AIM:

To perform Energy demand forecasting using different models and compare the results

DATASET:

For this analysis, I have considered only a single building energy data for 1 year:

Period: 01/08/2023 to 17/7/2024

Rows: 51796

Features: 4 – Timestamp, R[kWh], Y[kWh], B[kWh]

In a three-phase system, active power is measured separately for each of the three phases (**R**, **Y**, and **B**), and the total active power is the sum of the power consumed in all three phases.

The top 5 entries of the dataset are:

	TIME [UTC Seconds]	R[kW]	Y[kW]	B[kW]	Unnamed: 4
0	1690828200000	1.17	0.64	0.96	NaN
1	1690828800000	1.16	0.73	0.98	NaN
2	1690829400000	1.19	0.72	0.98	NaN
3	1690830000000	1.16	0.73	0.97	NaN
4	1690830600000	1.20	0.74	0.99	NaN

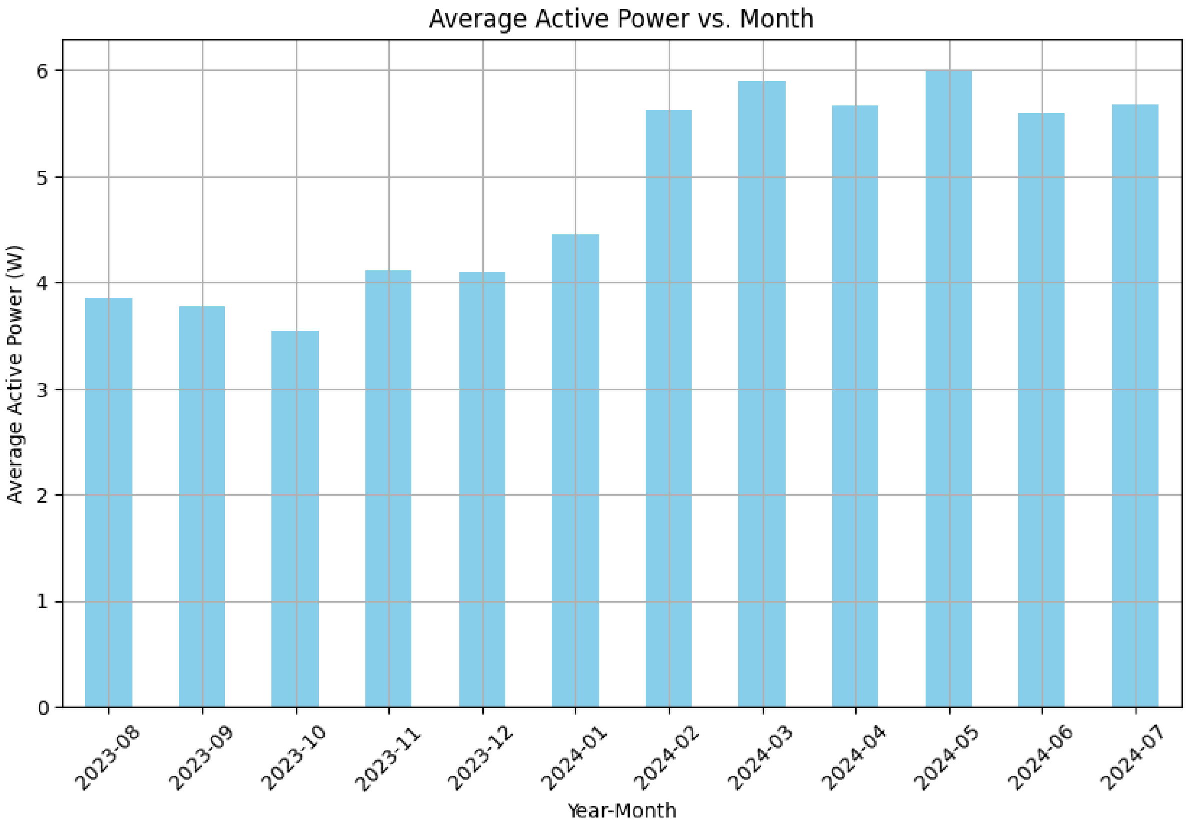
METHODOLOGY:

1. Data Cleaning and Pre-Processing
   - Converting the Time Column to Datetime format
   - Create a new column – 'Total Active Power' which is the sum of three columns – R+Y+B
   - Remove null values
   - Add new columns – Date, Year, Month, Hour, Year, Weekday, Is_Weekday
   - Plot a line plot with Date on X axis and Power on Y axis

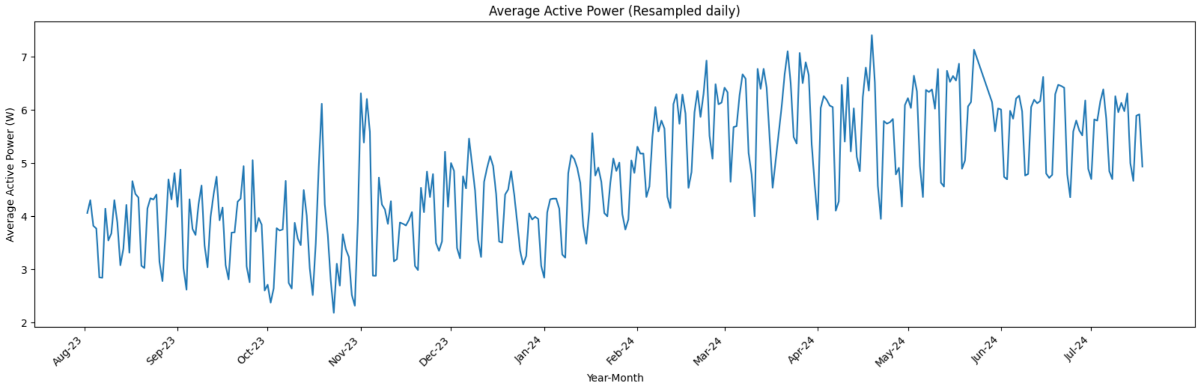
	R	Y	B	Active Power	Date	Hour	Month	Year	Weekday	Is_Weekday
TIME										
2023-08-01 00:00:00	1.17	0.64	0.96	2.77	2023-08-01	0	8	2023	1	1
2023-08-01 00:10:00	1.16	0.73	0.98	2.87	2023-08-01	0	8	2023	1	1
2023-08-01 00:20:00	1.19	0.72	0.98	2.89	2023-08-01	0	8	2023	1	1
2023-08-01 00:30:00	1.16	0.73	0.97	2.86	2023-08-01	0	8	2023	1	1
2023-08-01 00:40:00	1.20	0.74	0.99	2.93	2023-08-01	0	8	2023	1	1

2. TIMESERIES ANALYSIS

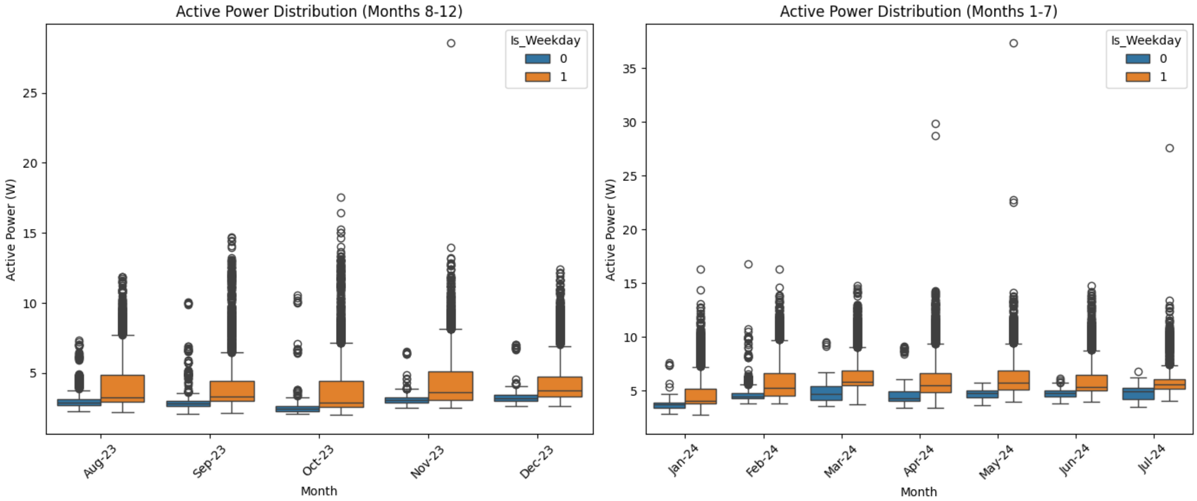
BAR PLOT



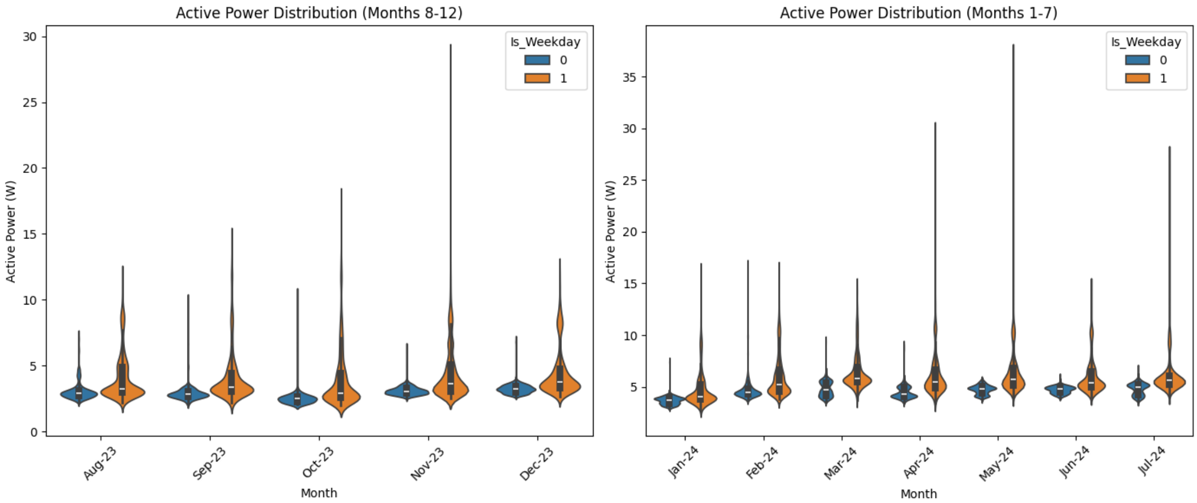
LINE PLOT



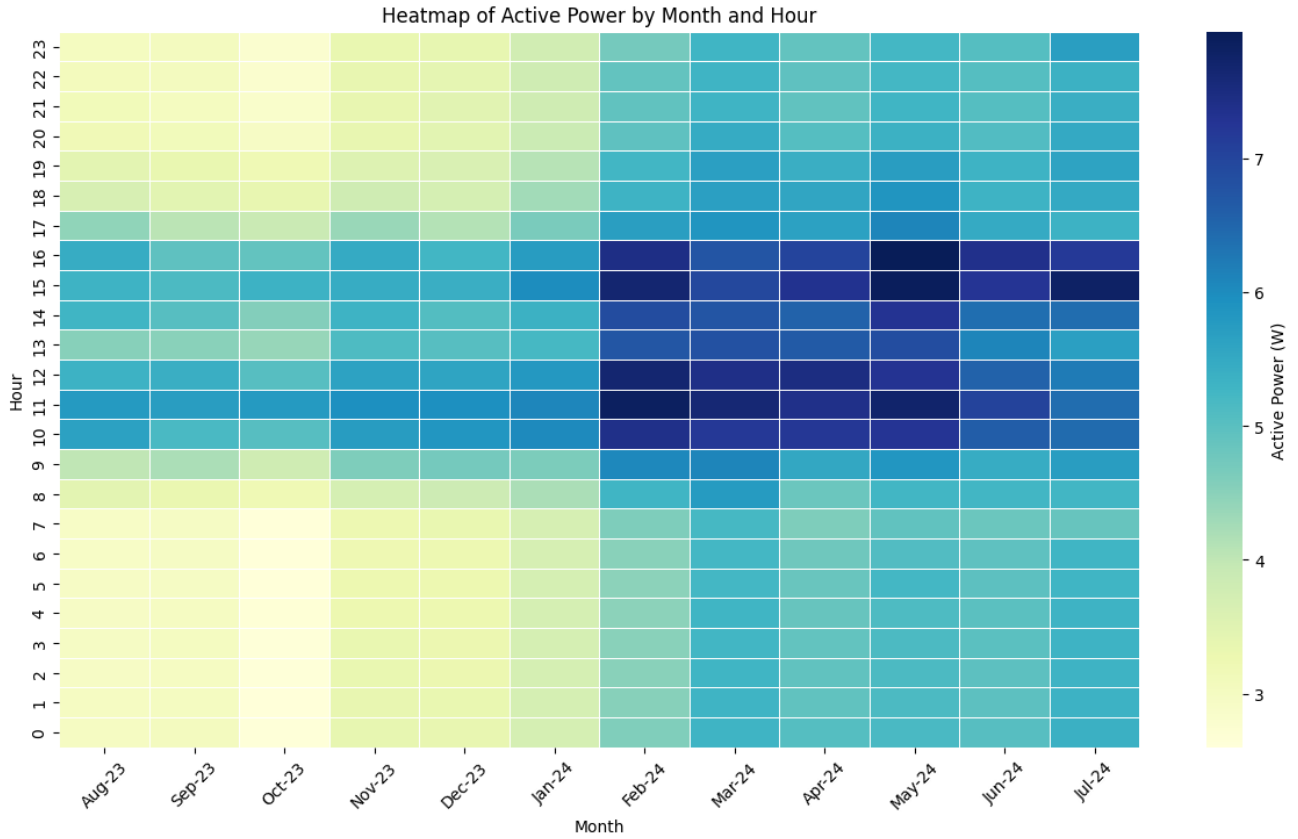
BOX PLOT



VIOLIN PLOT



HEATMAP



It can be observed that the energy consumption is more in the year 2021 and within the time from 9a.m to 5p.m

IDENTIFY OUTLIERS

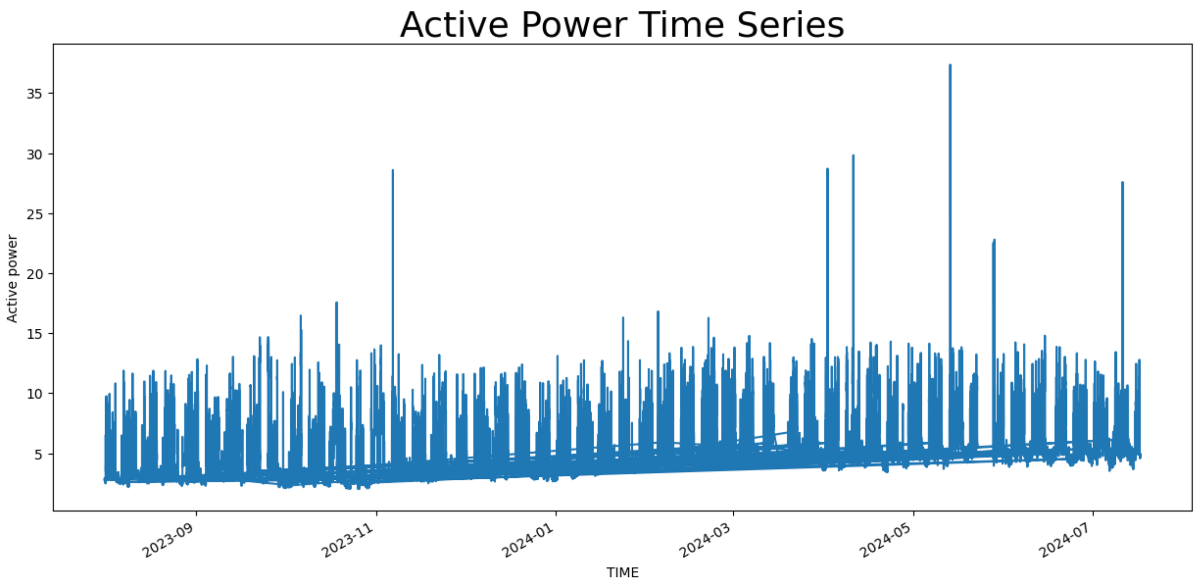
```
# Perform Rosner's test
outliers = rosner_test(sm_df['Active Power'])

# Display the outliers detected
print("Detected Outliers:", outliers)
```

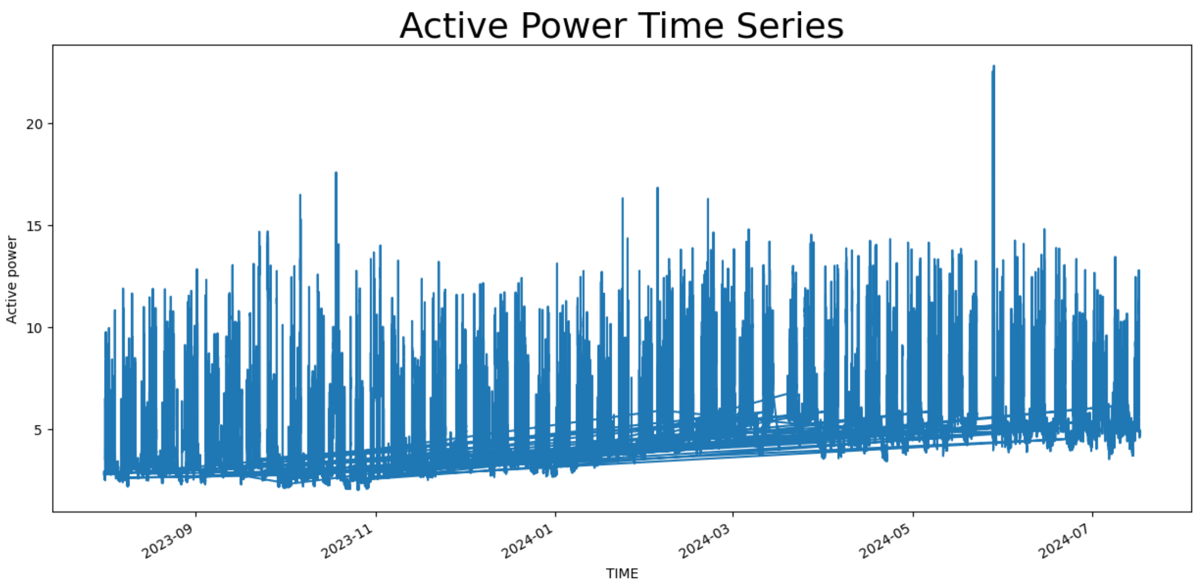
```
Detected Outliers: [37.36, 29.82, 28.7, 28.580000000000002, 27.59]
```

REMOVE OUTLIERS

Before



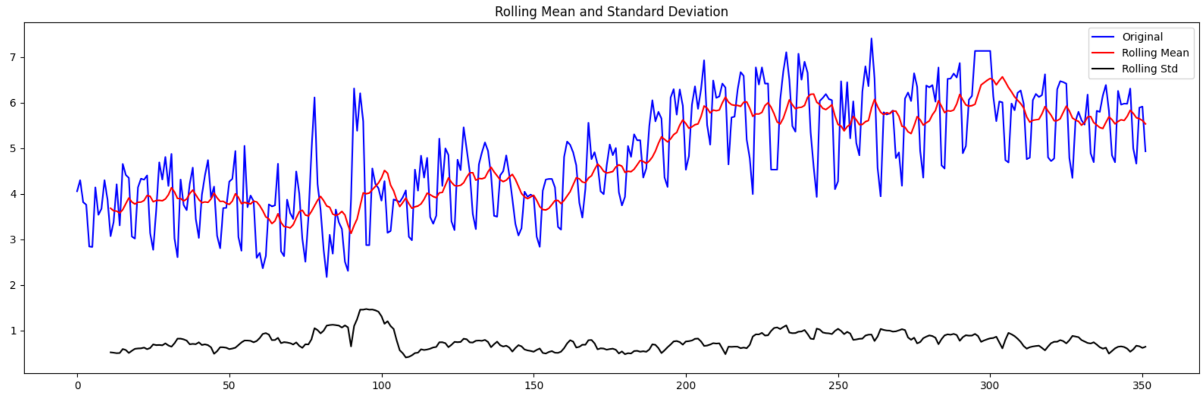
After



3. FORECASTING

ARIMA MODEL

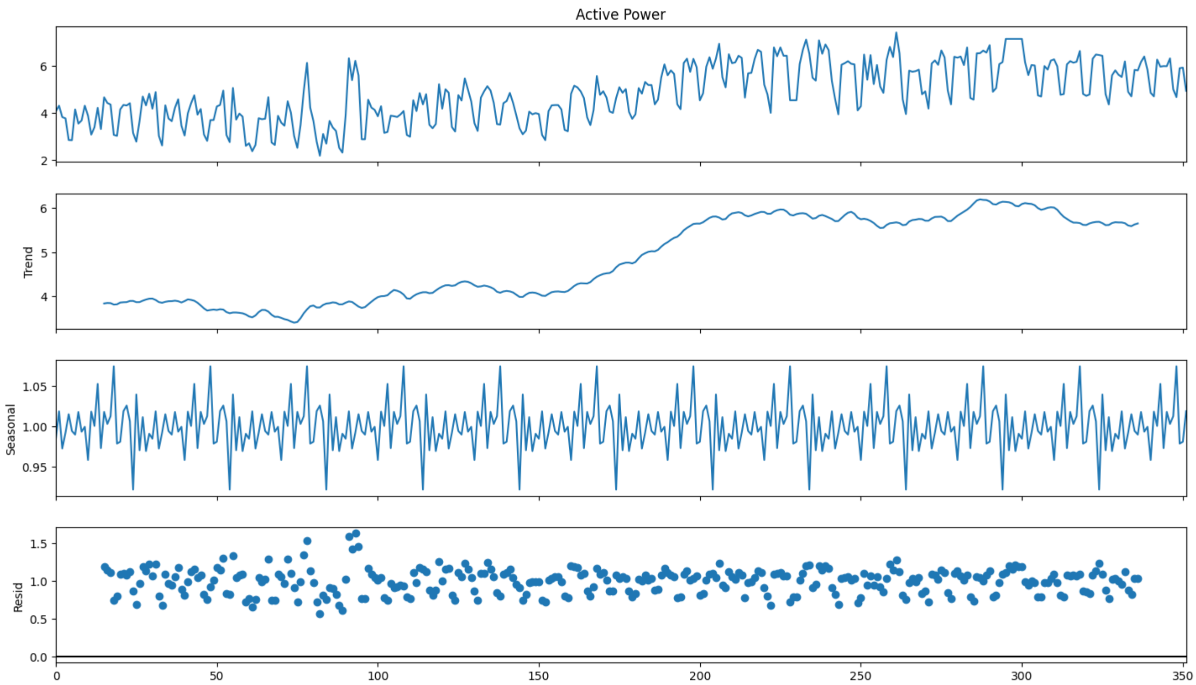
- Test for Stationary



Results of dickey fuller test

Test Statistics	-1.337126
p-value	0.612021
No. of lags used	15.000000
Number of observations used	336.000000
critical value (1%)	-3.449963
critical value (5%)	-2.870181
critical value (10%)	-2.571373

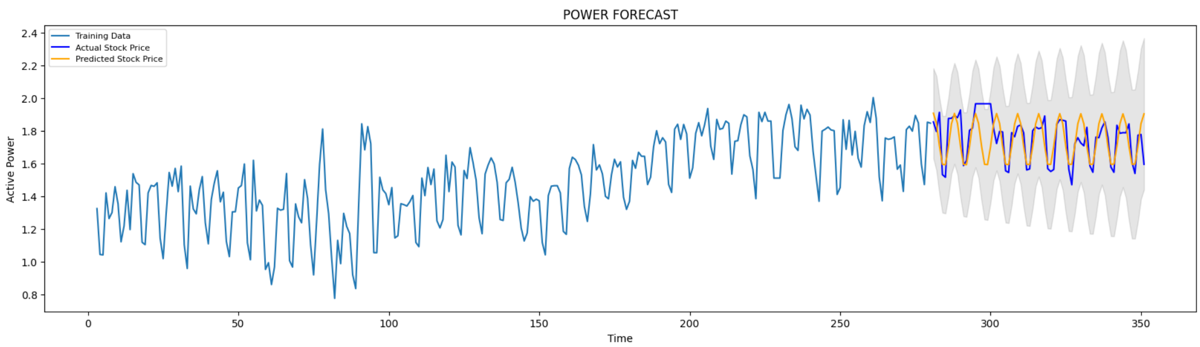
- Decomposing the time series



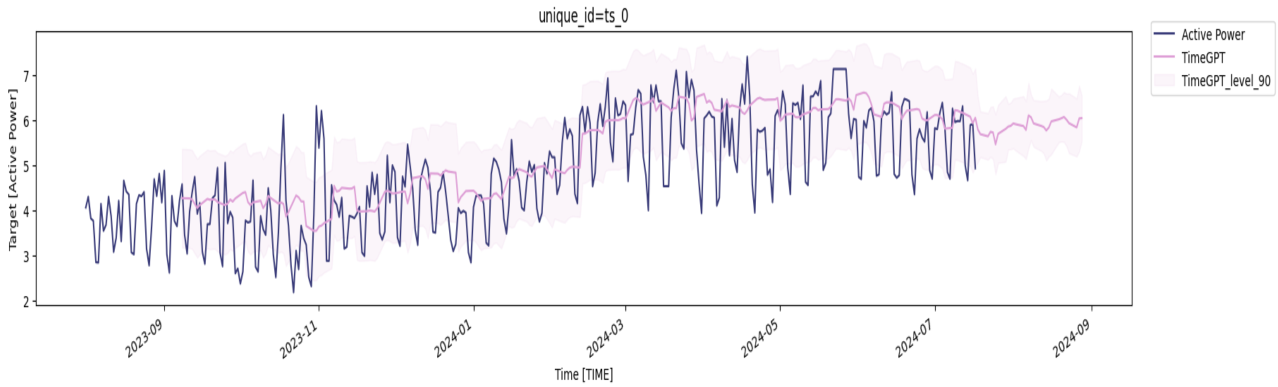
- Split into training and test set

  80% train data and 20% test data
- Use Auto Arima model to get the best parameters

  Best Model – (3,1,1)
- Fit the model and perform forecasting



4. TIMEGPT MODEL
    - Consider a forecast horizon of 30 days
    - Parameters:
      Confidence level: 90%
      Fine Tuning steps: 200
      Fine Tuning Loss: MAPE



RESULTS:

METRICS	TIMEGPT	ARIMA
MSE	0.4410	0.0145
MAE	0.4958	0.0887
RMSE	0.6641	0.1203
MAPE	10.12%	4.96%

Still work is going on to enhance the forecasting models.

References:

1. https://github.com/Vikasdubey0551/EDA_and_Timeseries-forecasting_power_consumption/blob/master/TimeseriesEDA-forecasting-model-comparision.ipynb
2. https://www.analyticsvidhya.com/blog/2021/07/stock-market-forecasting-using-time-series-analysis-with-arima-model/
3. https://docs.nixtla.io/docs/getting-started-timegpt_quickstart