ICS474: Big Data Analytics Course Project

Semester: 241

Section: 2

Student 1

Name: Ziyad Alshamrani

ID: 202021840

Student 2

Name: Elyas Alhabub

ID: 202036960

Used Dataset:

https://www.kaggle.com/datasets/jeanmidev/smart-meters-in-london

Contents

Project Objective 4

Report Contents 4

Part 1: Data Understanding and Exploration 4

1. Dataset Overview 4

2. Feature Description 5

3. Dataset Structure 6

4. Data Exploration 7

information_households Exploration 8

daily_dataset Exploration 10

weather_daily_darksky Exploration 13

acorn_details Exploration 21

uk_bank_holidays Exploration 26

5. Combining dataset and finding the target variable 27

6.Correlation Analysis with Target Variable 34

Part 2 35

7. Handling Missing Data 35

8. Feature Selection 36

9. Feature Scaling. 37

10. Encoding Categorical Variables 37

Part 3 37

11. Algorithm Selection 37

12. Data Splitting 38

13. Model Training and Evaluation 38

14. Performance Analysis 39

15. Model Improvement 40

16. Validation 42

17. Final Model Selection 42

Part 4 42

18. Data Distribution..42
19. Feature Importance..49
20. Model Performance Across Features 49

Project Objective

The purpose of this project is to explore the Smart meters in London dataset and finding a model that can predict the energy consumption in London for a given day.

Report Contents

The purpose of this report is to provide a description on the work done and the resaults found during the work of this project. Therefore, this report will not contain any code. For more information about the code, please refer to the Jupyter notebook and the readme file.

Part 1: Data Understanding and Exploration

This part will mainly focus on understanding the datasets available to use. We will not present the figures found in the jupyter notebook in this section here due to the large number of figures. Therefore, please refer to the jupyter notebook for the figures found here.

1. Dataset Overview

The dataset contains a reorganized version of data sourced from the London Data Store, showcasing energy consumption measurements for 5,567 London households involved in the UK Power Networks' Low Carbon London initiative from November 2011 to February 2014. The data specifically relates to electricity consumption recorded by smart meters.

The dataset helps analyze electricity consumption patterns at household level to identify peak usage times, optimize energy distribution, and reduce carbon emissions.

The data contains multiple datasets:

- **informations_households.csv**: This file contains detailed information about the households in the panel, such as their ACORN group and tariff type. It also specifies which block.csv.gz file stores their corresponding data.

- **halfhourly_dataset.zip**: A zip archive containing block files with half-hourly smart meter readings.

- **daily_dataset.zip**: A zip archive with block files providing daily aggregated data, including the number of measurements, minimum, maximum, mean, median, sum, and standard deviation.

• **acorn_details.csv**: This file provides details about the ACORN groups and their associated profiles, derived from an Excel spreadsheet. The first three columns represent studied attributes, and the ACORN-X column serves as an index for these attributes. Nationally, an index of 100 indicates the average, while a value of 150 suggests that 1.5 times more people in the ACORN group possess this attribute compared to the national average. More details can be found on the CACI website.

• **weather_daily_darksky.csv**: Contains daily weather data obtained from the Dark Sky API. Additional information on the parameters is available in the API documentation.

• **weather_hourly_darksky.csv**: Includes hourly weather data from the Dark Sky API, with parameter details provided in the API documentation.

2. Feature Description

• **Informations_households.csv**:

- **ACORN Group**: The socio-economic classification of the household, which is part of the ACORN segmentation system.
- **Tariff Type**: The electricity pricing plan or tariff that the household is on.
- **Block File**: Specifies the block.csv.gz file in which the corresponding household's data (electricity consumption) is stored.

• **halfhourly_dataset.zip**:

- **Smart Meter Readings**: Contains half-hourly electricity consumption readings for each household. These readings capture the household's electricity usage in 30-minute intervals, which helps in understanding consumption patterns throughout the day.

• **daily_dataset.zip**:

- **Number of Measurements**: The count of individual data points collected for each household daily.
- **Minimum Consumption**: The lowest electricity consumption value recorded for a household on a given day.
- **Maximum Consumption**: The highest electricity consumption value recorded for a household on a given day.
- **Mean Consumption**: The average electricity consumption for a household on a given day.
- **Median Consumption**: The middle value of the electricity consumption data for a household on a given day.
- **Sum Consumption**: The total electricity consumption for a household over the entire day.

- **Standard Deviation**: A measure of how much the electricity consumption deviates from the mean on a given day.

- **acorn_details.csv**:

  - **Attribute Columns**: The first three columns describe the attributes studied within each ACORN group (e.g., age, income, household size).
  - **ACORN-X**: The index for each attribute within an ACORN group. A value of 100 is the national average, and values greater than 100 indicate a higher prevalence of that attribute within the ACORN group compared to the national average.

- **weather_daily_darksky.csv**:

  - **Daily Weather Data**: Contains daily weather parameters such as temperature, humidity, precipitation, etc., obtained from the Dark Sky API. The parameters vary depending on the API data but typically include key weather statistics for each day.

- **weather_hourly_darksky.csv**:

  - **Hourly Weather Data**: Contains weather data on an hourly basis, including similar parameters to the daily dataset, such as temperature, humidity, and precipitation. This provides a more detailed, time-sensitive look at weather conditions throughout the day.

3. Dataset Structure

- **informations_households.csv**:

  - **Rows**: The file contains **5,567 rows**, corresponding to the 5,567 households in the panel.
  - **Columns**: The file has 5 columns
  - **Hierarchical Structure**: This file does not contain a hierarchical structure but rather flat, tabular data for each household.

- **halfhourly_dataset.zip**:

  - **Rows**: The data in the whole archive combined contains: 48 *5567 = 267,216 rows
  - **Columns**: There are 3 columns
  - **Hierarchical Structure**: The data is likely organized by **household ID** and **timestamp**, with each household's data stored in its own block file.

- **daily_dataset.zip**:

  - **Rows**: Each block file in this archive contains 25575 row
  - **Columns**: There are 9 columns

- **Hierarchical Structure**: Similar to the halfhourly dataset, the data is organized by **household ID** and **date**, with aggregated values for each day.

• **acorn_details.csv**:

- **Rows**: There 827 rows
- **Columns**: There are 20 columns
- **Hierarchical Structure**: There is a **group-level structure** here, where each ACORN group has its own set of attributes and indices. However, the file is mostly flat.

• **weather_daily_darksky.csv**:

- **Rows**: There are 883 rows
- **Columns**: There are 32 columns
- **Hierarchical Structure**: This file is flat, with no nested hierarchy, though it organizes data by **date**.

• **weather_hourly_darksky.csv**:

- **Rows**: 21166
- **Columns**: There are 12 columns
- **Hierarchical Structure**: The data is organized by **date** and **hour**, and for each combination, there are weather-related readings for each hour of the day.

4. Data Exploration

This section will combine the statistical summary, data distribution visualization, correlation analysis within each dataset, and outlier detection for This section will show the code and the result of Data exploration and understanding for the following datasets:

- information_households

- daily_dataset

- weather_daily_darksky

- acorn_details

- uk_bank_holidays

Note that we will not go over the other files for the following reasons:

- They represent the same dataset mentioned above except using different representation (for example : hours rather than day)

- There are too many files to explore and they represent the same data (for example:

daily_dataset has many subsets, each for a block in london). Therefore, we will consider block 0 only in this dataset.

The following content in this section will be a summary of the results found in the code. For more information, refer to the jupyter notebook. The information presented below are taken from the notebook.

information_households Exploration

```
==================================================
Analysis for Household Information
==================================================

1. Missing Values and Duplicates Analysis:

Missing Values:
Series([], dtype: int64)

Duplicate rows: 0

2. Statistical Summary:

Dataset Info:
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 5566 entries, 0 to 5565
Data columns (total 5 columns):
 #   Column         Non-Null Count  Dtype 
---  ------         --------------  ----- 
 0   LCLid          5566 non-null   object
 1   stdorToU       5566 non-null   object
 2   Acorn          5566 non-null   object
 3   Acorn_grouped  5566 non-null   object
 4   file           5566 non-null   object
dtypes: object(5)
memory usage: 217.6+ KB
None

Numerical Columns Summary:
            LCLid stdorToU    Acorn Acorn_grouped     file
count        5566     5566     5566          5566     5566
unique       5566        2       19             5      112
top     MAC005492      Std  ACORN-E      Affluent  block_0
```

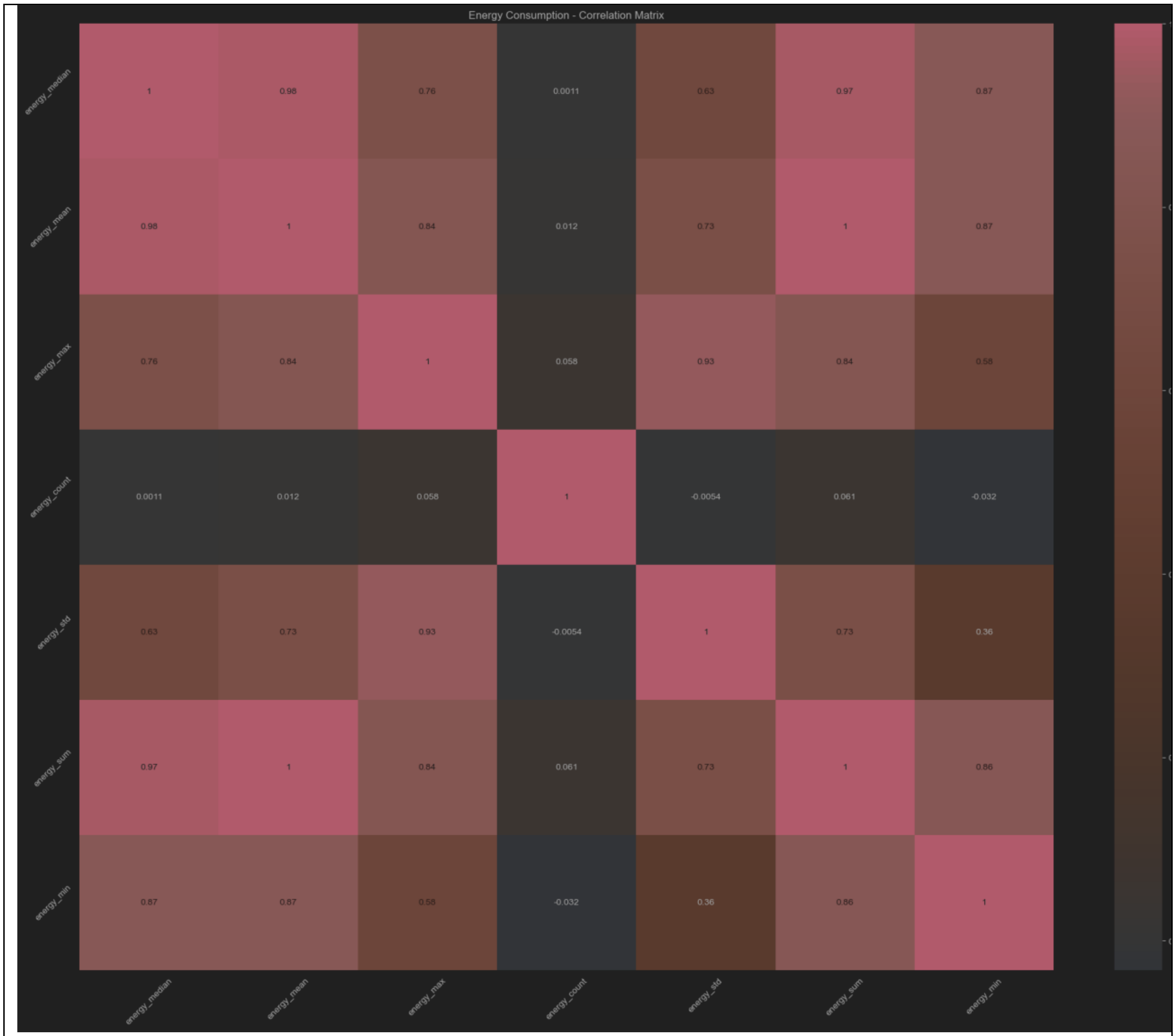
```
freq         1    4443    1567        2192      50

Categorical Columns Summary:

LCLid value counts:
LCLid
MAC005492    1
MAC003165    1
MAC004186    1
MAC004729    1
MAC000258    1
Name: count, dtype: int64

stdorToU value counts:
stdorToU
Std    4443
ToU    1123
Name: count, dtype: int64

Acorn value counts:
Acorn
ACORN-E    1567
ACORN-Q     831
ACORN-F     684
ACORN-H     455
ACORN-L     342
Name: count, dtype: int64

Acorn_grouped value counts:
Acorn_grouped
Affluent      2192
Adversity     1816
Comfortable    1507
ACORN-U         49
ACORN-           2
Name: count, dtype: int64

file value counts:
file
block_0     50
block_1     50
block_82    50
block_81    50
block_80    50
```

```
Name: count, dtype: int64

3. Distribution Plots: In notebook
4. Correlation Analysis cannot be done since no enough numerical columns for
correlation analysis

5. Outlier Analysis: No outliers
```

daily_dataset Exploration

```
==================================================
Analysis for Energy Consumption
==================================================

1. Missing Values and Duplicates Analysis:

Missing Values:
energy_std    78
dtype: int64

Duplicate rows: 0

2. Statistical Summary:

Dataset Info:
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 25574 entries, 0 to 25573
Data columns (total 9 columns):
 #   Column         Non-Null Count  Dtype
---  ------         --------------  -----
 0   LCLid          25574 non-null  object
 1   day            25574 non-null  datetime64[ns]
 2   energy_median  25574 non-null  float64
 3   energy_mean    25574 non-null  float64
 4   energy_max     25574 non-null  float64
 5   energy_count   25574 non-null  int64
 6   energy_std     25496 non-null  float64
 7   energy_sum     25574 non-null  float64
 8   energy_min     25574 non-null  float64
dtypes: datetime64[ns](1), float64(6), int64(1), object(1)
memory usage: 1.8+ MB
None
```

```
Numerical Columns Summary:
                         day  energy_median   energy_mean  \
count                  25574   25574.000000  25574.000000
mean   2013-05-21 02:39:00.689762816      0.366426      0.450346
min              2011-12-03 00:00:00      0.007000      0.012000
25%              2013-01-12 00:00:00      0.145500      0.213896
50%              2013-05-24 12:00:00      0.231500      0.318958
75%              2013-10-06 00:00:00      0.406500      0.528979
max              2014-02-28 00:00:00      5.522000      5.791125
std                              NaN      0.407172      0.421025

        energy_max  energy_count    energy_std    energy_sum    energy_min
count  25574.000000  25574.000000  25496.000000  25574.000000  25574.000000
mean       1.348149     47.807148      0.282705     21.543821      0.167271
min        0.012000      1.000000      0.002499      0.012000      0.000000
25%        0.732000     48.000000      0.138956     10.231000      0.050000
50%        1.173000     48.000000      0.243201     15.260000      0.091000
75%        1.762000     48.000000      0.384356     25.332750      0.159000
max        8.170999     48.000000      2.557372    277.973999      5.052000
std        0.910607      2.771494      0.201086     20.205007      0.255782

Categorical Columns Summary:

LCLid value counts:
LCLid
MAC000246    819
MAC004387    715
MAC004431    711
MAC004179    649
MAC004247    646
Name: count, dtype: int64

3. Plotting Distributions in notebook
4. Correlation analysis:
```



5. Outlier Analysis:

Outliers in energy_median:
Number of outliers: 2506
Percentage of outliers: 9.80%

Outliers in energy_mean:
Number of outliers: 1967
Percentage of outliers: 7.69%

Outliers in energy_max:
Number of outliers: 1017
Percentage of outliers: 3.98%

```
Outliers in energy_count:
Number of outliers: 288
Percentage of outliers: 1.13%

Outliers in energy_std:
Number of outliers: 696
Percentage of outliers: 2.72%

Outliers in energy_sum:
Number of outliers: 1969
Percentage of outliers: 7.70%

Outliers in energy_min:
Number of outliers: 2573
Percentage of outliers: 10.06%
```

weather_daily_darksky Exploration

```
==================================================
Analysis for Weather Data
==================================================

1. Missing Values and Duplicates Analysis:

Missing Values:
cloudCover    1
uvIndex       1
uvIndexTime   1
dtype: int64

Duplicate rows: 0

2. Statistical Summary:

Dataset Info:
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 882 entries, 0 to 881
Data columns (total 32 columns):
 #   Column                  Non-Null Count  Dtype
---  ------                  --------------  -----
 0   temperatureMax          882 non-null    float64
```

```
 1  temperatureMaxTime         882 non-null    object
 2  windBearing                882 non-null    int64
 3  icon                       882 non-null    object
 4  dewPoint                   882 non-null    float64
 5  temperatureMinTime         882 non-null    object
 6  cloudCover                 881 non-null    float64
 7  windSpeed                  882 non-null    float64
 8  pressure                   882 non-null    float64
 9  apparentTemperatureMinTime  882 non-null    object
 10  apparentTemperatureHigh     882 non-null    float64
 11  precipType                 882 non-null    object
 12  visibility                 882 non-null    float64
 13  humidity                   882 non-null    float64
 14  apparentTemperatureHighTime  882 non-null    object
 15  apparentTemperatureLow      882 non-null    float64
 16  apparentTemperatureMax      882 non-null    float64
 17  uvIndex                    881 non-null    float64
 18  time                       882 non-null    object
 19  sunsetTime                 882 non-null    object
 20  temperatureLow             882 non-null    float64
 21  temperatureMin             882 non-null    float64
 22  temperatureHigh            882 non-null    float64
 23  sunriseTime                882 non-null    object
 24  temperatureHighTime        882 non-null    object
 25  uvIndexTime                881 non-null    object
 26  summary                    882 non-null    object
 27  temperatureLowTime         882 non-null    object
 28  apparentTemperatureMin      882 non-null    float64
 29  apparentTemperatureMaxTime  882 non-null    object
 30  apparentTemperatureLowTime  882 non-null    object
 31  moonPhase                  882 non-null    float64
dtypes: float64(16), int64(1), object(15)
memory usage: 220.6+ KB
None

Numerical Columns Summary:
       temperatureMax  windBearing   dewPoint  cloudCover   windSpeed  \
count      882.000000   882.000000  882.000000  881.000000  882.000000
mean        13.660113   195.702948    6.530034    0.477605    3.581803
std          6.182744    89.340783    4.830875    0.193514    1.694007
min         -0.060000     0.000000   -7.840000    0.000000    0.200000
25%          9.502500   120.500000    3.180000    0.350000    2.370000
50%         12.625000   219.000000    6.380000    0.470000    3.440000
75%         17.920000   255.000000   10.057500    0.600000    4.577500
```

```
max      32.400000  359.000000  17.770000   1.000000   9.960000

      pressure  apparentTemperatureHigh  visibility   humidity  \
count  882.000000               882.000000  882.000000  882.000000
mean  1014.127540                12.723866   11.167143    0.781871
std     11.073038                 7.279168    2.466109    0.095348
min    979.250000                -6.460000    1.480000    0.430000
25%   1007.435000                 7.032500   10.327500    0.720000
50%   1014.615000                12.470000   11.970000    0.790000
75%   1021.755000                17.910000   12.830000    0.860000
max   1040.920000                32.420000   15.340000    0.980000

    apparentTemperatureLow  apparentTemperatureMax    uvIndex  \
count              882.000000              882.000000  881.000000
mean                 6.085045               12.929467    2.542565
std                  6.031967                7.105426    1.832985
min                 -8.880000               -4.110000    0.000000
25%                  1.522500                7.332500    1.000000
50%                  5.315000               12.625000    2.000000
75%                 11.467500               17.920000    4.000000
max                 20.540000               32.420000    7.000000

    temperatureLow  temperatureMin  temperatureHigh  \
count      882.000000      882.000000       882.000000
mean         7.709841        7.414161        13.542392
std          4.871004        4.888852         6.260196
min         -5.640000       -5.640000        -0.810000
25%          3.990000        3.705000         9.212500
50%          7.540000        7.100000        12.470000
75%         11.467500       11.277500        17.910000
max         20.540000       20.540000        32.400000

    apparentTemperatureMin   moonPhase
count              882.000000  882.000000
mean                 5.738039    0.500930
std                  6.048746    0.287022
min                 -8.880000    0.000000
25%                  1.105000    0.260000
50%                  4.885000    0.500000
75%                 11.277500    0.750000
max                 20.540000    0.990000

Categorical Columns Summary:
```

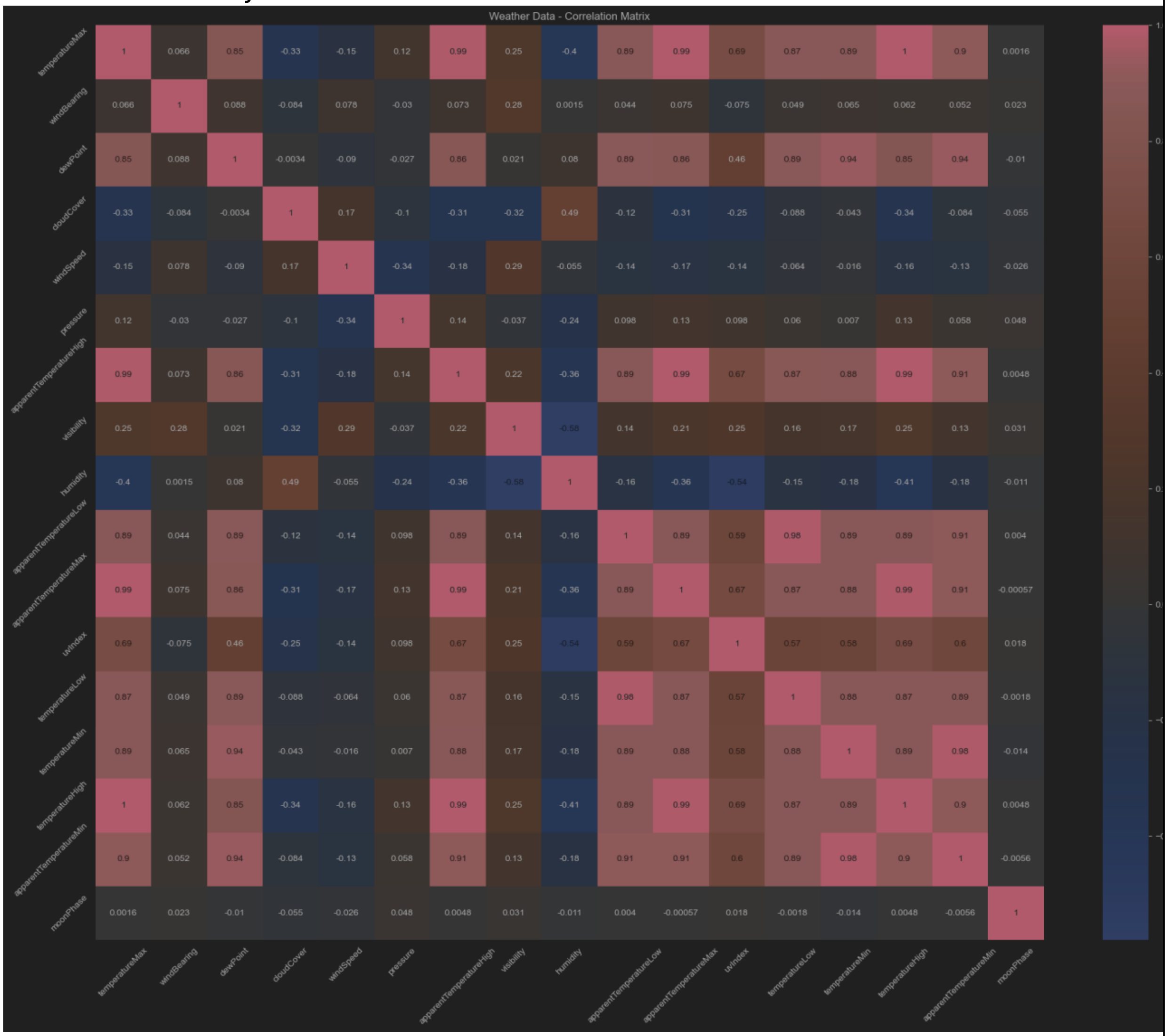
```
temperatureMaxTime value counts:
temperatureMaxTime
2011-11-11 23:00:00    1
2013-08-14 14:00:00    1
2013-04-02 15:00:00    1
2013-08-27 16:00:00    1
2013-02-03 22:00:00    1
Name: count, dtype: int64

icon value counts:
icon
partly-cloudy-day      619
wind                   124
fog                     91
partly-cloudy-night     33
cloudy                   9
Name: count, dtype: int64

temperatureMinTime value counts:
temperatureMinTime
2011-11-11 07:00:00    1
2013-08-14 05:00:00    1
2013-04-02 05:00:00    1
2013-08-27 06:00:00    1
2013-02-03 02:00:00    1
Name: count, dtype: int64

apparentTemperatureMinTime value counts:
apparentTemperatureMinTime
2011-11-11 07:00:00    1
2013-08-14 05:00:00    1
2013-04-02 05:00:00    1
2013-08-27 06:00:00    1
2013-02-03 04:00:00    1
Name: count, dtype: int64

precipType value counts:
precipType
rain    862
snow     20
Name: count, dtype: int64

apparentTemperatureHighTime value counts:
apparentTemperatureHighTime
```

```
2011-11-11 19:00:00    1
2013-08-14 14:00:00    1
2013-04-02 13:00:00    1
2013-08-27 16:00:00    1
2013-02-03 19:00:00    1
Name: count, dtype: int64

time value counts:
time
2011-11-11 00:00:00    1
2013-08-13 23:00:00    1
2013-04-01 23:00:00    1
2013-08-26 23:00:00    1
2013-02-03 00:00:00    1
Name: count, dtype: int64

sunsetTime value counts:
sunsetTime
2011-11-11 16:19:21    1
2013-08-14 19:27:07    1
2013-04-02 18:35:51    1
2013-08-27 19:00:14    1
2013-02-03 16:54:20    1
Name: count, dtype: int64

sunriseTime value counts:
sunriseTime
2011-11-11 07:12:14    1
2013-08-14 04:45:54    1
2013-04-02 05:34:21    1
2013-08-27 05:06:41    1
2013-02-03 07:36:47    1
Name: count, dtype: int64

temperatureHighTime value counts:
temperatureHighTime
2011-11-11 19:00:00    1
2013-08-14 14:00:00    1
2013-04-02 15:00:00    1
2013-08-27 16:00:00    1
2013-02-03 19:00:00    1
Name: count, dtype: int64

uvIndexTime value counts:
```

```
uvIndexTime
2011-11-11 11:00:00    1
2013-08-14 12:00:00    1
2013-04-02 11:00:00    1
2013-08-27 11:00:00    1
2013-02-03 10:00:00    1
Name: count, dtype: int64

summary value counts:
summary
Mostly cloudy throughout the day.    174
Partly cloudy throughout the day.    170
Partly cloudy until evening.         133
Mostly cloudy until evening.         118
Foggy in the morning.                 47
Name: count, dtype: int64

temperatureLowTime value counts:
temperatureLowTime
2011-11-11 19:00:00    1
2013-08-14 21:00:00    1
2013-04-03 05:00:00    1
2013-08-28 04:00:00    1
2013-02-04 03:00:00    1
Name: count, dtype: int64

apparentTemperatureMaxTime value counts:
apparentTemperatureMaxTime
2011-11-11 23:00:00    1
2013-08-14 14:00:00    1
2013-04-02 13:00:00    1
2013-08-27 16:00:00    1
2013-02-03 22:00:00    1
Name: count, dtype: int64

apparentTemperatureLowTime value counts:
apparentTemperatureLowTime
2011-11-11 19:00:00    1
2013-08-14 21:00:00    1
2013-04-03 04:00:00    1
2013-08-28 04:00:00    1
2013-02-04 03:00:00    1
Name: count, dtype: int64
```

3. Plotting Distributions : In notebook

4. Correlation analysis:



5. Outlier Analysis:

Outliers in temperatureMax:
Number of outliers: 2
Percentage of outliers: 0.23%

Outliers in windBearing:
Number of outliers: 0
Percentage of outliers: 0.00%

Outliers in dewPoint:
Number of outliers: 1
Percentage of outliers: 0.11%

Outliers in cloudCover:
Number of outliers: 3
Percentage of outliers: 0.34%

Outliers in windSpeed:
Number of outliers: 14
Percentage of outliers: 1.59%

Outliers in pressure:
Number of outliers: 11
Percentage of outliers: 1.25%

Outliers in apparentTemperatureHigh:
Number of outliers: 0
Percentage of outliers: 0.00%

Outliers in visibility:
Number of outliers: 67
Percentage of outliers: 7.60%

Outliers in humidity:
Number of outliers: 3
Percentage of outliers: 0.34%

Outliers in apparentTemperatureLow:
Number of outliers: 0
Percentage of outliers: 0.00%

Outliers in apparentTemperatureMax:
Number of outliers: 0
Percentage of outliers: 0.00%

Outliers in uvIndex:
Number of outliers: 0
Percentage of outliers: 0.00%

Outliers in temperatureLow:
Number of outliers: 0
Percentage of outliers: 0.00%

Outliers in temperatureMin:
Number of outliers: 0
Percentage of outliers: 0.00%

```
Outliers in temperatureHigh:
Number of outliers: 2
Percentage of outliers: 0.23%

Outliers in apparentTemperatureMin:
Number of outliers: 0
Percentage of outliers: 0.00%

Outliers in moonPhase:
Number of outliers: 0
Percentage of outliers: 0.00%
```

acorn_details Exploration

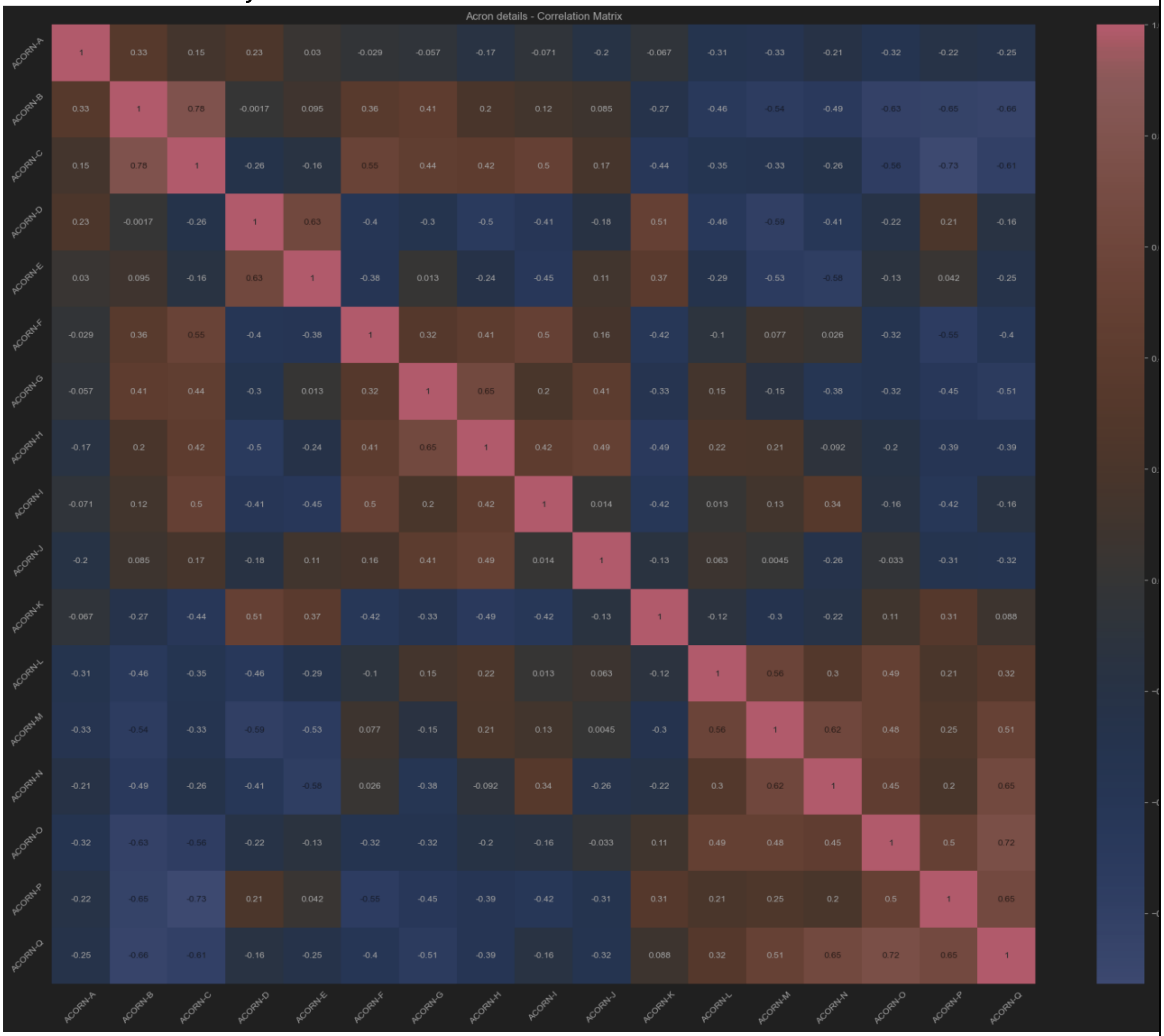
```
==================================================
Analysis for Acron details
==================================================

1. Missing Values and Duplicates Analysis:

Missing Values:
REFERENCE    1
dtype: int64

Duplicate rows: 0

2. Statistical Summary:

Dataset Info:
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 826 entries, 0 to 825
Data columns (total 20 columns):
 #   Column           Non-Null Count  Dtype
---  ------           --------------  -----
 0   MAIN CATEGORIES  826 non-null    object
 1   CATEGORIES       826 non-null    object
 2   REFERENCE        825 non-null    object
 3   ACORN-A          826 non-null    float64
 4   ACORN-B          826 non-null    float64
 5   ACORN-C          826 non-null    float64
 6   ACORN-D          826 non-null    float64
 7   ACORN-E          826 non-null    float64
```

```
 8   ACORN-F       826 non-null   float64
 9   ACORN-G       826 non-null   float64
 10  ACORN-H       826 non-null   float64
 11  ACORN-I       826 non-null   float64
 12  ACORN-J       826 non-null   float64
 13  ACORN-K       826 non-null   float64
 14  ACORN-L       826 non-null   float64
 15  ACORN-M       826 non-null   float64
 16  ACORN-N       826 non-null   float64
 17  ACORN-O       826 non-null   float64
 18  ACORN-P       826 non-null   float64
 19  ACORN-Q       826 non-null   float64
dtypes: float64(17), object(3)
memory usage: 129.2+ KB
None

Numerical Columns Summary:
           ACORN-A     ACORN-B     ACORN-C      ACORN-D     ACORN-E  \
count   826.000000  826.000000  826.000000   826.000000  826.000000
mean    131.313495  110.860256  100.080789   136.857507  117.894757
std     201.448212   42.464050   30.099529    97.740794   35.768807
min      12.000000    0.957011    0.281968     2.000000   21.000000
25%      87.000000   94.000000   86.000000    93.092150   99.000000
50%     104.000000  107.000000  100.000000   121.000000  117.000000
75%     128.000000  122.000000  113.000000   154.000000  135.000000
max    3795.000000  419.000000  272.000000  1159.034650  286.000000

          ACORN-F     ACORN-G     ACORN-H     ACORN-I     ACORN-J  \
count  826.000000  826.000000  826.000000  826.000000  826.000000
mean    95.574535  101.444276   97.298915   87.028545  104.216563
std     33.636661   21.798994   18.229234   30.337794   19.924033
min      0.000000    0.791419    1.155448    6.363259   16.050708
25%     81.000000   94.138076   91.000000   70.000000   97.000000
50%     98.000000  102.000000   99.000000   88.000000  105.000000
75%    108.000000  109.000000  105.000000  101.750000  115.000000
max    462.000000  295.000000  192.000000  410.000000  197.000000

           ACORN-K     ACORN-L     ACORN-M     ACORN-N     ACORN-O  \
count   826.000000  826.000000  826.000000  826.000000  826.000000
mean    127.482911   93.724209   91.410277   79.912379   95.579335
std      97.428159   22.177041   22.909602   33.995192   25.935770
min      17.000000    0.393546    0.714857    2.000000   11.000000
25%      85.000000   86.000000   82.000000   60.253502   86.000000
50%     109.000000   95.000000   93.000000   74.000000   96.000000
```

```
75%    144.000000  102.000000  101.000000   93.158386  104.000000
max   1821.000000  280.000000  161.000000  295.000000  252.000000

        ACORN-P     ACORN-Q
count  826.000000  826.000000
mean   100.141309   90.855423
std     37.210288   37.634017
min      9.000000    1.000000
25%     82.250000   71.250000
50%     96.000000   87.000000
75%    109.000000  101.000000
max    389.000000  326.000000

Categorical Columns Summary:

MAIN CATEGORIES value counts:
MAIN CATEGORIES
DIGITAL         372
FINANCE          94
LEISURE TIME     92
SHOPPING         44
POPULATION       35
Name: count, dtype: int64

CATEGORIES value counts:
CATEGORIES
Sites regularly visited                       85
Types of internet usage : Mobile Phone        46
Types of internet usage : Laptop or PC        45
Types of internet usage : Tablet / iPad       45
Purchased on the internet                     40
Name: count, dtype: int64

REFERENCE value counts:
REFERENCE
Mass Market    8
Premium        8
Value          8
Online         5
By phone       5
Name: count, dtype: int64
```

3. Plotting Distributions: In notebook

4. Correlation Analysis:



5. Outlier Analysis...

Outliers in ACORN-A:
Number of outliers: 80
Percentage of outliers: 9.69%

Outliers in ACORN-B:
Number of outliers: 97
Percentage of outliers: 11.74%

Outliers in ACORN-C:
Number of outliers: 72
Percentage of outliers: 8.72%

Outliers in ACORN-D:
Number of outliers: 50
Percentage of outliers: 6.05%

Outliers in ACORN-E:
Number of outliers: 50
Percentage of outliers: 6.05%

Outliers in ACORN-F:
Number of outliers: 67
Percentage of outliers: 8.11%

Outliers in ACORN-G:
Number of outliers: 95
Percentage of outliers: 11.50%

Outliers in ACORN-H:
Number of outliers: 77
Percentage of outliers: 9.32%

Outliers in ACORN-I:
Number of outliers: 29
Percentage of outliers: 3.51%

Outliers in ACORN-J:
Number of outliers: 65
Percentage of outliers: 7.87%

Outliers in ACORN-K:
Number of outliers: 52
Percentage of outliers: 6.30%

Outliers in ACORN-L:
Number of outliers: 85
Percentage of outliers: 10.29%

Outliers in ACORN-M:
Number of outliers: 99
Percentage of outliers: 11.99%

Outliers in ACORN-N:
Number of outliers: 40
Percentage of outliers: 4.84%

```
Outliers in ACORN-O:
Number of outliers: 105
Percentage of outliers: 12.71%

Outliers in ACORN-P:
Number of outliers: 83
Percentage of outliers: 10.05%

Outliers in ACORN-Q:
Number of outliers: 65
Percentage of outliers: 7.87%
```

uk_bank_holidays Exploration

```
==================================================
Analysis for Acron details
==================================================

1. Missing Values and Duplicates Analysis:

Missing Values:
Series([], dtype: int64)

Duplicate rows: 0

2. Statistical Summary:

Dataset Info:
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 25 entries, 0 to 24
Data columns (total 2 columns):
 #   Column         Non-Null Count  Dtype
---  ------         --------------  -----
 0   Bank holidays  25 non-null     object
 1   Type           25 non-null     object
dtypes: object(2)
memory usage: 532.0+ bytes
None

Numerical Columns Summary:
     Bank holidays      Type
```

```
count          25        25
unique         25        11
top      2012-12-26  Boxing Day
freq            1         3

Categorical Columns Summary:

Bank holidays value counts:
Bank holidays
2012-12-26    1
2013-06-05    1
2014-04-18    1
2014-04-21    1
2014-05-05    1
Name: count, dtype: int64

Type value counts:
Type
Boxing Day              3
Christmas Day           3
Summer bank holiday     3
Early May bank holiday  3
Easter Monday           3
Name: count, dtype: int64

3. Plotting Distributions In notebook

4. Correlation Analysis...
Not enough numerical columns for correlation analysis

5. Outlier Analysis: No outliers
```

5. Combining dataset and finding the target variable

The target variable of this project is the total energy consumption in London daily. However, this feature does not exist in the dataset, and it is not combined with other features like weather features. Therefore, we did the following transformation on the dataset to obtain a single dataframe that can be used in the next parts of this project:

- We combined all block data in daily_dataset (that has daily consumption for each house)

- We found the total energy consumption daily by summing all consumptions in a single day
- We cannot consider the acorn detail now since it was related to each house and block. Therefore, we will consider weather_daily_darksky dataset and uk_bank_holidays to see their effect on daily energy consumption in London.
- We combined all datasets together in one dataframe called "merged_dataset" based on date

The statistical analysis of merged_dataset is shown below. Other analysis are not done since it was done before.

```
==================================================
Analysis for Combined Dataset
==================================================

1. Missing Values and Duplicates Analysis:

Missing Values:
cloudCover      1
uvIndex         1
uvIndexTime     1
holiday       811
dtype: int64

Duplicate rows: 0

2. Statistical Summary:

Dataset Info:
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 829 entries, 0 to 828
Data columns (total 37 columns):
 #   Column                  Non-Null Count  Dtype
---  ------                  --------------  -----
 0   day                     829 non-null    datetime64[ns]
 1   energy_sum              829 non-null    float64
 2   house_number            829 non-null    int64
 3   avg_energy_per_house    829 non-null    float64
 4   temperatureMax          829 non-null    float64
 5   temperatureMaxTime      829 non-null    object
```

```
 6  windBearing            829 non-null    int64
 7  icon                   829 non-null    object
 8  dewPoint               829 non-null    float64
 9  temperatureMinTime     829 non-null    object
 10  cloudCover            828 non-null    float64
 11  windSpeed             829 non-null    float64
 12  pressure              829 non-null    float64
 13  apparentTemperatureMinTime  829 non-null    object
 14  apparentTemperatureHigh     829 non-null    float64
 15  precipType            829 non-null    object
 16  visibility            829 non-null    float64
 17  humidity              829 non-null    float64
 18  apparentTemperatureHighTime  829 non-null    object
 19  apparentTemperatureLow      829 non-null    float64
 20  apparentTemperatureMax      829 non-null    float64
 21  uvIndex               828 non-null    float64
 22  time                  829 non-null    object
 23  sunsetTime            829 non-null    object
 24  temperatureLow        829 non-null    float64
 25  temperatureMin        829 non-null    float64
 26  temperatureHigh       829 non-null    float64
 27  sunriseTime           829 non-null    object
 28  temperatureHighTime   829 non-null    object
 29  uvIndexTime           828 non-null    object
 30  summary               829 non-null    object
 31  temperatureLowTime    829 non-null    object
 32  apparentTemperatureMin      829 non-null    float64
 33  apparentTemperatureMaxTime  829 non-null    object
 34  apparentTemperatureLowTime  829 non-null    object
 35  moonPhase             829 non-null    float64
 36  holiday               18 non-null     object
dtypes: datetime64[ns](1), float64(18), int64(2), object(16)
memory usage: 239.8+ KB
None

Numerical Columns Summary:
                             day   energy_sum  house_number  \
count                        829   829.000000    829.000000
mean   2013-01-09 23:40:53.558504192  42854.268973   4234.314837
min          2011-11-23 00:00:00     90.385000     13.000000
25%          2012-06-17 00:00:00  34421.895002   4084.000000
50%          2013-01-10 00:00:00  45814.022999   5138.000000
75%          2013-08-05 00:00:00  58757.548990   5367.000000
max          2014-02-28 00:00:00  82650.492003   5541.000000
```

```
std              NaN  20143.330599  1789.999539

    avg_energy_per_house  temperatureMax  windBearing   dewPoint  \
count         829.000000      829.000000   829.000000 829.000000
mean           10.354520       13.680543   197.945718   6.544753
min             0.208997       -0.060000     0.000000  -7.840000
25%             8.565412        9.320000   130.000000   3.110000
50%            10.321918       12.570000   221.000000   6.440000
75%            11.832222       18.090000   256.000000  10.220000
max            15.940238       32.400000   359.000000  17.770000
std             1.887923        6.332020    89.062878   4.917198

    cloudCover   windSpeed     pressure  ...  visibility   humidity  \
count  828.000000  829.000000   829.000000  ...  829.000000 829.000000
mean     0.479638    3.606273  1014.034946  ...   11.296767   0.780531
min      0.000000    0.200000   979.250000  ...    1.480000   0.430000
25%      0.350000    2.370000  1007.400000  ...   10.540000   0.720000
50%      0.470000    3.480000  1014.410000  ...   12.040000   0.790000
75%      0.600000    4.600000  1021.600000  ...   12.860000   0.850000
max      1.000000    9.960000  1040.920000  ...   15.340000   0.980000
std      0.191571    1.712262    11.127844  ...    2.320175   0.094711

    apparentTemperatureLow  apparentTemperatureMax    uvIndex  \
count             829.000000              829.000000 828.000000
mean                6.134415               12.913329   2.591787
min                -8.880000               -4.110000   0.000000
25%                 1.430000                7.170000   1.000000
50%                 5.370000               12.570000   2.000000
75%                11.650000               18.120000   4.000000
max                20.540000               32.420000   7.000000
std                 6.154799                7.282502   1.866793

    temperatureLow  temperatureMin  temperatureHigh  \
count      829.000000      829.000000       829.000000
mean         7.763257        7.453209        13.558432
min         -5.640000       -5.640000        -0.810000
25%          3.980000        3.640000         9.100000
50%          7.540000        7.060000        12.410000
75%         11.650000       11.460000        18.080000
max         20.540000       20.540000        32.400000
std          4.962144        4.984435         6.411557

    apparentTemperatureMin  moonPhase
count             829.000000 829.000000
```

```
mean            5.766743   0.500796
min            -8.880000   0.000000
25%             1.070000   0.250000
50%             4.890000   0.500000
75%            11.460000   0.750000
max            20.540000   0.990000
std             6.173678   0.288515

[8 rows x 21 columns]

Categorical Columns Summary:

temperatureMaxTime value counts:
temperatureMaxTime
2011-11-23 14:00:00    1
2013-06-02 15:00:00    1
2013-05-23 14:00:00    1
2013-05-24 15:00:00    1
2013-05-25 17:00:00    1
Name: count, dtype: int64

icon value counts:
icon
partly-cloudy-day      594
wind                   122
fog                     69
partly-cloudy-night     31
cloudy                   8
Name: count, dtype: int64

temperatureMinTime value counts:
temperatureMinTime
2011-11-23 07:00:00    1
2013-06-02 04:00:00    1
2013-05-23 05:00:00    1
2013-05-24 06:00:00    1
2013-05-25 04:00:00    1
Name: count, dtype: int64

apparentTemperatureMinTime value counts:
apparentTemperatureMinTime
2011-11-23 07:00:00    1
2013-06-02 05:00:00    1
2013-05-23 05:00:00    1
```

```
2013-05-24 06:00:00    1
2013-05-25 04:00:00    1
Name: count, dtype: int64

precipType value counts:
precipType
rain    809
snow     20
Name: count, dtype: int64

apparentTemperatureHighTime value counts:
apparentTemperatureHighTime
2011-11-23 14:00:00    1
2013-06-02 15:00:00    1
2013-05-23 14:00:00    1
2013-05-24 15:00:00    1
2013-05-25 17:00:00    1
Name: count, dtype: int64

time value counts:
time
2011-11-23 00:00:00    1
2013-06-01 23:00:00    1
2013-05-22 23:00:00    1
2013-05-23 23:00:00    1
2013-05-24 23:00:00    1
Name: count, dtype: int64

sunsetTime value counts:
sunsetTime
2011-11-23 16:03:50    1
2013-06-02 20:10:20    1
2013-05-23 19:58:11    1
2013-05-24 19:59:32    1
2013-05-25 20:00:51    1
Name: count, dtype: int64

sunriseTime value counts:
sunriseTime
2011-11-23 07:32:38    1
2013-06-02 03:49:29    1
2013-05-23 03:59:07    1
2013-05-24 03:57:58    1
2013-05-25 03:56:51    1
```

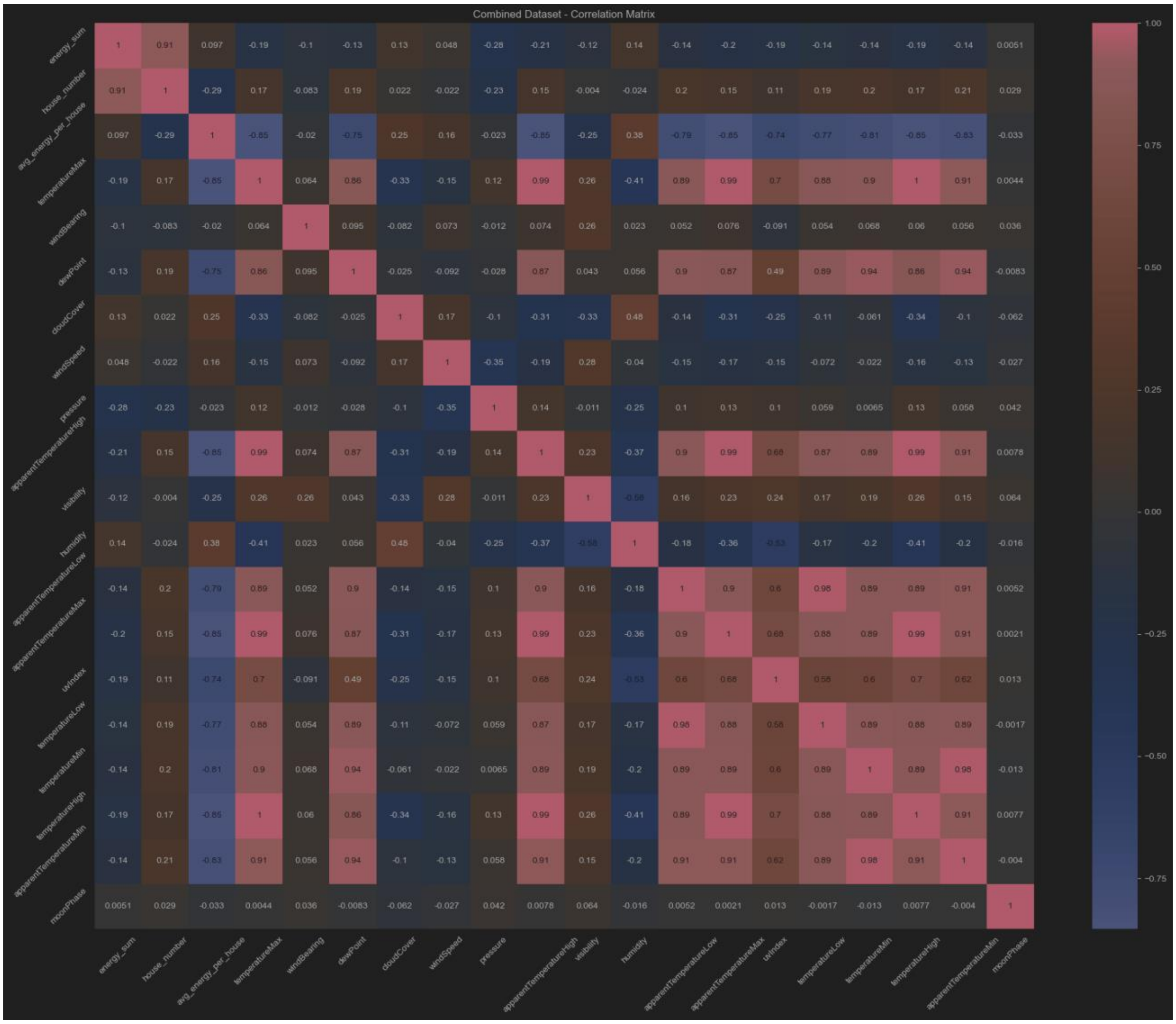
```
Name: count, dtype: int64

temperatureHighTime value counts:
temperatureHighTime
2011-11-23 14:00:00    1
2013-06-02 15:00:00    1
2013-05-23 14:00:00    1
2013-05-24 15:00:00    1
2013-05-25 17:00:00    1
Name: count, dtype: int64

uvIndexTime value counts:
uvIndexTime
2011-11-23 10:00:00    1
2013-06-01 11:00:00    1
2013-05-22 11:00:00    1
2013-05-23 13:00:00    1
2013-05-24 11:00:00    1
Name: count, dtype: int64

summary value counts:
summary
Partly cloudy throughout the day.    167
Mostly cloudy throughout the day.    167
Partly cloudy until evening.         130
Mostly cloudy until evening.         111
Foggy in the morning.                 36
Name: count, dtype: int64

temperatureLowTime value counts:
temperatureLowTime
2011-11-23 22:00:00    1
2013-06-03 03:00:00    1
2013-05-24 06:00:00    1
2013-05-25 04:00:00    1
2013-05-26 05:00:00    1
Name: count, dtype: int64

apparentTemperatureMaxTime value counts:
apparentTemperatureMaxTime
2011-11-23 14:00:00    1
2013-06-02 15:00:00    1
2013-05-23 14:00:00    1
2013-05-24 15:00:00    1
```

```
2013-05-25 17:00:00    1
Name: count, dtype: int64

apparentTemperatureLowTime value counts:
apparentTemperatureLowTime
2011-11-23 22:00:00    1
2013-06-03 03:00:00    1
2013-05-24 06:00:00    1
2013-05-25 04:00:00    1
2013-05-26 05:00:00    1
Name: count, dtype: int64

holiday value counts:
holiday
Good Friday            2
Early May bank holiday    2
Summer bank holiday      2
Easter Monday          2
Christmas Day          2
Name: count, dtype: int64
```

6.Correlation Analysis with Target Variable

The following heatmap shows the correlation matrix of all the features in the merged dataset. Since our concern is the target variable, we can see from the matrix that the number of houses has the highest correlation with it (correlation: 0.91), then air pressure (correlation: 0.28).

This indicates that the major change in the energy consumption daily in londo os related to the number of consuming houses in the city.



Part 2

7. Handling Missing Data

The columns that has missing data in merged dataset are:

- Column: cloudCover, Type: float64, Missing Values: 1
- Column: uvIndex, Type: float64, Missing Values: 1
- Column: uvIndexTime, Type: object, Missing Values: 1
- Column: holiday, Type: object, Missing Values: 811

So, to handle the missing values we will do the following:

- For numerical columns: we will replace the missing values with the mean value

- For categorical columns: we will replace the missing values with the mode

8. Feature Selection

We considered all the columns that has time and date (except the day column), and the columns that has summery of the data or meaningless information for our prediction target, which is predicting the energy consumption in London for a given day.

The columns that were excluded from the dataset:

- "house_number"
- "avg_energy_per_house"
- "temperatureMaxTime"
- "temperatureMinTime"
- "apparentTemperatureMinTime"
- "apparentTemperatureHighTime"
- "time"
- "sunsetTime"
- "sunriseTime"
- "temperatureHighTime"
- "uvIndexTime"
- "temperatureLowTime"
- "apparentTemperatureMaxTime"
- "apparentTemperatureLowTime"
- "icon"
- "summary"
- "precipType"

The remaining columns are with their type:

- "day" (Type: datetime64[ns])
- "energy_sum" (Type: float64)
- "windBearing" (Type: int64)
- "dewPoint" (Type: float64)
- "cloudCover" (Type: float64)
- "windSpeed" (Type: float64)
- "pressure" (Type: float64)
- "apparentTemperatureHigh" (Type: float64)
- "visibility" (Type: float64)
- "humidity" (Type: float64)
- "apparentTemperatureLow" (Type: float64)
- "apparentTemperatureMax" (Type: float64)

- "uvIndex" (Type: float64)
- "temperatureLow" (Type: float64)
- "temperatureMin" (Type: float64)
- "temperatureHigh" (Type: float64)
- "moonPhase" (Type: float64)
- "holiday" (Type: object)

9. Feature Scaling.

Since the features have different ranges, it must be scaled using same scaler. The reason behind that lies in the effect that different scales will result in biased results. Take linear regression for example, if two features have same range but different scales, they will affect the target variable differently. For this reason, we will use StandardScaler to scale the data before training the model. Moreover, the scaler will be applying to the training data only.

10. Encoding Categorical Variables

As can be seen from the remaining column above, only “holiday” and “day” needs encoding.

For “holiday”, we used label encoding since it has many categories.

For “day”, we set the first data to zero and took it as a reference for any other date we have to convert the dates into numerical labels. So, if “11-11-2011” is day zero, then next day is day 1 and so on.

Part 3

11. Algorithm Selection

The target variable in our data is numerical, which is energy_sum. Therefore, the best algorithm type to use in our case is regression algorithms to estimate the values in the future. The regression algorithms that will be evaluated here are:

- Linear Regressor
- Ridge
- Decision Tree

- Random Forest
- ARD Regressor

12. Data Splitting

After we found the energy sum per day in London, the amount of data decreased and it has now 829 row. Therefore, it is better to use K-fold since Ensures all data is utilized for training and testing. Provides a robust evaluation by averaging over multiple splits. Mitigates bias from a single random split. Moreover, this will prevent overfitting and underfitting.
We will use k = 10

13. Model Training and Evaluation

We will train and test each model 10 time, one for each fold then will find the average results. Since we are evaluating regression, the best metrics to evaluate the mode are the following:

.R^2 (R-squared)

- **Definition**: R^2 is the proportion of variance in the dependent variable that is explained by the independent variables in the model. It measures the goodness of fit.
- **Range**: R^2 ranges from **0 to 1**, where:
    - **0** means the model does not explain any of the variance in the data.
    - **1** means the model explains all the variance in the data.
    - **Negative R^2** can occur when the model is worse than a horizontal line (i.e., predicting the mean of the target variable).

- **Interpretation**: A higher R^2 value indicates a better fit of the model to the data. It tells you how well the model captures the underlying trend in the data.

.RMSE (Root Mean Squared Error)

- **Definition**: RMSE is a measure of the average magnitude of errors between the predicted and actual values. It gives an idea of how far the predictions are from the true values, in the same units as the target variable.
- **Range**: RMSE can take values from **0 to infinity**:
    - **0** means perfect predictions (no error).
    - Larger RMSE values indicate worse performance (larger errors).

- **Interpretation**: RMSE gives the magnitude of the average error in the model's predictions, so a lower RMSE indicates better predictive accuracy.

14. Performance Analysis

After training and testing the model. We evaluated their performance using R-Squared and RMSE for each fold, then found each regressor average results. The best regressor we found was “Random Forest Regressor”. The evaluation results are shown below:

```
Evaluating Linear Regressor...
  Fold 1: R2 = 0.5620, RMSE = 13367.2539
  Fold 2: R2 = 0.6218, RMSE = 12722.5192
  Fold 3: R2 = 0.5574, RMSE = 13078.4331
  Fold 4: R2 = 0.6354, RMSE = 13271.5477
  Fold 5: R2 = 0.5595, RMSE = 12728.9554
  Fold 6: R2 = 0.3968, RMSE = 16234.2055
  Fold 7: R2 = 0.6312, RMSE = 12188.4621
  Fold 8: R2 = 0.3684, RMSE = 12272.5248
  Fold 9: R2 = 0.6694, RMSE = 11879.8458
  Fold 10: R2 = 0.6270, RMSE = 12956.6421
Average R2: 0.5629
Average RMSE: 13070.0390

Evaluating Ridge...
  Fold 1: R2 = 0.5689, RMSE = 13260.4899
  Fold 2: R2 = 0.6190, RMSE = 12768.9201
  Fold 3: R2 = 0.5547, RMSE = 13118.8110
  Fold 4: R2 = 0.6290, RMSE = 13388.4692
  Fold 5: R2 = 0.5642, RMSE = 12661.5600
  Fold 6: R2 = 0.3947, RMSE = 16261.9510
  Fold 7: R2 = 0.6341, RMSE = 12140.2972
  Fold 8: R2 = 0.3696, RMSE = 12260.5709
  Fold 9: R2 = 0.6697, RMSE = 11874.9603
  Fold 10: R2 = 0.6327, RMSE = 12858.2925
Average R2: 0.5637
Average RMSE: 13059.4322

Evaluating Decision Tree...
  Fold 1: R2 = 0.9856, RMSE = 2423.2459
  Fold 2: R2 = 0.9861, RMSE = 2443.1223
  Fold 3: R2 = 0.9692, RMSE = 3449.0484
  Fold 4: R2 = 0.9849, RMSE = 2704.4660
  Fold 5: R2 = 0.9841, RMSE = 2421.5884
  Fold 6: R2 = 0.9007, RMSE = 6585.1163
```

```
  Fold 7: R2 = 0.9821, RMSE = 2682.4636
  Fold 8: R2 = 0.9720, RMSE = 2583.6519
  Fold 9: R2 = 0.9834, RMSE = 2663.5342
  Fold 10: R2 = 0.9796, RMSE = 3028.7490
Average R2: 0.9728
Average RMSE: 3098.4986

Evaluating Random Forest...
  Fold 1: R2 = 0.9935, RMSE = 1627.4600
  Fold 2: R2 = 0.9919, RMSE = 1866.8543
  Fold 3: R2 = 0.9800, RMSE = 2781.3784
  Fold 4: R2 = 0.9915, RMSE = 2023.9988
  Fold 5: R2 = 0.9906, RMSE = 1860.7451
  Fold 6: R2 = 0.9160, RMSE = 6059.0811
  Fold 7: R2 = 0.9819, RMSE = 2699.7087
  Fold 8: R2 = 0.9799, RMSE = 2189.7034
  Fold 9: R2 = 0.9942, RMSE = 1573.4874
  Fold 10: R2 = 0.9875, RMSE = 2373.0873
Average R2: 0.9807
Average RMSE: 2505.5504

Evaluating ARD Regressor...
  Fold 1: R2 = 0.5624, RMSE = 13360.4705
  Fold 2: R2 = 0.5940, RMSE = 13180.6979
  Fold 3: R2 = 0.5475, RMSE = 13224.2060
  Fold 4: R2 = 0.6060, RMSE = 13796.6219
  Fold 5: R2 = 0.5719, RMSE = 12548.6155
  Fold 6: R2 = 0.4145, RMSE = 15994.5834
  Fold 7: R2 = 0.6276, RMSE = 12247.1456
  Fold 8: R2 = 0.3498, RMSE = 12451.2972
  Fold 9: R2 = 0.6663, RMSE = 11936.8131
  Fold 10: R2 = 0.6014, RMSE = 13394.1477
Average R2: 0.5542
Average RMSE: 13213.4599
```

15. Model Improvement

The best approach here to improve the model performance is using hyperparameter Tuning. We already considered different types of regression models, scaled the data before training, and did some feature engineering to the data (like creating energy sum feature). Further feature engineering might be helpful. However, this requires expertise in the domain of this study and time as well. So, the best choice for our case is to try hyper parameter tuning and compare the results with our result. We Implemented a code for tuning the hyperparameter of the “Random Forest Regressor” using GridSearchCV, which exhaustively searches through a manually specified set of hyperparameters. method. Then we evaluated the trained tuned regressor performance. However, the regressor performance did not improve with this tuning.

The parameters we used here are:

```
param_grid = {
   'n_estimators': [50, 100, 150],
   'max_depth': [None, 10, 20, 30],
   'min_samples_split': [2, 5, 10],
   'min_samples_leaf': [1, 2, 4]
}
```

The evaluation results:

```
Performing GridSearchCV for hyperparameter tuning...
Fitting 5 folds for each of 108 candidates, totalling 540 fits
Best Parameters: {'max_depth': 20, 'min_samples_leaf': 2, 'min_samples_split': 2,
'n_estimators': 50}

Evaluating the tuned Random Forest Regressor...
  Fold 1: R2 = 0.9931, RMSE = 1677.6458
  Fold 2: R2 = 0.9924, RMSE = 1802.4279
  Fold 3: R2 = 0.9780, RMSE = 2913.1780
  Fold 4: R2 = 0.9913, RMSE = 2055.6853
  Fold 5: R2 = 0.9907, RMSE = 1853.6584
  Fold 6: R2 = 0.9154, RMSE = 6078.3362
  Fold 7: R2 = 0.9833, RMSE = 2593.7013
  Fold 8: R2 = 0.9775, RMSE = 2317.4910
  Fold 9: R2 = 0.9925, RMSE = 1787.8751
  Fold 10: R2 = 0.9881, RMSE = 2317.4134
Average R2: 0.9802
Average RMSE: 2539.7412
```

16. Validation

We have already done this part in the data splitting using cross-validation to ensure the model generalize well on the data.

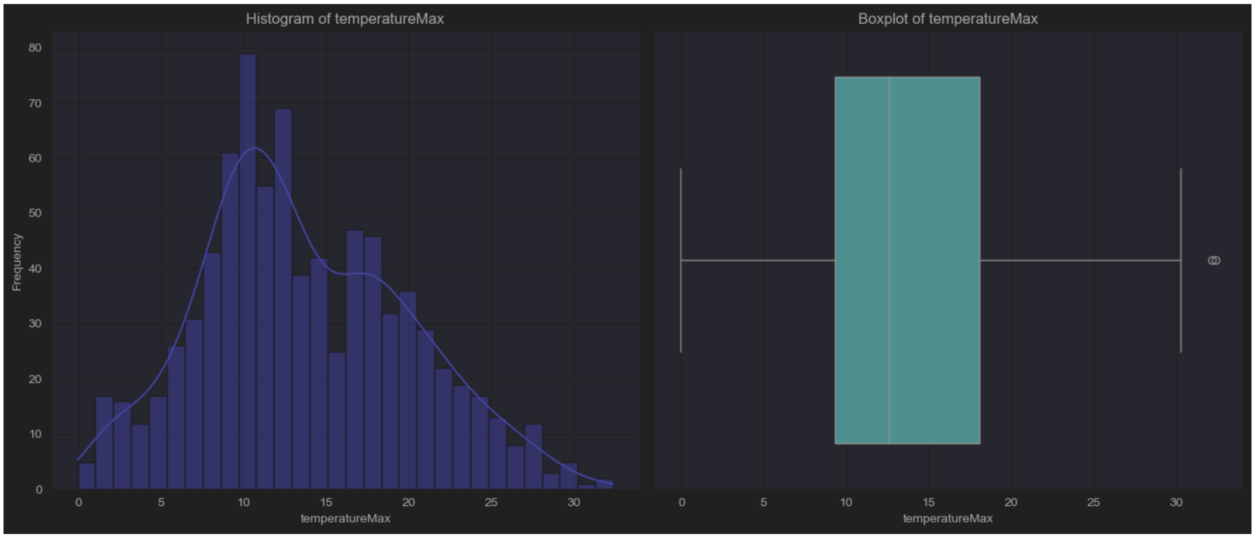
17. Final Model Selection

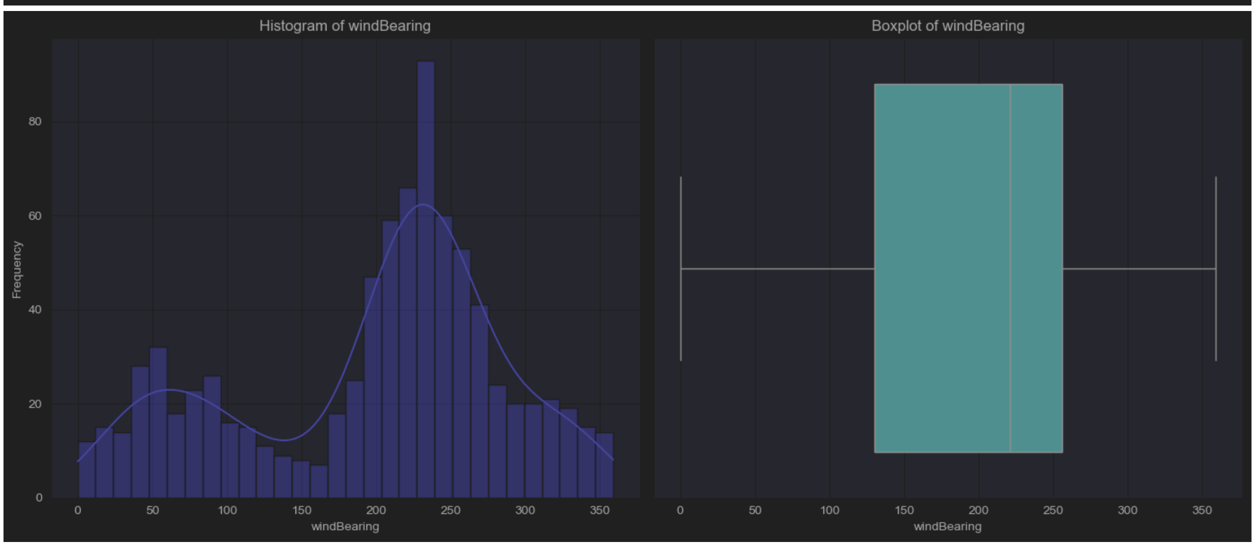
Based on the evaluation result we found earlier, the best model to choose is “Random Forest Regression”

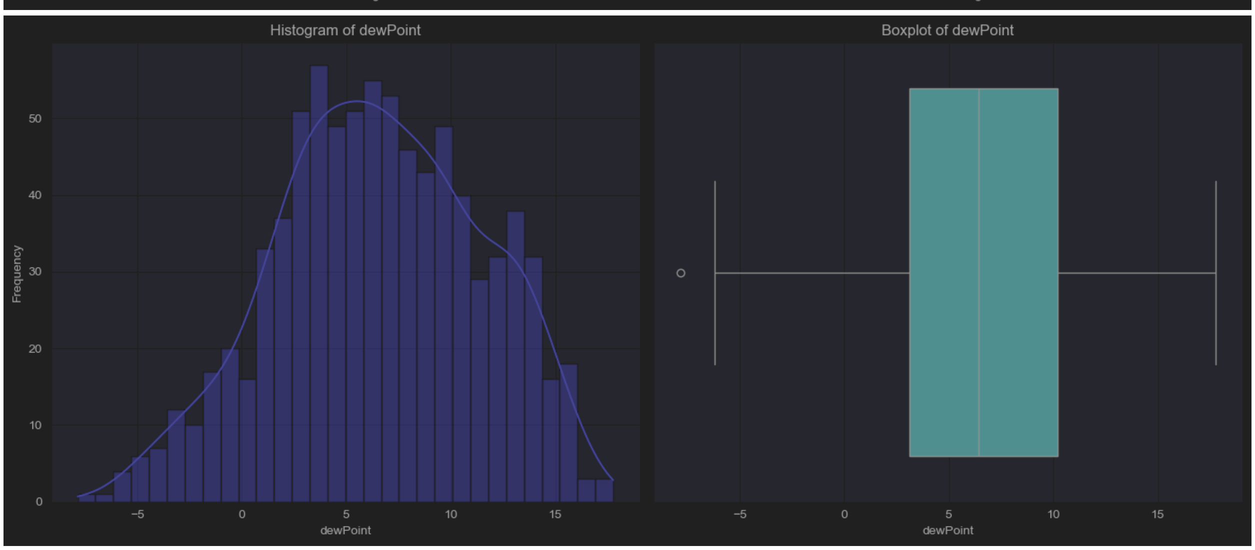
Part 4

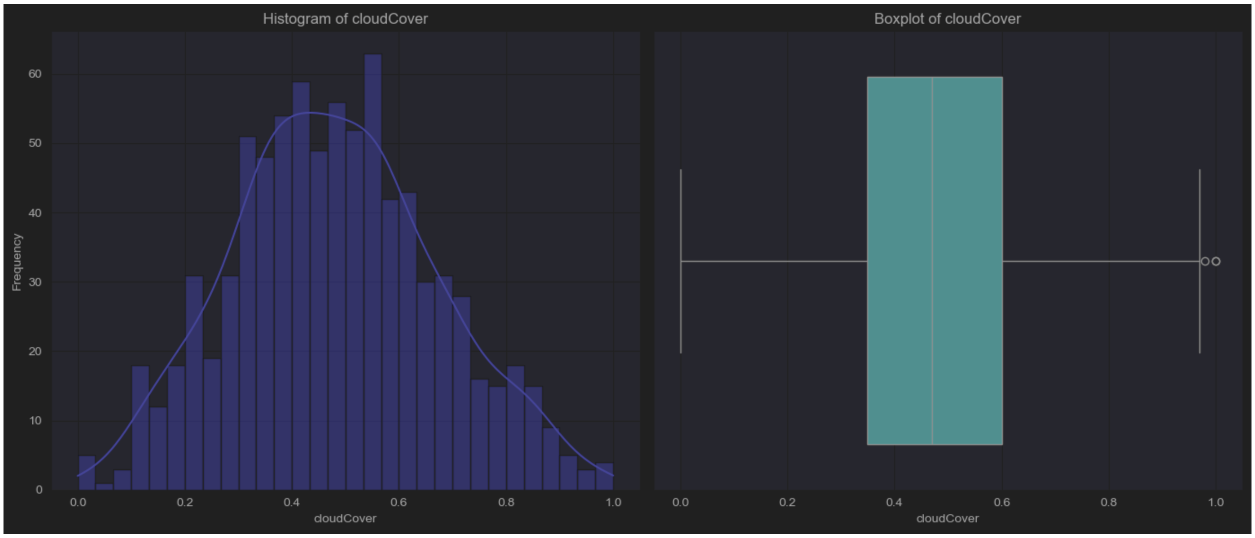
18. Data Distribution

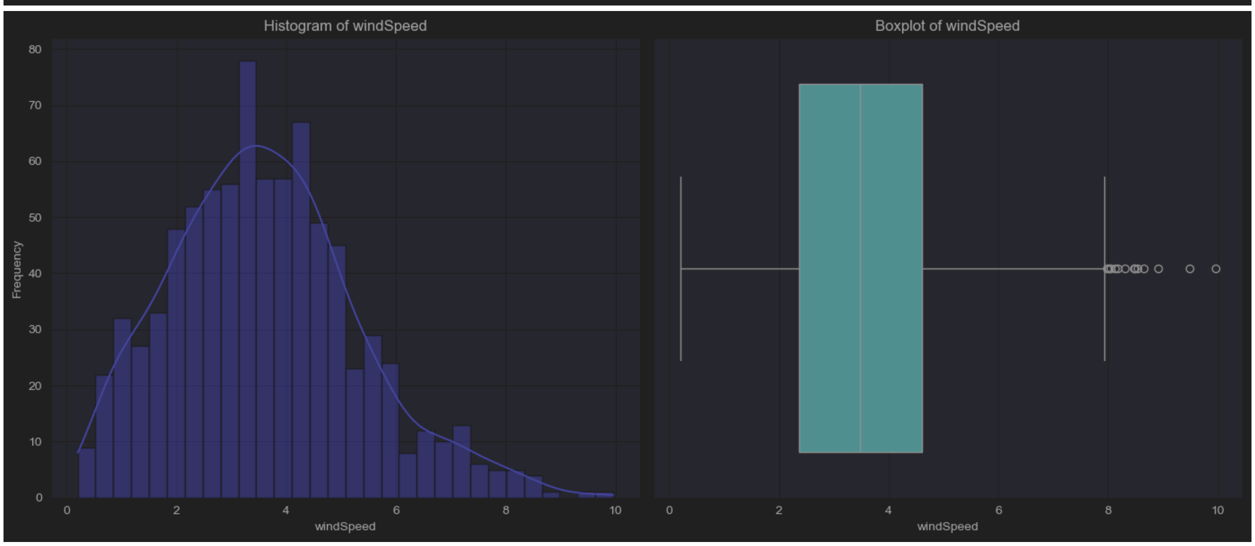
We are going to visualize the data before scaling or encoding. The day will not be visualized since it is like an index to the data only.

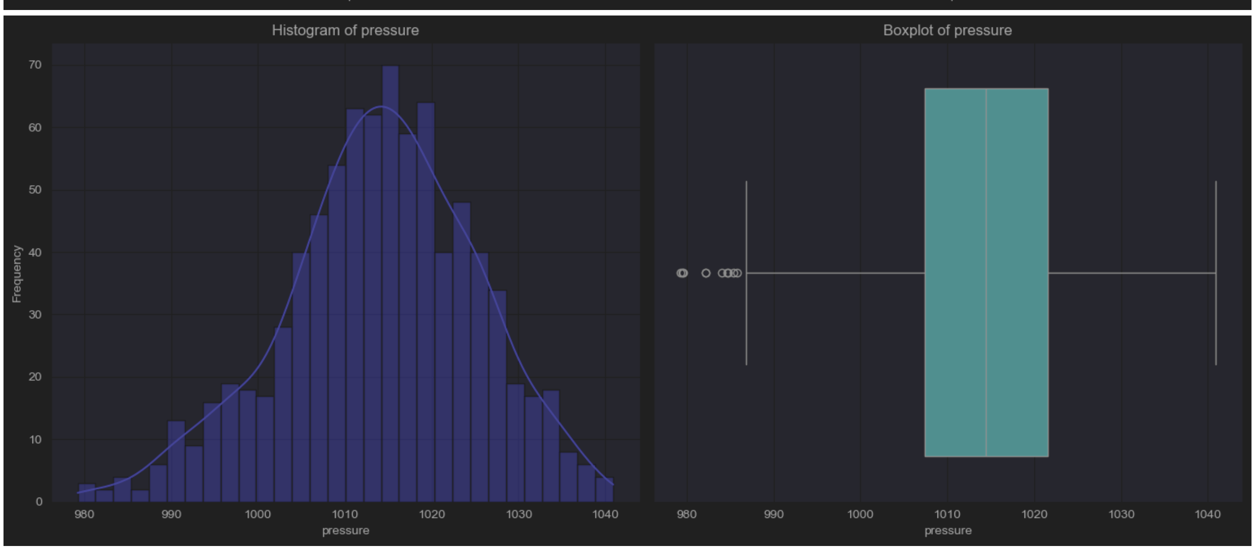

Histogram of temperatureMax
Frequency
temperatureMax
Boxplot of temperatureMax
temperatureMax

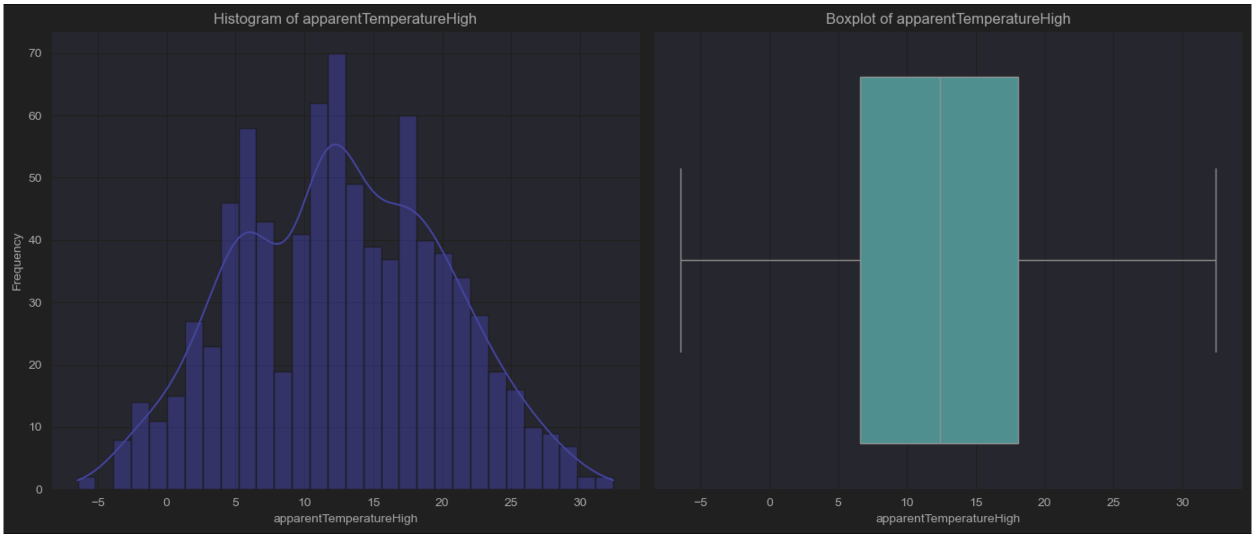

Histogram of windBearing
Frequency
windBearing
Boxplot of windBearing
windBearing


Histogram of dewPoint
Frequency
dewPoint
Boxplot of dewPoint
dewPoint

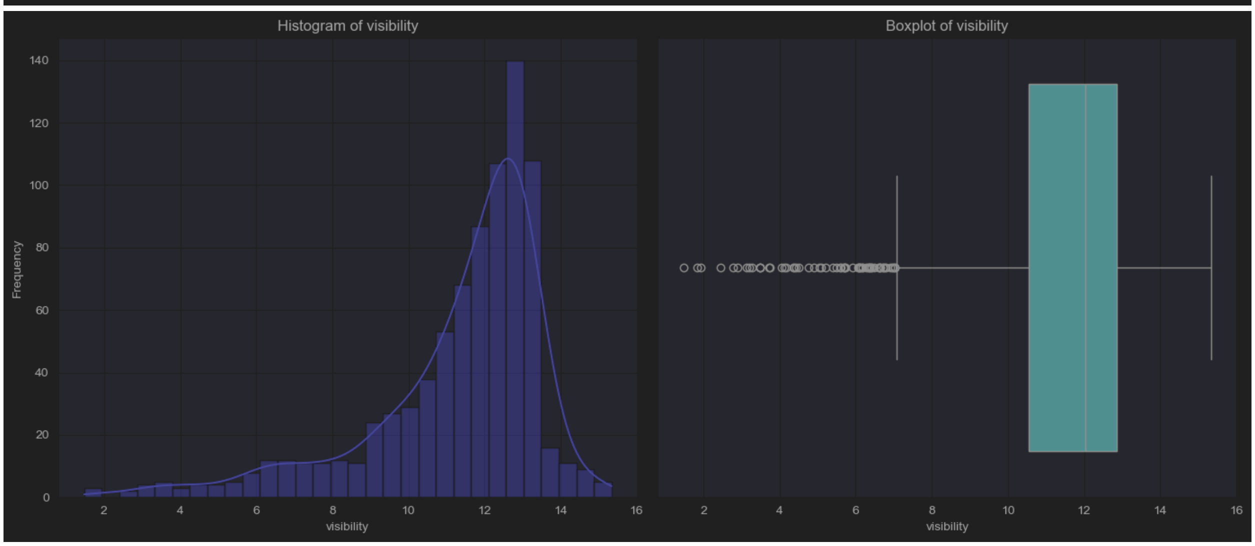

Histogram of cloudCover
Boxplot of cloudCover
Frequency
0
10
20
30
40
50
60
0.0
0.2
0.4
0.6
0.8
1.0
cloudCover


Histogram of windSpeed
Boxplot of windSpeed
Frequency
0
10
20
30
40
50
60
70
80
0
2
4
6
8
10
windSpeed

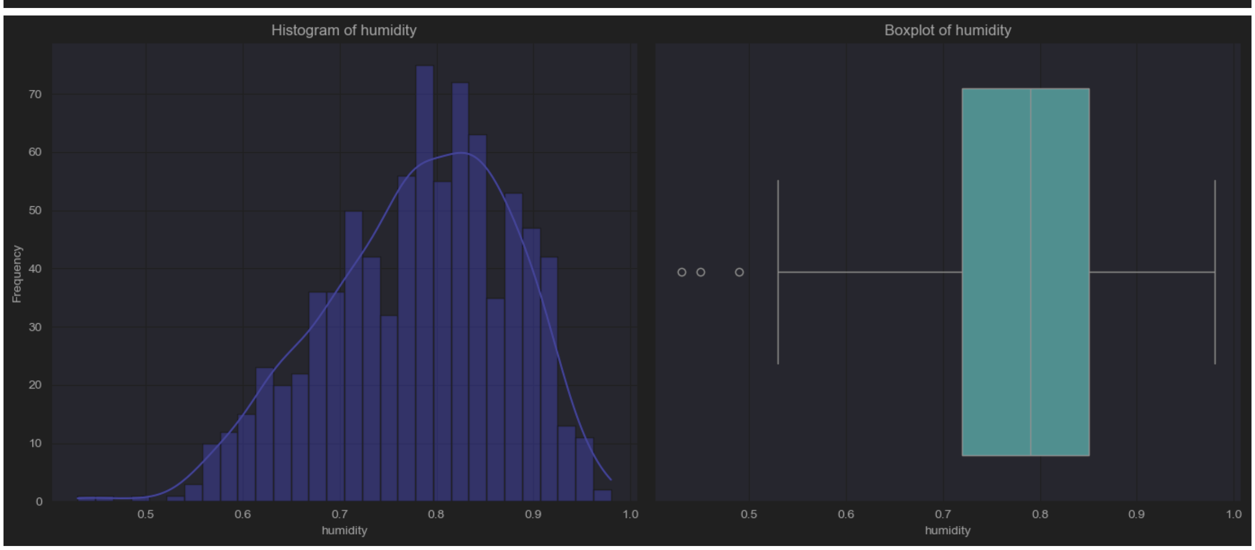

Histogram of pressure
Boxplot of pressure
Frequency
0
10
20
30
40
50
60
70
980
990
1000
1010
1020
1030
1040
pressure



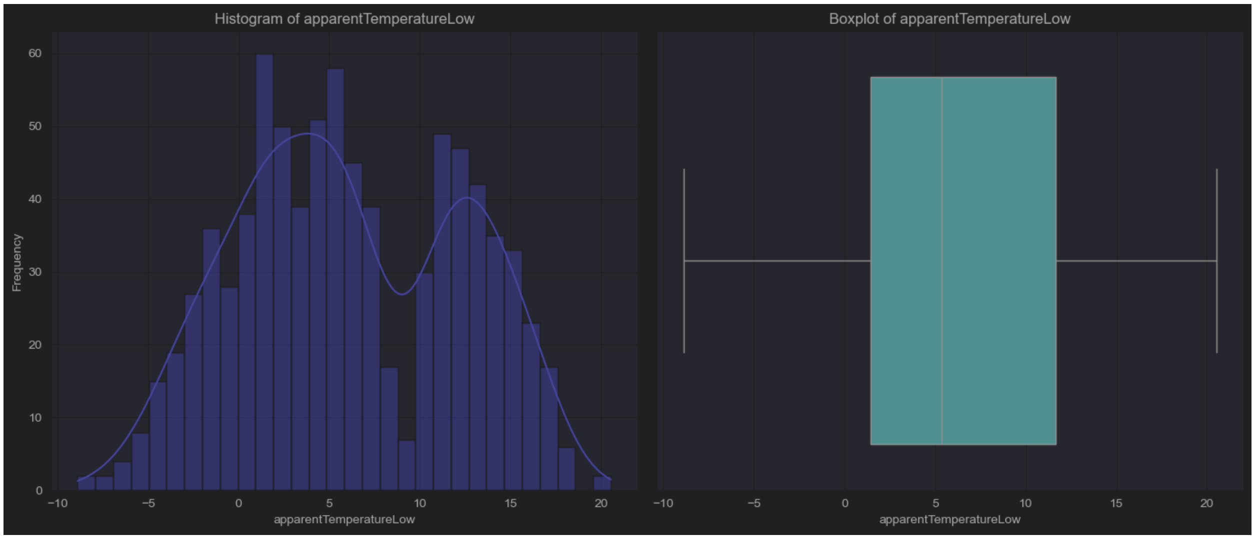
Histogram of apparentTemperatureHigh
Frequency
0
10
20
30
40
50
60
70
−5
0
5
10
15
20
25
30
apparentTemperatureHigh
Boxplot of apparentTemperatureHigh
−5
0
5
10
15
20
25
30
apparentTemperatureHigh

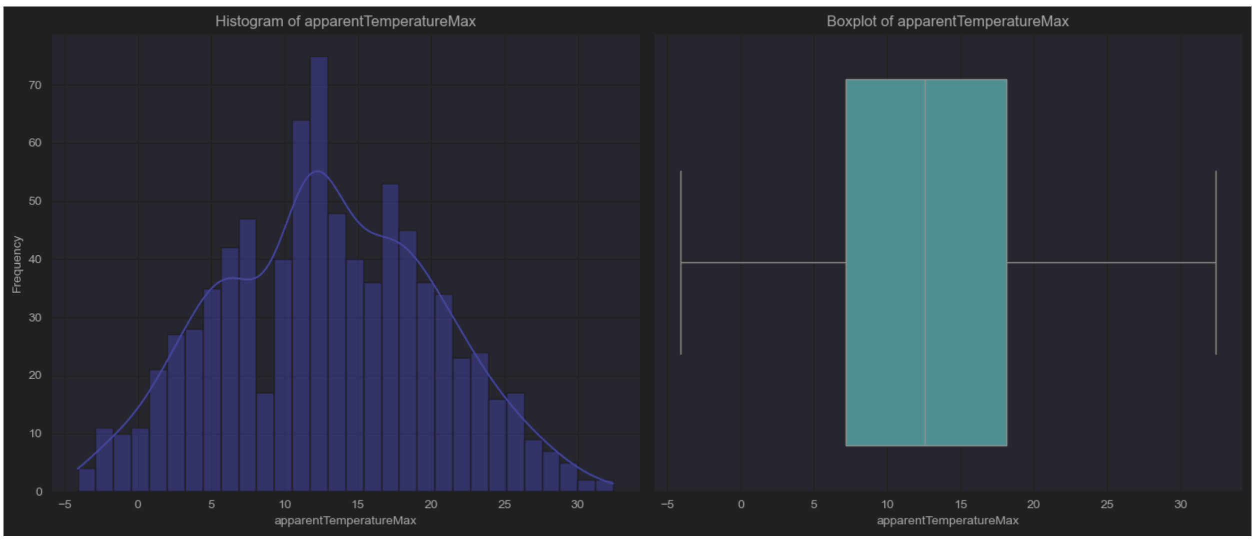


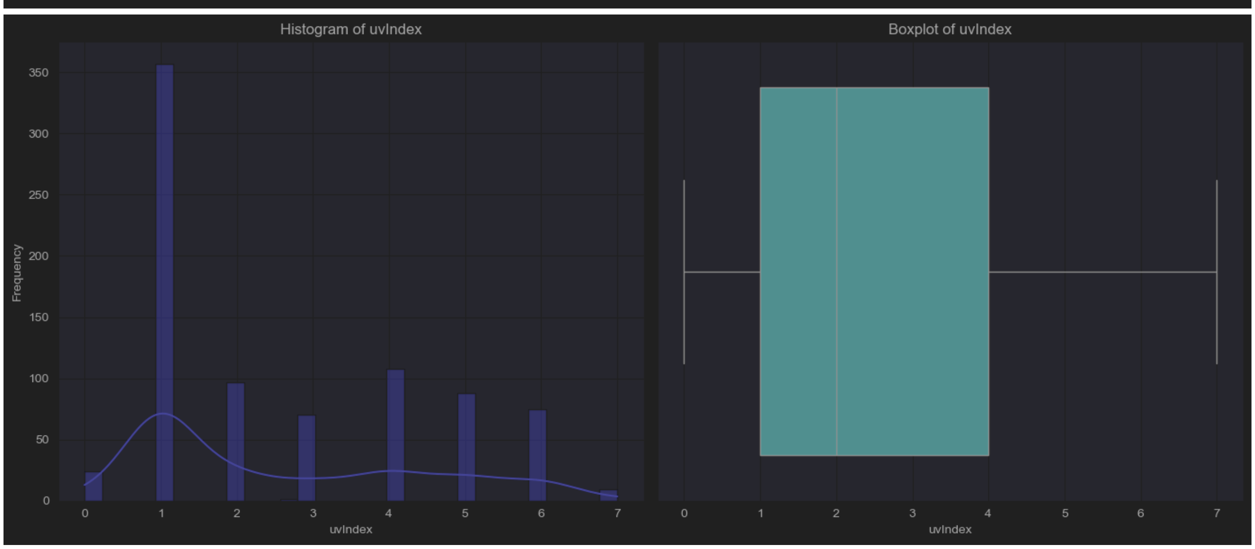
Histogram of visibility
Frequency
0
20
40
60
80
100
120
140
2
4
6
8
10
12
14
16
visibility
Boxplot of visibility
2
4
6
8
10
12
14
16
visibility

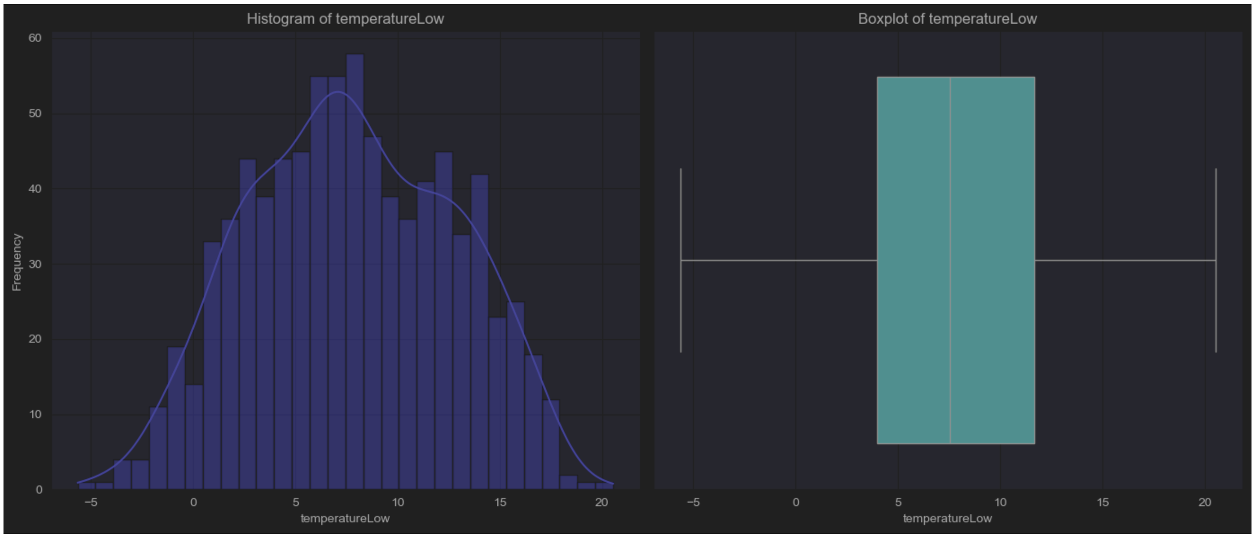


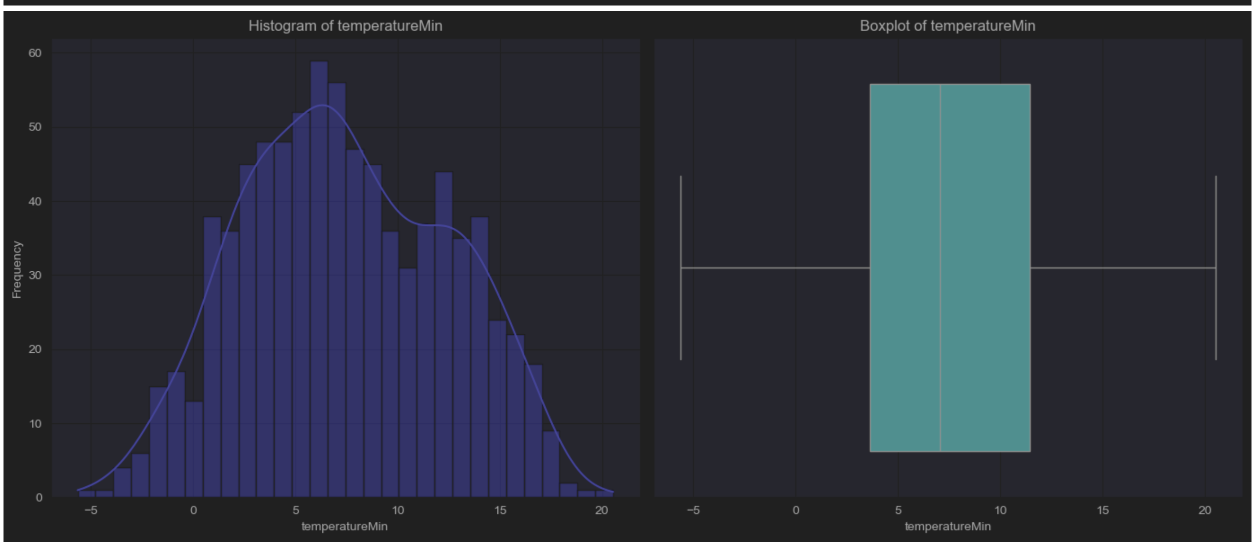
Histogram of humidity
Frequency
0
10
20
30
40
50
60
70
0.5
0.6
0.7
0.8
0.9
1.0
humidity
Boxplot of humidity
0.5
0.6
0.7
0.8
0.9
1.0
humidity

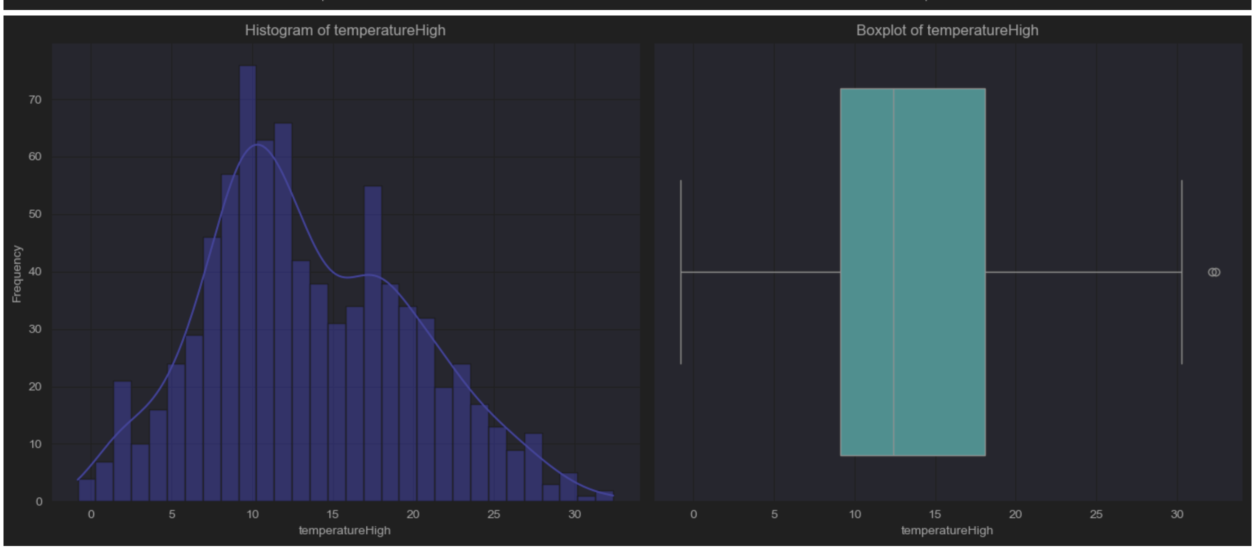

Histogram of apparentTemperatureLow
Frequency
apparentTemperatureLow
Boxplot of apparentTemperatureLow
apparentTemperatureLow

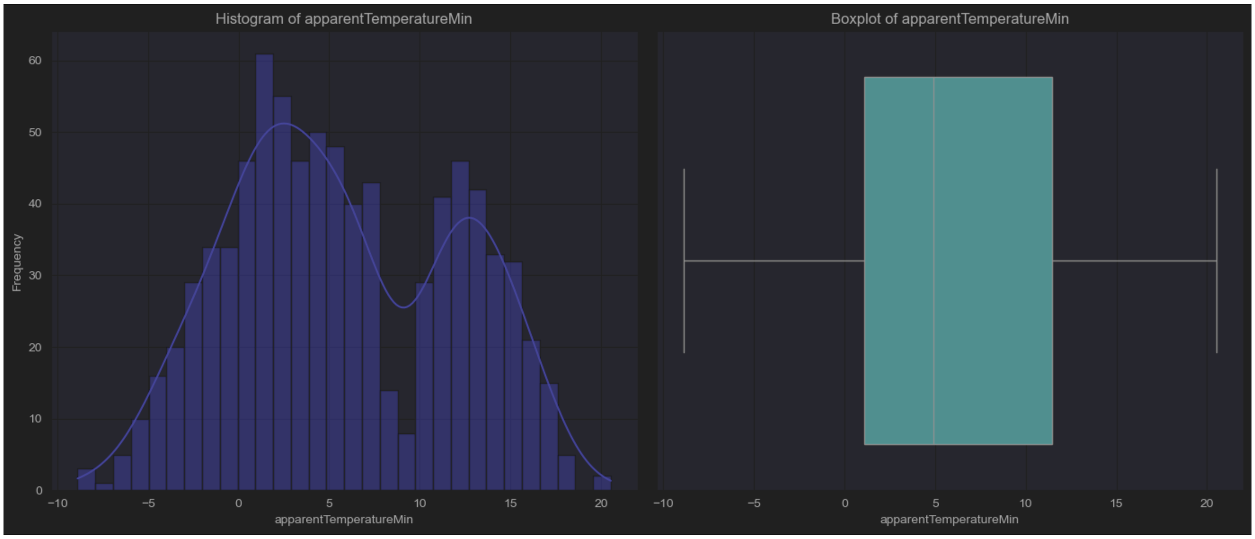

Histogram of apparentTemperatureMax
Frequency
apparentTemperatureMax
Boxplot of apparentTemperatureMax
apparentTemperatureMax

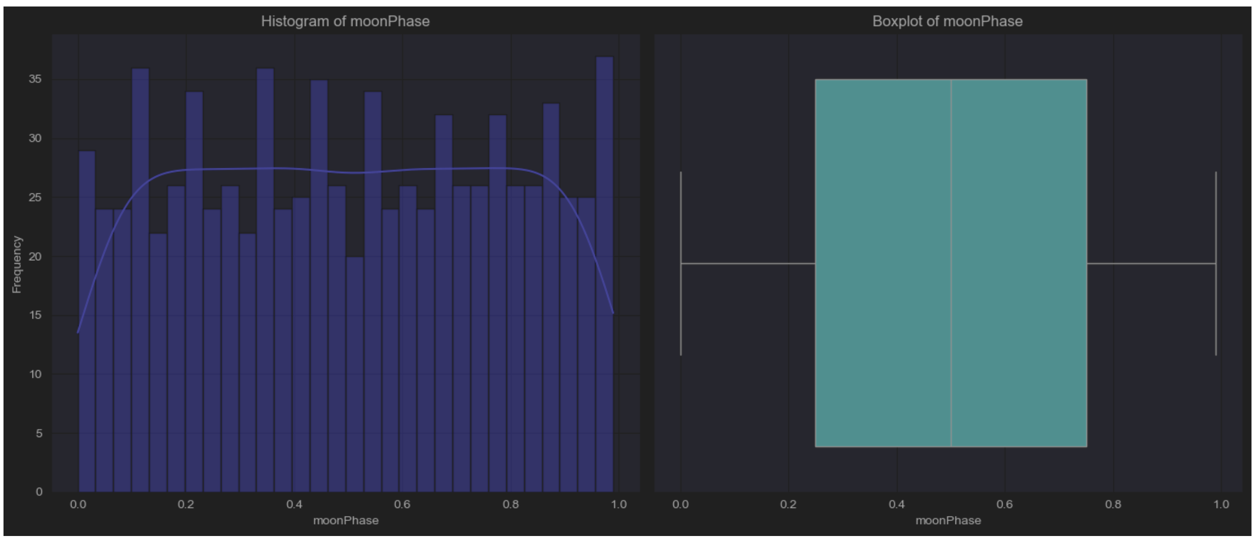

Histogram of uvIndex
Frequency
uvIndex
Boxplot of uvIndex
uvIndex

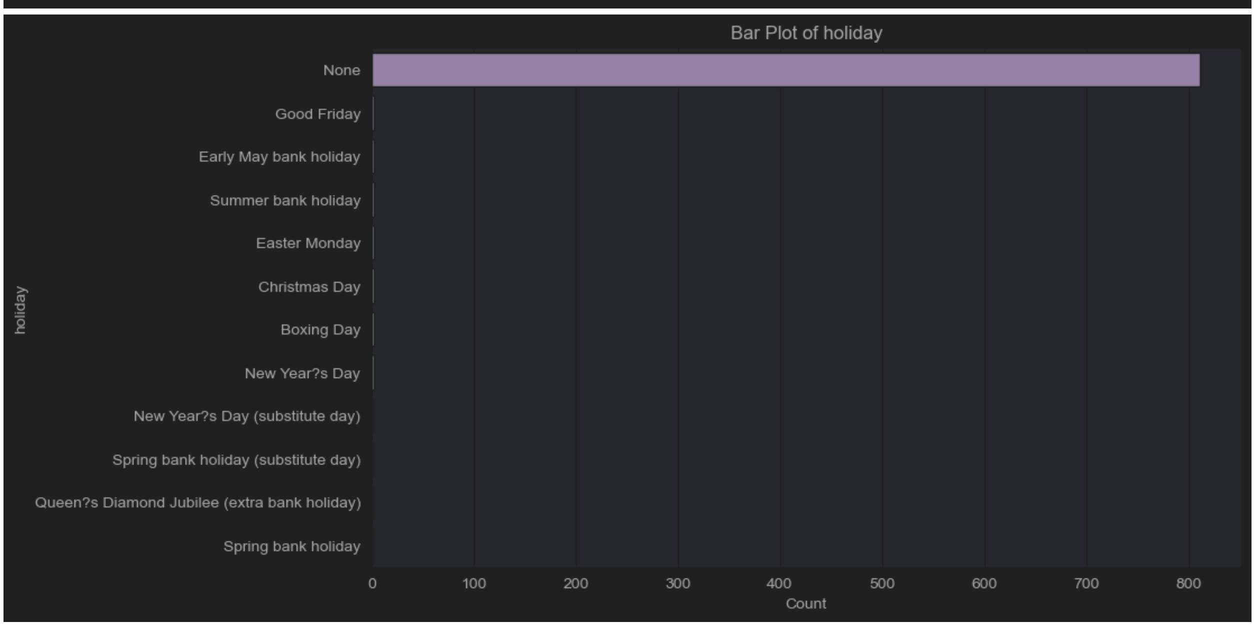

Histogram of temperatureLow
Frequency
temperatureLow
Boxplot of temperatureLow
temperatureLow
0
10
20
30
40
50
60
−5
0
5
10
15
20


Histogram of temperatureMin
Frequency
temperatureMin
Boxplot of temperatureMin
temperatureMin
0
10
20
30
40
50
60
−5
0
5
10
15
20


Histogram of temperatureHigh
Frequency
temperatureHigh
Boxplot of temperatureHigh
temperatureHigh
0
10
20
30
40
50
60
70
0
5
10
15
20
25
30

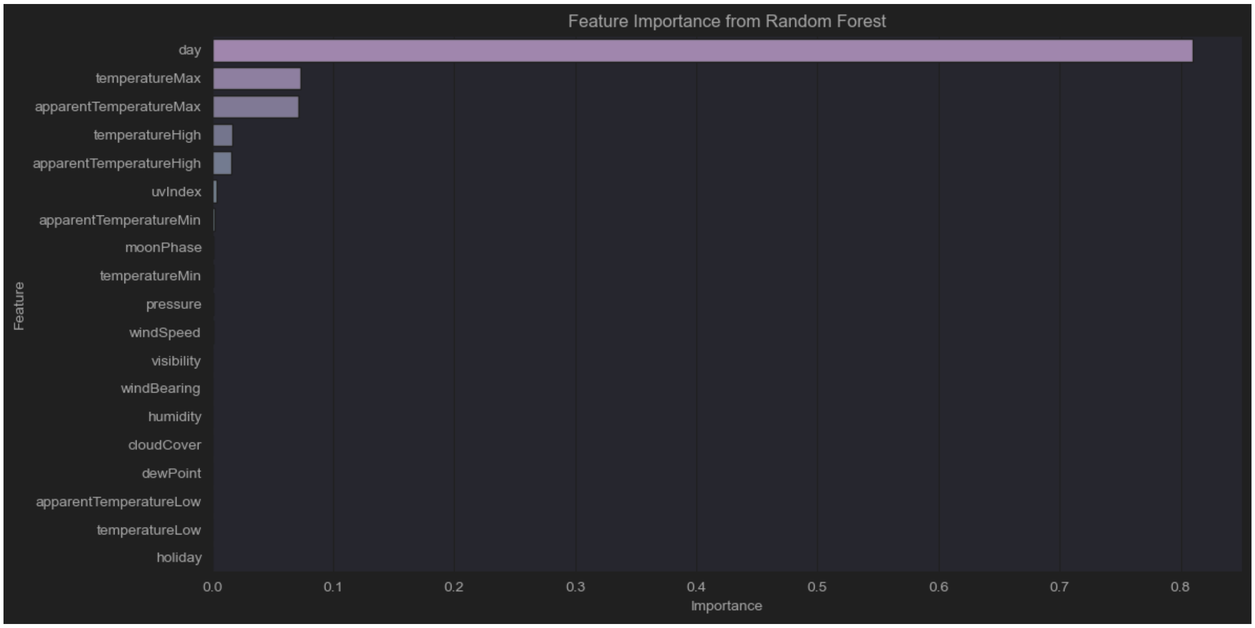

Histogram of apparentTemperatureMin
Frequency
0
10
20
30
40
50
60
−10
−5
0
5
10
15
20
apparentTemperatureMin
Boxplot of apparentTemperatureMin
−10
−5
0
5
10
15
20
apparentTemperatureMin


Histogram of moonPhase
Frequency
0
5
10
15
20
25
30
35
0.0
0.2
0.4
0.6
0.8
1.0
moonPhase
Boxplot of moonPhase
0.0
0.2
0.4
0.6
0.8
1.0
moonPhase


Bar Plot of holiday
holiday
None
Good Friday
Early May bank holiday
Summer bank holiday
Easter Monday
Christmas Day
Boxing Day
New Year?s Day
New Year?s Day (substitute day)
Spring bank holiday (substitute day)
Queen?s Diamond Jubilee (extra bank holiday)
Spring bank holiday
0
100
200
300
400
500
600
700
800
Count

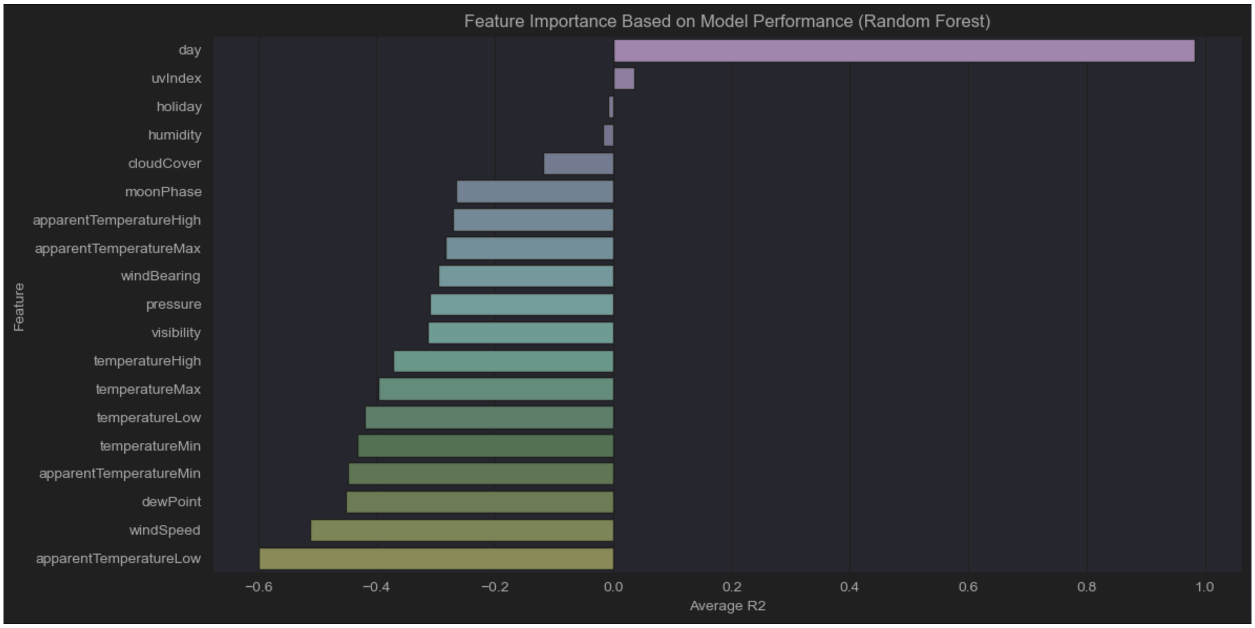
19. Feature Importance

Feature importance refers to how valuable each feature is in predicting the target variable. Tree-based models (like the Random Forest Regressor) can easily provide feature importance by evaluating the reduction in impurity (entropy) or variance due to each feature during training. The following plot show each feature importance in the model:



20. Model Performance Across Features

In this section, we evaluated the contribution of each feature individually to the overall performance of the model. The following figure shows the contribution of each feature in percentage with negative, if it decreases the performance, or positive if it increases it.



Feature Importance Based on Model Performance (Random Forest)
day
uvIndex
holiday
humidity
cloudCover
moonPhase
apparentTemperatureHigh
apparentTemperatureMax
windBearing
pressure
visibility
temperatureHigh
temperatureMax
temperatureLow
temperatureMin
apparentTemperatureMin
dewPoint
windSpeed
apparentTemperatureLow
Feature
−0.6
−0.4
−0.2
0.0
0.2
0.4
0.6
0.8
1.0
Average R2