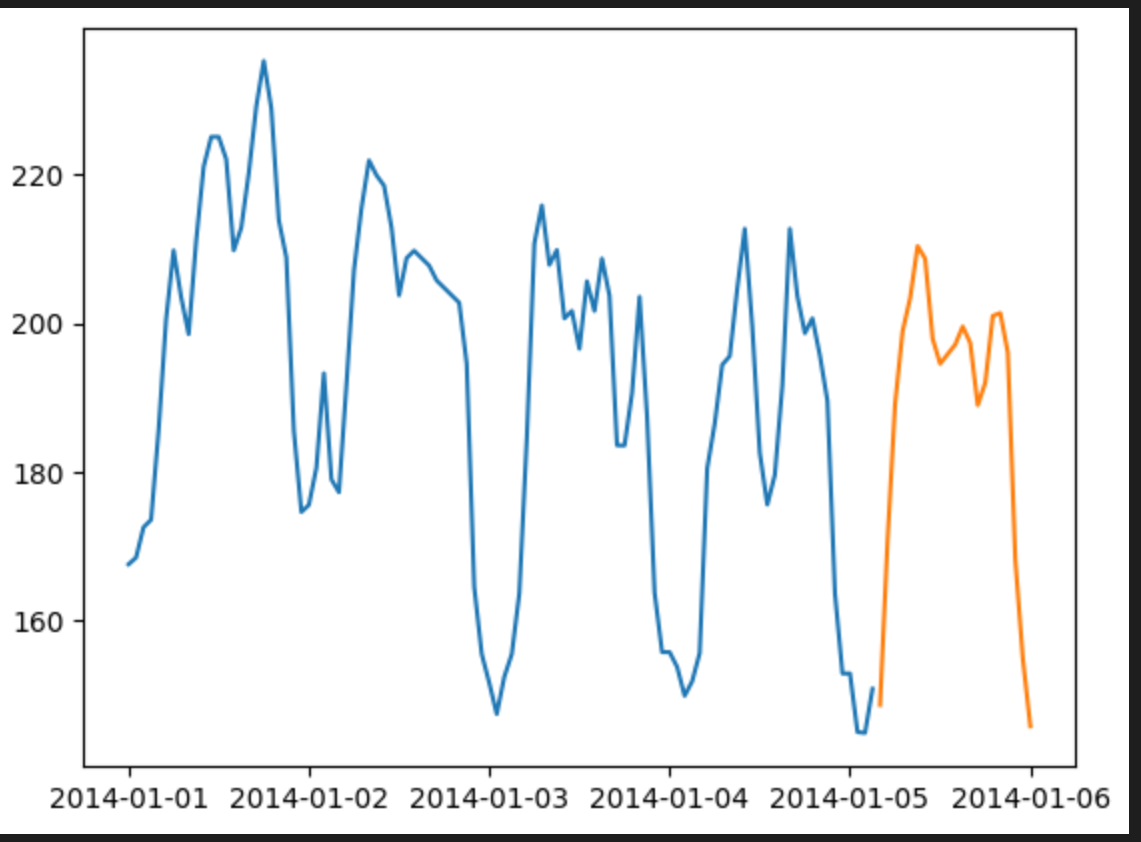
To train the arima model on the seasonal data, first we need to make sure that the data is stationary. Stationary means when the mean and variance of the dataset isn’t changing w.r.t time. In our test we observed that the dataset is indeed stationary

## Step 5: Train the arima model and use hyper-param tuning to get best param-set

Seasonality means the repetition in the pattern of the time series data. But, how do we know after how many days or months the pattern is repeating? To answer this, we will use several values for seasonality and train Arima model to get the best results. Then we will do the same for the trend and error component of the time series.

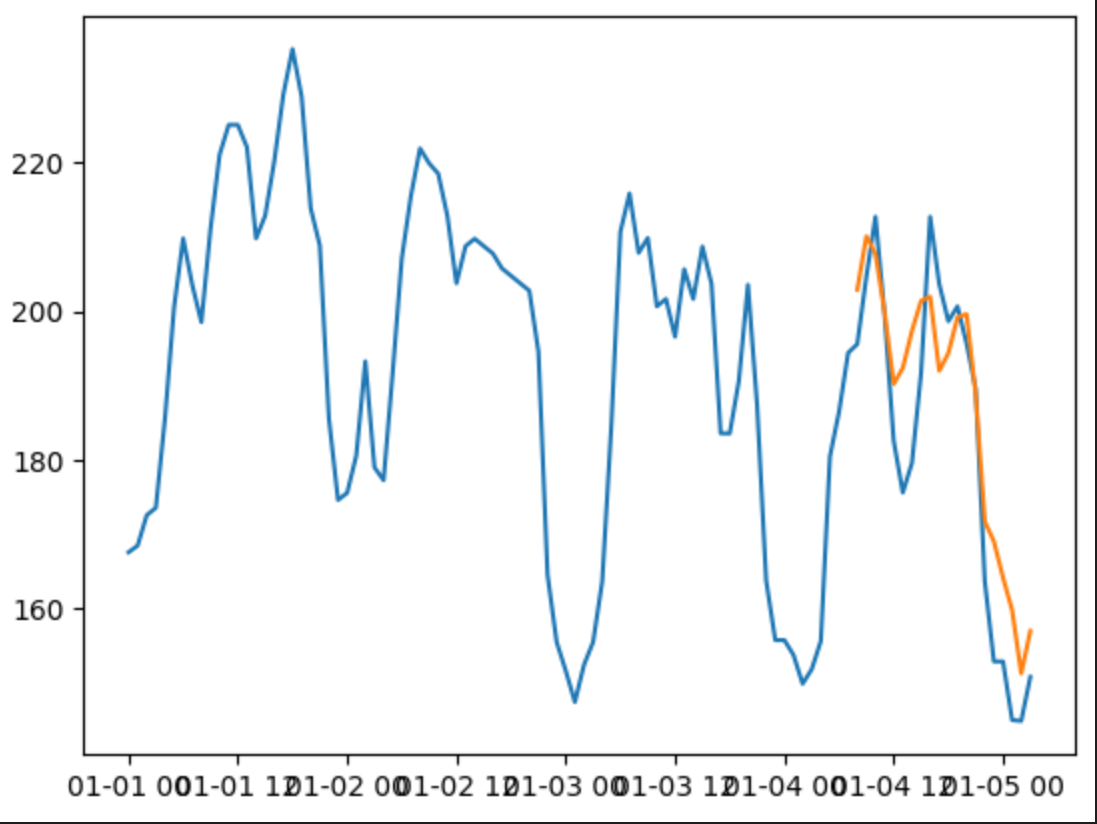
## Step 6: Reconstruct the time series for the forecasted error, trend and seasonality

Since, the seasonal decomposition is a additive process, after forecasting the error, trend and seasons, we will simply add the values to reconstruct the predicted time series. Here is a sample of the forecasted time series:

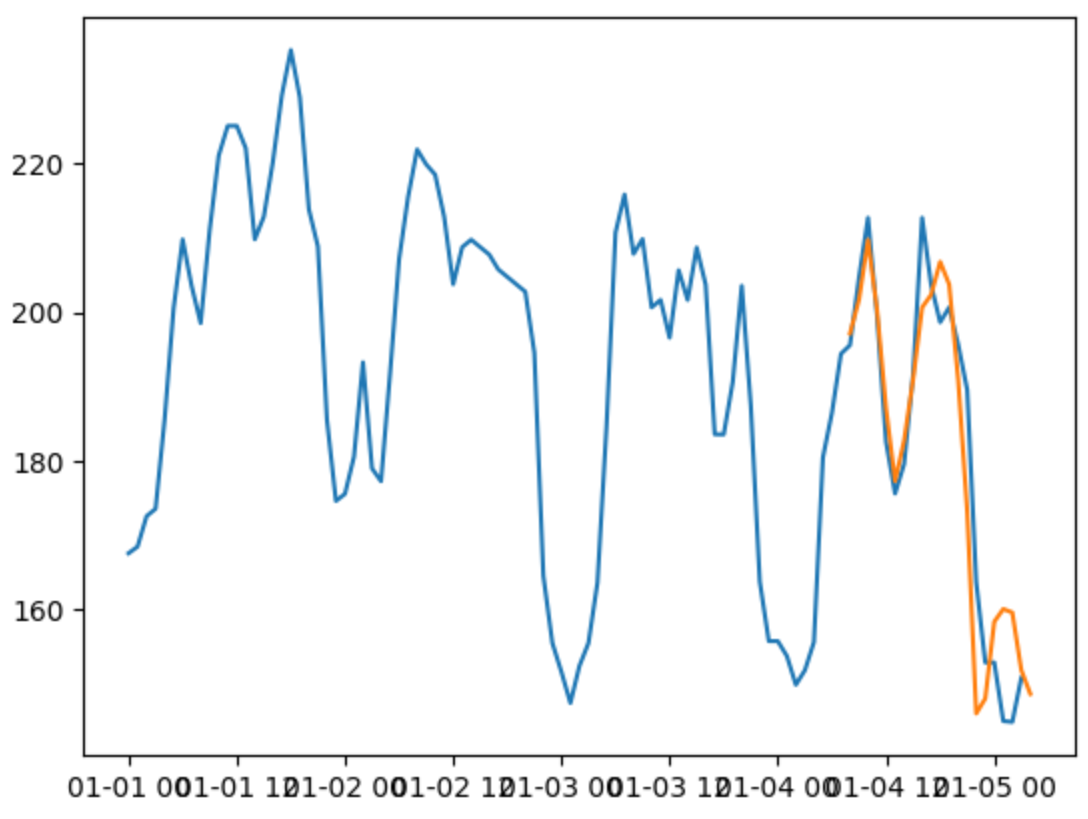


## Step 7: Comparison between Arima only vs ETS-Arima on a small sample of data

**ARIMA only**



**ETS-ARIMA model**



## Step 8: Diagram for ETS-Arima on the full dataset

