AIR_QUALITY_TURIN_VS_HELSINKI_OCT1TO7_10_2023

October 8, 2023

1 Comparing The Air Quality Of Two European Cities 1 - 7 Oct 2023

[1]:
```
import matplotlib.pyplot as plt
import pandas as pd
import numpy as np
import pandas as pd
from sklearn.linear_model import LinearRegression
from sklearn.model_selection import train_test_split
from sklearn.ensemble import RandomForestRegressor
```

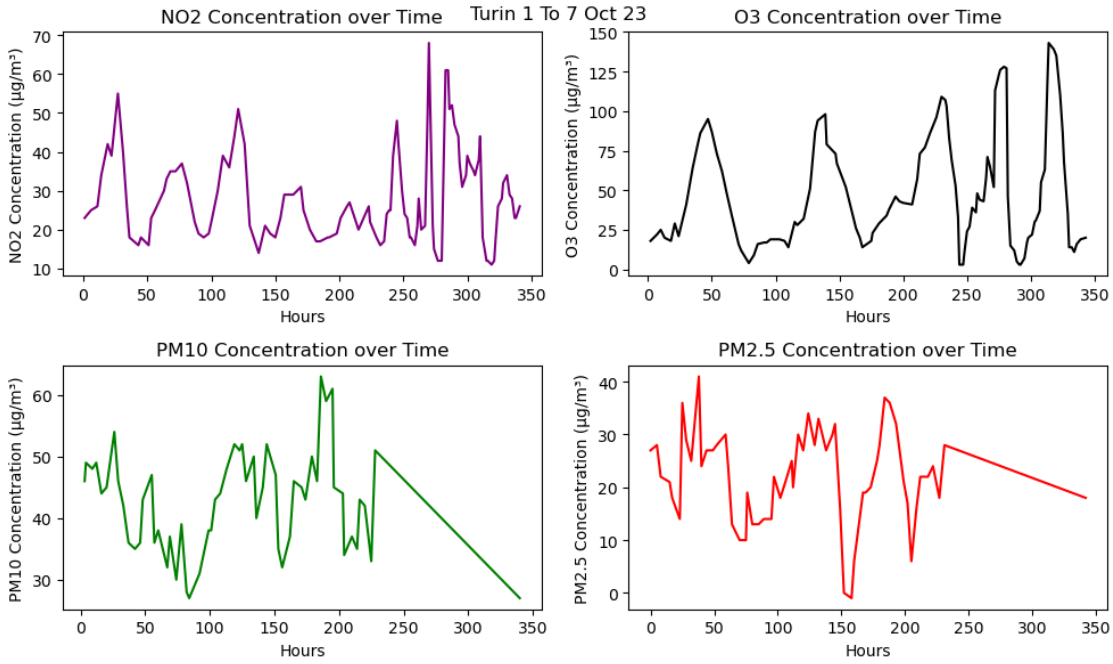
2 Turin Italy Air Quality From 01/10/2023 <=> 07/10/2023

[2]:
```
# Read the CSV Dataframe
df = pd.read_csv("/home/zorinosgh/Desktop/CLIMATE_CHANGE_MODELS/
 ↪Air_Quality_Turin_Oct_First_Week.csv")

# Get the unique values of the parameters column
parameters = df['parameter'].unique()

# Print the unique values of the parameters column
print(parameters)

# Get the data type of the values column
values_dtype = df['value'].dtype

# Print the data type of the values column
print(values_dtype)

# Get the date and time of the first recording
first_recording_datetime = df['dateLocal'].iloc[0]

# Print the date and time of the first recording
print(first_recording_datetime)
```

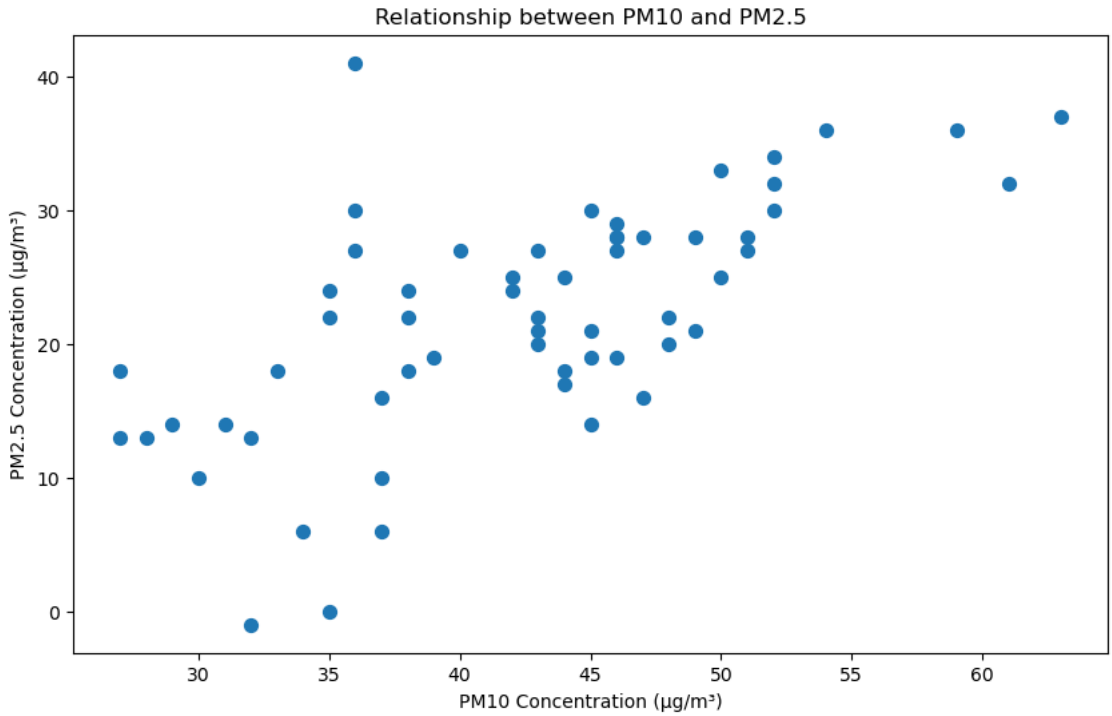
```
['pm25' 'no2' 'o3' 'pm10']
```

```
int64
2023-10-07T05:00:00+02:00
```

[3]:
```python
# Create a figure and a set of subplots
fig, axs = plt.subplots(2, 2, figsize=(10, 6))

# Plot the NO2 concentration
axs[0, 0].plot(df['value'][df['parameter'] == 'no2'], color='purple')
axs[0, 0].set_xlabel('Hours')
axs[0, 0].set_ylabel('NO2 Concentration (µg/m³)')
axs[0, 0].set_title('NO2 Concentration over Time')

# Plot the O3 concentration
axs[0, 1].plot(df['value'][df['parameter'] == 'o3'], color='black')
axs[0, 1].set_xlabel('Hours')
axs[0, 1].set_ylabel('O3 Concentration (µg/m³)')
axs[0, 1].set_title('O3 Concentration over Time')

# Plot the PM10 concentration
axs[1, 0].plot(df['value'][df['parameter'] == 'pm10'], color='green')
axs[1, 0].set_xlabel('Hours')
axs[1, 0].set_ylabel('PM10 Concentration (µg/m³)')
axs[1, 0].set_title('PM10 Concentration over Time')

# Plot the PM2.5 concentration
axs[1, 1].plot(df['value'][df['parameter'] == 'pm25'], color='red')
axs[1, 1].set_xlabel('Hours')
axs[1, 1].set_ylabel('PM2.5 Concentration (µg/m³)')
axs[1, 1].set_title('PM2.5 Concentration over Time')

# Adjust the subplot layout
fig.tight_layout()

#Show plot title
plt.suptitle("Turin 1 To 7 Oct 23 ")

# Show the plot
plt.show()
```

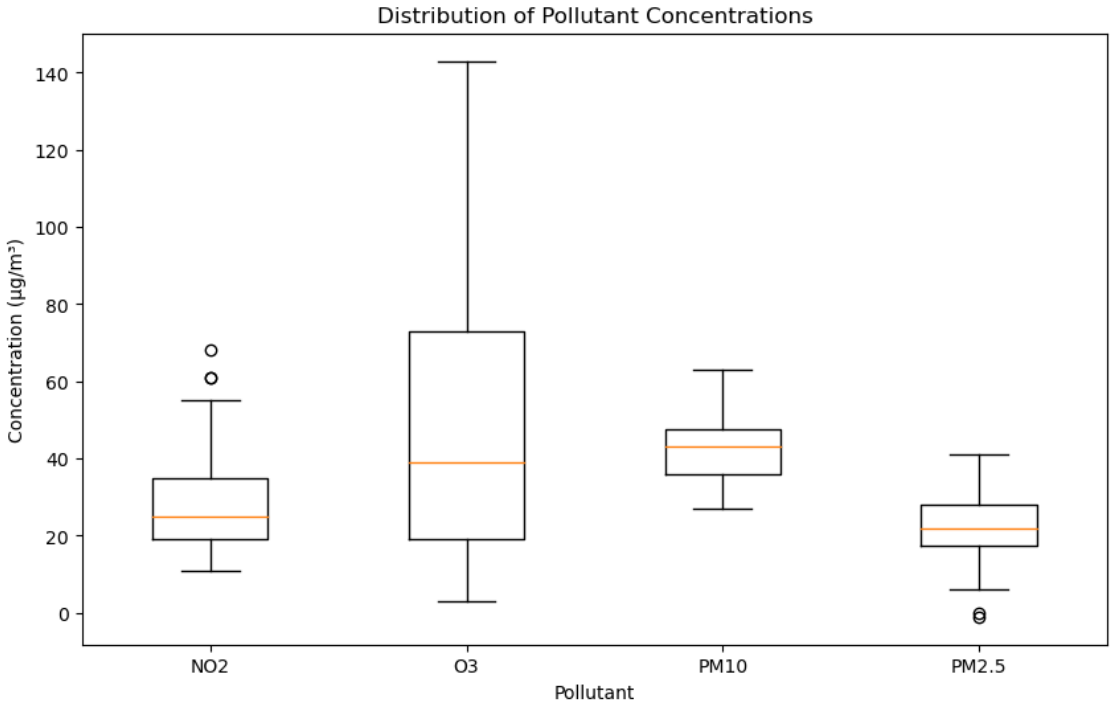


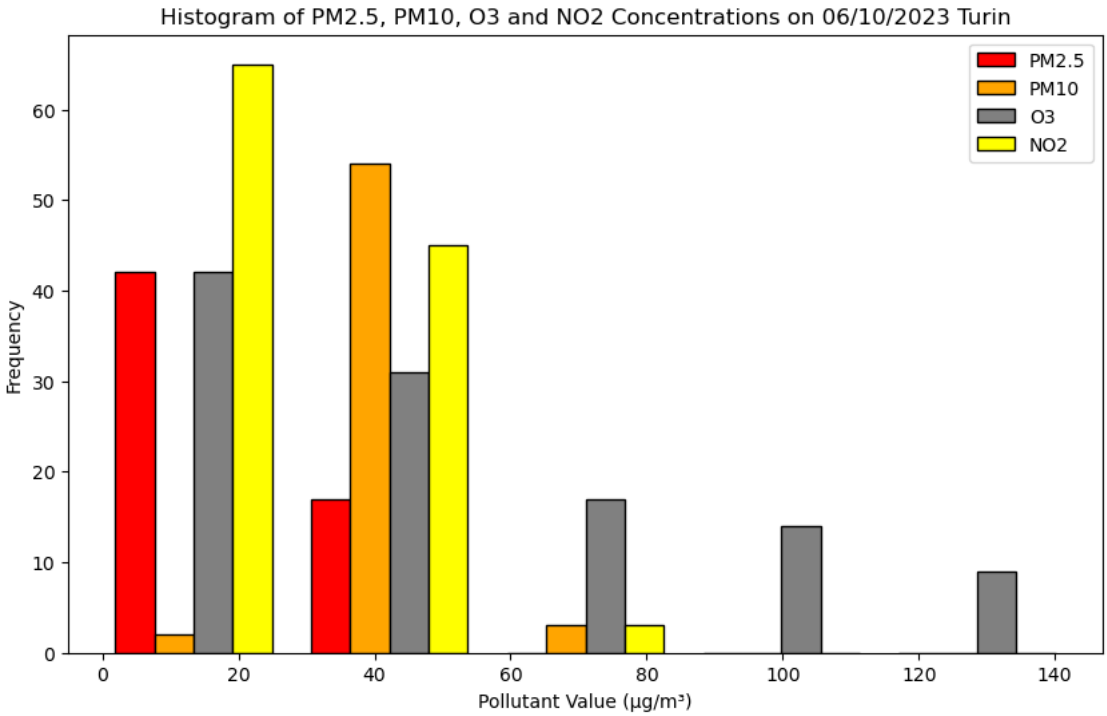
[4]:
```
# Create a scatter plot of PM10 and PM2.5
plt.figure(figsize=(10, 6))
plt.scatter(df['value'][df['parameter'] == 'pm10'], df['value'][df['parameter']␣
↪== 'pm25'], marker='o', s=50)
plt.xlabel('PM10 Concentration (µg/m³)')
plt.ylabel('PM2.5 Concentration (µg/m³)')
plt.title('Relationship between PM10 and PM2.5')
plt.show()
```



[5]:
```
# Create a box plot of all 4 pollutants
plt.figure(figsize=(10, 6))
plt.boxplot([df['value'][df['parameter'] == 'no2'], df['value'][df['parameter']␣
 ↪== 'o3'], df['value'][df['parameter'] == 'pm10'],␣
 ↪df['value'][df['parameter'] == 'pm25']], labels=['NO2', 'O3', 'PM10', 'PM2.
 ↪5'])
plt.xlabel('Pollutant')
plt.ylabel('Concentration (µg/m³)')
plt.title('Distribution of Pollutant Concentrations')
plt.show()

# Get the values of the 4 pollutants
no2 = df['value'][df['parameter'] == 'no2']
o3 = df['value'][df['parameter'] == 'o3']
pm25 = df['value'][df['parameter'] == 'pm25']
pm10 = df['value'][df['parameter'] == 'pm10']

# Create a list of the four pollutants
pollutants = ['PM2.5', 'PM10', 'O3', 'NO2']

# Create a list of the colors to use for the histogram
colors = ['red', 'orange', 'grey', 'yellow']

# Create the histogram
```

```
plt.figure(figsize=(10, 6))
plt.hist([pm25, pm10, o3, no2], bins=5, edgecolor='black', color=colors)

# Create a legend
plt.legend(pollutants, loc='upper right')

# Set the labels for the x- and y-axes
plt.xlabel('Pollutant Value (µg/m³)')
plt.ylabel('Frequency')

# Set the title of the plot
plt.title('Histogram of PM2.5, PM10, O3 and NO2 Concentrations on 06/10/2023␣
↪Turin')

# Show the plot
plt.show()
```





[6]:
```
# Calculate the average concentration of each pollutant
no2_avg = df['value'][df['parameter'] == 'no2'].mean()
o3_avg = df['value'][df['parameter'] == 'o3'].mean()
pm10_avg = df['value'][df['parameter'] == 'pm10'].mean()
pm25_avg = df['value'][df['parameter'] == 'pm25'].mean()

# Calculate the standard deviation of the concentration of each pollutant
no2_std = df['value'][df['parameter'] == 'no2'].std()
o3_std = df['value'][df['parameter'] == 'o3'].std()
pm10_std = df['value'][df['parameter'] == 'pm10'].std()
pm25_std = df['value'][df['parameter'] == 'pm25'].std()

# Create a list of pollutant names and their average and standard deviation
pollutant_info = [['NO2', no2_avg, no2_std], ['O3', o3_avg, o3_std], ['PM10',␣
↪pm10_avg, pm10_std], ['PM2.5', pm25_avg, pm25_std]]

# Print the pollutant information in a table
print('Pollutant | Average Concentration (µg/m³) | Standard Deviation (µg/m³)␣
↪|')
print('------- | ------- | ------- |')
for pollutant in pollutant_info:
    print(f'{pollutant[0]} | {pollutant[1]} | {pollutant[2]} |')
```

```
Pollutant | Average Concentration (µg/m³) | Standard Deviation (µg/m³) |
```

------- | ------- | ------- |
NO2 | 27.97345132743363 | 11.859640575882883 |
O3 | 49.610619469026545 | 37.30866273121302 |
PM10 | 42.389830508474574 | 8.287957506282817 |
PM2.5 | 22.084745762711865 | 8.783385836031218 |

[7]:
```
# Define the WHO air quality standards for the three pollutants
who_standards = {
    'no2': 25,
    'o3' : 60,
    'pm25': 5,
    'pm10': 20
}

# Check the average concentrations of PM1, PM2.5, and PM10 against the WHO air␣
 ↪quality standards
def check_pollutant_concentrations(pollutant, concentration):
    who_standard = who_standards[pollutant]
    if concentration > who_standard:
        print('Warning: The average concentration of {} exceeds the WHO air␣
 ↪quality standard of {} µg/m³.'.format(pollutant, who_standard))

# Check the average concentrations of all 4 pollutants
check_pollutant_concentrations('no2', no2_avg)
check_pollutant_concentrations('o3', o3_avg)
check_pollutant_concentrations('pm25', pm25_avg)
check_pollutant_concentrations('pm10', pm10_avg)
```

Warning: The average concentration of no2 exceeds the WHO air quality standard of 25 µg/m³.
Warning: The average concentration of pm25 exceeds the WHO air quality standard of 5 µg/m³.
Warning: The average concentration of pm10 exceeds the WHO air quality standard of 20 µg/m³.

[8]:
```
who_pm10 = 20
who_pm25 = 5
who_no2 = 25
who_o3 = 60

WHO_Diff_NO2 = no2_avg - who_no2
WHO_Diff_O3 = o3_avg - who_o3
WHO_Diff_PM10 = pm10_avg - who_pm10
WHO_Diff_PM25 = pm25_avg - who_pm25


print("NO2 exceeds WHO standards by:", WHO_Diff_NO2, "µg/m³")
```

```
print("O3 exceeds WHO standards by:", WHO_Diff_O3, "µg/m³")
print("PM10 exceeds WHO standards by:", WHO_Diff_PM10, "µg/m³")
print("PM2.5 exceeds WHO standards by:", WHO_Diff_PM25, "µg/m³")

def check_air_quality(o3_avg , no2_avg, pm25_avg, pm10_avg):
    """
    Checks the air quality based on the average concentrations of PM1, PM2.5,␣
 ↪and PM10.

    Args:
        pm1_avg: The average concentration of PM1 in µg/m³.
        pm25_avg: The average concentration of PM2.5 in µg/m³.
        pm10_avg: The average concentration of PM10 in µg/m³.

    Returns:
        A string indicating the air quality, either "Okay" or "Not Okay".
    """

    who_pm10 = 20
    who_pm25 = 5
    who_no2 = 25
    who_o3 = 60

    if (o3_avg < who_o3) and (pm25_avg < who_pm25) and (pm10_avg < who_pm10)␣
 ↪and (no2_avg < who_no2):
        return "GOOD!"
    else:
        return "BAD!!!"

NO2_Diff_Per_TO = (WHO_Diff_NO2 / who_no2)*100
O3_Diff_Per_TO = (WHO_Diff_O3 / who_o3)*100
PM10_Diff_Per_TO = (WHO_Diff_PM10 / who_pm10)*100
PM25_Diff_Per_TO = (WHO_Diff_PM25 / who_pm25)*100

NO2_Diff_Per_TO = round(NO2_Diff_Per_TO,0)
O3_Diff_Per_TO = round(O3_Diff_Per_TO,0)
PM10_Diff_Per_TO = round(PM10_Diff_Per_TO,0)
PM25_Diff_Per_TO = round(PM25_Diff_Per_TO,0)

# Check the air quality
air_quality = check_air_quality(no2_avg, o3_avg, pm25_avg, pm10_avg)

# Print the air quality
print("The air quality in Turin From 1 To 7 Oct 2023 is:", air_quality)
```

```
NO2 exceeds WHO standards by: 2.9734513274336294 µg/m³
O3 exceeds WHO standards by: -10.389380530973455 µg/m³
```

```
PM10 exceeds WHO standards by: 22.389830508474574 µg/m³
PM2.5 exceeds WHO standards by: 17.084745762711865 µg/m³
The air quality in Turin From 1 To 7 Oct 2023 is: BAD!!!
```

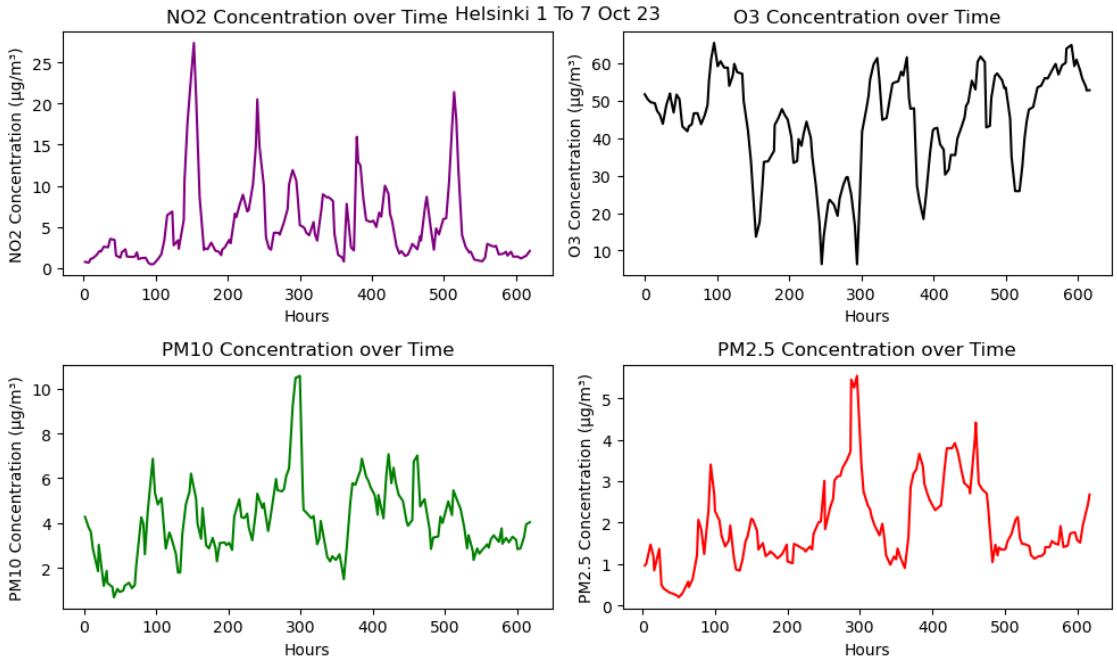
3 Helsinki Finland Air Quality From 01/10/2023 <=> 07/10/2023

[9]:
```
# Read the CSV Dataframe
df = pd.read_csv("/home/zorinosgh/Desktop/CLIMATE_CHANGE_MODELS/
 ↪Air_Quality_Helsinki_Oct_First_Week.csv")

# Get the unique values of the parameters column
parameters = df['parameter'].unique()

# Print the unique values of the parameters column
print(parameters)

# Get the data type of the values column
values_dtype = df['value'].dtype

# Print the data type of the values column
print(values_dtype)

# Get the date and time of the first recording
first_recording_datetime = df['dateLocal'].iloc[0]

# Print the date and time of the first recording
print(first_recording_datetime)
```

```
['o3' 'pm10' 'no2' 'pm25']
float64
2023-10-07T23:00:00+03:00
```

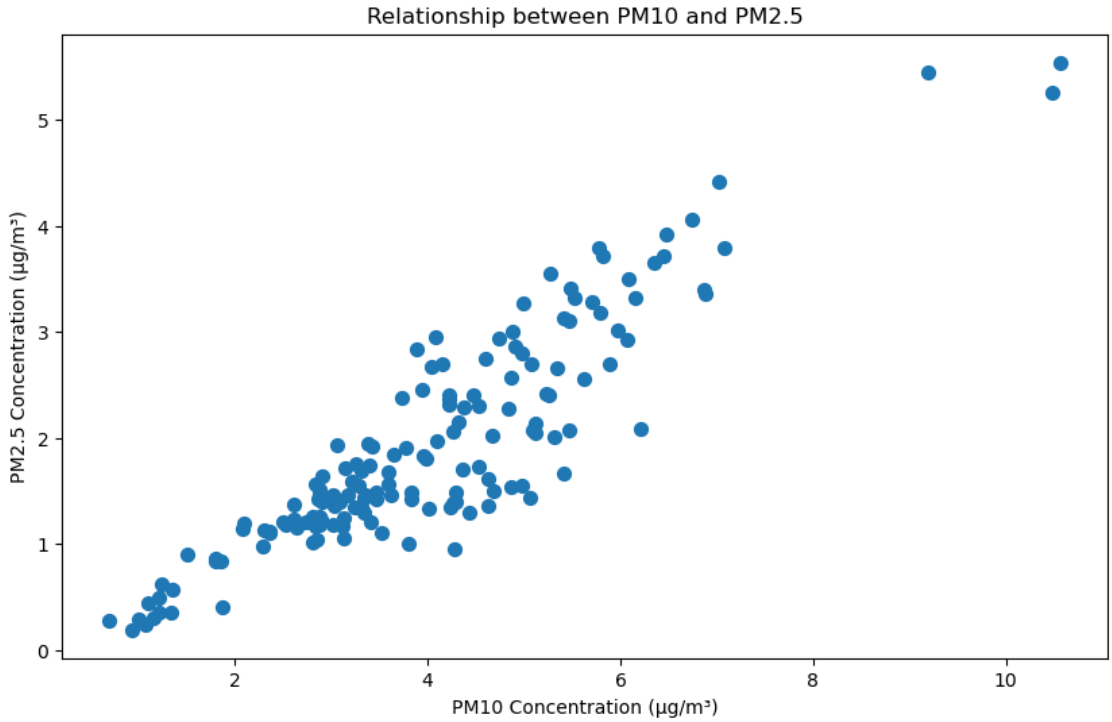
[10]:
```
# Create a figure and a set of subplots
fig, axs = plt.subplots(2, 2, figsize=(10, 6))

# Plot the NO2 concentration
axs[0, 0].plot(df['value'][df['parameter'] == 'no2'], color='purple')
axs[0, 0].set_xlabel('Hours')
axs[0, 0].set_ylabel('NO2 Concentration (µg/m³)')
axs[0, 0].set_title('NO2 Concentration over Time')

# Plot the O3 concentration
axs[0, 1].plot(df['value'][df['parameter'] == 'o3'], color='black')
axs[0, 1].set_xlabel('Hours')
axs[0, 1].set_ylabel('O3 Concentration (µg/m³)')
```

```
axs[0, 1].set_title('O3 Concentration over Time')

# Plot the PM10 concentration
axs[1, 0].plot(df['value'][df['parameter'] == 'pm10'], color='green')
axs[1, 0].set_xlabel('Hours')
axs[1, 0].set_ylabel('PM10 Concentration (µg/m³)')
axs[1, 0].set_title('PM10 Concentration over Time')

# Plot the PM2.5 concentration
axs[1, 1].plot(df['value'][df['parameter'] == 'pm25'], color='red')
axs[1, 1].set_xlabel('Hours')
axs[1, 1].set_ylabel('PM2.5 Concentration (µg/m³)')
axs[1, 1].set_title('PM2.5 Concentration over Time')

# Adjust the subplot layout
fig.tight_layout()

#Show plot title
plt.suptitle("Helsinki 1 To 7 Oct 23 ")

# Show the plot
plt.show()
```



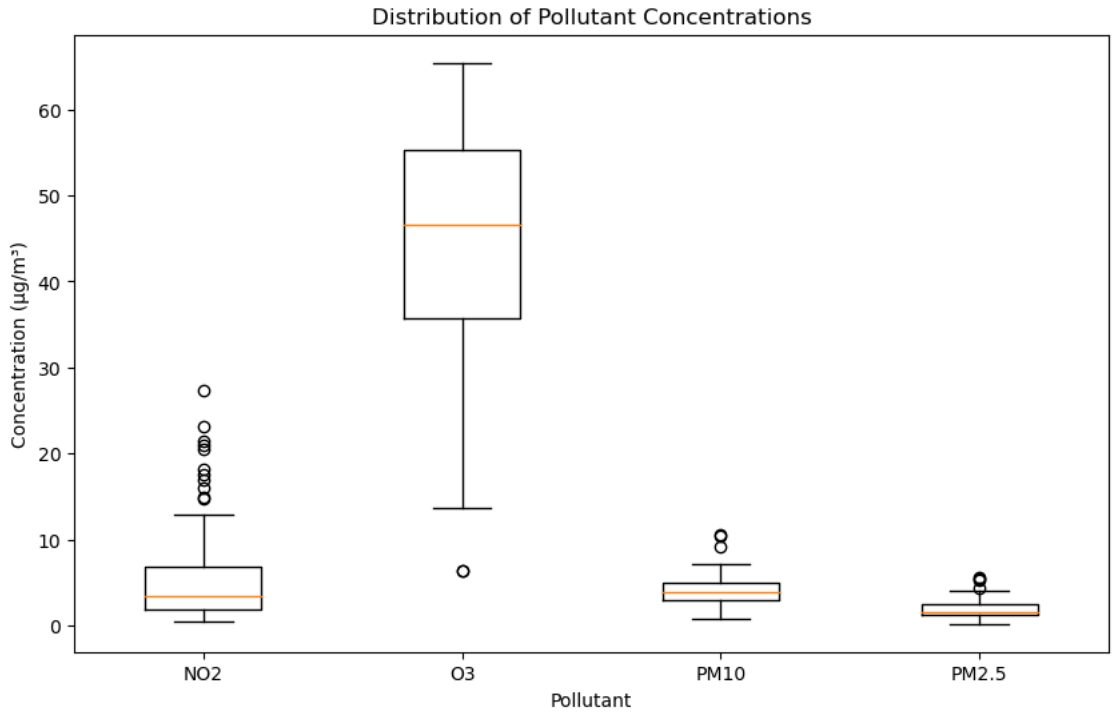
[11]:
```
# Create a scatter plot of PM10 and PM2.5
plt.figure(figsize=(10, 6))
```

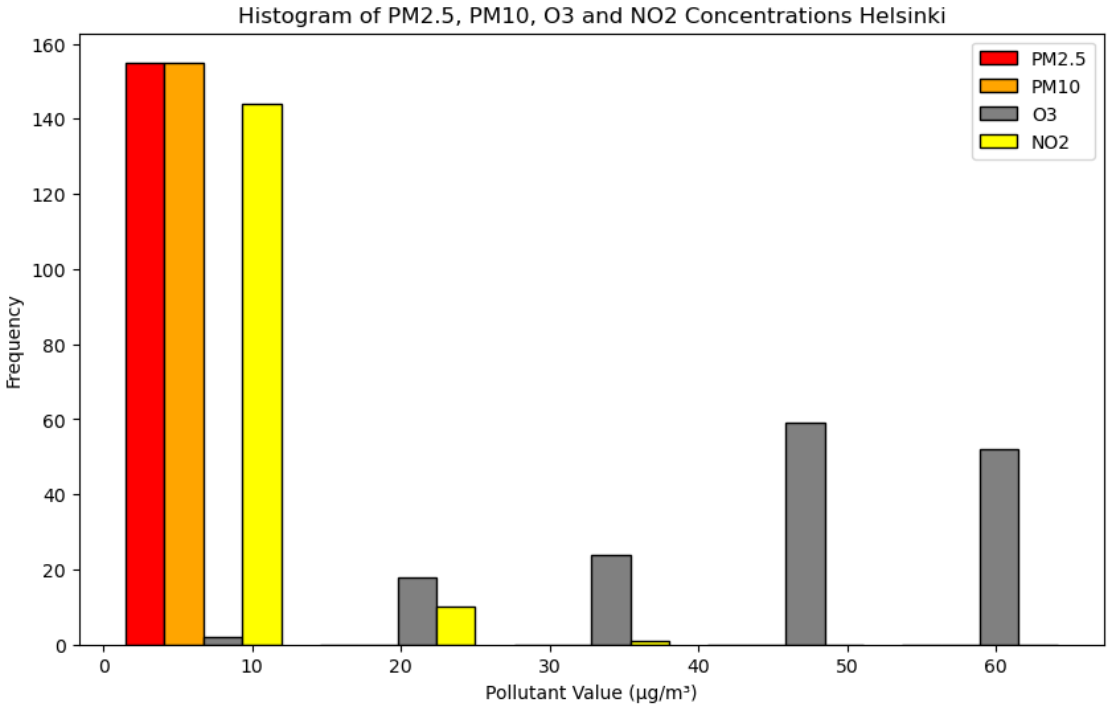
```
plt.scatter(df['value'][df['parameter'] == 'pm10'], df['value'][df['parameter']␣
 ↪== 'pm25'], marker='o', s=50)
plt.xlabel('PM10 Concentration (µg/m³)')
plt.ylabel('PM2.5 Concentration (µg/m³)')
plt.title('Relationship between PM10 and PM2.5')
plt.show()
```



[12]:
```
# Create a box plot of all 4 pollutants
plt.figure(figsize=(10, 6))
plt.boxplot([df['value'][df['parameter'] == 'no2'], df['value'][df['parameter']␣
 ↪== 'o3'], df['value'][df['parameter'] == 'pm10'],␣
 ↪df['value'][df['parameter'] == 'pm25']], labels=['NO2', 'O3', 'PM10', 'PM2.
 ↪5'])
plt.xlabel('Pollutant')
plt.ylabel('Concentration (µg/m³)')
plt.title('Distribution of Pollutant Concentrations')
plt.show()

# Get the values of the 4 pollutants
no2 = df['value'][df['parameter'] == 'no2']
o3 = df['value'][df['parameter'] == 'o3']
pm25 = df['value'][df['parameter'] == 'pm25']
pm10 = df['value'][df['parameter'] == 'pm10']
```

```
# Create a list of the four pollutants
pollutants = ['PM2.5', 'PM10', 'O3', 'NO2']

# Create a list of the colors to use for the histogram
colors = ['red', 'orange', 'grey', 'yellow']

# Create the histogram
plt.figure(figsize=(10, 6))
plt.hist([pm25, pm10, o3, no2], bins=5, edgecolor='black', color=colors)

# Create a legend
plt.legend(pollutants, loc='upper right')

# Set the labels for the x- and y-axes
plt.xlabel('Pollutant Value (µg/m³)')
plt.ylabel('Frequency')

# Set the title of the plot
plt.title('Histogram of PM2.5, PM10, O3 and NO2 Concentrations Helsinki')

# Show the plot
plt.show()
```





[13]:
```
# Calculate the average concentration of each pollutant
no2_avg = df['value'][df['parameter'] == 'no2'].mean()
o3_avg = df['value'][df['parameter'] == 'o3'].mean()
pm10_avg = df['value'][df['parameter'] == 'pm10'].mean()
pm25_avg = df['value'][df['parameter'] == 'pm25'].mean()

# Calculate the standard deviation of the concentration of each pollutant
no2_std = df['value'][df['parameter'] == 'no2'].std()
o3_std = df['value'][df['parameter'] == 'o3'].std()
pm10_std = df['value'][df['parameter'] == 'pm10'].std()
pm25_std = df['value'][df['parameter'] == 'pm25'].std()

# Create a list of pollutant names and their average and standard deviation
pollutant_info = [['NO2', no2_avg, no2_std], ['O3', o3_avg, o3_std], ['PM10',␣
 ↪pm10_avg, pm10_std], ['PM2.5', pm25_avg, pm25_std]]

# Print the pollutant information in a table
print('Pollutant | Average Concentration (µg/m³) | Standard Deviation (µg/m³)␣
 ↪|')
print('------- | ------- | ------- |')
for pollutant in pollutant_info:
    print(f'{pollutant[0]} | {pollutant[1]} | {pollutant[2]} |')
```

```
Pollutant | Average Concentration (µg/m³) | Standard Deviation (µg/m³) |
```

```
------- | ------- | ------- |
NO2 | 5.169032818064516 | 4.981478366750061 |
O3 | 44.47634996774194 | 13.131068841902279 |
PM10 | 3.9844512941935486 | 1.6782884493422237 |
PM2.5 | 1.9222977658064517 | 1.034627247419987 |
```

[14]:
```
# Define the WHO air quality standards for the three pollutants
who_standards = {
    'no2': 25,
    'o3' : 60,
    'pm25': 5,
    'pm10': 20
}

# Check the average concentrations of PM1, PM2.5, and PM10 against the WHO air␣
 ↪quality standards
def check_pollutant_concentrations(pollutant, concentration):
    who_standard = who_standards[pollutant]
    if concentration > who_standard:
        print('Warning: The average concentration of {} exceeds the WHO air␣
 ↪quality standard of {} µg/m³.'.format(pollutant, who_standard))

# Check the average concentrations of all 4 pollutants
check_pollutant_concentrations('no2', no2_avg)
check_pollutant_concentrations('o3', o3_avg)
check_pollutant_concentrations('pm25', pm25_avg)
check_pollutant_concentrations('pm10', pm10_avg)
```

[15]:
```
who_pm10 = 20
who_pm25 = 5
who_no2 = 25
who_o3 = 60

WHO_Diff_NO2 = no2_avg - who_no2
WHO_Diff_O3 = o3_avg - who_o3
WHO_Diff_PM10 = pm10_avg - who_pm10
WHO_Diff_PM25 = pm25_avg - who_pm25


print("NO2 exceeds WHO standards by:", WHO_Diff_NO2, "µg/m³")
print("O3 exceeds WHO standards by:", WHO_Diff_O3, "µg/m³")
print("PM10 exceeds WHO standards by:", WHO_Diff_PM10, "µg/m³")
print("PM2.5 exceeds WHO standards by:", WHO_Diff_PM25, "µg/m³")

def check_air_quality(o3_avg , no2_avg, pm25_avg, pm10_avg):
    """
```

```
    Checks the air quality based on the average concentrations of PM1, PM2.5,␣
 ↪and PM10.

    Args:
        pm1_avg: The average concentration of PM1 in µg/m³.
        pm25_avg: The average concentration of PM2.5 in µg/m³.
        pm10_avg: The average concentration of PM10 in µg/m³.

    Returns:
        A string indicating the air quality, either "Okay" or "Not Okay".
    """

    who_pm10 = 20
    who_pm25 = 5
    who_no2 = 25
    who_o3 = 60

    if (o3_avg > who_o3) and (pm25_avg > who_pm25) and (pm10_avg > who_pm10)␣
 ↪and (no2_avg > who_no2):
        return "BAD!!!"
    else:
        return "GOOD!"

NO2_Diff_Per_HEL = (WHO_Diff_NO2 / who_no2)*100
O3_Diff_Per_HEL = (WHO_Diff_O3 / who_o3)*100
PM10_Diff_Per_HEL = (WHO_Diff_PM10 / who_pm10)*100
PM25_Diff_Per_HEL = (WHO_Diff_PM25 / who_pm25)*100

NO2_Diff_Per_HEL = round(NO2_Diff_Per_HEL,0)
O3_Diff_Per_HEL = round(O3_Diff_Per_HEL,0)
PM10_Diff_Per_HEL = round(PM10_Diff_Per_HEL,0)
PM25_Diff_Per_HEL = round(PM25_Diff_Per_HEL,0)

# Check the air quality
air_quality = check_air_quality(no2_avg, o3_avg, pm25_avg, pm10_avg)

# Print the air quality
print("The air quality in Helsinki From 1 To 7 Oct 2023 is:", air_quality)
```

```
NO2 exceeds WHO standards by: -19.830967181935485 µg/m³
O3 exceeds WHO standards by: -15.523650032258061 µg/m³
PM10 exceeds WHO standards by: -16.015548705806452 µg/m³
PM2.5 exceeds WHO standards by: -3.0777022341935485 µg/m³
The air quality in Helsinki From 1 To 7 Oct 2023 is: GOOD!
```

[16]:

```
print("The Difference Between the WHO Standards & Average Concentration for NO2␣
 ↪from 01 - 07 Oct 2023 is", NO2_Diff_Per_TO,"%", "For Turin", "While",␣
 ↪NO2_Diff_Per_HEL, "%", "For Helsiniki")
```

```
The Difference Between the WHO Standards & Average Concentration for NO2 from 01
- 07 Oct 2023 is 12.0 % For Turin While -79.0 % For Helsiniki
```

[17]:
```
print("The Difference Between the WHO Standards & Average Concentration for␣
 ↪Ozone from 01 - 07 Oct 2023 is", O3_Diff_Per_TO,"%", "For Turin", "While",␣
 ↪O3_Diff_Per_HEL, "%", "For Helsiniki")
```

```
The Difference Between the WHO Standards & Average Concentration for Ozone from
01 - 07 Oct 2023 is -17.0 % For Turin While -26.0 % For Helsiniki
```

[18]:
```
print("The Difference Between the WHO Standards & Average Concentration for␣
 ↪PM10 from 01 - 07 Oct 2023 is", PM10_Diff_Per_TO,"%", "For Turin", "While",␣
 ↪PM10_Diff_Per_HEL, "%", "For Helsiniki")
```

```
The Difference Between the WHO Standards & Average Concentration for PM10 from
01 - 07 Oct 2023 is 112.0 % For Turin While -80.0 % For Helsiniki
```

[19]:
```
print("The Difference Between the WHO Standards & Average Concentration for PM2.
 ↪5 from 01 - 07 Oct 2023 is", PM25_Diff_Per_TO,"%", "For Turin", "While",␣
 ↪PM25_Diff_Per_HEL, "%", "For Helsiniki")
```

```
The Difference Between the WHO Standards & Average Concentration for PM2.5 from
01 - 07 Oct 2023 is 342.0 % For Turin While -62.0 % For Helsiniki
```

[ ]:
```

```