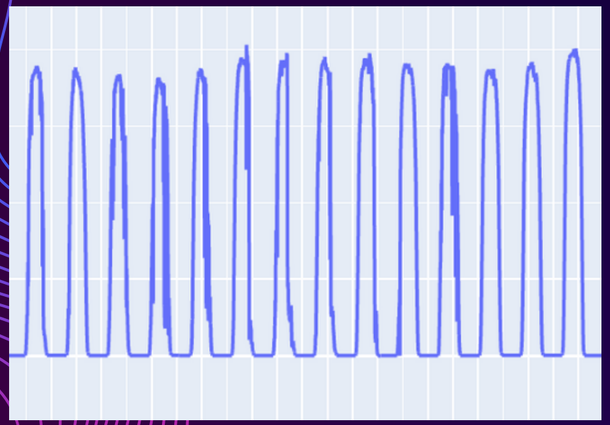
Description

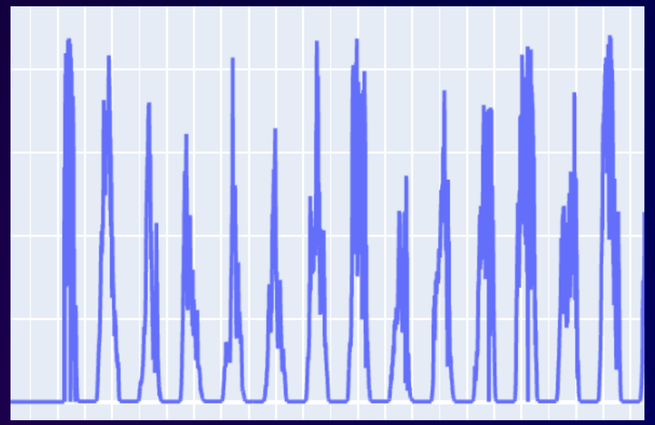
The given data consists of current from a transformer site which measures after every 5 minutes from 23/12/2018 to 4/10/2019 of total of 285 days.

All the days can be classified into three categories:

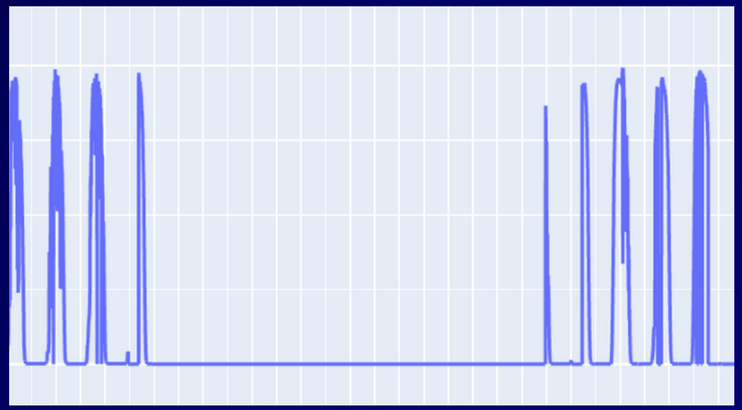
- Good Days: These days exhibit minimal data noise and a scarcity of outliers.
- Bad Days: These days are marked by higher data noise, possibly from numerous errors.
- Missing Days: There are many days in the data where data is missing for the whole day.



Relatively Good days



Bad Days



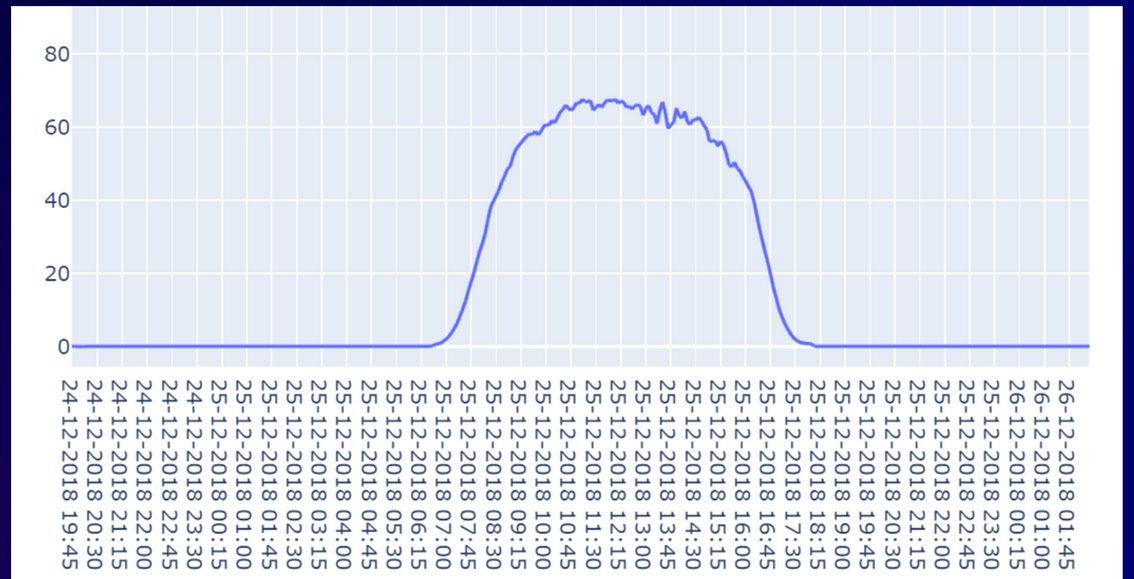
Missing Days

Description

- Classify days as good, bad or missing
- Some days are relatively good ,we must find and fix them.
- Then train a ML mode on 80%datal and test on the 20% data.
- Use that model to fix the bad and missing days

Trend Of The Data Observed

- A noticeable pattern emerges as the current readings begin to ascend between 5-6 am and decline between 6-7 pm.



Steps for solving the problem

Managing Outliers

Making ML model

Finding Missing Value

Finding Good Days

Fixing bad/missing days

Steps for solving the problem

1. **Finding Missing Days:** Initially, we created an additional column for dates using the “partition” function of strings. Then, group the data by date to calculate the corresponding mean for each date. Any date with a mean value of 0 is considered a missing day.

```
grouped_df = df.groupby("Date")
grouped_df.describe()
```

```
mean_days =grouped_df.mean(["HT R Phase Current"])
mean_days[mean_days==0 ].dropna().index
```

Steps for solving the problem

2.**Finding the outliers and removing them:** Initially, create an additional column named "Difference" to record the difference between the current reading and the reading at the subsequent time. When it exceeds 30, it is regarded as an outlier, and in such cases, replace these outlier values with the mean current reading for that specific time, considering all days.

We opted for the threshold of 30 because the mean of the current reading at a specific time, considering all days, typically falls within the range of 30 to 40. Setting the threshold at 40 could pose issues for days when the readings remain at 0 continuously for extended periods as then the value after 2 or 3 time donot change.

Steps for solving the problem

EXPONENTIAL MOVING AVERAGE: For detecting the outliers we created a column named “Moving Average” in which we stored the exponential moving average with a **window 6** and then created a difference column which stores the absolute difference of the current and the moving average.

The reason of using exponential moving average was it gave **more weight to the recent values**.

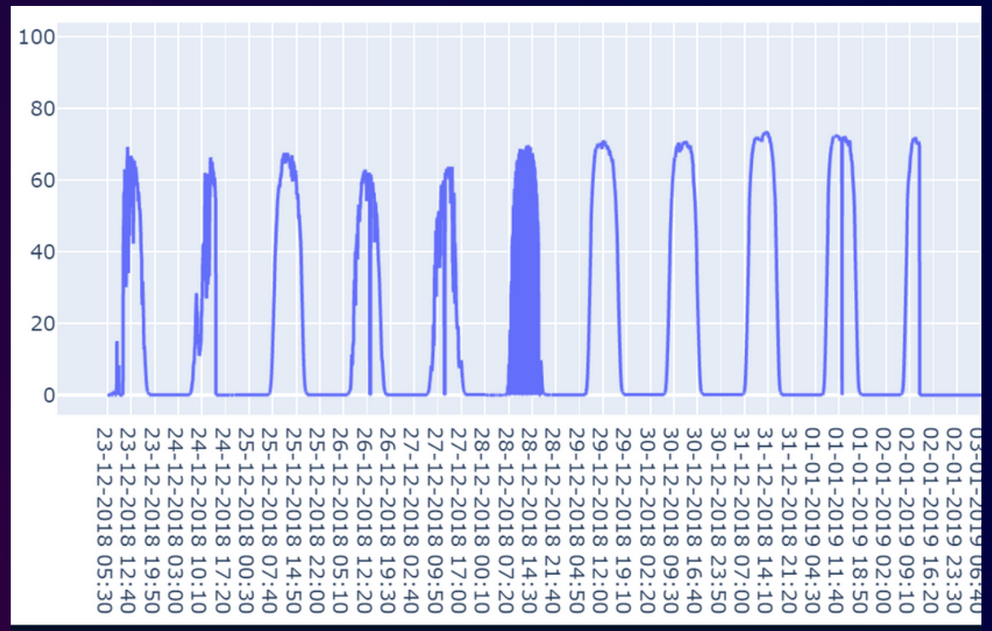
Then we did set a threshold of 25, i.e if for a particular row the difference exceeds threshold, then we considered it as outlier.

In [31]:
```
df['moving average']=df.iloc[:,1].ewm(span=6,adjust=False).mean()
df["difference"]=abs(df["HT R Phase Current"]-df["moving average"])
df.head()
```

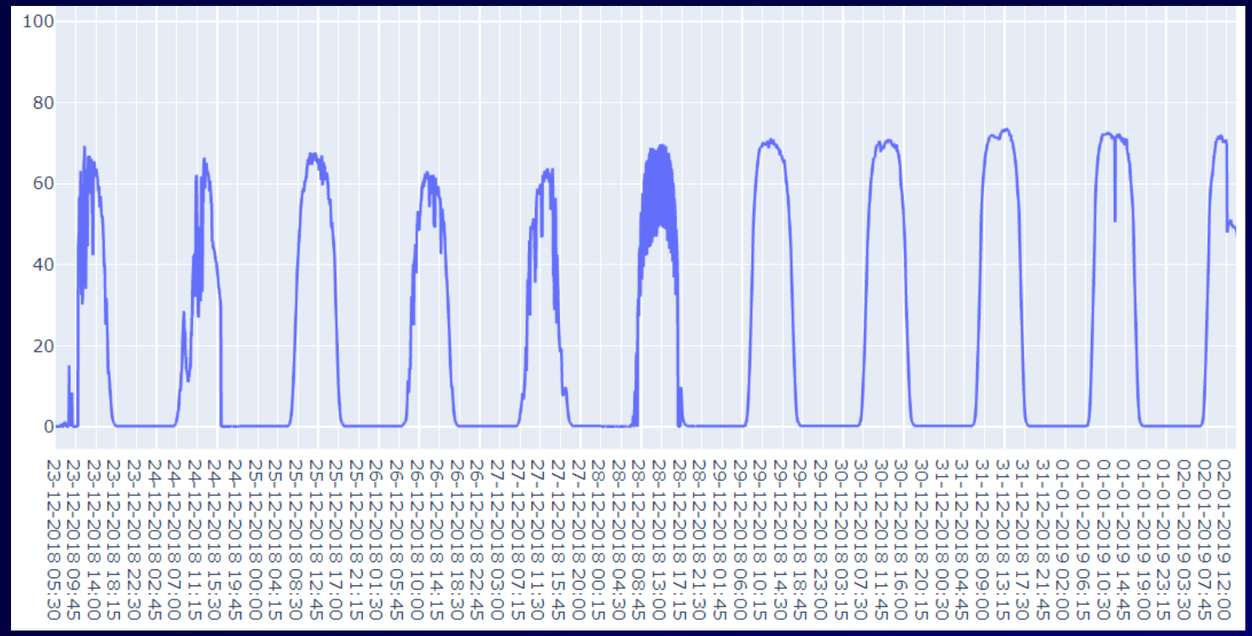
Out[31]:

	Timestamp	HT R Phase Current	Time	moving average	difference
0	23-12-2018 05:30	0.0	05:30	0.0	0.0
1	23-12-2018 05:35	0.0	05:35	0.0	0.0
2	23-12-2018 05:40	0.0	05:40	0.0	0.0
3	23-12-2018 05:45	0.0	05:45	0.0	0.0
4	23-12-2018 05:50	0.0	05:50	0.0	0.0

In [18]:
```
#function for detecting and replacing outliers
def outlier(i):
    if(df.iloc[i,3]>25):
        df.iloc[i,1]=df.iloc[i,2]
for i in range (0,len(df)):
    outlier(i)
```



Before replacing the outlierss



After replacing the outlierss

3. **Finding good days:** First check the difference between the consecutive points and if it is greater than 5 and it happens less than then we classify it as the relatively good days. To refine this classification, further examination of the interquartile range is performed. Notably, for the days classified as good, it is observed that the interquartile range exceeds 1. Consequently, from the previous list, dates with an interquartile range less than 1 are filtered out. The remaining dates are identified as good days.

4. **Making the ML model:**

While constructing our model, we recognise that the current reading is contingent on the timestamp, particularly the time of day and month it is registered. Hence, our first step in model development involves transforming the categorical data into a numerical format. To achieve this, we leverage the OneHotEncoder class from the scikit-learn library. One-hot encoding represents categorical variables as numeric values within a machine-learning model. This process yields indicator variables for months and times utilised in model creation.

Using the list of good days, we then make a new data frame for the model.
After that, we split 80% of the data for training the model, and the rest, 20%, is kept for testing the model. This gives us different metrics which we can use to validate our model. We then see which of the parameters have a p-value more than 0.05, drop those variables and again run the model till there is no variable with p-value greater than 0.05

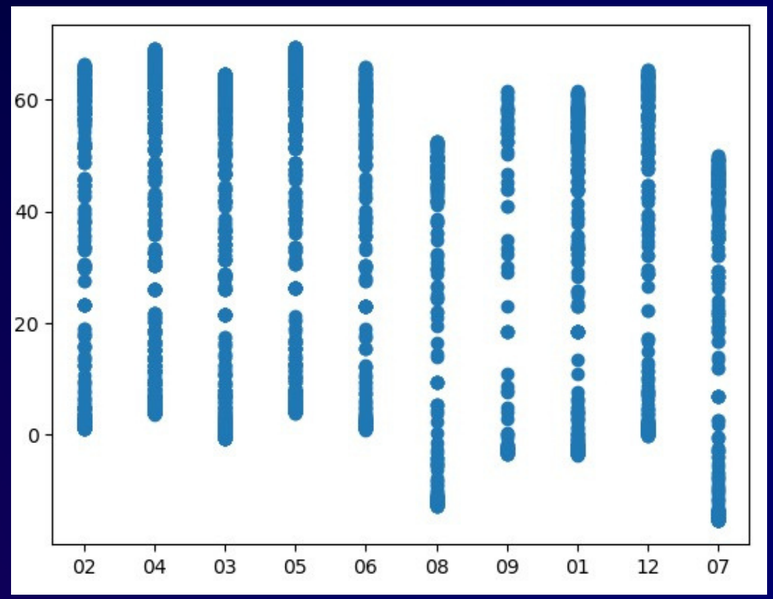
Metrics of the train data:

OLS Regression Results

Dep. Variable:	HT R Phase Current	R-squared:	0.809
Model:	OLS	Adj. R-squared:	0.806
Method:	Least Squares	F-statistic:	294.3
Date:	Thu, 12 Oct 2023	Prob (F-statistic):	0.00
Time:	21:51:05	Log-Likelihood:	-81481.
No. Observations:	20505	AIC:	1.635e+05
Df Residuals:	20214	BIC:	1.659e+05
Df Model:	290		
Covariance Type:	nonrobust		

- Scatter plot of the model on the test data:

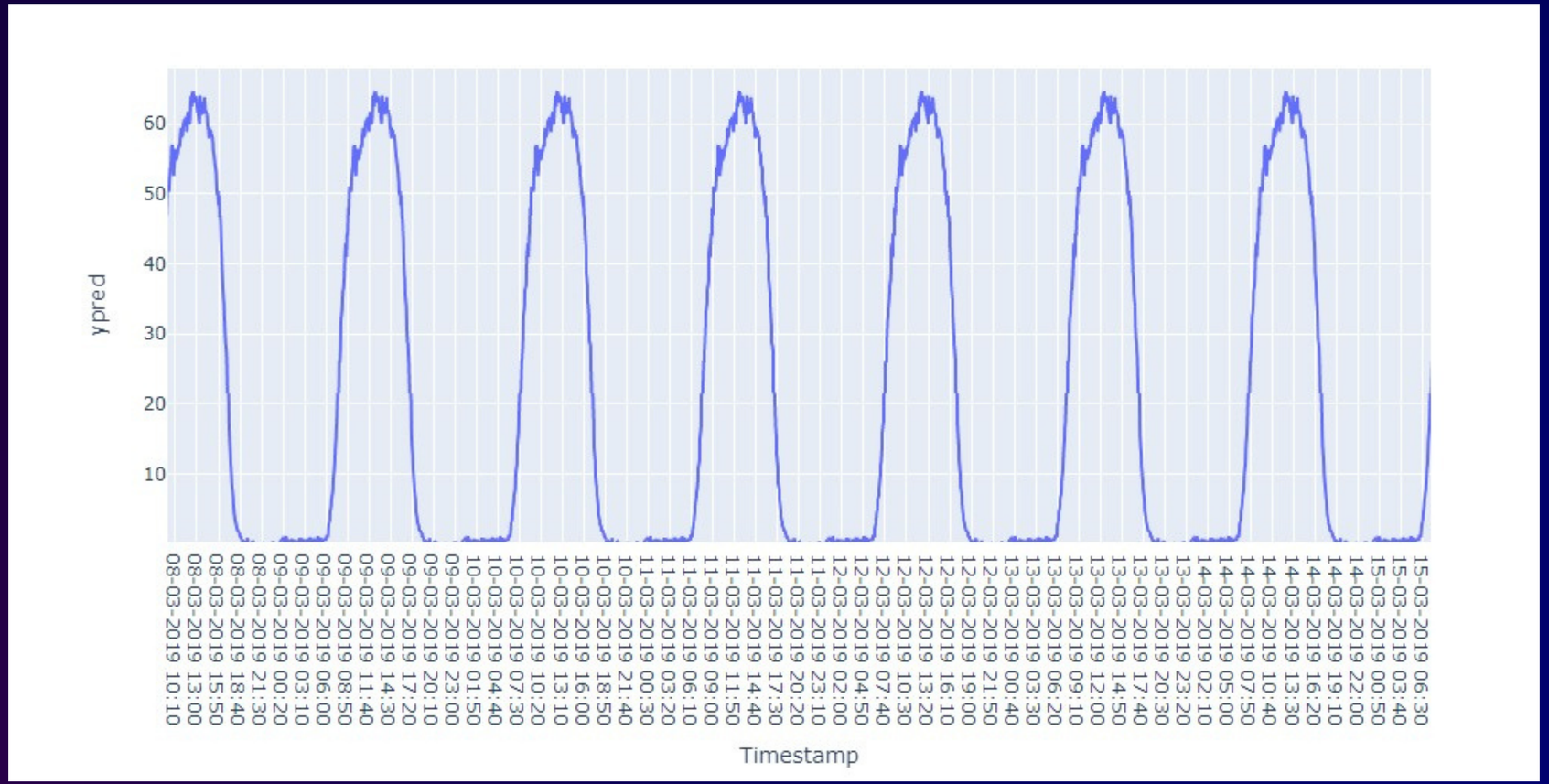
This is the scatter plot with x axis as the months and y_predicted



5. **Using the model to fix “bad” and “missing” days:**

With the model we established earlier, we address and rectify all the dataset's problematic or missing data points in the dataset.

Final predicted model

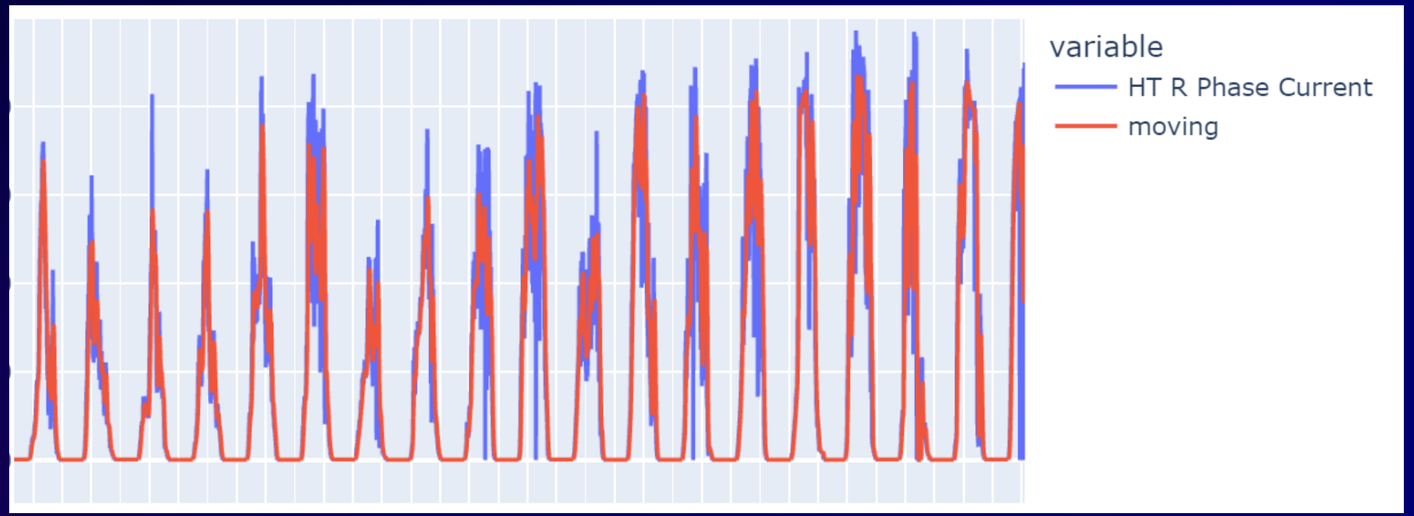


Different approaches used:

1. **Using Moving averages:** Initially, we created the moving average column, followed by calculating the difference between the actual current reading and the moving average. For a given day, if this difference exceeds 7 on more than ten occasions, it is classified as a bad day. Replacing the outlier defined by the difference between consecutive days greater than 30 we replace ethem by the moving average value at that given time.

```
df["moving"] =df["HT R Phase Current"].rolling(window =8).mean()
df.fillna(value=0, inplace =True)
df.head(5)
```

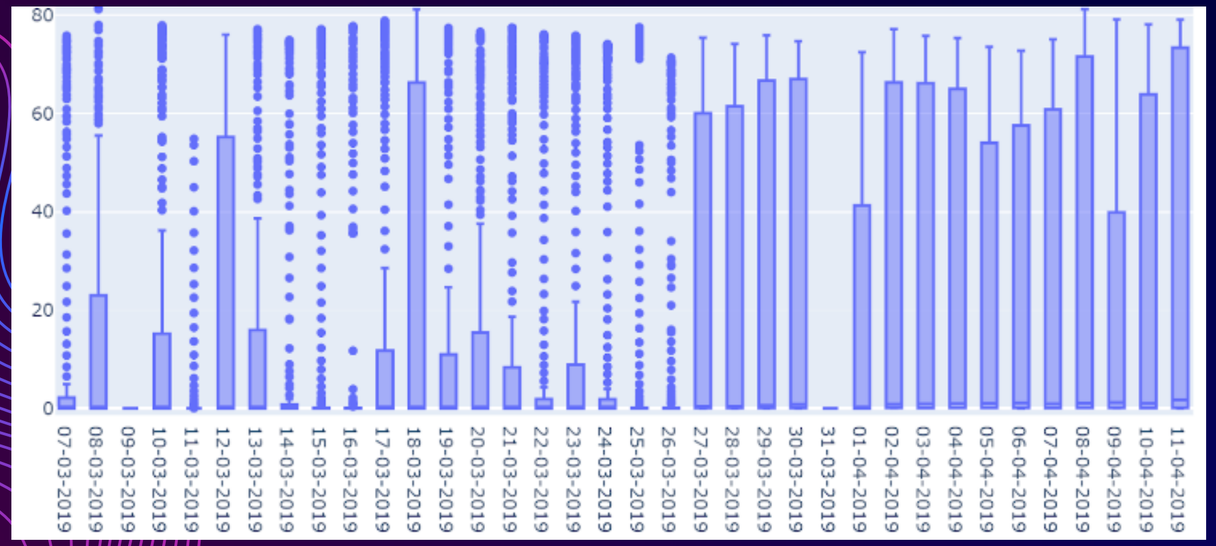
	Timestamp	HT R Phase Current	Date	DIfference	moving
0	23-12-2018 05:30	0.0	23-12-2018	0.0	0.0
1	23-12-2018 05:35	0.0	23-12-2018	0.0	0.0
2	23-12-2018 05:40	0.0	23-12-2018	0.0	0.0
3	23-12-2018 05:45	0.0	23-12-2018	0.0	0.0
4	23-12-2018 05:50	0.0	23-12-2018	0.0	0.0

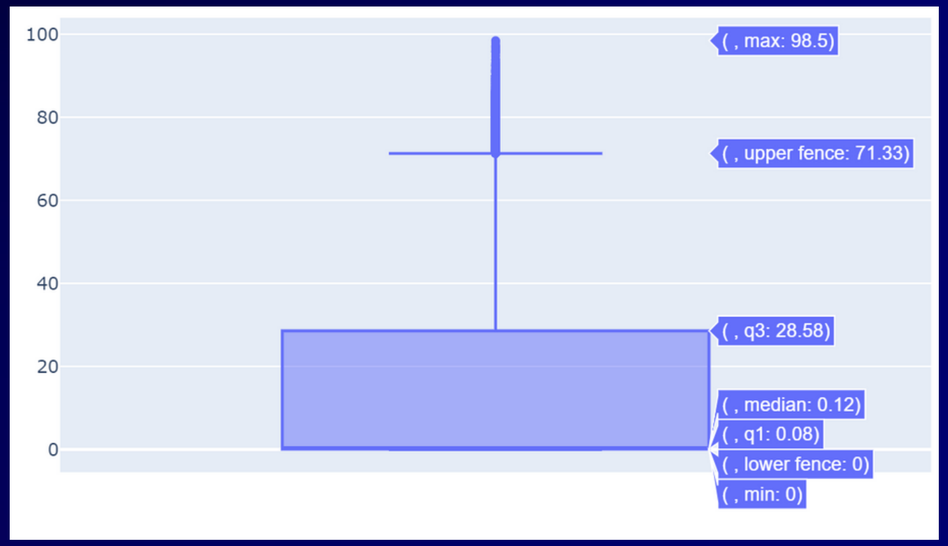


```
def remove_outlier(i):
    if abs(good_df.iloc[i , 5] )>30:
        good_df.replace(good_df.iloc[i,1] , good_df.iloc[i,4] , inplace=True)

for i in range(22176):
    remove_outlier(i)
```

2.**Making the box plots for different days:** Created box plots for each and every date and for whole datbase in plotly library and seeing the interquartile range using it.





The second box plot gives the different statistics for the data and the outliers

Another model used:

- We made an Multiple Linear Regression model taking the MONTH and its 5 powers and “Time in minutes” and its 6 powers as the input features.
- Took HT R Phase Current as the dependent variable and applied the model on it to predict the current.
- Got the **R-squared** value of around **0.54** and **F-statistics** value of 2782.

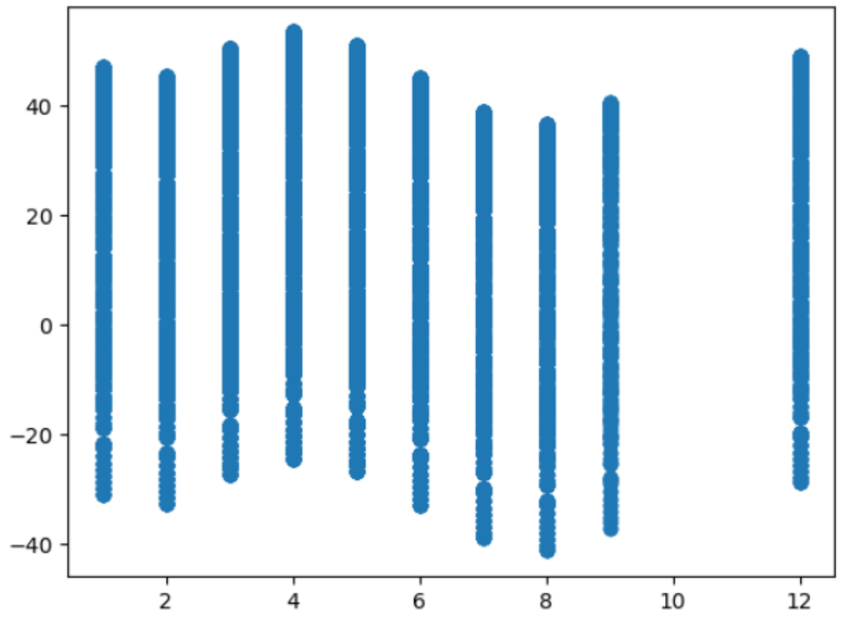
OLS Regression Results

Dep. Variable:	HT R Phase Current	R-squared:	0.544
Model:	OLS	Adj. R-squared:	0.544
Method:	Least Squares	F-statistic:	2782.
Date:	Thu, 12 Oct 2023	Prob (F-statistic):	0.00
Time:	21:49:48	Log-Likelihood:	-1.1293e+05
No. Observations:	25632	AIC:	2.259e+05
Df Residuals:	25620	BIC:	2.260e+05
Df Model:	11		
Covariance Type:	nonrobust		

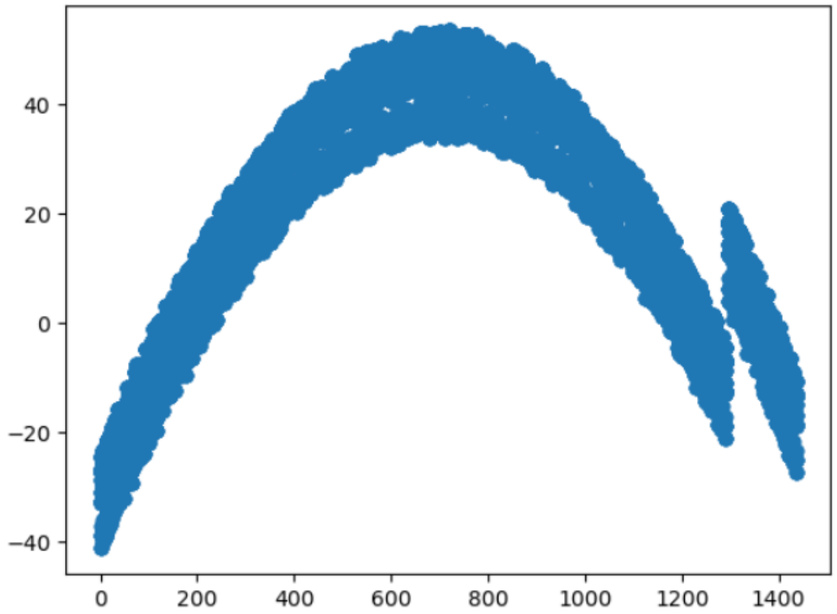
Another model used:

Scatter Plot of The variation of the current as predicted by our MLR Model vs the MONTH.

```
In [50]: plt.scatter(X['MONTH'],y_pred)
Out[50]: <matplotlib.collections.PathCollection at 0x1b46adce460>
```




```
In [51]: plt.scatter(X['Time_in_Minutes'],y_pred)
Out[51]: <matplotlib.collections.PathCollection at 0x1b46addeee0>
```



Scatter Plot of Current as Predicted by MLR model vs the Time in Minutes.

Summary

- Identified 89 good days and 38 days with missing data in the provided dataset.
- Significant number of outliers were eliminated from the dataset.
- Metrics for the training model: We use input axis as time and month of the timestamp.
- Created ML model and used it on the given datset to predict the data