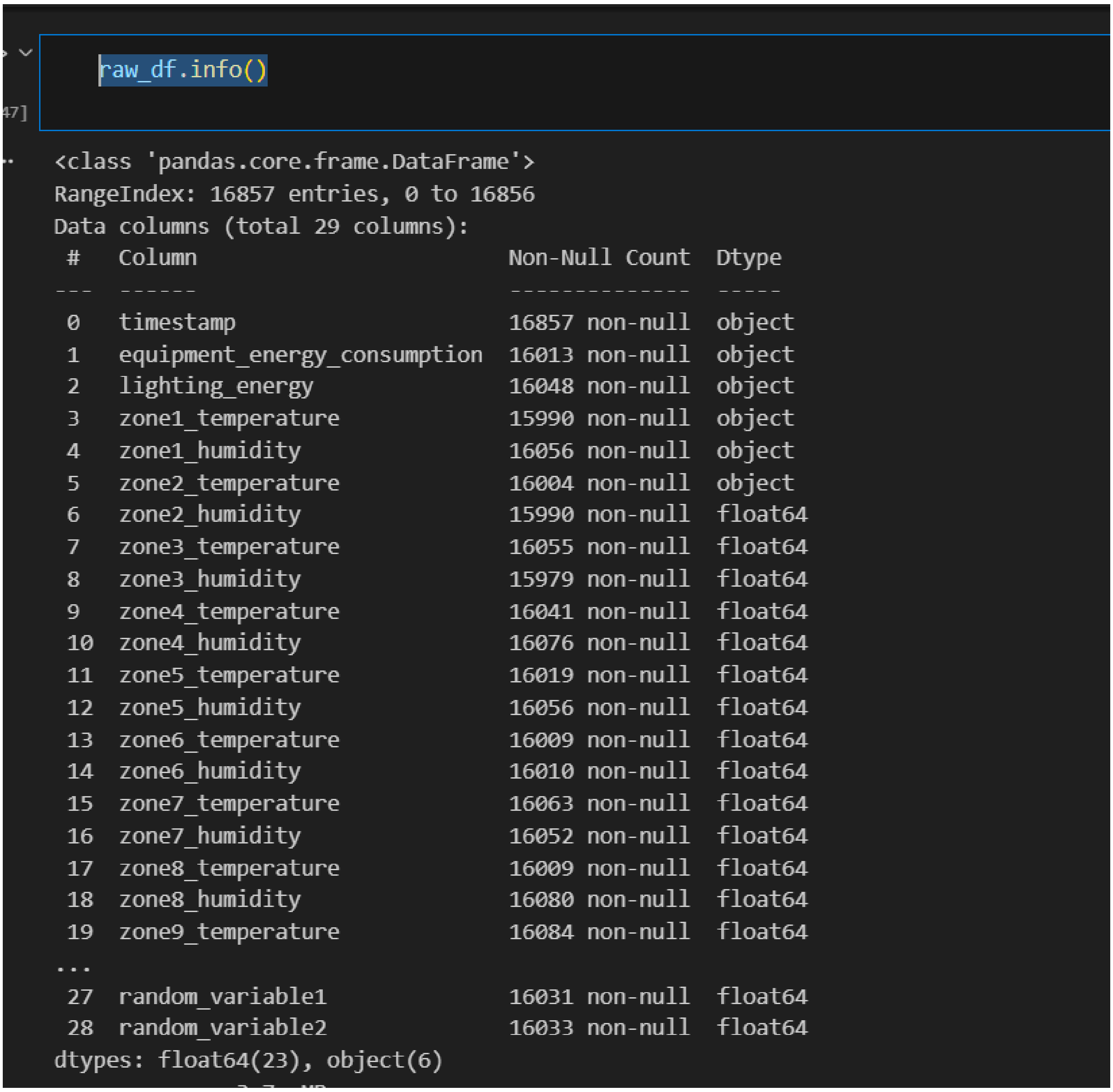
Documentation –

Problem Statement - Your assignment is to develop a machine learning model that can accurately predict the energy consumption of industrial equipment (equipment_energy_consumption) based on the data collected from the factory's sensor network. This will help the facility managers optimize their operations for energy efficiency and cost reduction.

Solution – So starting with data set first and foremost step is to check the data – and analyse it.

So, I have created 2 juypter notebooks one is for Feature Exploration and other one is for model building.



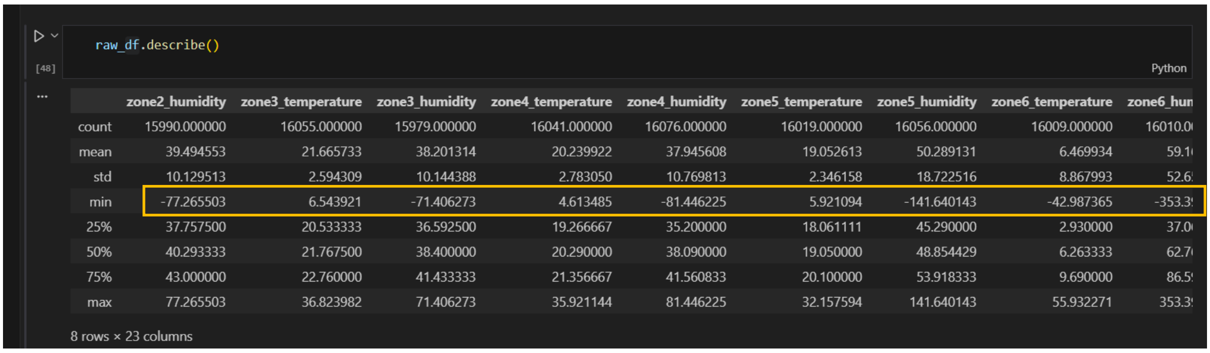
```
raw_df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 16857 entries, 0 to 16856
Data columns (total 29 columns):
 #   Column                        Non-Null Count  Dtype
---  ------                        --------------  -----
 0   timestamp                     16857 non-null  object
 1   equipment_energy_consumption  16013 non-null  object
 2   lighting_energy               16048 non-null  object
 3   zone1_temperature             15990 non-null  object
 4   zone1_humidity                16056 non-null  object
 5   zone2_temperature             16004 non-null  object
 6   zone2_humidity                15990 non-null  float64
 7   zone3_temperature             16055 non-null  float64
 8   zone3_humidity                15979 non-null  float64
 9   zone4_temperature             16041 non-null  float64
 10  zone4_humidity                16076 non-null  float64
 11  zone5_temperature             16019 non-null  float64
 12  zone5_humidity                16056 non-null  float64
 13  zone6_temperature             16009 non-null  float64
 14  zone6_humidity                16010 non-null  float64
 15  zone7_temperature             16063 non-null  float64
 16  zone7_humidity                16052 non-null  float64
 17  zone8_temperature             16009 non-null  float64
 18  zone8_humidity                16080 non-null  float64
 19  zone9_temperature             16084 non-null  float64
...
 27  random_variable1              16031 non-null  float64
 28  random_variable2              16033 non-null  float64
dtypes: float64(23), object(6)
```

Details about the dataset

Issues in the Dataset

So starting with the data info we can see that are some of the null values are present in the dataset which is not good for our model.



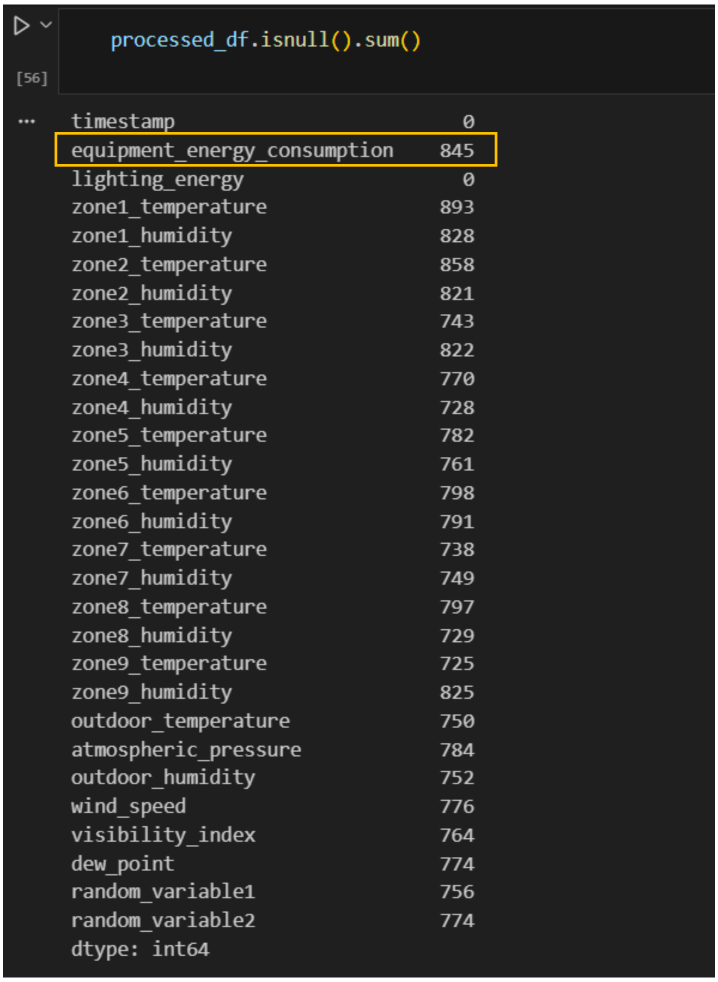
```python
raw_df.describe()
```

[48] Python

	zone2_humidity	zone3_temperature	zone3_humidity	zone4_temperature	zone4_humidity	zone5_temperature	zone5_humidity	zone6_temperature	zone6_hun
count	15990.000000	16055.000000	15979.000000	16041.000000	16076.000000	16019.000000	16056.000000	16009.000000	16010.0
mean	39.494553	21.665733	38.201314	20.239922	37.945608	19.052613	50.289131	6.469934	59.1
std	10.129513	2.594309	10.144388	2.783050	10.769813	2.346158	18.722516	8.867993	52.6
min	-77.265503	6.543921	-71.406273	4.613485	-81.446225	5.921094	-141.640143	-42.987365	-353.3
25%	37.757500	20.533333	36.592500	19.266667	35.200000	18.061111	45.290000	2.930000	37.0
50%	40.293333	21.767500	38.400000	20.290000	38.090000	19.050000	48.854429	6.263333	62.7
75%	43.000000	22.760000	41.433333	21.356667	41.560833	20.100000	53.918333	9.690000	86.5
max	77.265503	36.823982	71.406273	35.921144	81.446225	32.157594	141.640143	55.932271	353.3

8 rows × 23 columns

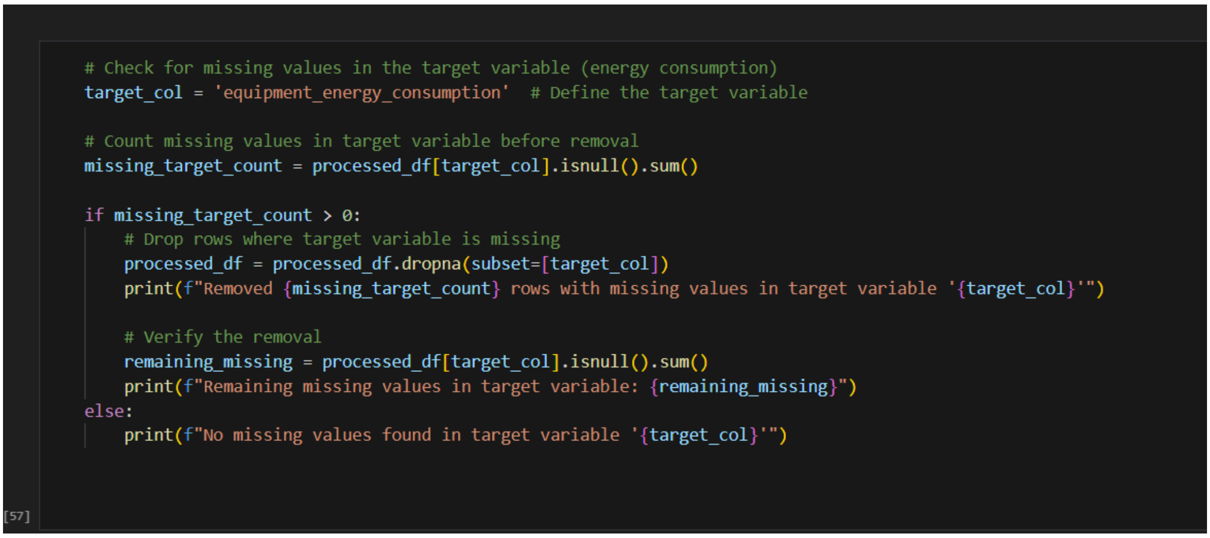
Also, After performing describe method, some values are invalid as well – for example as humidity is the %age but values are negative, also energy consumption and as well as lighting energy are also in negative.



```python
processed_df.isnull().sum()
```

[56]

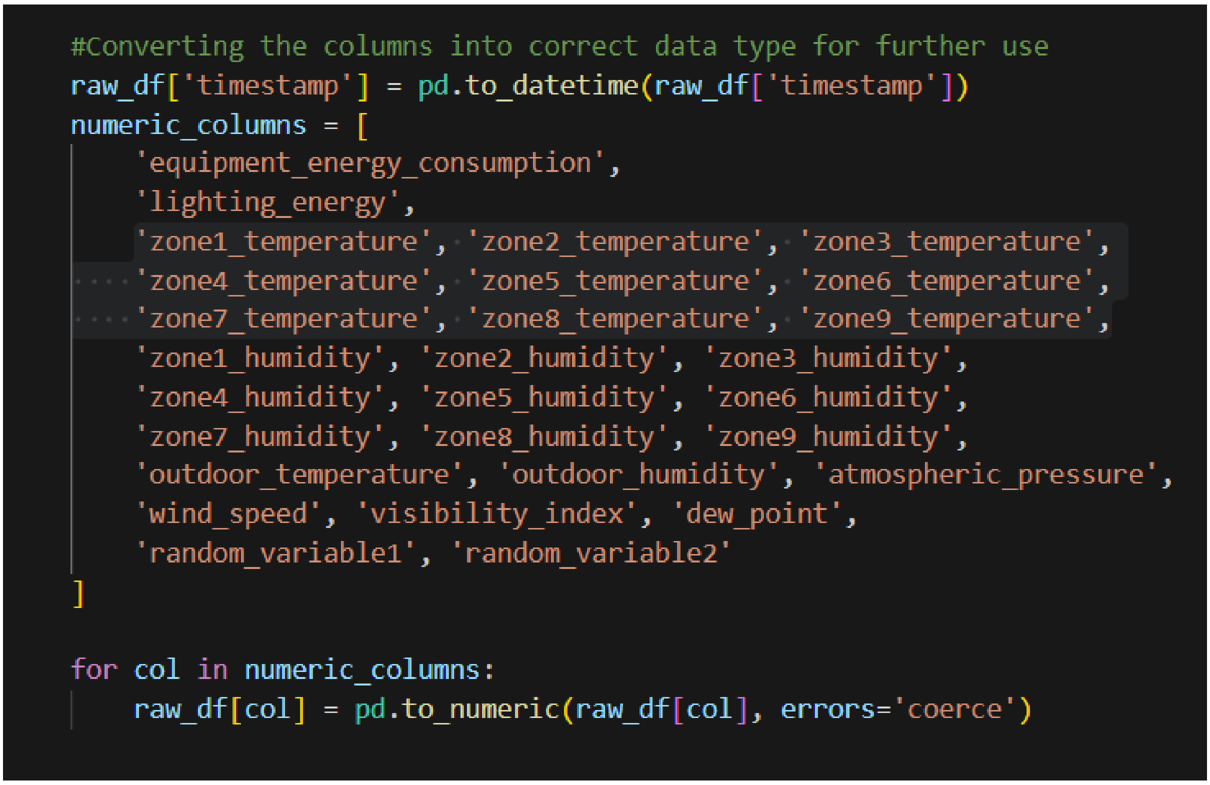
```
timestamp                      0
equipment_energy_consumption   845
lighting_energy                0
zone1_temperature            893
zone1_humidity               828
zone2_temperature            858
zone2_humidity               821
zone3_temperature            743
zone3_humidity               822
zone4_temperature            770
zone4_humidity               728
zone5_temperature            782
zone5_humidity               761
zone6_temperature            798
zone6_humidity               791
zone7_temperature            738
zone7_humidity               749
zone8_temperature            797
zone8_humidity               729
zone9_temperature            725
zone9_humidity               825
outdoor_temperature          750
atmospheric_pressure         784
outdoor_humidity             752
wind_speed                   776
visibility_index             764
dew_point                    774
random_variable1             756
random_variable2             774
dtype: int64
```



```
# Check for missing values in the target variable (energy consumption)
target_col = 'equipment_energy_consumption'  # Define the target variable

# Count missing values in target variable before removal
missing_target_count = processed_df[target_col].isnull().sum()

if missing_target_count > 0:
    # Drop rows where target variable is missing
    processed_df = processed_df.dropna(subset=[target_col])
    print(f"Removed {missing_target_count} rows with missing values in target variable '{target_col}'")

    # Verify the removal
    remaining_missing = processed_df[target_col].isnull().sum()
    print(f"Remaining missing values in target variable: {remaining_missing}")
else:
    print(f"No missing values found in target variable '{target_col}'")
```

[57]

Preprocessing



```
#Converting the columns into correct data type for further use
raw_df['timestamp'] = pd.to_datetime(raw_df['timestamp'])
numeric_columns = [
    'equipment_energy_consumption',
    'lighting_energy',
    'zone1_temperature', 'zone2_temperature', 'zone3_temperature',
    'zone4_temperature', 'zone5_temperature', 'zone6_temperature',
    'zone7_temperature', 'zone8_temperature', 'zone9_temperature',
    'zone1_humidity', 'zone2_humidity', 'zone3_humidity',
    'zone4_humidity', 'zone5_humidity', 'zone6_humidity',
    'zone7_humidity', 'zone8_humidity', 'zone9_humidity',
    'outdoor_temperature', 'outdoor_humidity', 'atmospheric_pressure',
    'wind_speed', 'visibility_index', 'dew_point',
    'random_variable1', 'random_variable2'
]

for col in numeric_columns:
    raw_df[col] = pd.to_numeric(raw_df[col], errors='coerce')
```

Step – converting to numeric column

Step 1 – Handling outliers –

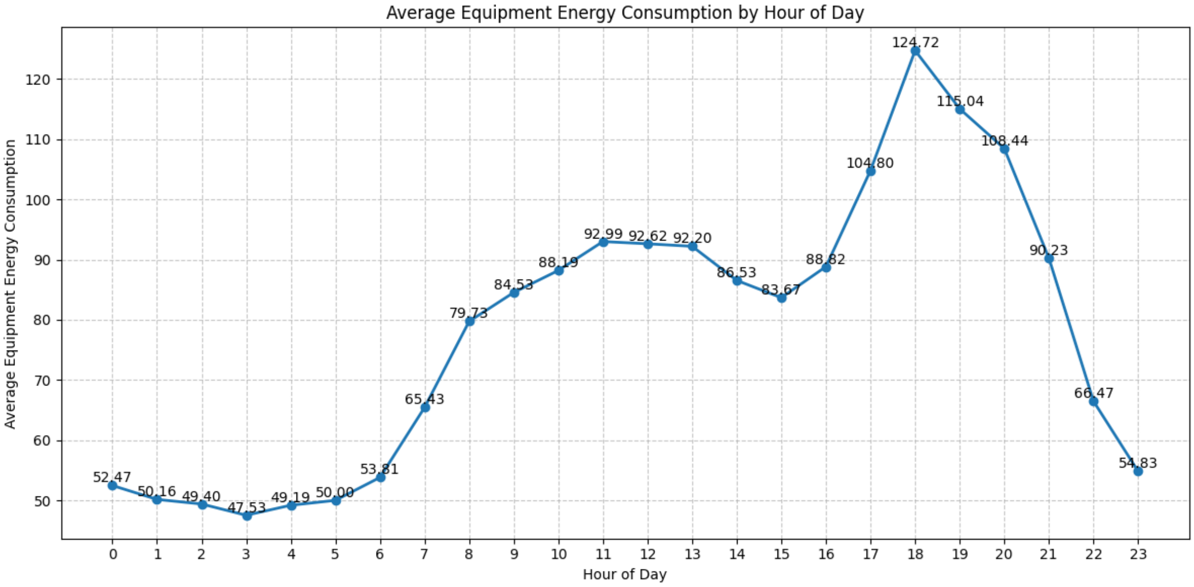
- For the target variable I did cap it above positive range i.e. above 10 wh as mentioned in the data description.
- For the lighting energy I removed the negatives values as they mostly invalid.
- And for other values I removed the negative values – (humidty) and for outliers I used IQR method to cap them

STEP 2 – Handling Missing data –

- As every column contain missing data and most importantly our target column also contain the null values which is big problem as we don't have target we can't train our model. So I removed the rows where the target value is missing.
- For rest rows I used Imputation methods – For normal curve I used mean and for other I have used median

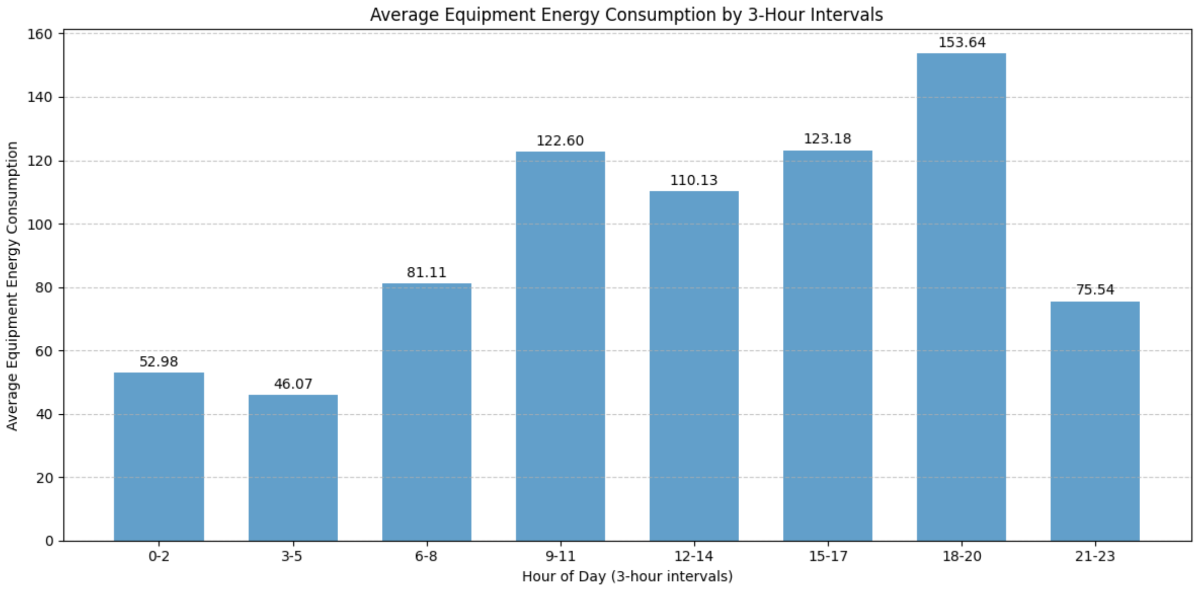
Now Understanding Relationship between features and target-

1. I tried plotting the energy consumption throughout the day – on hourly basis

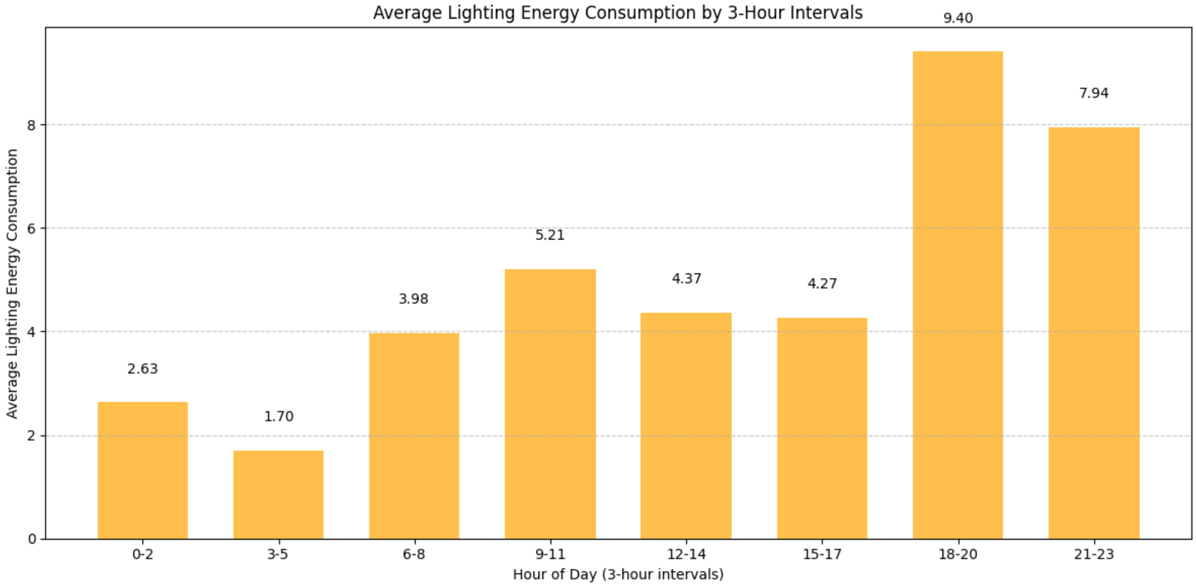


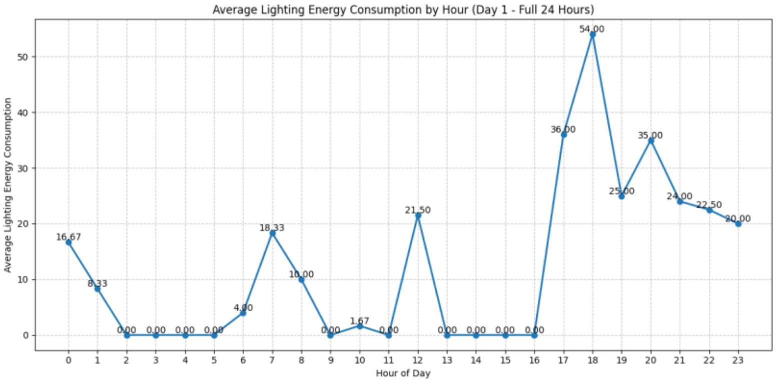
Insights – I observed that the factory mainly runs from morning 7 to night around 10 – 11 pm and rest it is on standby mode.

2. Also plotted on 3 hours window as well



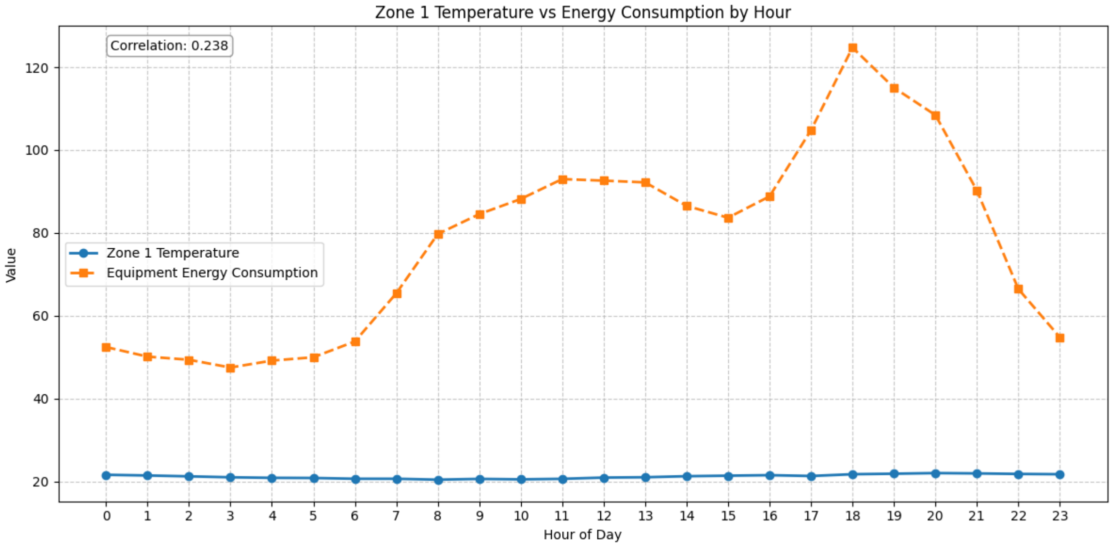
3. Same with the lighting energy –





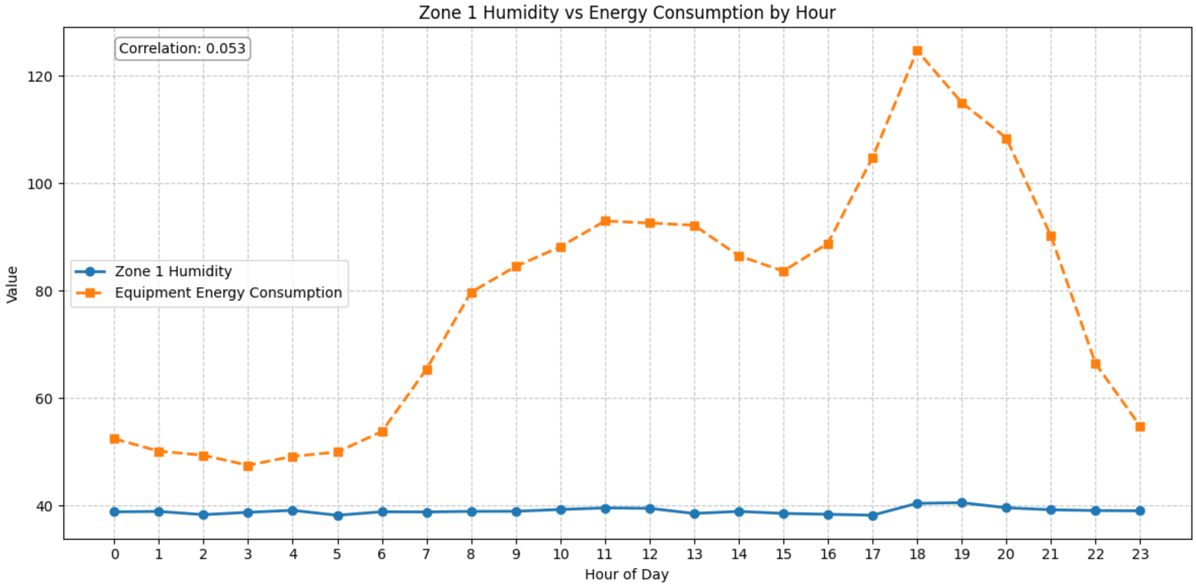
Insights – Lighting energy is mostly active in late evening which relates to low visibility in the factory from around – evening – 5 pm

4. Now I try to find the relation between features and target variable – firstly I tried with individual zone temperature and humidity –

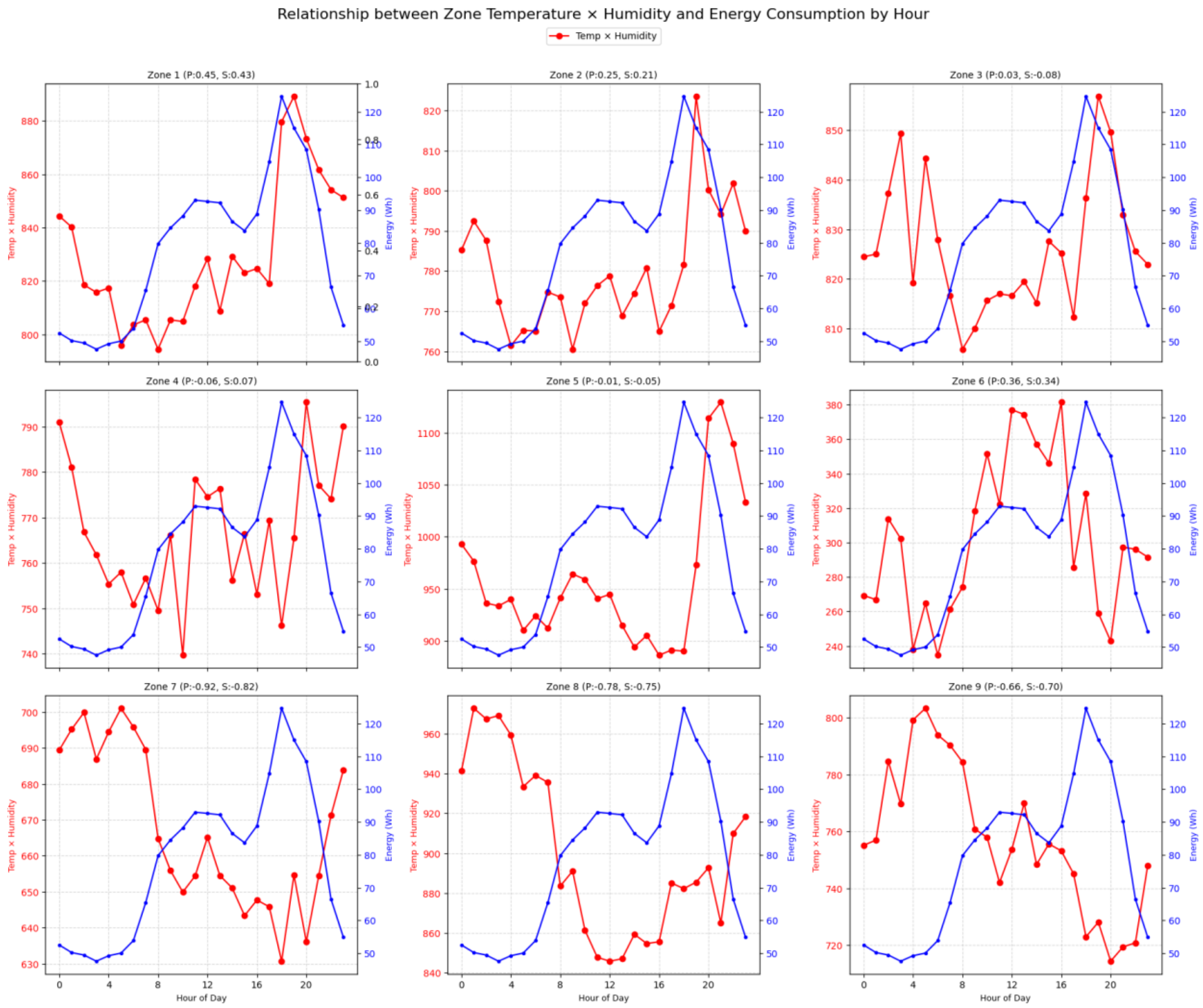


As you can see the don't have much relation and correlation is also coming low –

Same with humidity -

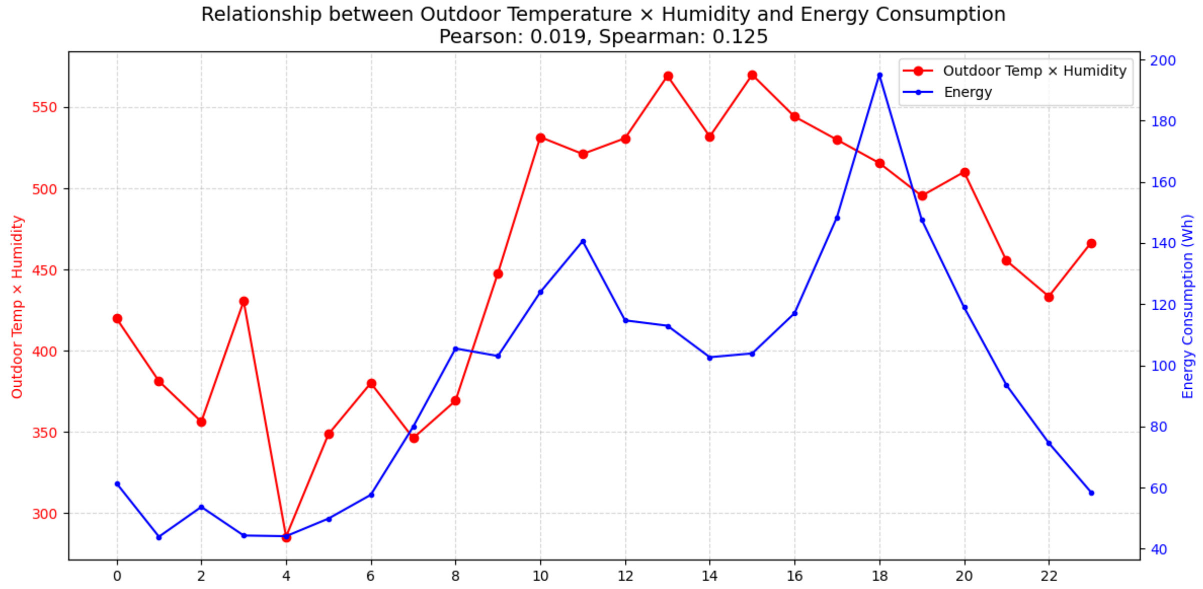


5. Then I tried to multiply those feature zone wise like zone1_temp * zone1_humidity -> so that I can have single value for single zone – then plotted against the energy consumption.



And as you can see there is some relation between them.

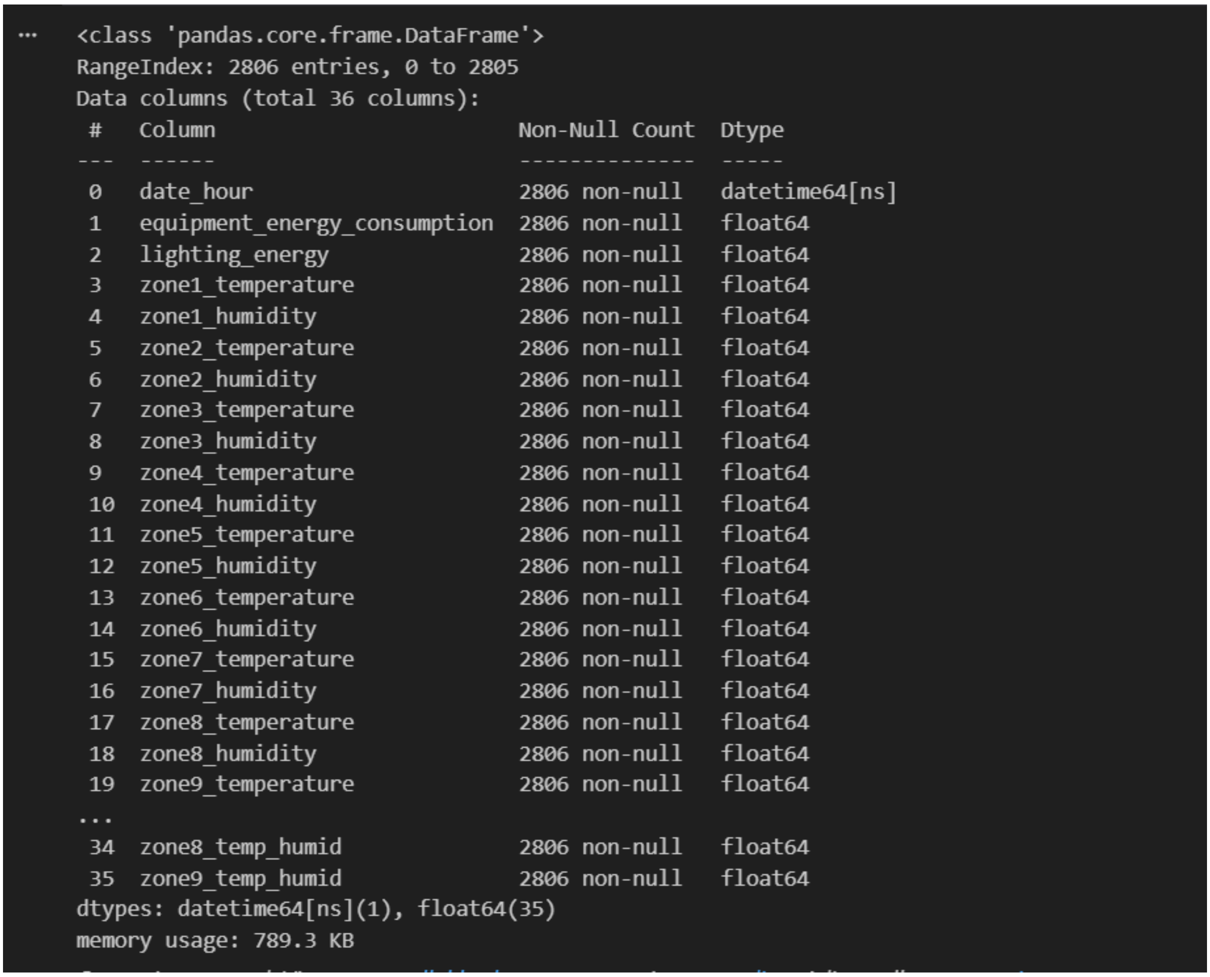
6. Then I tried to check as well for the outside temp and humidity as well –



It doesn’t show much relation.

Feature engineering –

Then I tried to convert the minutes data to hourly data for more smoother curve so that model can easily adapt the data and patterns in it.



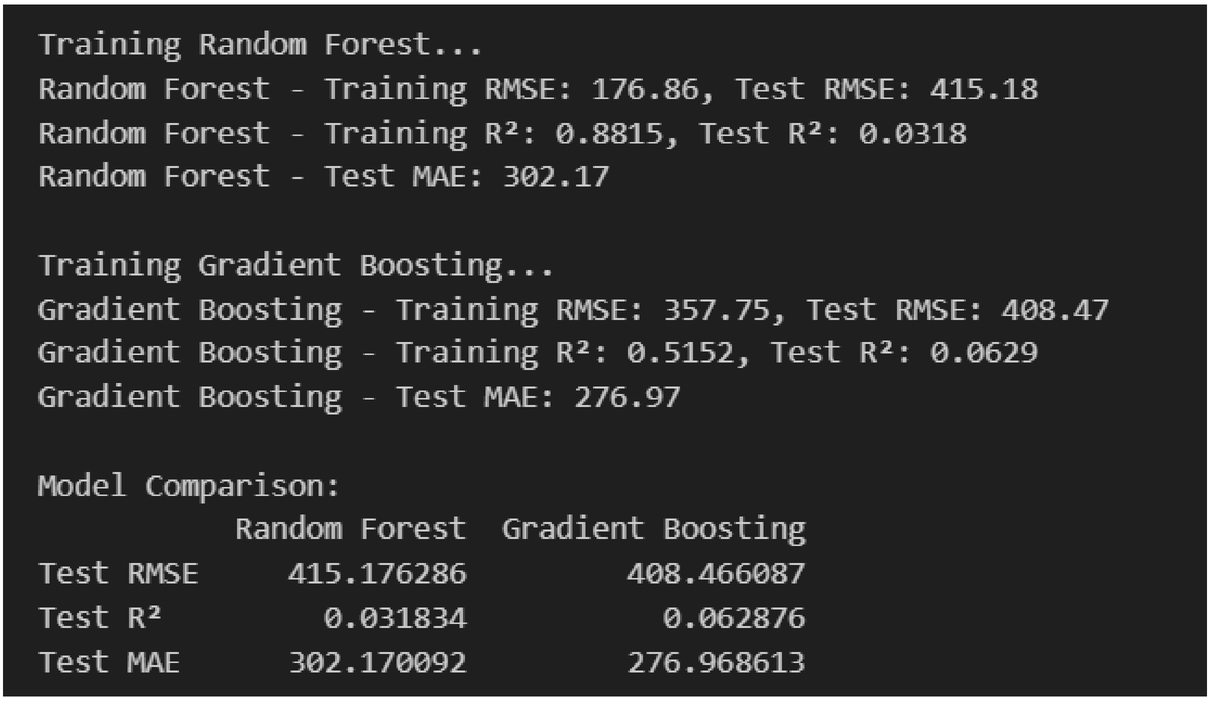
```
...  <class 'pandas.core.frame.DataFrame'>
     RangeIndex: 2806 entries, 0 to 2805
     Data columns (total 36 columns):
      #   Column                        Non-Null Count  Dtype
     ---  ------                        --------------  -----
      0   date_hour                     2806 non-null   datetime64[ns]
      1   equipment_energy_consumption  2806 non-null   float64
      2   lighting_energy               2806 non-null   float64
      3   zone1_temperature             2806 non-null   float64
      4   zone1_humidity                2806 non-null   float64
      5   zone2_temperature             2806 non-null   float64
      6   zone2_humidity                2806 non-null   float64
      7   zone3_temperature             2806 non-null   float64
      8   zone3_humidity                2806 non-null   float64
      9   zone4_temperature             2806 non-null   float64
      10  zone4_humidity                2806 non-null   float64
      11  zone5_temperature             2806 non-null   float64
      12  zone5_humidity                2806 non-null   float64
      13  zone6_temperature             2806 non-null   float64
      14  zone6_humidity                2806 non-null   float64
      15  zone7_temperature             2806 non-null   float64
      16  zone7_humidity                2806 non-null   float64
      17  zone8_temperature             2806 non-null   float64
      18  zone8_humidity                2806 non-null   float64
      19  zone9_temperature             2806 non-null   float64
     ...
      34  zone8_temp_humid              2806 non-null   float64
      35  zone9_temp_humid              2806 non-null   float64
     dtypes: datetime64[ns](1), float64(35)
     memory usage: 789.3 KB
```

Due to which my data got reduced – from 16 K to 3K around..

Then as this is a time series data I tried to split the data into X and y Dataset – not in random manner as order matters in this.

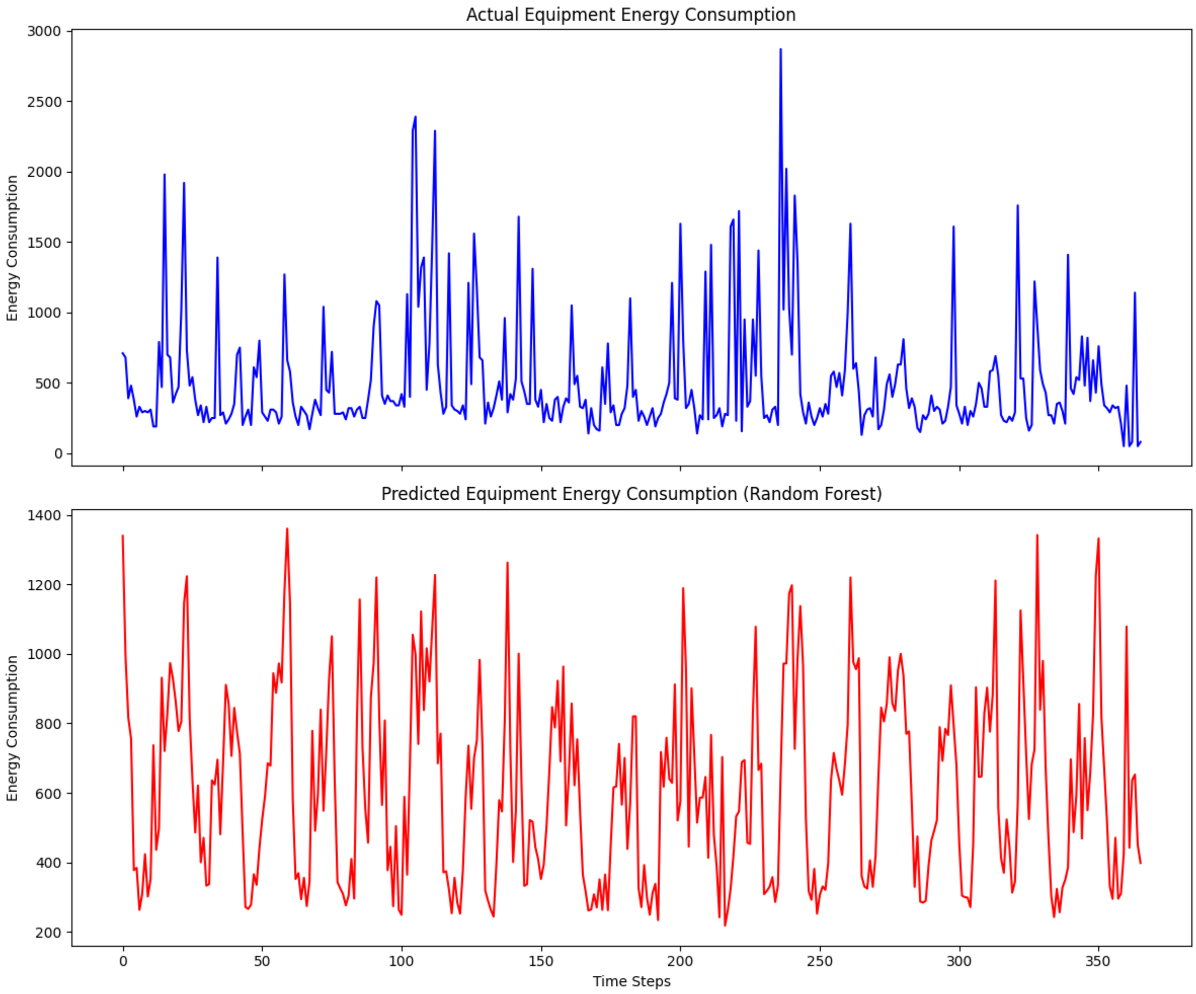
ML Model –

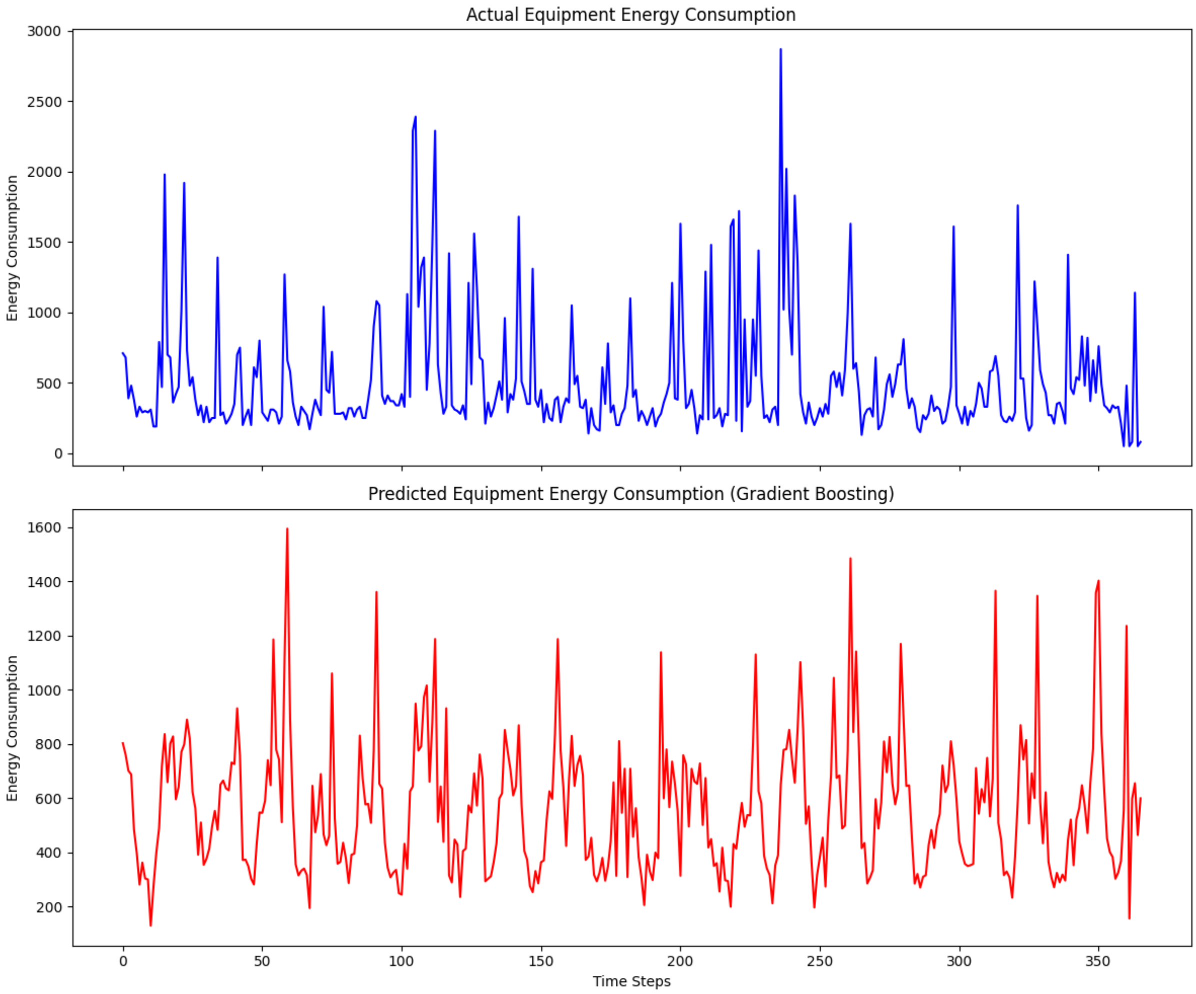
After That I used two models – Random Forest Regressor, and Gradient Boosting.



```
Training Random Forest...
Random Forest - Training RMSE: 176.86, Test RMSE: 415.18
Random Forest - Training R²: 0.8815, Test R²: 0.0318
Random Forest - Test MAE: 302.17

Training Gradient Boosting...
Gradient Boosting - Training RMSE: 357.75, Test RMSE: 408.47
Gradient Boosting - Training R²: 0.5152, Test R²: 0.0629
Gradient Boosting - Test MAE: 276.97

Model Comparison:
           Random Forest  Gradient Boosting
Test RMSE     415.176286         408.466087
Test R²         0.031834           0.062876
Test MAE      302.170092         276.968613
```

These are the values and accuracy score which I got – I know these are not got enough. But this what I can think of – And I tried. Maybe there is something I miss of But little bit of guidance can help me to solve this issue.


Actual Equipment Energy Consumption
Energy Consumption
3000
2500
2000
1500
1000
500
0
Predicted Equipment Energy Consumption (Random Forest)
Energy Consumption
1400
1200
1000
800
600
400
200
0
50
100
150
200
250
300
350
Time Steps



Thank you – I tried

Shubham Mahobia