**Analysis and Visualization of Air Quality Data**

**Introduction**

This report aims to provide a comprehensive analysis and visualization of the Air Quality dataset.

The dataset includes various air quality metrics measured in a certain region.

The key objectives of this assignment are to:

1. Preprocess the data to handle missing values and convert necessary columns.

2. Conduct Exploratory Data Analysis (EDA) to understand the dataset better.

3. Visualize the data to gain insights into the patterns and relationships among different air quality metrics.

**Data Preprocessing**

1. Loading the Data

The data is loaded from an Excel file using the `pandas` library.

Code:

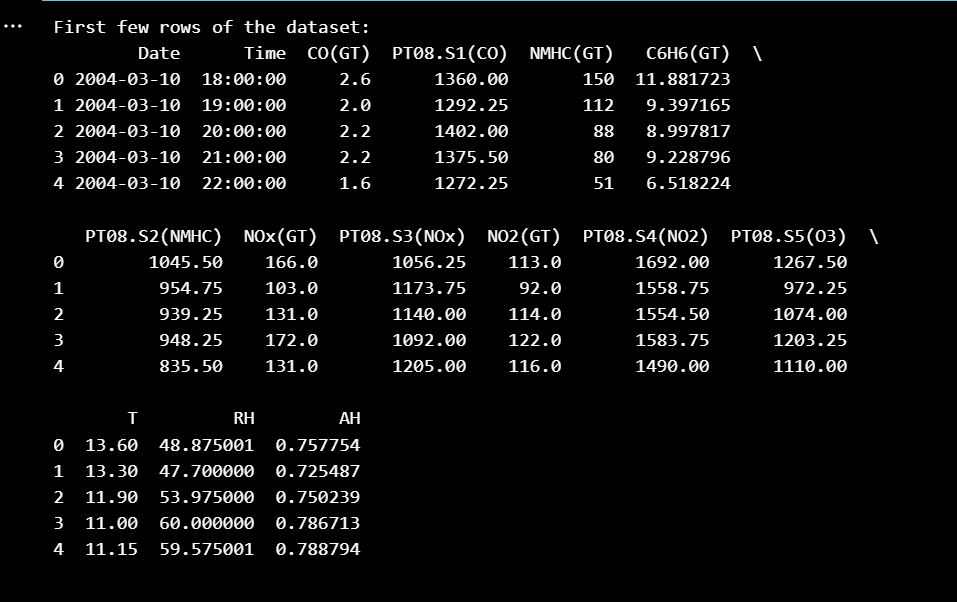
import pandas as pd

import matplotlib.pyplot as plt

import seaborn as sns

Loading Data

df = pd.read\_excel('/content/AirQualityUCI.xlsx')

1. Initial Data Inspection

The first few rows of the dataset were displayed to understand the structure and content of the data.

Code:

print("First few rows of the dataset:")

print(df.head())

1. Date and Time Conversion

The 'Date' and 'Time' columns were combined into a single 'Datetime' column to facilitate time series analysis.

Code:

Convert 'Date' and 'Time' columns to strings, then combine them into a single 'Datetime' column

df['Datetime'] = pd.to\_datetime(df['Date'].astype(str) + ' ' + df['Time'].astype(str))

df.drop(['Date', 'Time'], axis=1, inplace=True)

Set the 'Datetime' column as the index

df.set\_index('Datetime', inplace=True)

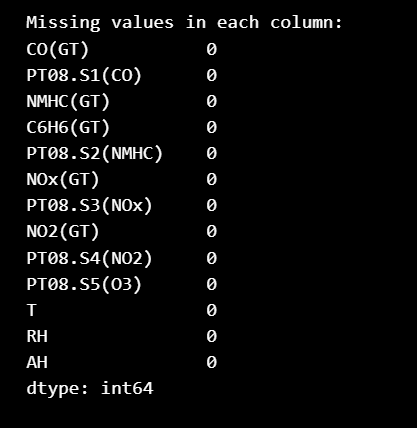
1. Handling Missing Values

The dataset was checked for missing values. Any missing values in numeric columns were filled with the median of the respective columns.

Code:

# Check for missing values

print("\nMissing values in each column:")

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

# Fill missing values with the median of each column (numeric columns only)

numeric\_columns = df.select\_dtypes(include=['float64', 'int64']).columns

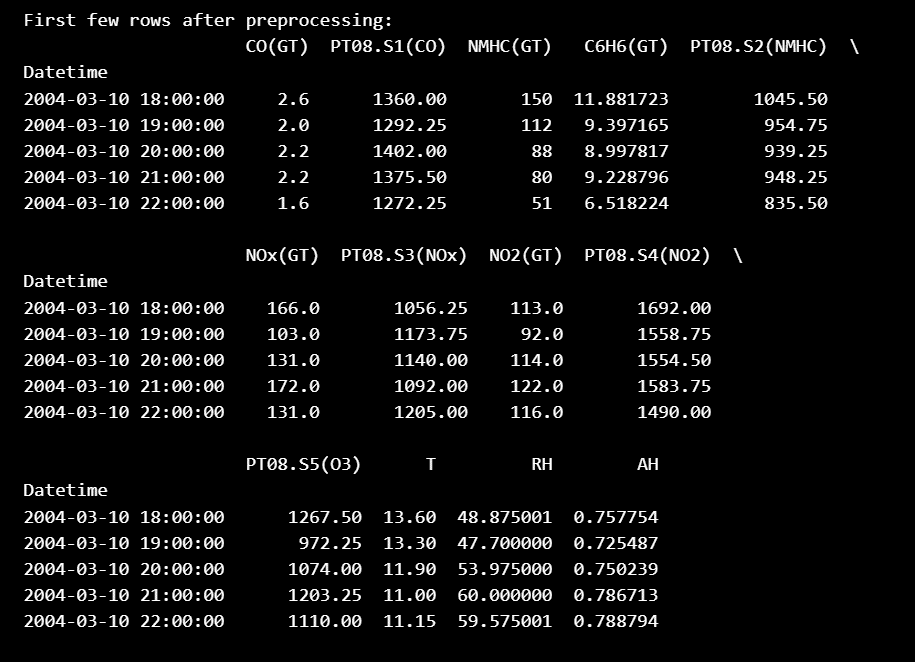
df[numeric\_columns] = df[numeric\_columns].fillna(df[numeric\_columns].median())

1. Post-Processing Inspection

# The first few rows of the dataset after preprocessing were displayed to verify the changes.

Code:

print("\nFirst few rows after preprocessing:")

print(df.head())

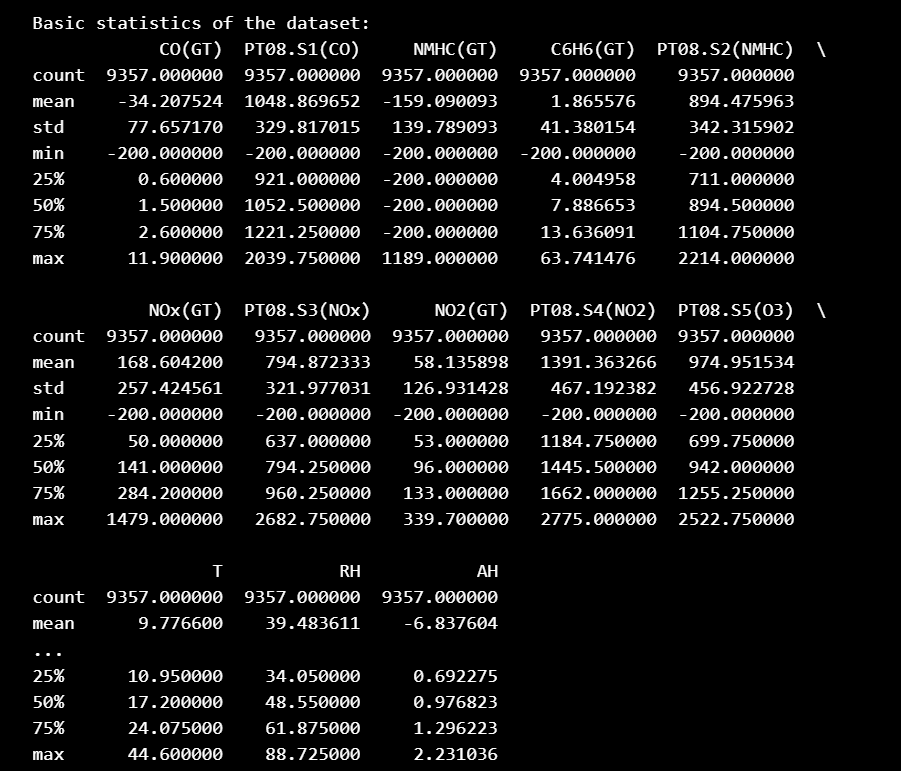
1. Exploratory Data Analysis (EDA)

# Basic Statistics: Descriptive statistics were computed to provide a summary of the dataset.

Code:

print("\nBasic statistics of the dataset:")

print(df.describe())



1. Histograms

# Histograms for each numeric column were plotted to understand the distribution of values.

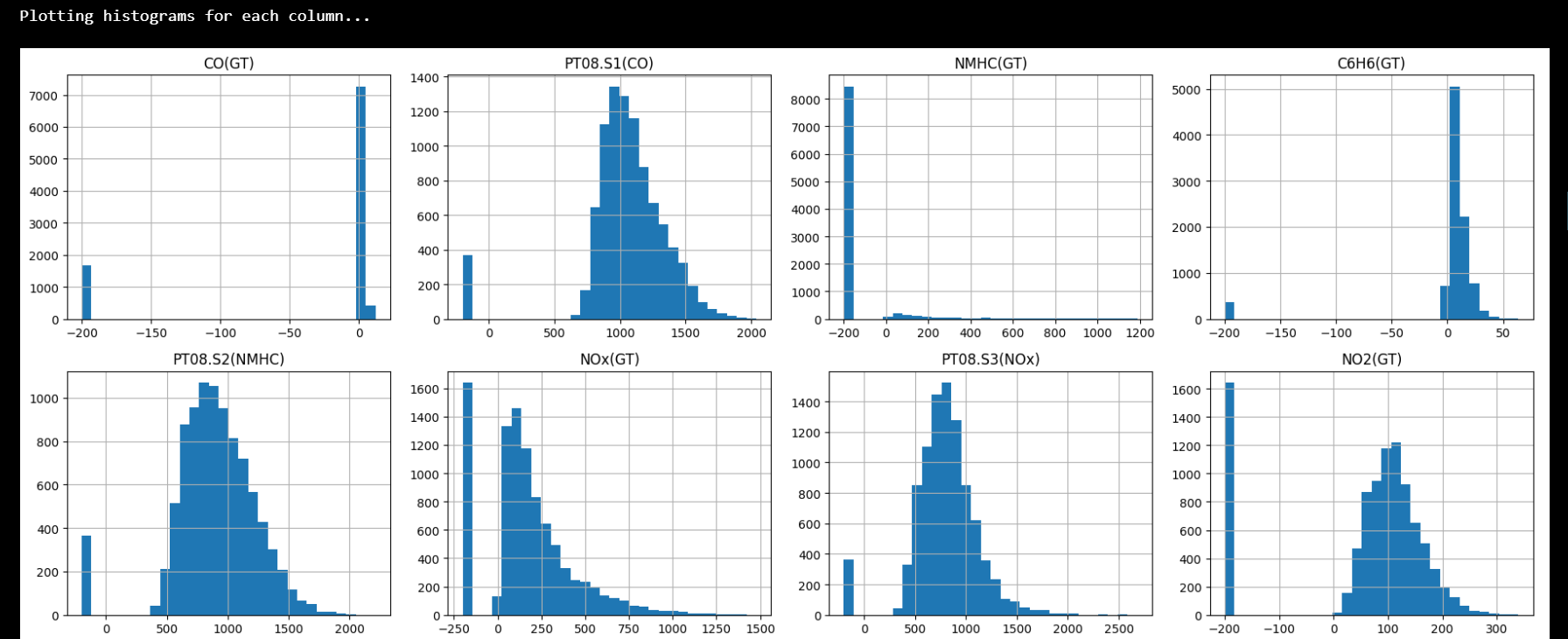
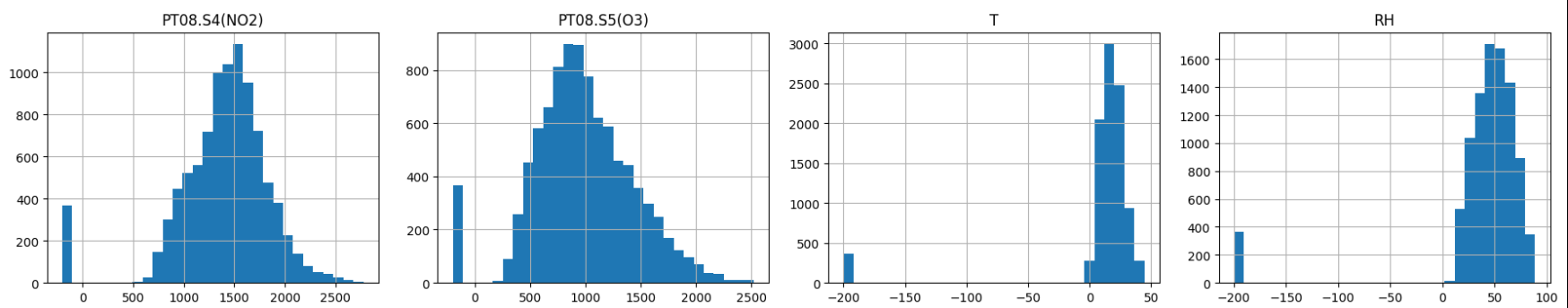
Code:

print("\nPlotting histograms for each column...")

df.hist(bins=30, figsize=(20, 15))

plt.tight\_layout()

plt.show()



1. Correlation Heatmap

# A heatmap of the correlation matrix was plotted to identify relationships between different variables.

Code:

print("\nPlotting correlation heatmap...")

plt.figure(figsize=(12, 10))

sns.heatmap(df.corr(), annot=True, fmt='.2f', cmap='coolwarm')

plt.show()

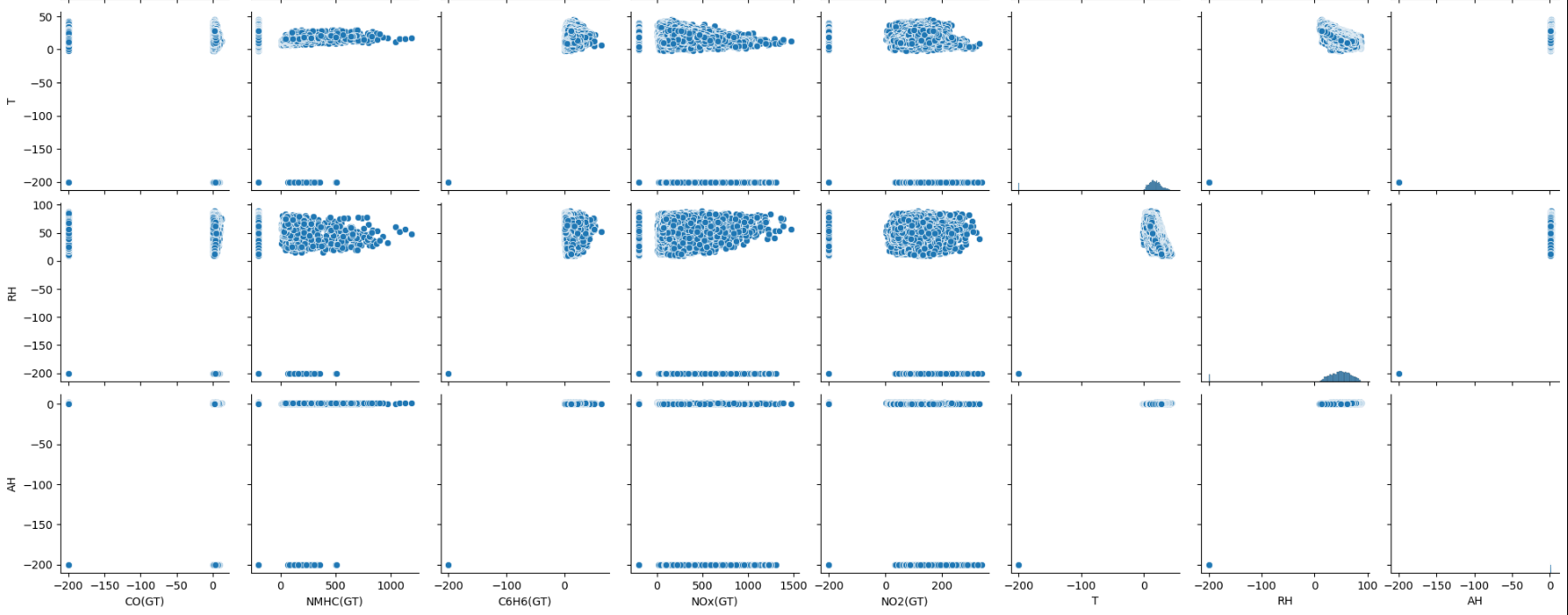
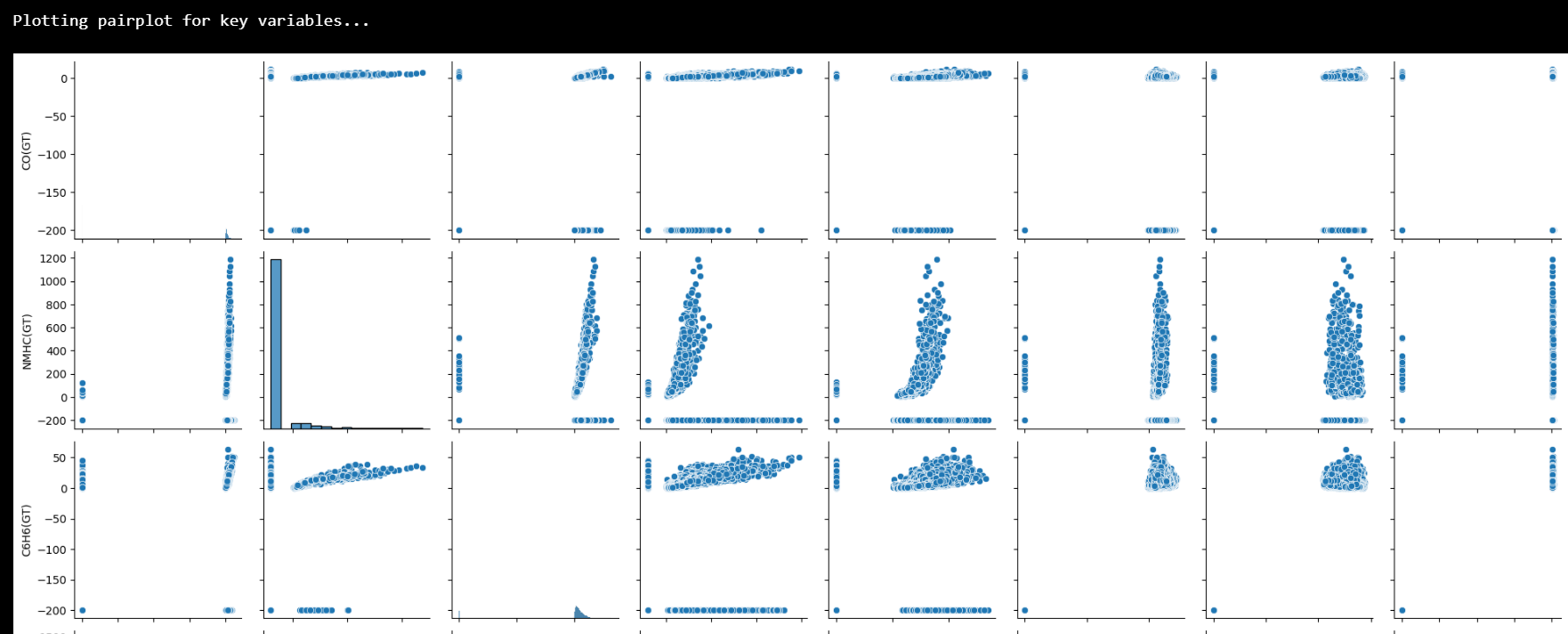
1. Pair Plot

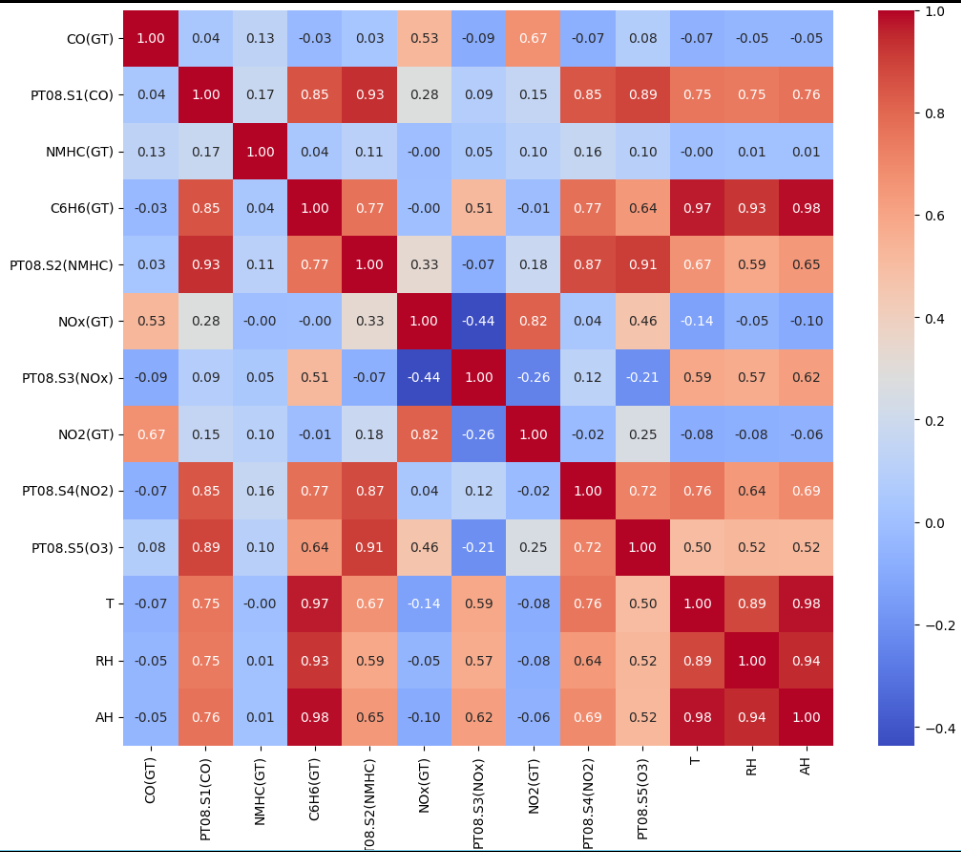
# A pair plot for a subset of key variables was generated to explore their pairwise relationships.

Code:

print("\nPlotting pairplot for key variables...")

sns.pairplot(df[['CO(GT)', 'NMHC(GT)', 'C6H6(GT)', 'NOx(GT)', 'NO2(GT)', 'T', 'RH', 'AH']])

****

**Correlation Heatmap**

1. Data Visualization
2. Time Series Plot

# A time series plot was created to visualize the trends of key pollutants over time.

Code:

print("\nPlotting time series for key pollutants...")

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

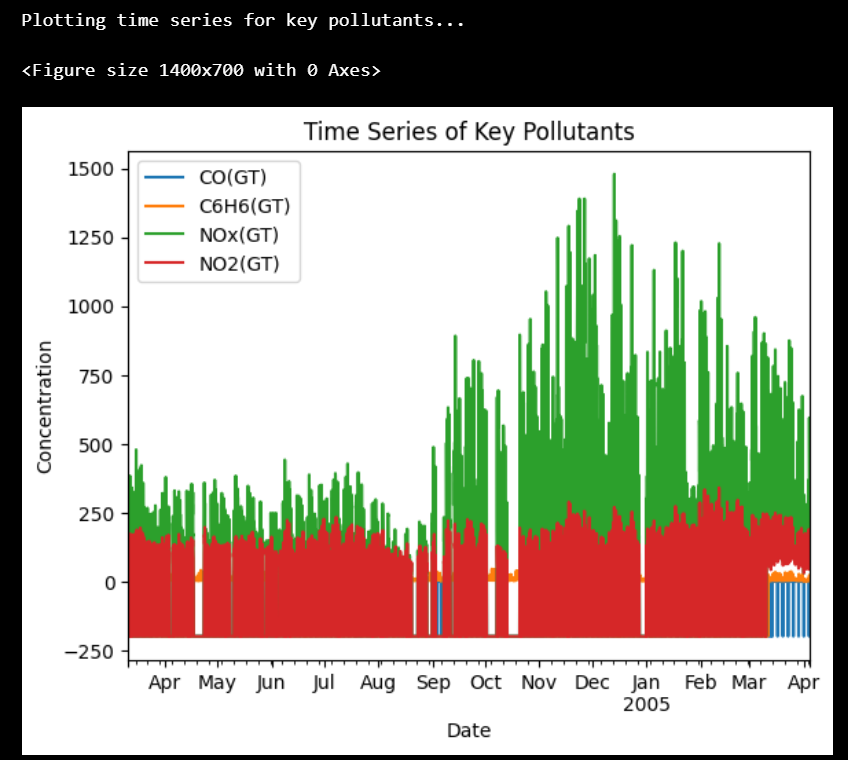
df[['CO(GT)', 'C6H6(GT)', 'NOx(GT)', 'NO2(GT)']].plot()

plt.title('Time Series of Key Pollutants')

plt.xlabel('Date')

plt.ylabel('Concentration')

plt.legend()

plt.show()

1. Scatter Plot

A scatter plot between Temperature (T) and CO levels (CO(GT)) was generated to investigate their relationship.

Code:

print("\nPlotting scatter plot between Temperature and CO levels...")

plt.figure(figsize=(10, 6))

sns.scatterplot(x='T', y='CO(GT)', data=df)

plt.title('Scatter Plot between Temperature and CO Levels')

plt.xlabel('Temperature (T)')

plt.ylabel('CO(GT)')

plt.show()

1. Monthly Analysis

The CO(GT) levels were analyzed on a monthly basis using different types of plots.

Line Plot

A line plot was created to show the average CO(GT) levels for each month.

Code:

print("\nPlotting line plot for monthly average CO(GT) levels...")

monthly\_avg = df['CO(GT)'].resample('M').mean()

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

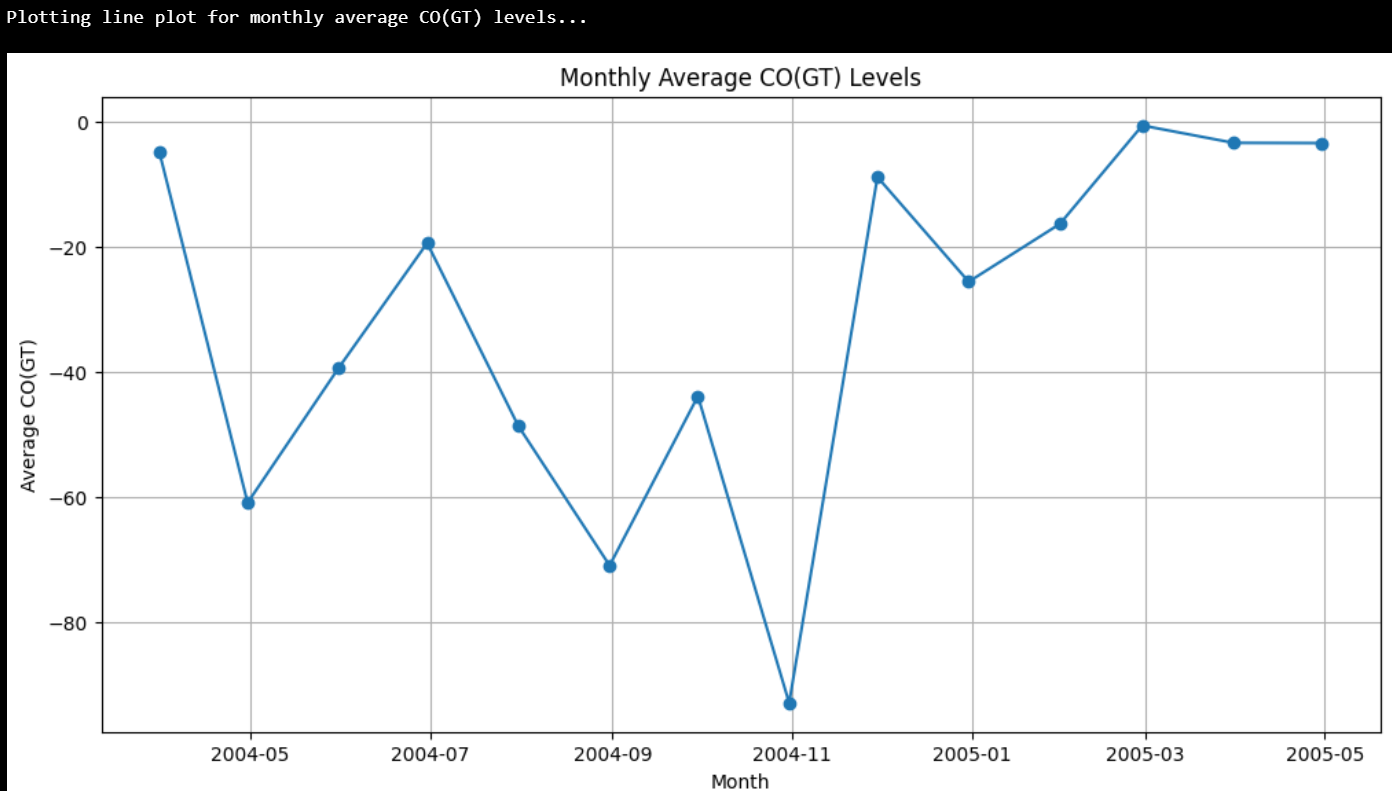
plt.plot(monthly\_avg.index, monthly\_avg.values, marker='o')

plt.title('Monthly Average CO(GT) Levels')

plt.xlabel('Month')

plt.ylabel('Average CO(GT)')

plt.grid(True)

plt.show()

Conclusion

This assignment involved loading, preprocessing, and analyzing an air quality dataset. Through various visualizations, several insights were gained:

1. Trends and Distributions: Histograms and time series plots revealed the distributions and trends of key pollutants.

2. Correlations: The correlation heatmap and pair plot helped identify relationships between different air quality metrics.

3. Monthly Analysis: Various plots (box plot, line plot, bar plot, and violin plot) provided a comprehensive view of CO(GT) levels across different months.

The visualizations generated in this report can be used to inform further analysis and decision-making related to air quality management.