Data Analysis and Machine Learning Assignment

Name: Mohammed AL shuaili

ID: 20106181

Date: 26/11/2024

1. Question 1:

1.1) Electricity and Gas usage 2023/2024: These datasets record daily gas and electricity consumption for the year 2023/2024, with usage measured at 15-minute intervals throughout each day and structured as follows:

Rows: 365 entries, one for each day of the year.
Columns (98 columns):

- Date: Represents the day (Interval attribute as the difference between dates has meaning and we can add days to a date and dates have meaningful distances).
- Values: Likely represents the total number of readings ((integer) Ratio because it has a meaningful zero and supports operations like addition, subtraction, multiplication, and division).
- Time Interval Columns (e.g., 00:00, 00:15, ...): Represent electricity usage in kilowatts (or similar units) at specific 15-minute intervals for 24 hours (Ratio).

The datasets have Some missing values at some intervals and date. This may indicate issues like incomplete measurements or errors during data collection.

1.2) If the dataset is recorded for urban households, we can apply analysis to understand how urban households can reduce energy and gas waste by identifying inefficiencies in daily consumption patterns and correcting abnormal usage behaviours. By analysing electricity and gas consumption at 60-minute intervals, we can identify periods of excessive usage that may indicate inefficiencies, such as:

- Excessive consumption during off-peak hours: Identifying when gas or electricity is being used unnecessarily during periods of low activity (e.g., late-night hours when households are inactive).
- Identifying equipment inefficiencies: Spotting unusual consumption patterns that could suggest faulty or inefficient appliances (e.g., malfunctioning heating systems or lights left on).
- Behavioural analysis: Understanding how human behaviours, such as leaving lights or heating systems running unnecessarily, contribute to higher and unnecessary energy consumption.

Correcting these inefficiencies can lead to significant energy savings, reduced overall consumption, and the analysis can also provide actionable insights and recommendations for households to optimize their energy usage, and reduce unnecessary costs.

1.3) This can be broken into multiple steps:

1) Data Cleansing:

Data cleansing ensures that the data is accurate, complete, and consistent before analysis. The steps involved in this process are:

a) Handling Missing Values:

- Identify Missing Data: Check for NaN, None, or empty values in the dataset.
  - Example: in the Electricity dataset 2023, to find the missing data the following code is used

```
missing_values = Edf23.isnull().sum()
# Filter columns with missing values only
columns_with_missing = missing_values[missing_values > 0]
columns_with_missing.sort_values(ascending=False)
```

This shows that all columns include missing data. Then to get an overall idea about these columns we can run.

```
missing_columns = Edf23.columns[Edf23.isnull().sum() > 0]
# Describe only those columns with missing data
missing_data_description =
Edf23[missing_columns].describe(include='all')
missing_data_description
```

Each column can then be analysed, and missing values can be imputed by identifying the mode for that specific time interval. This approach is most appropriate since these values represent units of electricity and replacing them with the mode (being the most frequently occurring unit) provides a practical solution.

The same can be applied to Gas usage 2023. However, Looking at Electricity/Gas Usage 2024, we can see that the datasets are incomplete. This is expected, as the datasets were recorded before the end of 2024. However, we must replace these values or remove them to prevent them from causing errors in the data summary and visualizations. In this case the best way is to remove these values as we can gain a rough idea of the dataset using the recorded values and just predicting these values won't give us much information.

b) Identifying and Handling Outliers:
To identify outliers we can use boxplot, Jitter plot or histograms, but we must first reshape the dataset to prepare it for visualisation.

- Handling Outliers:
    - We can either remove them, cap them at a specific value, or transform the data.

2) Data Reshaping:

Data reshaping is the process of transforming the data into a format that suits the analysis or model you're working with. So, for the above issue the best suitable way is to merge gas with electricity. This is because the goal is to reduce overall energy consumption and combining both energy types into a single dataset allows for a more comprehensive view of the household's total energy usage.

a) Chage columns to represent hourly instead of every 15 minutes. Considering the below, since the values are similar across each hour, it makes sense to only consider hourly readings. (the standard deviation is small)

	00:00	00:15	00:30	00:45	01:00
count	365.000000	365.000000	365.000000	365.000000	365.000000
mean	30.276712	30.284932	30.241096	30.197260	30.191781
std	3.159258	3.195278	3.192016	3.160452	3.491188
min	2.000000	2.000000	3.000000	3.000000	3.000000
25%	29.000000	29.000000	29.000000	29.000000	29.000000
50%	31.000000	31.000000	30.000000	31.000000	31.000000
75%	32.000000	32.000000	32.000000	32.000000	32.000000
max	38.000000	35.000000	36.000000	35.000000	35.000000

b) We apply the same process to the Gas dataset to ensure consistency, and then merge the gas data with the electricity data for analysis. Using this merged dataset, we can compare gas and electricity usage at any given time throughout 2023. Additionally, we can calculate the total consumption at each time point, providing an estimate of the overall units consumed, whether from gas or electricity

1.4) The head of the dataset

Date	02:00_Gas	05:00_Gas	08:00_Gas	11:00_Gas	14:00_Gas	17:00_Gas	20:00_Gas	23:00_Gas	02:00_Electricity	0
1/1/2023	22	0	0	0	0	0	0	0	27	
2/1/2023	0	0	22	22	11	11	22	0	28	
3/1/2023	0	0	187	253	121	88	0	0	28	
4/1/2023	0	0	110	187	154	110	0	0	26	
5/1/2023	0	0	110	198	132	121	0	0	25	
6/1/2023	0	0	187	33	33	121	0	0	30	
7/1/2023	0	0	0	0	0	0	0	22	27	
8/1/2023	0	0	0	0	22	0	0	0	26	
9/1/2023	0	0	143	242	198	110	0	0	30	
10/1/2023	0	0	209	132	110	88	0	0	29	

General summary statistics with description (hourly):

a) Gas usage 2024:

- The provided dataset summarizes gas usage in 2024, measured at specific time intervals throughout the day (e.g., 00:00, 00:15, etc.). The statistics include key metrics such as the number of observations (`count`), mean (`mean`), standard deviation (`std`), and percentiles (`min`, `25%`, `50%`, `75%`, `max`) for each time interval.

- Each interval has a consistent number of observations, with a `count` of **236** across all time periods (after cleaning). The mean gas usage varies significantly throughout the day, starting at 10.30 units at midnight (00:00), peaking at 101.84 (102) units at 07:30, and tapering off to lower values like 9.18 (9) units by 22:30. This trend suggests higher consumption during the morning hours, likely coinciding with household activities, and reduced usage during nighttime.

- The variability in gas usage, represented by the standard deviation (`std`), also fluctuates. For instance, at 00:00, the standard deviation is 14 units, increasing significantly to 86 units at 08:45, indicating more inconsistent consumption patterns during peak hours. The minimum usage (`min`) across all intervals is consistently 0 units**,** highlighting periods of no consumption.

- The dataset's percentiles provide further insights. For example, the 25th percentile (`25%`) remains 0 units for many intervals, showing that at least a quarter of the readings indicate no gas usage. However, during active periods like 07:30, the 75th percentile (`75%`) rises to 154 units, and the maximum (`max`) reaches 550 units at 10:00, showing periods of exceptionally high consumption.

- This dataset captures both inactive and peak usage periods, making it valuable for identifying inefficiencies in gas consumption. However, the irregular intervals (e.g., 01:15, 02:30) may require standardisation for more comprehensive time-based analysis.

b) Electricity usage 2024:

- Standard deviation (`std`) shows variability in electricity usage. During the peak hours of 12:00 PM to 2:00 PM, variability is highest (e.g., 16.55 units at 12:00 PM), but it significantly decreases by late evening, reaching 2.87 units at 11:00 PM.

- Minimum (`min`) and Maximum (`max`) values show a wide range of usage, from as low as 8 units at noon to as high as 81 units at 11:00 AM.

- The 25% percentile consumption is steady at 36 units from 12:00 PM to 2:00 PM, dropping to 33 units by late evening (11:00 PM).
- The 50th percentile (median) peaks at 64 units at 12:00 PM, gradually declining through the afternoon and evening to 34 units by 11:00 PM.
- The 75th percentile, indicative of higher consumption, mirrors this pattern, peaking at 69 units from 12:00 PM to 2:00 PM and dropping to 36 units by 11:00 PM.

C) Electricity usage 2023:

- This dataset highlights consistent patterns in mean consumption, with a gradual increase from 30.28 (30) units at midnight to a peak of 58.40 (58) units by 12:00 PM. A slight dip follows, with the mean at 58.22 (58) units by 2:00 PM.

Key Observations:

- Standard deviation (std) reflects higher variability in late mornings and afternoons, peaking at 20.63 units at noon.

- Minimum (min) values range from 2 units (e.g., 12:00 PM) to 11 units (9:00 AM).

- Median (50%) values steadily climb, reaching 68 units at noon, indicating the central tendency aligns with peak usage times.

- Maximum (max) usage occurs at 12:00 PM (86 units), showcasing the busiest period.

This data indicates a similar daily cycle to 2024, with low usage overnight and peaks during late mornings and midday

D) Gas usage 2023:

- This data illustrates a marked rise in consumption during morning hours and a peak in late mornings, followed by a decline towards the afternoon and evening.

Key Observations:

- Mean Consumption:
- Low overnight: Starts at 2.50 (3) units (00:00), increasing to 5.39 (5) units (06:00).
- Peak usage: 71.03 units (08:00) and 86.28 units (09:00), with a gradual decline to 54.82 units (14:00) and 23.7 at (17:00).
- Standard Deviation (std): Significant variability observed during the peak hours (08:00–09:00, 68.17–93.81 units).
- Minimum (min) values: Gas usage is 0 units throughout all time intervals for at least some data points.
- Median (50%) values: Peaks at 55 units (08:00), then declines gradually.
- Maximum (max) values: Highest usage at 330 units (08:00), tapering off to 275 units by 14:00.

This dataset suggests high morning demand, likely reflecting heating or cooking activities, with usage tapering off as the day progresses.

Now, let's examine each dataset.

Plotting the Electricity mean values (200 samples, first and last) (hourly)

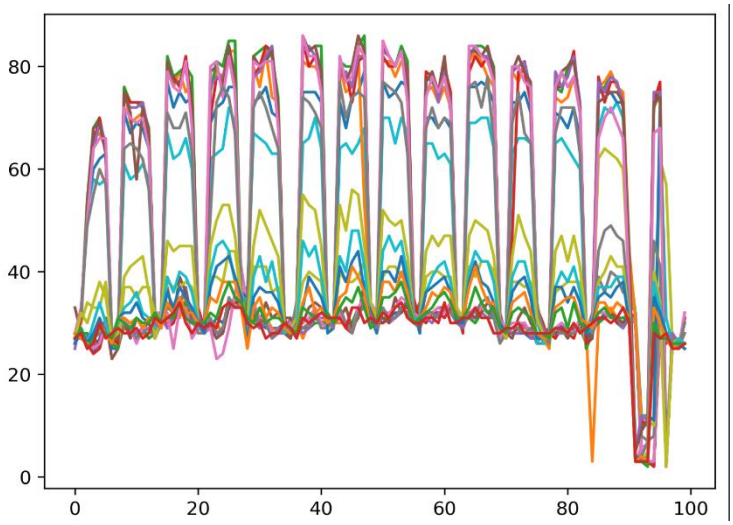

Figure 1 Electricity usage 2023 first 100 samples

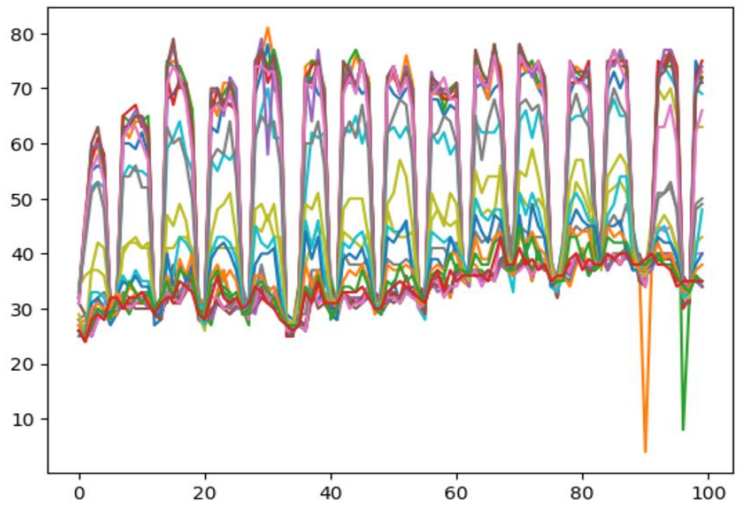

Figure 2 Electricity usage 2024 first 100 samples

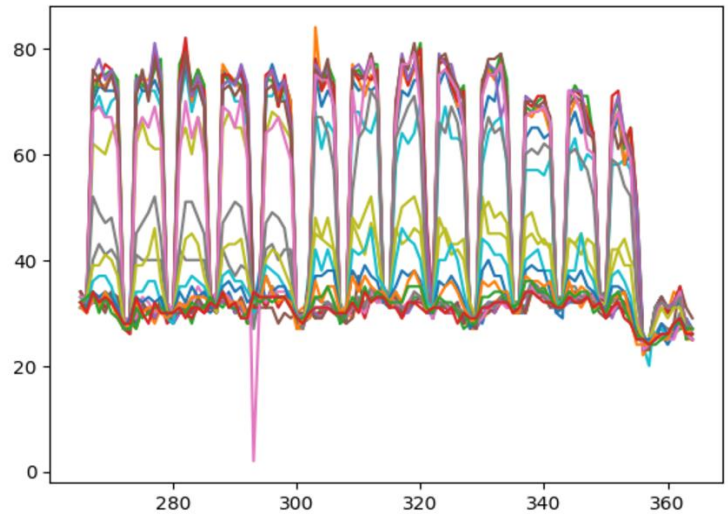

Figure 3 Electricity usage 2023 last 100 samples

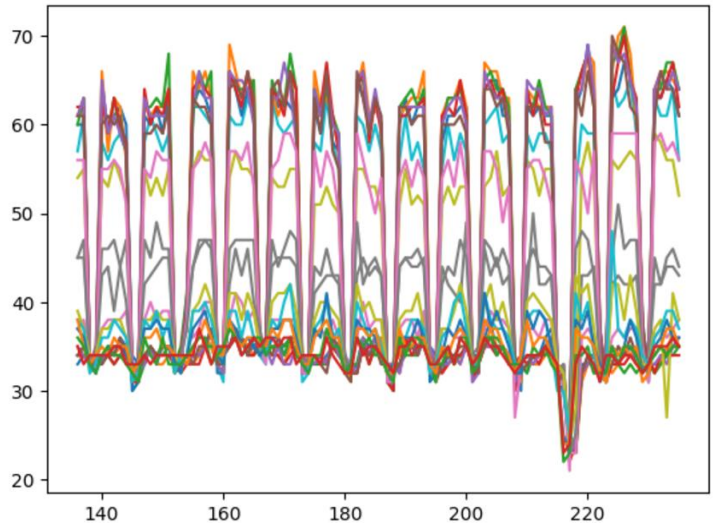

Figure 4 Electricity usage 2024 last 100 samples

Analysing the mean values for electricity consumption in 2023 and 2024, we observe that weekdays generally exhibit higher consumption compared to weekends. Additionally, several unexpected values are apparent, particularly in Figures 3, 2, and 1. These values could either be normal variations or potential outliers.

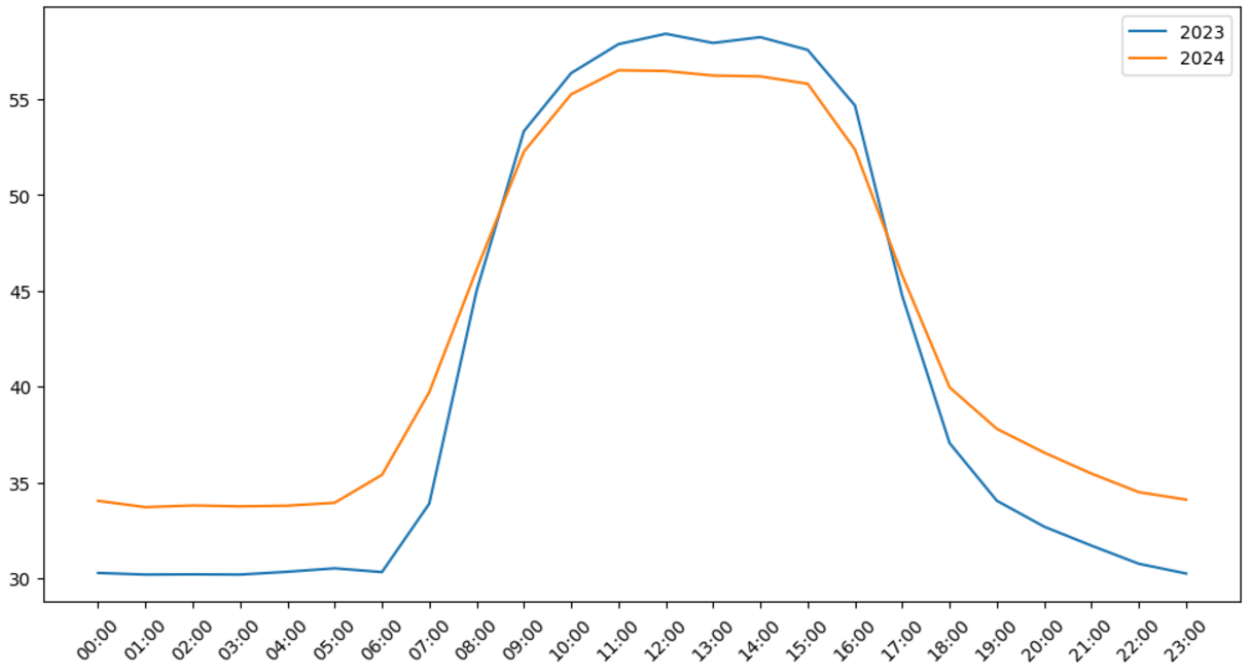

Figure 5 the mean values of electricity usage throughout a day

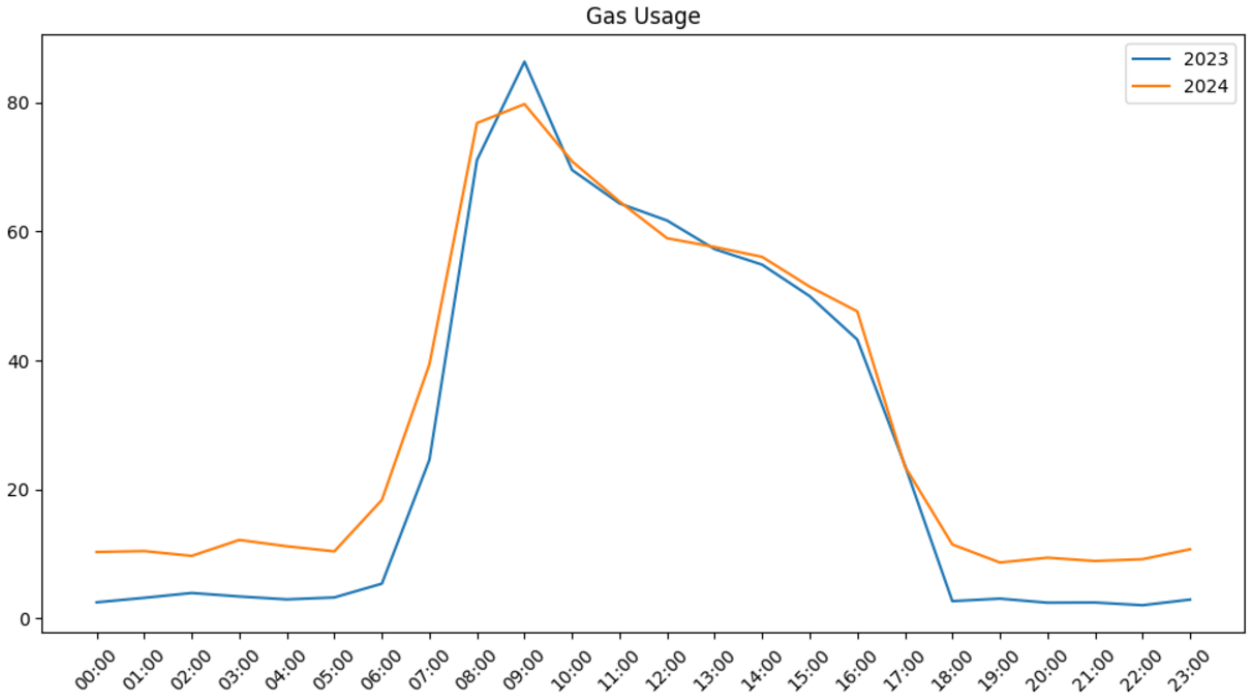

Figure 6 shows the mean values of Gas usage over the course of a day

Now let's examine the Density:

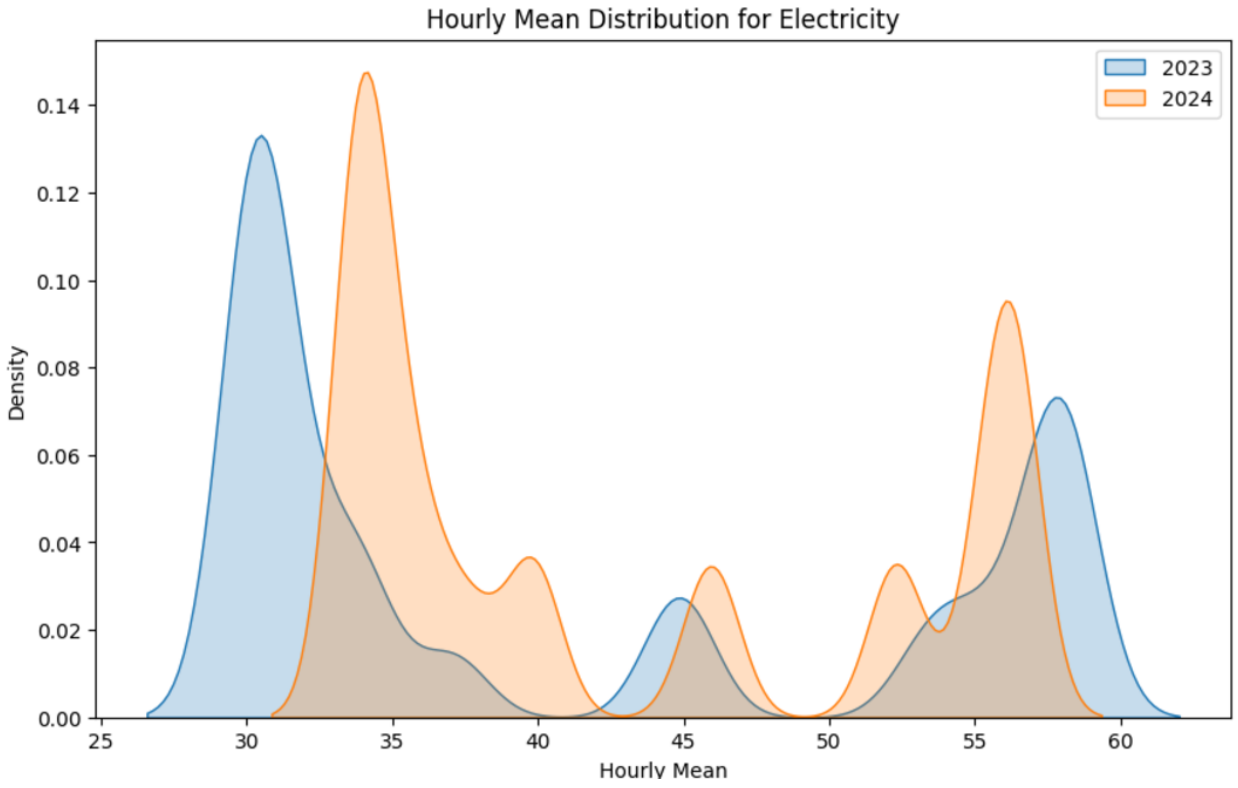

Figure 7 shows Hourly mean distribution for electricity

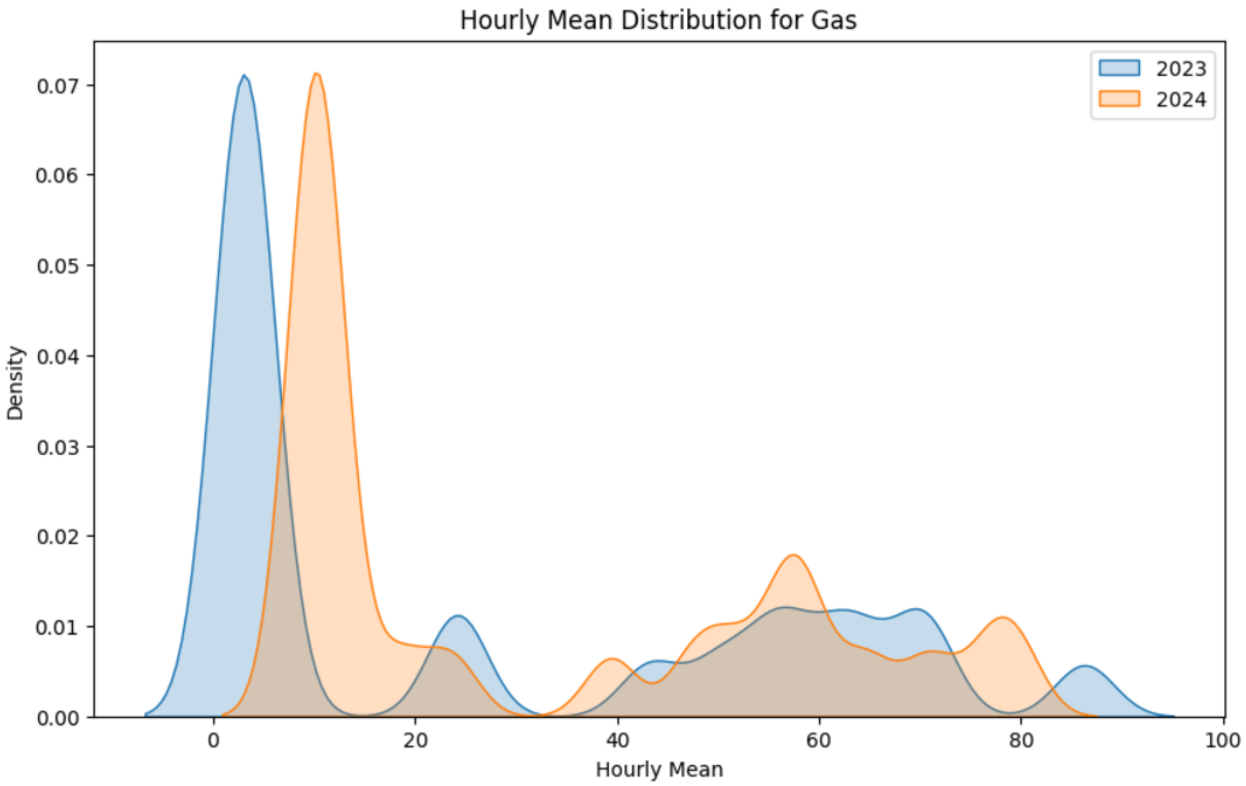

Figure 8 Shows hourly mean distribution for gas usage

Figure 7 and Figure 8 represent kernel density estimators for electricity and gas mean hourly values, respectively. From the figures, we can identify high density around 31 and 35 units of electricity in 2023 and 2024, reflecting dominant consumption patterns across these periods. For gas, notable peaks are observed near 10 and 18 units in 2023 and 2024, suggesting shifts in usage patterns. The differences in density curves indicate variations in distribution between the two years, highlighting potential changes in energy demand over time.

In order to create a boxplot to detect outliers and gain more information about the datasets, we can create a new data frame with the mean value of each as follows:

```
data_for_boxplot = pd.DataFrame({
    'Year': ['2023'] * len(hourly_means_2023_electricity) + ['2024'] *
len(hourly_means_2024_electricity),
    'Electricity': hourly_means_2023_electricity.tolist() +
hourly_means_2024_electricity.tolist(),
    'Gas': hourly_means_2023_gas.tolist() + hourly_means_2024_gas.tolist()
})
```

The new DataFrame, `data_for_boxplot`, is created to organize the data for effective visualization. The `Year` column is included to indicate whether the data corresponds to 2023 or 2024, allowing for clear differentiation between the two years. The `Electricity` and `Gas` columns store the

respective hourly mean values for each year, which are converted to lists using the `.tolist()` method and then concatenated to combine data from both years into a single structured format.

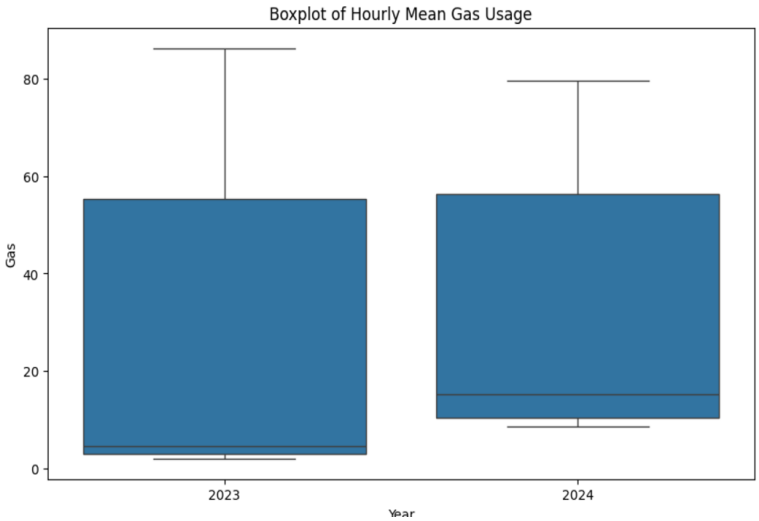

Figure 9 boxplots of hourly mean gas usage in 2023/24

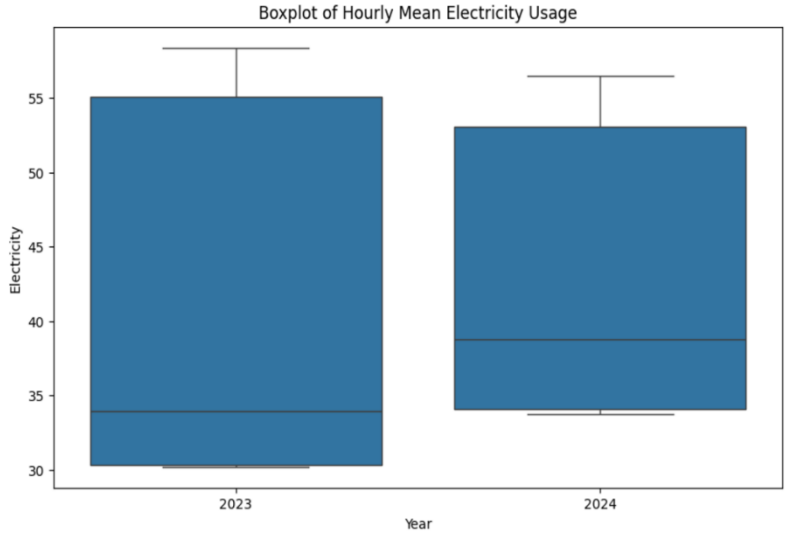

Figure 10 boxplots of hourly mean Electricity usage in 2023/24

Figures 9 and 10 present boxplots of hourly mean electricity and gas usage, respectively. From Figure 9, electricity usage in 2023 shows a slightly broader interquartile range compared to 2024, suggesting greater variability in consumption during this period. The median values are relatively stable between the two years, with a small shift observed toward slightly higher usage in 2024. In Figure 10, gas usage shows that the median value in 2024 is higher than in 2023, alongside a noticeable narrowing of the interquartile range, reflecting a more uniform distribution of hourly

gas consumption in 2024. These visualisations highlight potential improvements in energy efficiency or shifts in consumption behaviour over time.

1.5) To address the challenge of helping urban households reduce energy and gas waste a strategic use of visualisation techniques can be highly effective. Visualisations can translate complex data into understandable insights for the general audience, prompting actionable changes in consumption patterns.

A) Understand the Data Distribution using Fig 7 and 8:

- The kernel density plot for electricity usage reveals distinct patterns for 2023 and 2024. In both years, the highest density is observed around 31–35 units, indicating consistent demand in this range. However, in 2024, an additional peak is noticeable near 55 units, suggesting increased consumption during certain hours. This shift may be attributed to new appliances, behavioral changes, or increased energy usage during specific periods. The concentration of high-density values around certain ranges could indicate inefficiencies, such as simultaneous use of multiple high-energy appliances or unoptimized energy schedules.
- For gas usage, the density plot shows a more efficient and concentrated usage pattern for 2023, with values primarily below 10 units. In contrast, 2024 exhibits a broader distribution with higher peaks in the 10–20 range and irregular spikes up to 40–60 units. This suggests an increase in variability and possible inefficiencies in gas consumption, such as seasonal heating demand or system malfunctions. The spread in 2024 highlights the need to investigate abnormal patterns, optimize heating systems, or address potential leaks to reduce waste and improve efficiency.

B) Compare Energy Usage by Day

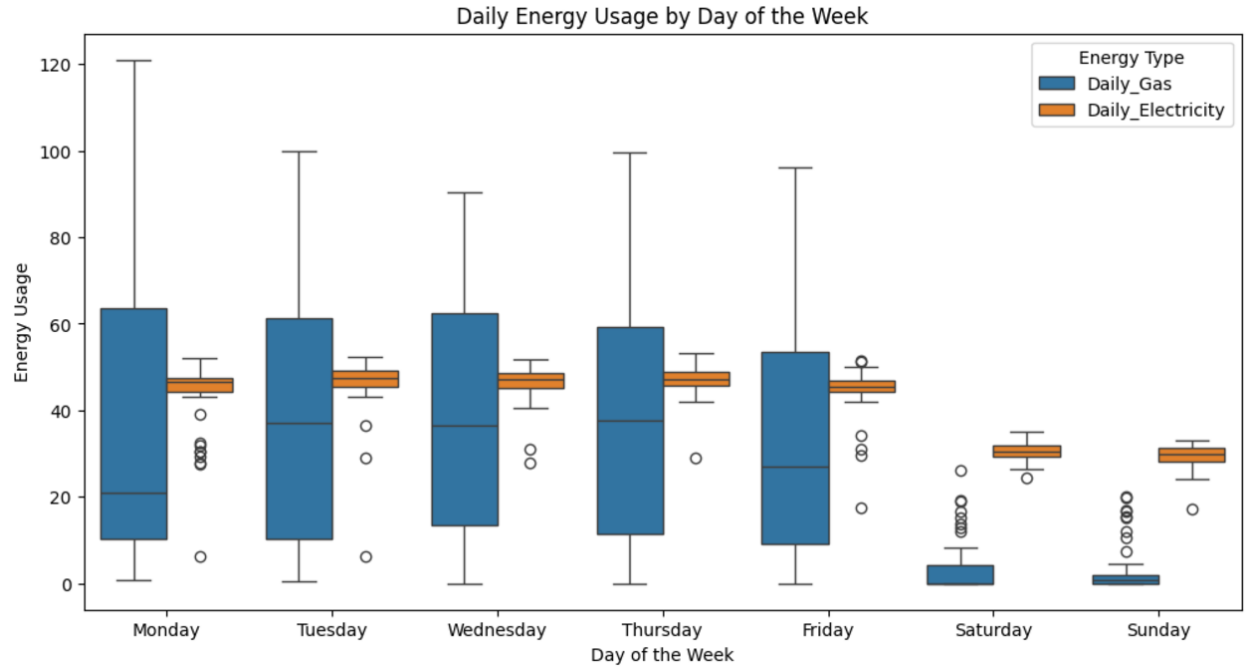

Figure 11Comparison of Daily Mean Gas and Electricity Usage Across the Week in 2023

This boxplot provides a detailed comparison of daily mean energy usage for gas (blue) and electricity (orange) across the days of the week (in 2023). Gas usage shows a clear pattern of higher consumption and variability on weekdays (Monday to Friday), with the median usage ranging from 20 to 40 units. The interquartile range (IQR) is also broader on weekdays, indicating a wider spread in consumption levels, while outliers suggest occasional spikes in usage. On weekends (Saturday and Sunday), gas usage drops significantly, with medians around 3 units, minimal variability, and tighter IQRs, reflecting reduced heating or cooking activities.

Electricity usage, on the other hand, remains consistent throughout the week, with median values around 30–50 units across all days. The IQR for electricity is much narrower than that for gas, indicating stable and predictable consumption patterns. A slight decrease in electricity usage at weekends suggests a shift in activities, possibly due to more time spent at home. This contrast between gas and electricity usage highlights opportunities to investigate and optimise weekday gas consumption while maintaining efficient and stable electricity use.

Figure12 illustrates the daily mean gas (blue) and electricity (orange) usage across the days of the week for 2024. Gas usage shows a consistent pattern of higher median values during weekdays (Monday to Friday), with medians ranging from 30 to 50 units and significant variability, as indicated by the wider interquartile ranges and presence of outliers. On weekends (Saturday and Sunday), gas usage drops sharply, with medians below 10 units and minimal variability. Electricity usage remains stable throughout the week, with median values consistently around 40–45 units, and a slight decrease on Weekends. The narrower interquartile ranges and fewer outliers for electricity indicate predictable and consistent consumption

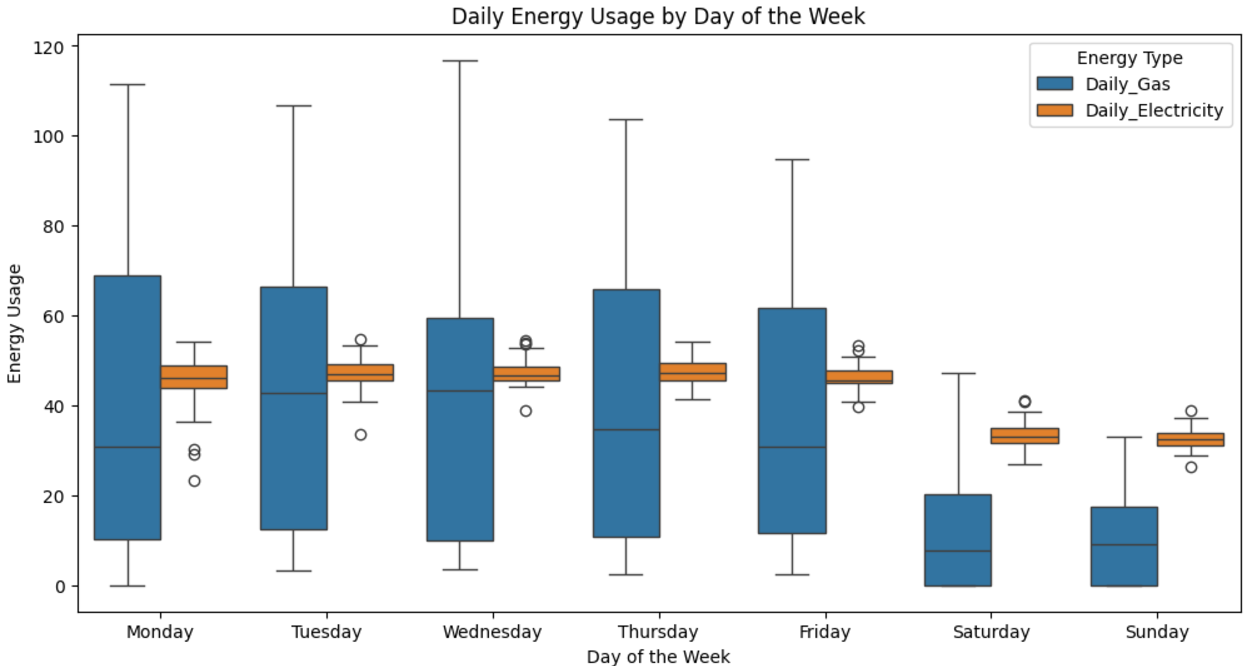

Figure 12 Comparison of Daily Mean Gas and Electricity Usage Across the Week in 2024

The comparison shows that gas usage in 2024 had slightly higher weekend medians and more variability than 2023, while electricity usage remained stable across both years. Since 2023 provides clear patterns for analysis, it is sufficient for identifying trends and inefficiencies.

C) Joint Relationships Between Gas and Electricity

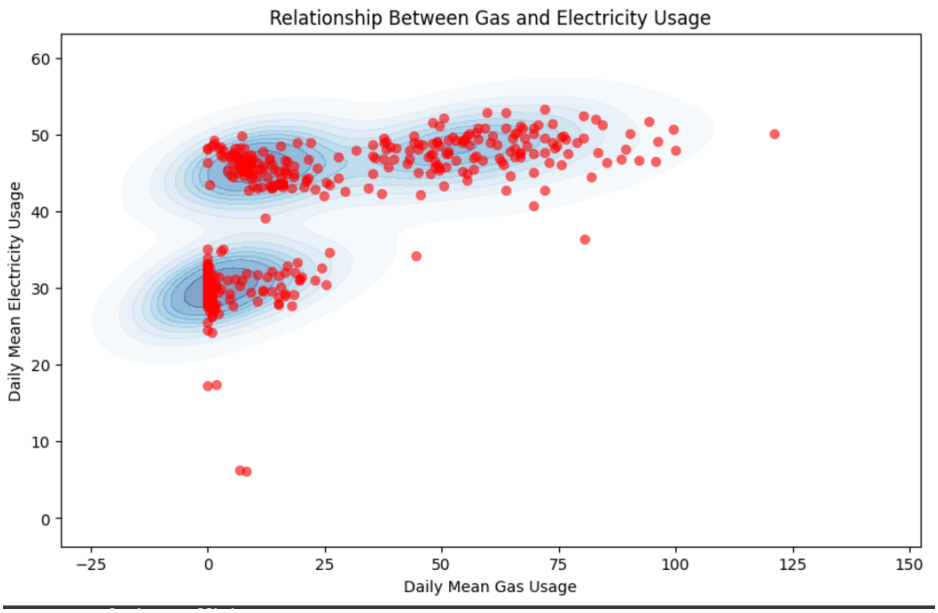

Figure 13 Joint Relationship Between Daily Mean Gas and Electricity Usage

The scatter plot shows a moderate positive correlation between gas and electricity usage (Pearson: 0.60, Spearman: 0.67). Two dense clusters indicate consistent simultaneous usage at low and moderate levels, while higher gas usage aligns with increased electricity usage. Outliers may highlight abnormal patterns or isolated inefficiencies, suggesting opportunities to optimise combined energy consumption.

Message for a General Audience

Our energy consumption analysis highlights key patterns that can help households reduce waste and improve efficiency:

1. Electricity Usage: Electricity consumption remains consistent throughout the week, with slightly lower usage on weekends. This stability suggests well-regulated usage patterns but also an opportunity to further optimise weekday energy use by reducing non-essential appliances.
2. Gas Usage: Gas consumption is higher and more variable on weekdays, likely due to heating or cooking needs, and drops significantly on weekends. This pattern indicates potential inefficiencies during the week that can be addressed by better scheduling and system maintenance.
3. Combined Usage: A moderate positive correlation between gas and electricity usage suggests simultaneous usage of both energy types. This could indicate inefficiencies in heating or other systems that rely on both. Outliers point to unusual consumption that may require investigation.

Actionable Insights

- Reduce Weekday Gas Waste: Identify and fix inefficiencies in heating systems or cooking routines to lower gas usage during the week.
- Optimize Combined Usage: Inspect systems that rely on both gas and electricity (e.g., water heaters) to ensure they are running efficiently and only when needed.
- Weekend Electricity Management: Consider limiting the use of non-essential appliances during weekends to reduce unnecessary electricity consumption.

By adopting these practices, households can cut costs, improve energy efficiency, and contribute to a more sustainable future.

Q2

2.1) **Project Air View Dublin Dataset Description:**

1. Road Data File:

- Purpose: Records details about the streets and highways in Dublin where air quality measurements were collected.
- Rows: 24694 entries, each representing a unique road segment.
- Columns:
    - Road Metadata:
        - road_id, osm_id, osm_code, osm_fclass, osm_name, osm_ref, osm_oneway, osm_maxspeed, osm_layer, osm_bridge, osm_tunnel: Represent OpenStreetMap (OSM) identifiers, road characteristics, and traffic rules.
        - Example: osm_fclass (nominal).(road type, e.g., service or residential) and osm_maxspeed (maximum allowed speed in km/h).
        - osm_layer (Ordinal): Represents vertical positioning (e.g., ground level, bridge, tunnel).
        - osm_name (Nominal): Name of the road (if available).
    - Geometric Data:
        - the_geom: (Ratio) Encoded as LINESTRING, specifies the geographical coordinates of road segments.
    - Pollutant Data:
        - For each pollutant (e.g., NO2, NO, CO2, CO, O3, PM2.5), the file includes:
            - points: (Ratio) Number of measurement points.
            - drives: (Ratio) Number of times the road was driven during measurements.
            - ug/m3 or mg/m3: (Ratio) Average concentration values for each pollutant.
- Observations:
    - Some values, such as pollutant concentrations, are negative, possibly indicating calibration artifacts or errors.
    - Missing data for some pollutants or roads could suggest limited coverage or measurement challenges.

2. Measurements File:

- Purpose: Captures second-by-second air quality measurements recorded by the electric Street View car.
- Rows: Each row corresponds to a 1-second measurement.
- Columns:
    - Timestamp and Location:
        - gps_timestamp: (Interval) Timestamp of the measurement in UTC.
        - latitude, longitude: (Ratio) Geographic coordinates of the measurement location.

- Pollutant Concentrations:
  - NO_ugm3, NO2_ugm3, O3_ugm3: (Ratio) Nitrogen Oxide, Nitrogen Dioxide, and Ozone levels in micrograms per cubic meter.
  - CO_mgm3, CO2_mgm3: (Ratio) Carbon Monoxide and Carbon Dioxide levels in milligrams per cubic meter.
  - PM25_ugm3: Concentration of Particulate Matter (PM2.5) in micrograms per cubic meter.
- Particle Counts:
  - PMch1_perL to PMch6_perL: Size-resolved particles count per Liter, representing particles of different diameters.
- Observations:
  - Some measurements (e.g., O3_ugm3, CO2_mgm3) contain NaN, indicating missing data for certain pollutants.
  - Variability in measurements reflects dynamic air quality conditions influenced by location, time, and traffic.

General Observations:

- Coverage: Measurements are representative of daytime, weekday air quality in urban Dublin streets.
- Granularity: Data allows for both street-level spatial analysis and second-level temporal analysis.
- Data Quality:
  - Negative pollutant values and missing data points suggest the need for preprocessing and calibration.
  - High temporal and spatial resolution provide valuable insights for urban air quality studies and policymaking.

2.2) Real-World Problems Using the Air View Dublin Dataset

1. Regression Problem: Predicting PM2.5 Concentrations Based on Other Pollutants

- Problem Statement: Use machine learning regression to predict PM25_ugm3 concentrations based on features like NO2_ugm3, O3_ugm3, CO_mgm3, and CO2_mgm3.
- Goal: Provide actionable insights for urban planners to identify high-risk areas and helps in understanding how pollutants like NO2 and CO contribute to PM2.5 formation, which is crucial for targeted air quality management strategies.
- Features (Independent Variables):
  - Road metadata: osm_fclass, osm_maxspeed, osm_oneway.
  - Traffic-related attributes: NO2points, NO2drives.
  - Geometric attributes: the_geom.
  - Additional pollutant concentrations: CO2_mgm3, PM25_ugm3.

- The target for this regression problem is to predict the average NO_2 concentration, represented by the dependent variable NO2_ugm3. The approach involves training regression models such as Linear Regression or Random Forest Regressor to estimate NO_2 concentrations accurately. This model has practical applications, including forecasting pollutant levels for roads not covered in the dataset and evaluating the potential impact of proposed traffic management changes on air quality.

---

2. Classification Problem: Categorizing Roads Based on Pollution Risk

- Problem Statement: Classify roads into pollution risk categories (e.g., "Low", "Moderate", "High") based on pollutant concentrations of NO2.
- Goal: Helps policymakers identify areas with hazardous air quality for prioritizing monitoring and public health interventions.
- Features (Independent Variables):
    - Road metadata: osm_fclass, osm_maxspeed, osm_oneway.
    - Pollutant attributes: NO2_ugm3, CO_mgm3, PM25_ugm3, CO2_mgm3.
    - Traffic-related attributes: NO2drives, COdrives, PM25drives.

- The target for this classification problem is to categorise areas based on pollution risk, represented as a categorical variable with labels such as "Low" (pollutants below a threshold), "Moderate" (pollutants within a mid-range), and "High" (pollutants exceeding safety limits). The approach involves applying classification algorithms such as Decision Trees, Logistic Regression, or Gradient Boosting to assign risk levels effectively. Domain knowledge will be crucial in defining appropriate risk thresholds to ensure accurate classification. This model can be applied to identify areas requiring air quality monitoring.

2.3) apply pre-processing techniques:

a) Handle missing data.

To handle missing values in the dataset, numeric missing values in particle count columns were replaced with their respective column mean to ensure data continuity without introducing bias. For pollutant columns, missing values were addressed using linear interpolation, leveraging temporal trends for a more accurate estimation. Rows with missing timestamps or critical spatial data, such as latitude and longitude, were dropped to maintain the integrity and alignment of the dataset.

b) The dataset was reshaped to display aggregated minute-level data by truncating timestamps to the nearest minute and calculating the mean for all numerical columns. This provides a summarized view of pollutant levels and other measurements for each minute. In addition, in the road data for categorical columns (e.g., osm_fclass), the mode imputation is used for missing data.

2.4) Divide the dataset into training, testing and validation datasets and explain your approach.

Classify each minute's air quality based on NO2_ugm3 levels into "Low", "Moderate", or "High" categories, using threshold values derived from air quality standard

Steps**:**

1. Create a target column, Pollution_Level, based on the NO2_ugm3 values. The new column Pollution_Level is added, categorising each row based on the NO2_ugm3 value. This can be done as follows: first Define thresholds for pollution levels :

- Low: NO2_ugm3 < -10
- Moderate: -10 ≤ NO2_ugm3 ≤ 40
- High: NO2_ugm3 > 40

```
from sklearn.model_selection import train_test_split

# Define features (X) and target (y)
X = minute_data[['NO_ugm3', 'NO2_ugm3', 'O3_ugm3', 'CO_mgm3', 'CO2_mgm3']]
y = minute_data['Pollution_Level']

# Step 1: Split into training (70%) and remaining (30%)
X_train, X_temp, y_train, y_temp = train_test_split(X, y, test_size=0.3, stratify=y)

# Step 2: Split remaining data into validation (15%) and testing (15%)
X_val, X_test, y_val, y_test = train_test_split(X_temp, y_temp, test_size=0.5, stratify=y_temp)

# Print dataset sizes
print(f"Training Set: {X_train.shape}, {y_train.shape}")
print(f"Validation Set: {X_val.shape}, {y_val.shape}")
print(f"Testing Set: {X_test.shape}, {y_test.shape}")

```

Split the Dataset:

The dataset is divided into training (70%), validation (15%), and testing (15%) subsets to ensure an excellent model training, tuning, and evaluation.

Stratify During Splitting:

Stratified sampling is applied to preserve the class proportions of Pollution_Level across all subsets, ensuring balanced representation in training, validation, and testing data.

2.5) For this part, let's use the minute obtained data after reshaping and cleansing.

a) Using the describe function, summary statistics for the dataset were generated to provide a comprehensive overview of each column. The dataset contains 86,416 records, with key pollutant variables such as NO2_ugm3, CO2_mgm3, and PM25_ugm3 showing varying distributions. For example, the mean value for NO2_ugm3 is 14.92, with a wide range from -220.19 to 1098.71, indicating the presence of outliers or highly polluted regions. The median (50%) value of PM25_ugm3 is 6.30, with a maximum of 250.76, suggesting significant variability. The geographic range is also evident, with latitude values spanning from 53.30 to 53.41 and longitude from -6.38 to -6.14. The hour column shows hourly data, ranging from 6 to 19, reflecting the daytime measurement schedule.

b) using boxplot:

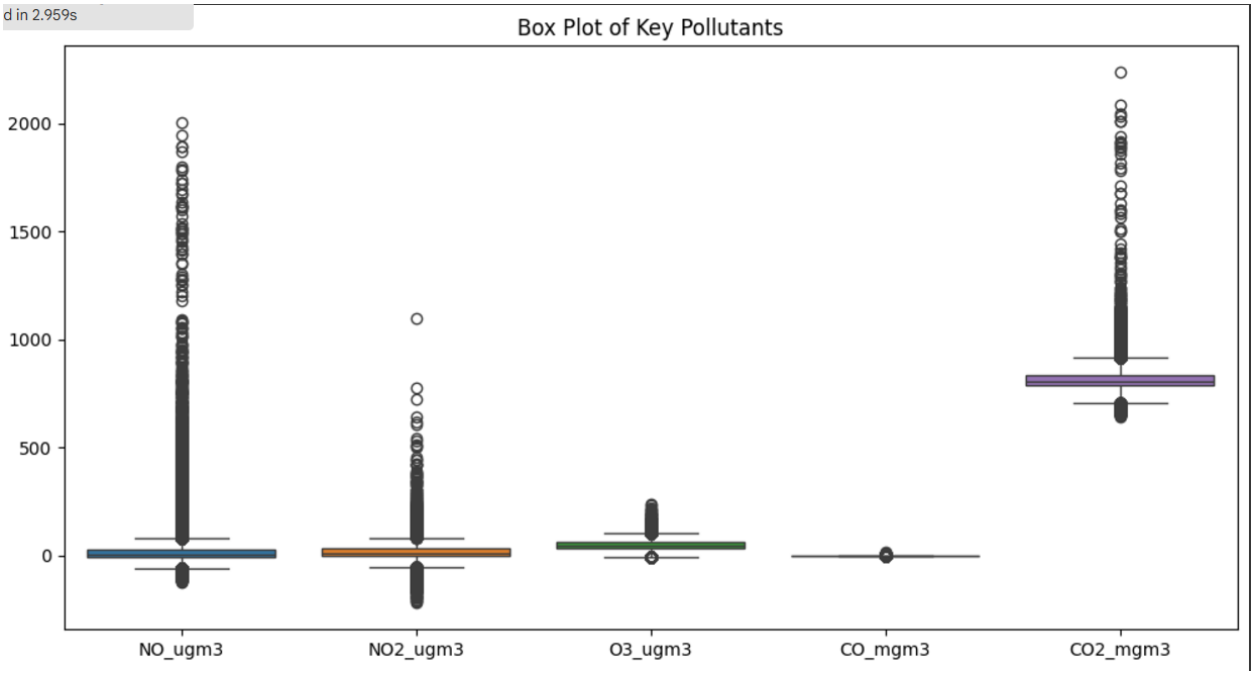

Figure 14 Distribution of Key Pollutants with Outlier Analysis

The box plot visualizes the distribution of key pollutants (NO_ugm3, NO2_ugm3, O3_ugm3, CO_mgm3, and CO2_mgm3), highlighting the interquartile range and the presence of significant outliers. Pollutants like NO_ugm3 and NO2_ugm3 show extreme variability, while CO_mgm3 and O3_ugm3 have more compact distributions. This plot reveals important patterns and anomalies in the pollutant data.

C) scatter plot:

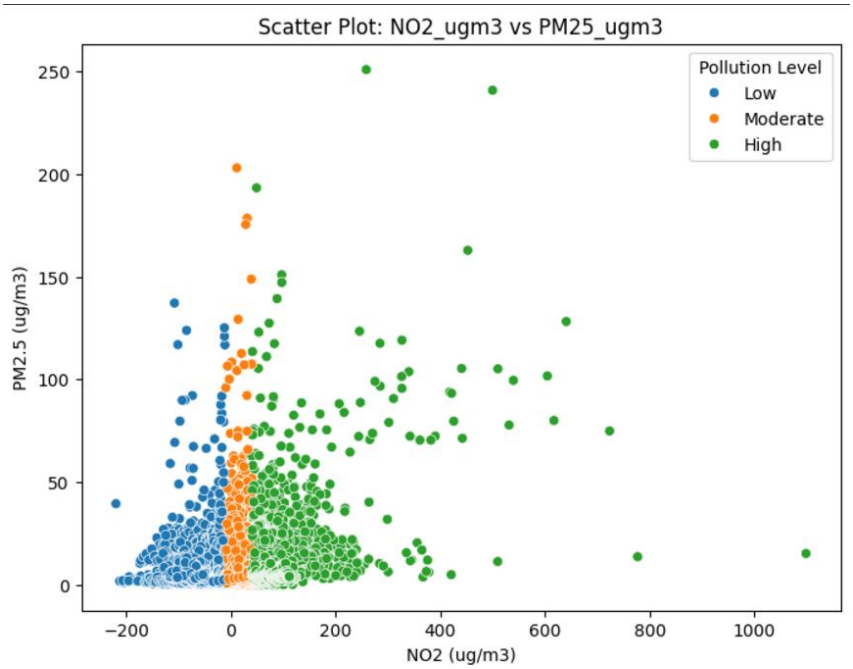

Figure 15 Scatter Plot of NO2 Concentration vs PM2.5 Levels by Pollution Category

This scatter plot visualises the relationship between NO2_ugm3 and PM25_ugm3 concentrations, categorized by pollution levels (Low, Moderate, High). It shows how higher NO2_ugm3 values often correspond to increased PM25_ugm3, with distinct clustering patterns for each pollution level. This indicates a potential correlation between the two pollutants and their impact on air quality.

d) histogram

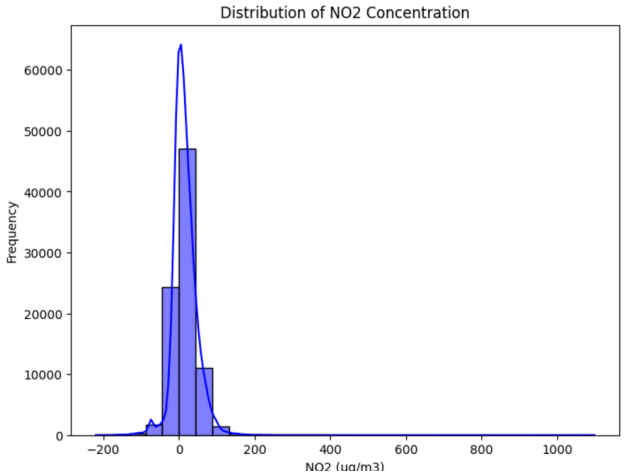

Figure 16 Histogram of NO2 Concentration with Distribution Curve

This histogram illustrates the distribution of NO2_ugm3 concentrations, showing a high frequency of values around 0 to 100 µg/m³, with a long right tail indicating the presence of outliers at higher concentrations. The curve overlay highlights the skewed nature of the data, suggesting variability in NO2 levels across the dataset.

2.6) To train the model for the regression problem, the dataset is divided into features (X) and the target variable (y). The features (X) consist of key pollutants such as NO_ugm3 (Nitric Oxide), NO2_ugm3 (Nitrogen Dioxide), O3_ugm3 (Ozone), and CO2_mgm3, which serve as independent variables. The target variable (y) is PM25_ugm3, representing the concentration of fine particulate matter (PM2.5). This structure allows the regression model to learn the relationship between various pollutant levels and PM2.5 concentrations, enabling accurate predictions.

Then The data is split into training (70%), validation (15%), and testing (15%) sets. The training set is used to train the model, the validation set to fine-tune it, and the testing set to evaluate its performance on unseen data. Then the data can be trained using algorithms like linear regression or nearest neighbour regression.

For linear regression

```
 1. import pandas as pd
 2. from sklearn.model_selection import train_test_split
 3. from sklearn.linear_model import LinearRegression
 4. from sklearn.metrics import mean_squared_error, r2_score
 5. X = minute_data[['NO_ugm3', 'NO2_ugm3', 'O3_ugm3', 'CO_mgm3', 'CO2_mgm3']]
 6. y = minute_data['PM25_ugm3']
 7. # Split into training (70%), validation (15%), and testing (15%)
 8. X_train, X_temp, y_train, y_temp = train_test_split(X, y, test_size=0.3)
 9. X_val, X_test, y_val, y_test = train_test_split(X_temp, y_temp, test_size=0.5)
10. # train the model
11. linear_model = LinearRegression().fit(X_train, y_train)
12. # Predict on validation set
13. y_val_pred = linear_model.predict(X_val)
14.
```

For neighbour regression with k =6

```
from sklearn.neighbors import KNeighborsRegressor
# Select features and target
X = minute_data[['NO_ugm3', 'NO2_ugm3', 'O3_ugm3', 'CO_mgm3', 'CO2_mgm3']]
y = minute_data['PM25_ugm3']
# Split into training and testing data (70% training, 15% validation, 15% testing)
X_train, X_temp, y_train, y_temp = train_test_split(X, y, test_size=0.3)
X_val, X_test, y_val, y_test = train_test_split(X_temp, y_temp, test_size=0.5)

# Initialize the KNeighborsRegressor
knn_model = KNeighborsRegressor(n_neighbors=300)

# Fit the model on the training data
knn_model.fit(X_train, y_train)
# Predict on validation data
y_val_pred = knn_model.predict(X_val)

# Calculate evaluation metrics
val_mse = mean_squared_error(y_val, y_val_pred)
val_r2 = r2_score(y_val, y_val_pred)

```

In nearest neighbor regression, a small k can lead to overfitting and high variance as the model becomes sensitive to noise, while a large k increases bias by over smoothing predictions and ignoring local patterns. Choosing optimal k balances bias and variance for better performance.

2.7)

logistic regression is used to classify roads into pollution risk categories ("Low," "Moderate," "High") based on pollutant concentrations like NO2_ugm3, O3_ugm3, and PM25_ugm3. The target variable, Pollution_Level, was derived from NO2_ugm3 thresholds, making it suitable for a supervised learning approach. Logistic regression was chosen for its simplicity, interpretability, and effectiveness in multi-class classification tasks.

```
import pandas as pd
from sklearn.preprocessing import LabelEncoder
from sklearn.model_selection import train_test_split
from sklearn.linear_model import LogisticRegression
from sklearn.metrics import classification_report, accuracy_score
import matplotlib.pyplot as plt

features = ['NO_ugm3', 'NO2_ugm3', 'O3_ugm3', 'CO_mgm3', 'PM25_ugm3']
target = 'Pollution_Level'
X = minute_data[features]
y = minute_data[target]

# Encode Pollution_Level
le = LabelEncoder()
y = le.fit_transform(y)

# Train-Test split
```

```
18. X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2, random_state=42)
19.
20. # Train model
21. model = LogisticRegression(max_iter=1000, random_state=0)
22. model.fit(X_train, y_train)
23. # Make predictions
24. y_pred = model.predict(X_test)
25. # Evaluate the model
26. print("Accuracy:", accuracy_score(y_test, y_pred))
27.
```

The dataset was preprocessed by encoding the target variable, splitting it into training and testing sets, and training the model on selected features. The model achieved 90% accuracy, with balanced precision and recall across all categories, demonstrating its reliability. Logistic regression was an ideal choice due to its computational efficiency and ability to highlight feature contributions, making it a practical tool for pollution risk classification.

2.8)

To evaluate the trained logistic regression model, we used the following techniques:

1. Accuracy: Calculated as the proportion of correctly predicted instances.

   0.9996528581346911

2. Confusion Matrix: Visualised the model's ability to distinguish between "Low," "Moderate," and "High" pollution levels and identified potential misclassifications.

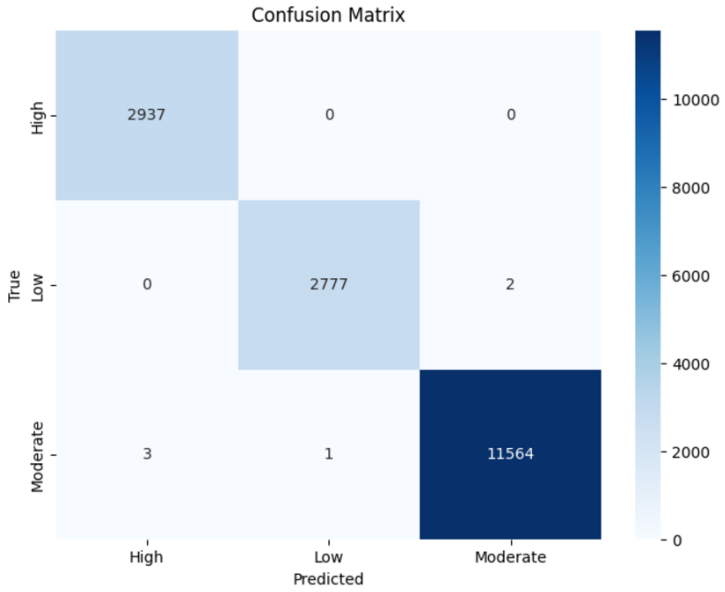

Figure 17 Confusion Matrix: Actual vs Predicted Pollution Levels

For both the linear model and the nearest neighbor model, the Mean Squared Error (MSE) was used as the evaluation metric. The MSE values for these models ranged between 35 and 39, indicating their respective levels of predictive accuracy.

Q3)

3.1)

The Mars Surface dataset contains 6,691 images spanning 24 classes, captured by the Mars Science Laboratory (Curiosity rover) using three instruments: Mastcam Right eye, Mastcam Left eye, and MAHLI. These images, reduced to approximately 256x256 pixels, are "browse" versions of the originals and were collected from sols 3 to 1060 (August 2012 to July 2015). To model operational usage, the dataset is split into training, validation, and testing sets based on the Martian day (sol) of acquisition. This approach mimics how the image archive would grow over time. The dataset spans a variety of classes, such as "apxs," "ground," "sun," "wheel," and more, with mappings between class IDs and names provided for reference.

Each image is labeled and distributed across the dataset splits: train-calibrated-shuffled.txt, val-calibrated-shuffled.txt, and test-calibrated-shuffled.txt. These labels ensure balanced representation and enable model evaluation. The dataset is designed to support the development of machine learning models for Mars imagery classification, with a focus on operational scenarios.

3.2)

1. Clustering Problem:

Problem Statement: Identify distinct surface patterns or textures on Mars by clustering the images without prior labels.

Objective: Using unsupervised learning, group images based on similar visual features (e.g., textures, colours, or structures) to uncover natural groupings or patterns on the Martian surface. This analysis could help scientists discover new terrain types or geological formations that were not previously categorised.

Application: Such clustering could assist in prioritising areas of interest for further exploration, determining the diversity of Martian landscapes, and aiding resource mapping for future missions.

---

2. Classification Problem:

Problem Statement: Develop a system to automatically classify Mars surface images into their respective 24 predefined categories.

Objective: Train a supervised machine learning model using the provided training labels to identify the objects or features in each image, such as "drill holes," "ground," or "wheel."

Application: Automating image classification would significantly speed up data processing for researchers by enabling efficient indexing and retrieval of images based on their content. This could support mission operations by quickly identifying features critical to rover navigation, equipment monitoring, or geological analysis.

3.3)

a) Data Cleansing: Ensure all images are valid and non-corrupt

```
3. # Remove corrupted or unreadable images
 4. clean_images = []
 5. clean_filenames = []
 6.
 7. for img, filename in zip(images, filenames):
 8.     try:
 9.         # Ensure the image is not empty or corrupted
10.         if img is not None and img.shape[-1] == 3:  # Check if it has 3 color channels
11.             clean_images.append(img)
12.             clean_filenames.append(filename)
13.     except Exception as e:
14.         print(f"Error loading image {filename}: {e}")
15.
```

b) Reshaping and Resizing Standardize all images to a fixed smaller size 224x224 pixels for easier analysis.

```
 1. from skimage.transform import resize
 2.
 3. # Define target dimensions
 4. target_height, target_width = 224, 224
 5.
 6. # Resize all images
 7. resized_images = []
 8. for img in clean_images:
 9.     resized = resize(img, (target_height, target_width), anti_aliasing=True)
10.     resized_images.append(resized)
11.
12. print(f"Resized images to {target_height}x{target_width}")
13.
```

Then we can display them to check if it caused any error for example the following will display the first 5 images.

```
1. import matplotlib.pyplot as plt
2.
3. # Display the first 5 resized images
4. for i in range(5):
```

```
    plt.imshow(resized_images[i])  # Show the image
    plt.title(f"Image {i+1}")      # Add a title for reference
    plt.axis('off')                # Turn off axis
    plt.show()

```

Figure 18 an example of the resized images

Next, we can split the resized images into three separate sets manage memory usage more effectively in Colab.

3.4)