**ACADEMIC TASK-2**

**CSE375**

(**DATA SCIENCE TOOLBOX: PYTHON** )

**COMPUTER SCIENCE AND ENGINEERING**

Submitted by:-

Name:- Alok Kumar Mahato

Registration number:- 12325978

Roll No:- 19

Section:- K23GN

**Submitted to :-**

**Dr. Aashima**



**Submitted in complete fulfilment of the requirements**

**for the award of degree of**

**B. Tech CSE**

**School of Computer Science**

**Lovely Professional University Phagwara, Punjab**

**Air Quality Data Analysis**

**DECLARATION**

I am Alok Kumar, a student of Bachelor of Technology under CSE discipline at Lovely Professional University, Punjab, hereby declare that all the information furnished in this project report is based on Air Quality data analysis

Acknowledgement

I would like to express my sincere gratitude to Ashima Mam and all those who supported me during this air quality data analysis project. I am especially thankful for the availability of the Air Quality UCI dataset from the UCI Machine Learning Repository, which served as the foundation for this analysis. I also acknowledge the developers of powerful Python libraries such as Pandas, NumPy, Matplotlib, Seaborn, and Scikit-learn, which made data processing, visualization, and modelling possible. This project has deepened my understanding of environmental data and highlighted the importance of air quality monitoring in ensuring public health and sustainability.

Table of Content

Index Page no.

**1.INTRODUCTION -------------------------------- 1**

**2. Objectives----------------------------------------------- 1**

**3. Source of dataset----------------------------------- 2**

**4. EDA Process in Air Quality Dataset--------- 3**

**5. EDA Process in Air Quality Dataset----------3**

**i. Introduction--------------------------------------------4**

**ii. General Description-----------------------------------4**

**iii. Specific Requirements, Functions, and Formulas---4**

**iv. Analysis Results---------------------------------------------4**

1. **Dataset Information-------------------------------------5**
2. **Handling Missing Data----------------------------------5**
3. **3. Objective-Based Analysis---------------------------5**

**v. Visualization-------------------------------------------------6**

**a. Pollution by Hour (Hourly Average Trends)-------7**

**b. Multivariate Time Series Plot (Daily Trends)-----8**

**c. Correlation and Covariance Analysis--------------9**

**d. Outlier Detection and Visualization-----------------9**

**6. Programing Code in IDEL-----------------------------10-17**

**7. Output of Code in IDEL-------------------------------18-19**

**8.OUTPUT IN VISUALIZATION GRAPH FORM--------19**

* **BAR CHAT------------------------------------------------20**
* **LINE CHART---------------------------------------------20**
* **TIME SERIES PLOT OF 7 AIR QUALITY VARIABL-21**
* **CORRELATION HEATMAP OF AIR QUALITY VARIABLES-----------------------------------------------22**
* **CORRELATION HEATMAP OF AIR QUALITY VARIABLES-------------------------------------------------22**

**9. Conclusion-----------------------------------------------23**

**10. Future scope------------------------------------------23**

**11. References----------------------------------------------24**

**1. Introduction**

Air quality data analysis plays a vital role in understanding the impact of environmental conditions on public health and the ecosystem. This project focuses on analyzing the AirQualityUCI dataset, which contains hourly air pollutant measurements collected in an Italian city. By leveraging data science tools and techniques, this analysis aims to uncover trends, detect anomalies, and evaluate pollutant levels such as CO, NOx, and O3. Through visualization and statistical modeling, the goal is to gain meaningful insights into the quality of air over time and provide a foundation for informed environmental decisions and policies.

**2. Objectives:-**

1. **To understand and explore the AirQualityUCI dataset**  
   Gain familiarity with the dataset's structure, types of variables, and the nature of the collected data.
2. **To clean and preprocess the dataset**  
   Handle missing or invalid values (e.g., -200), convert data types, and prepare the dataset for analysis.
3. **To analyze trends in pollutant levels over time**  
   Examine how concentrations of pollutants like CO, NOx, NO₂, and Benzene vary by hour, day, or season.
4. **To assess correlations between pollutants and meteorological factors**

**1**

1. Investigate how temperature, humidity, and other environmental factors influence air quality.
2. **To visualize air quality patterns using appropriate graphs and charts**  
   Create informative visualizations (line plots, heatmaps, histograms, etc.) to convey key findings clearly.

3.Source of dataset

Link:-

<https://archive.ics.uci.edu/dataset/360/air+quality>

Info:-

The dataset used for this analysis is the **AirQualityUCI** dataset, which is publicly available on the **UCI Machine Learning Repository** and was originally collected from an air quality chemical multi-sensor device located in an Italian city over a period from March 2004 to February 2005.

**4. EDA Process in Air Quality Dataset**

* **Data Loading:** Import the Air Quality dataset and check its basic structure (rows, columns, and data types).
* **Data Cleaning:** Handle missing values, correct data types, and remove or treat outliers if necessary.
* **Statistical Summary:** Calculate summary statistics like mean, median, mode, standard deviation, and range to understand data distribution.
* **Visualization of Distributions:** Use histograms and box plots to analyze the spread and detect any skewness or outliers in pollutant concentrations.
* **Correlation Analysis:** Create a correlation matrix and heatmap to identify relationships between different pollutants and sensor readings.
* **Time-Series Analysis:** Plot line graphs or time series charts to track trends and patterns in air quality data over time.
* **Comparative Plots:** Use scatter plots and bar charts to compare pollutant levels across different time intervals or conditions.
* **Insights & Observations:** Summarize key findings from visual and statistical analysis to guide further modeling or reporting steps.

**5.Analysis on Dataset Air Quality dataset**

**i. Introduction**

This analysis uses the Air Quality UCI dataset to explore air pollution patterns using exploratory data analysis techniques. The dataset consists of hourly averaged responses from an array of air quality chemical sensors.

**ii. General Description**

We examine:

The structure and contents of the dataset

* Handle missing data
* Perform time-based pollution analysis
* Detect trends using time series
* Evaluate feature correlations
* Detect outliers

**iii. Specific Requirements, Functions, and Formulas**

* Pandas, NumPy: Data manipulation
* Matplotlib, Seaborn: Visualization
* Functions used:
  + df.info(), df.describe(), df.isnull(), df.fillna(), df.corr(), df.cov()
  + groupby(), plot(), sns.heatmap(), sns.boxplot(), sns.barplot()
* Formulas:
  + Outlier detection: IQR = Q3 - Q1; Outliers = Values < Q1 - 1.5×IQR or > Q3 + 1.5×IQR

**iv. Analysis Results**

1. Dataset Information

* Displays the head, descriptive statistics, column info, and number of unique values.

**Code:-**

**print("\nINFO ABOUT DATASET")**

**print(df.head())**

**print(df.describe())**

**print(df.info())**

**print(df.columns)**

**print(df.nunique())**

**2. Handling Missing Data**

* Shows null values, drops unnecessary columns, fills missing values with mean.

Code:-

print("\nHANDLING MISSING DATA")

print(df.isnull())

print(df.isnull().sum())

print(df.dropna())

df.drop(columns=['Date'], inplace=True)

filled\_df = df.fillna(df.mean(numeric\_only=True))

**3. Objective-Based Analysis**

**a. Finding Most Missing Values**

**Code:-**

most\_missing = df.isnull().sum().sort\_values(ascending=False)

print("\nMost Missing Values:\n", most\_missing)

b. Filtering Data (e.g., CO > 5)

**code:-**

high\_co = df[df['CO(GT)'] > 5]

print("\nHigh CO Data:\n", high\_co)

c. Imputing Missing Values

code:-

filled\_df = df.fillna(df.mean(numeric\_only=True))

print("\nFilled Missing Data:\n", filled\_df)

**v. Visualization**

**1. Pollution by Hour (Hourly Average Trends)**

Code:-

df = pd.read\_excel("E:\\project py\\AirQualityUCI\_dirty.xlsx")

df.columns = df.columns.str.strip()

df["DateTime"] = pd.to\_datetime(df["Date"].astype(str) + " " + df["Time"].astype(str), errors='coerce')

df.drop(columns=["Date", "Time"], inplace=True)

df["Hour"] = df["DateTime"].dt.hour

pollutants = ["CO(GT)", "NOx(GT)", "NO2(GT)", "C6H6(GT)", "NMHC(GT)"]

hourly\_avg = df.groupby("Hour")[pollutants].mean()

plt.figure(figsize=(10, 5))

hourly\_avg.plot()

plt.title("Average Pollutant Levels by Hour")

plt.xlabel("Hour of Day")

plt.ylabel("Pollutant Concentration")

plt.grid(True)

plt.tight\_layout()

plt.show()

2. Multivariate Time Series Plot (Daily Trends)

Code:-

df = pd.read\_excel("E:\\project py\\AirQualityUCI\_dirty.xlsx")

df.columns = df.columns.str.strip()

df["DateTime"] = pd.to\_datetime(df["Date"].astype(str) + " " + df["Time"].astype(str), errors='coerce')

df.set\_index("DateTime", inplace=True)

columns\_to\_plot = ["CO(GT)", "C6H6(GT)", "NOx(GT)", "NO2(GT)", "T", "RH", "AH"]

df\_plot = df[columns\_to\_plot].dropna()

plt.figure(figsize=(14, 7))

for col in columns\_to\_plot:

plt.plot(df\_plot.index, df\_plot[col], label=col)

plt.title("Time Series Plot of 7 Air Quality Variables")

plt.xlabel("DateTime")

plt.ylabel("Values")

plt.legend()

plt.grid(True)

plt.tight\_layout()

plt.show()

3. Correlation and Covariance Analysis

Code:-

correlation = df.corr(numeric\_only=True)

print("\nCorrelation Matrix:\n", correlation)

sns.heatmap(correlation, cmap="Blues", annot=True, linewidth=3, fmt=".2f")

plt.title("Correlation Heatmap of Air Quality Variables")

plt.tight\_layout()

plt.show()

covariance = df.cov(numeric\_only=True)

print("\nCovariance Matrix:\n", covariance)

4. Outlier Detection and Visualization

Code:-

numeric = df.select\_dtypes(include=['number'])

Q1 = numeric.quantile(0.25)

Q3 = numeric.quantile(0.75)

IQR = Q3 - Q1

lower\_bound = Q1 - 1.5 \* IQR

upper\_bound = Q3 + 1.5 \* IQR

outliers = ((numeric < lower\_bound) | (numeric > upper\_bound)).sum()

print("\nOutliers Count:\n", outliers)

plt.figure(figsize=(12, 6))

sns.barplot(x=outliers.index, y=outliers.values)

plt.xticks(rotation=45, ha="right")

plt.title("Number of Outliers per Numeric Feature")

plt.xlabel("Feature")

plt.ylabel("Outlier Count")

plt.tight\_layout()

plt.show()

plt.figure(figsize=(10, 5))

sns.boxplot(data=numeric)

plt.title("Boxplot of Numeric Features for Outlier Detection")

plt.grid(True)

plt.tight\_layout()

plt.show()

**6.Programing Code in IDEL**

import numpy as np

import pandas as pd

import seaborn as sns

import matplotlib.pyplot as plt

df=pd.read\_excel("E:\\project py\\AirQualityUCI\_dirty.xlsx")

print(df)

#Info About Dataset

print("\n INFO ABOUT DATASET")

print(df.head())

print(df.describe())

print(df.info())

print(df.columns)

print(df.nunique())

#Handling Missing Data

print("\n HANDLING MISSING DATA")

print(df.isnull())

print(df.isnull().sum)

print(df.dropna())

print(df.drop(columns=['Date'], inplace=True))

# 5 Objective

print("\n 5 OBJECTIVE")

# 1 Finding most missing value

most\_missing = df.isnull().sum().sort\_values(ascending=False)

print(most\_missing)

# 2 Filtering the Data

high\_co = df[df['CO(GT)'] > 5]

print(high\_co)

# 3 Imputing the Data

filled\_df = df.fillna(df.mean(numeric\_only=True))

print(filled\_df)

# 4 Visulation of data to check the pollution by Hours Increasing and Decrasing

df=pd.read\_excel("E:\\project py\\AirQualityUCI\_dirty.xlsx")

print(df)

# Strip any whitespace from column names

df.columns = df.columns.str.strip()

# Combine Date and Time into a single datetime column

df["DateTime"] = pd.to\_datetime(df["Date"].astype(str) + " " + df["Time"].astype(str), errors='coerce')

# Drop the original Date and Time columns

df.drop(columns=["Date", "Time"], inplace=True)

# Extract hour and day of week from DateTime

df["Hour"] = df["DateTime"].dt.hour

df["DayOfWeek"] = df["DateTime"].dt.day\_name()

# Display the DateTime breakdown

print(df[["DateTime", "Hour", "DayOfWeek"]].head())

# Define list of pollutant columns

pollutants = ["CO(GT)", "NOx(GT)", "NO2(GT)", "C6H6(GT)", "NMHC(GT)"]

# Compute average pollutant levels by hour

hourly\_avg = df.groupby("Hour")[pollutants].mean()

plt.figure(figsize=(10, 5))

hourly\_avg.plot()

plt.title("Average Pollutant Levels by Hour")

plt.xlabel("Hour of Day")

plt.ylabel("Pollutant Concentration")

plt.grid(True)

plt.tight\_layout()

plt.show()

# 5 Multivariate Time Series Visualization to detecting pollution peaking at certain hours or days

df=pd.read\_excel("E:\\project py\\AirQualityUCI\_dirty.xlsx")

df.columns = df.columns.str.strip()

# Combine Date and Time into a single DateTime column and set it as index

df["DateTime"] = pd.to\_datetime(df["Date"].astype(str) + " " + df["Time"].astype(str), errors='coerce')

df.set\_index("DateTime", inplace=True)

# Select 7 variables to plot

columns\_to\_plot = ["CO(GT)", "C6H6(GT)", "NOx(GT)", "NO2(GT)", "T", "RH", "AH"]

df\_plot = df[columns\_to\_plot].dropna()

# Plot all selected variables on one graph

plt.figure(figsize=(14, 7))

for col in columns\_to\_plot:

plt.plot(df\_plot.index, df\_plot[col], label=col)

plt.title("Time Series Plot of 7 Air Quality Variables")

plt.xlabel("DateTime")

plt.ylabel("Values")

plt.legend()

plt.grid(True)

plt.tight\_layout()

plt.show()

# Finding the Relation Btween

print("\n FINDINNG THE RELATION BTWEEN")

correlation = df.corr(numeric\_only=True)

print(correlation)

sns.heatmap(correlation, cmap="Blues", annot=True, linewidth=3, fmt=".2f")

plt.title("Correlation Heatmap of Air Quality Variables")

plt.tight\_layout()

plt.show()

covariance=df.cov(numeric\_only=True)

print(covariance)

numeric = df.select\_dtypes(include=['number'])

print(numeric)

Q1 = df[numeric.columns].quantile(0.25)

print(Q1)

Q3 = df[numeric.columns].quantile(0.75)

print(Q3)

IQR = Q3-Q1

print(IQR)

lower\_bound = Q1-1.5\*IQR

print(lower\_bound)

upper\_bound = Q3+1.5\*IQR

print(upper\_bound)

outliers = ((df[numeric.columns] < lower\_bound) | (df[numeric.columns] > upper\_bound)).sum()

print(outliers)

# Creating Bar plot

plt.figure(figsize=(12, 6))

sns.barplot(data = outliers)

plt.xticks(rotation=45, ha="right")

plt.title("Number of Outliers per Numeric Feature")

plt.xlabel("Feature")

plt.ylabel("Outlier Count")

plt.tight\_layout()

#plt.grid(True)

plt.show()

# Create Box plot

plt.figure(figsize=(10, 5))

sns.boxplot(data=outliers)

plt.title("Outlier Count per Numeric Feature")

plt.ylabel("Number of Outliers")

plt.grid(True)

plt.tight\_layout()

plt.show()

A screenshot of a computer

AI-generated content may be incorrect.7.Output:-

A screenshot of a computer

AI-generated content may be incorrect.

A screenshot of a computer

AI-generated content may be incorrect.

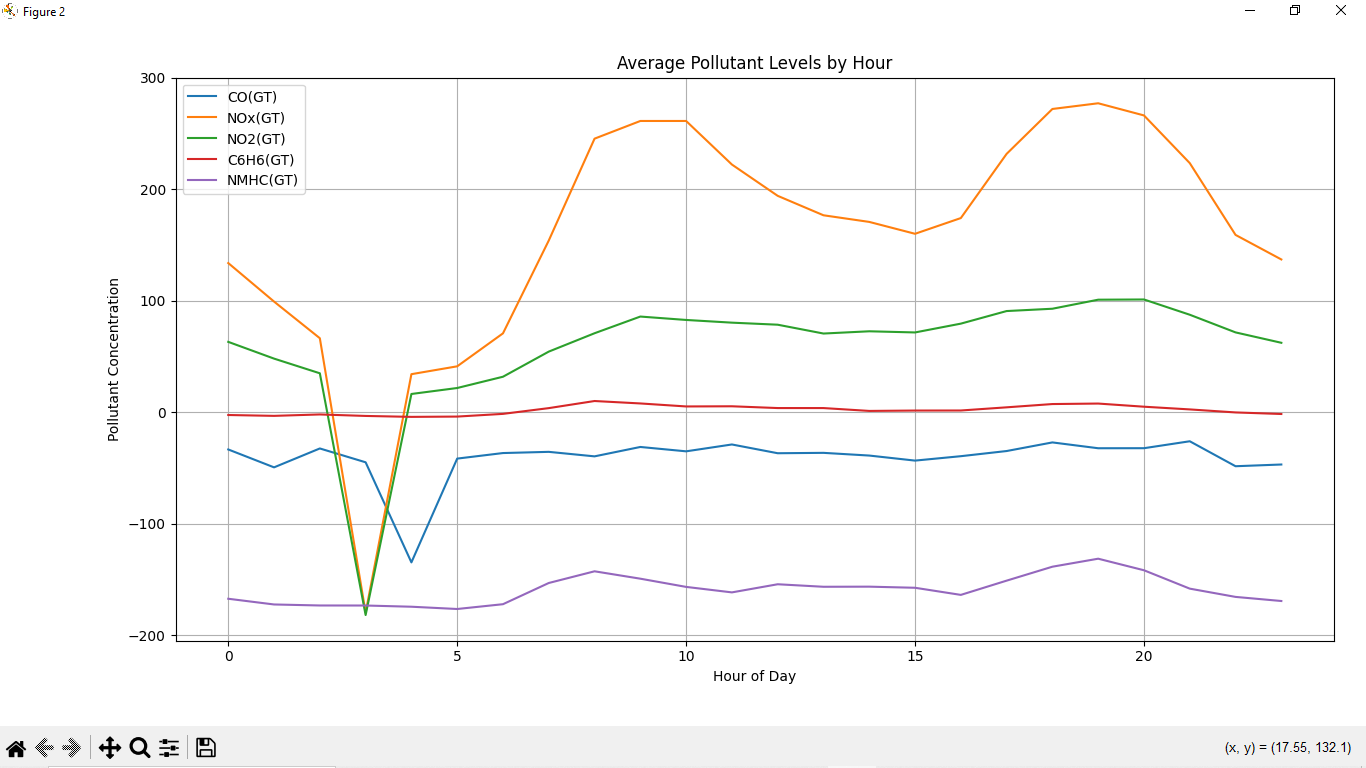
**8.OUTPUT IN VISUALIZATION GRAPH FORM**

* **BAR CHAT**

A graph of blue rectangular bars

AI-generated content may be incorrect.

* **LINE CHART**

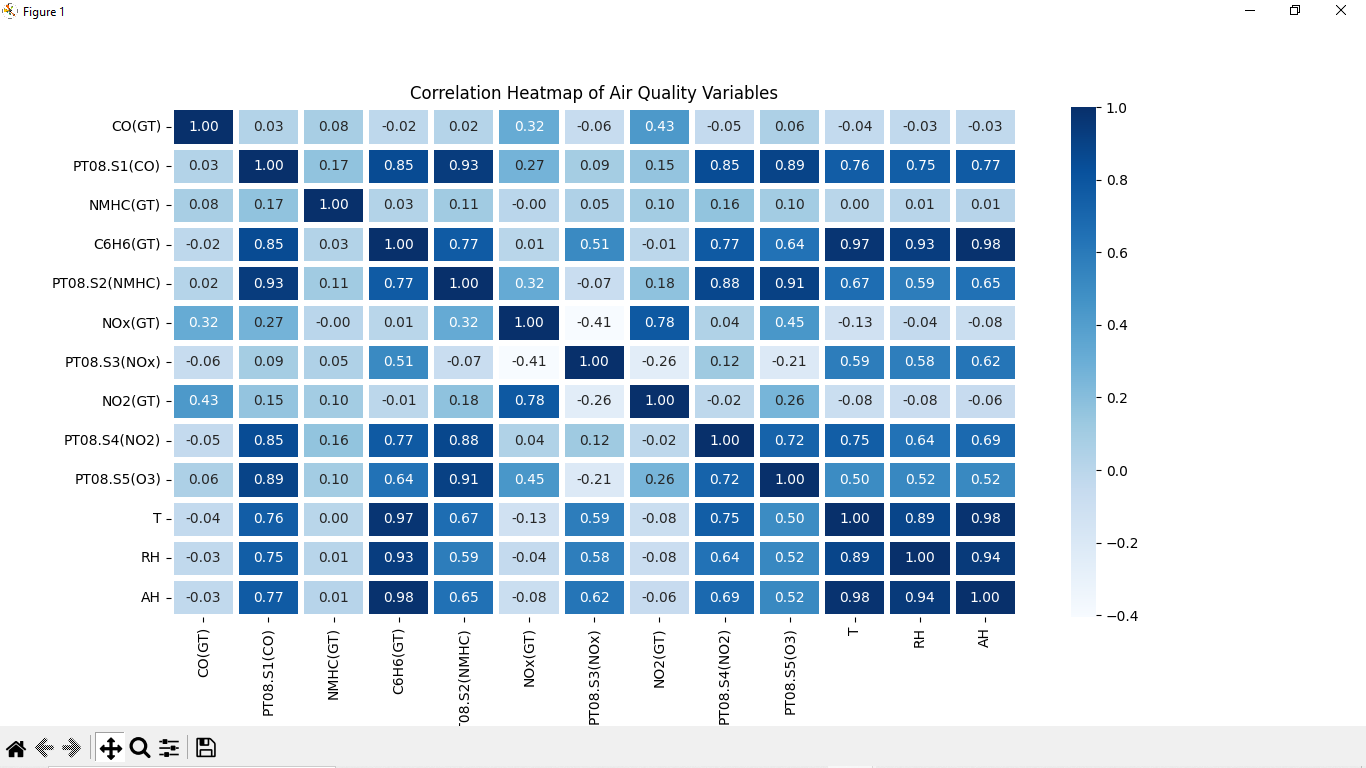


* **TIME SERIES PLOT OF 7 AIR QUALITY VARIABLES**

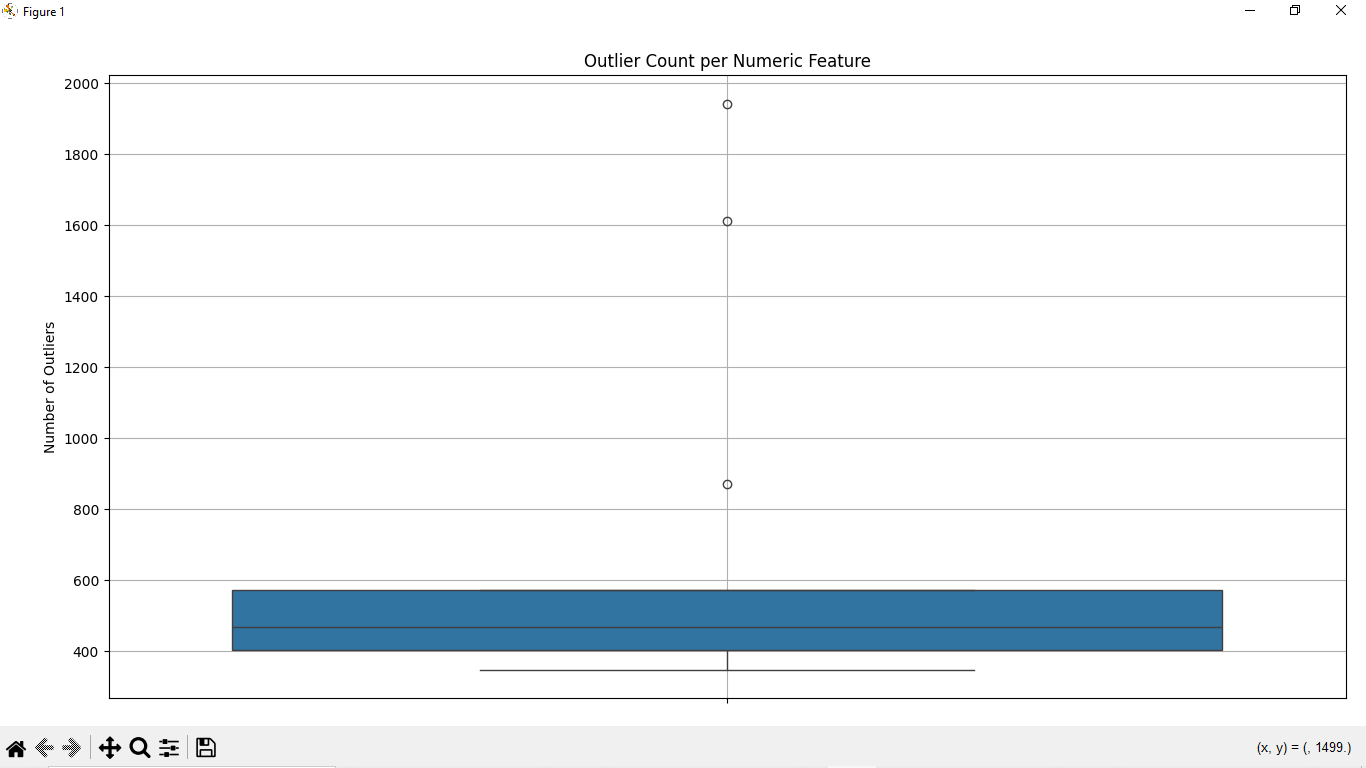
A graph showing a number of data

AI-generated content may be incorrect.

* **CORRELATION HEATMAP OF AIR QUALITY VARIABLES**



* **OUTLIER COUNT PER NUMBERIC FEATURE**



**9.Conclusion**

The analysis of the Air Quality UCI dataset revealed important insights into air pollution levels and the relationships between various pollutants and meteorological factors.

By handling missing data, we ensured the dataset's integrity, and through hourly pollutant analysis, we identified peak pollution times, which are crucial for air quality control. The multivariate time series analysis highlighted the dynamic relationships between pollutants and environmental conditions such as temperature, humidity, and absolute humidity. Additionally, correlation and covariance analyses showed significant relationships among pollutants, indicating shared sources or influences.

Outlier detection further helped in identifying anomalies, which could suggest measurement errors or extreme pollution events. Overall, the findings underscore the importance of data preprocessing and visualization in drawing actionable conclusions for understanding and managing air quality.

**10.Future scope**

The future scope of the Air Quality UCI dataset lies in its potential to drive more advanced research and applications for improving air quality and public health.

One area for development is integrating real-time data collection from sensors to create predictive models for air pollution forecasting, enabling timely alerts for high pollution levels. Additionally, incorporating machine learning techniques could allow for more accurate classification of pollution sources and better understanding of complex pollutant interactions.

Further, the dataset could be expanded by integrating socio-economic and health-related data to assess the impacts of air pollution on various demographics and regions.

Finally, advancing data visualization techniques could provide more accessible and actionable insights for policy makers, environmentalists, and urban planners to design effective interventions and regulations.

**11.References**

- Archive.ics.uci – for Dataset of Air quality

-IDEL – For Codeing

-Chatgpt – for Rectifying the error in Code

-Google – for implanting and finding the new code for project