Air quality index

In [1]:
```
import pandas as pd

# Not limiting the column number when displaying dataframe
pd.set_option("display.max_columns", None)
```

In [2]:
```
df = pd.read_csv("C:/Users/nisho/Documents/SEM 5/ML and core applications/Air_Quality.c
df.head()
```

Out[2]:

	id	country	state	city	station	pollutant_id	last_update	pollutant_min	pollutan
0	1	India	Andhra_Pradesh	Amaravati	Secretariat, Amaravati - APPCB	PM2.5	21-10-2021 01:00:00	69.0	
1	2	India	Andhra_Pradesh	Amaravati	Secretariat, Amaravati - APPCB	PM10	21-10-2021 01:00:00	82.0	
2	3	India	Andhra_Pradesh	Amaravati	Secretariat, Amaravati - APPCB	NO2	21-10-2021 01:00:00	10.0	
3	4	India	Andhra_Pradesh	Amaravati	Secretariat, Amaravati - APPCB	NH3	21-10-2021 01:00:00	4.0	
4	5	India	Andhra_Pradesh	Amaravati	Secretariat, Amaravati - APPCB	SO2	21-10-2021 01:00:00	16.0	

In [3]:
```
df.tail()
```

Out[3]:

	id	country	state	city	station	pollutant_id	last_update	pollutant_min	pollutant_
1831	1832	India	West_Bengal	Kolkata	Victoria, Kolkata - WBPCB	NO2	21-10-2021 01:00:00	10.0	
1832	1833	India	West_Bengal	Kolkata	Victoria, Kolkata - WBPCB	NH3	21-10-2021 01:00:00	1.0	
1833	1834	India	West_Bengal	Kolkata	Victoria, Kolkata - WBPCB	SO2	21-10-2021 01:00:00	6.0	

	id	country	state	city	station	pollutant_id	last_update	pollutant_min	pollutant_
1834	1835	India	West_Bengal	Kolkata	Victoria, Kolkata - WBPCB	CO	21-10-2021 01:00:00	34.0	
1835	1836	India	West_Bengal	Kolkata	Victoria, Kolkata - WBPCB	OZONE	21-10-2021 01:00:00	10.0	1

In [4]:
```python
print(df.columns)
```
```
Index(['id', 'country', 'state', 'city', 'station', 'pollutant_id',
       'last_update', 'pollutant_min', 'pollutant_max', 'pollutant_avg'],
      dtype='object')
```

In [5]:
```python
rows = df.shape[0]
cols = df.shape[1]

print("Before cleaning, there are " + str(rows) + " rows and " + str(cols) + " columns
```
```
Before cleaning, there are 1836 rows and 10 columns in this dataframe.
```

In [6]:
```python
dupRows = df.duplicated().sum()
print("There are " + str(dupRows) + " duplicated rows in the dataframe.")
```
```
There are 0 duplicated rows in the dataframe.
```

In [7]:
```python
df.isnull().sum()
```
Out[7]:
```
id                0
country           0
state             0
city              0
station           0
pollutant_id      0
last_update       0
pollutant_min    98
pollutant_max    98
pollutant_avg    98
dtype: int64
```

In [8]:
```python
df.nunique()
```
Out[8]:
```
id               1836
country             1
state              26
city              142
station           281
pollutant_id        7
last_update         1
pollutant_min     149
pollutant_max     340
```

```
pollutant_avg     237
dtype: int64
```

In [9]:

```
df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 1836 entries, 0 to 1835
Data columns (total 10 columns):
 #   Column         Non-Null Count  Dtype  
---  ------         --------------  -----  
 0   id             1836 non-null   int64  
 1   country        1836 non-null   object 
 2   state          1836 non-null   object 
 3   city           1836 non-null   object 
 4   station        1836 non-null   object 
 5   pollutant_id   1836 non-null   object 
 6   last_update    1836 non-null   object 
 7   pollutant_min  1738 non-null   float64
 8   pollutant_max  1738 non-null   float64
 9   pollutant_avg  1738 non-null   float64
dtypes: float64(3), int64(1), object(6)
memory usage: 143.6+ KB
```

In [10]:

```
df.dtypes.value_counts()
```

Out[10]:

```
object     6
float64    3
int64      1
dtype: int64
```

In [11]:

```
df.describe()
```

Out[11]:

	id	pollutant_min	pollutant_max	pollutant_avg
count	1836.000000	1738.000000	1738.000000	1738.000000
mean	918.500000	28.414269	96.873418	54.100690
std	530.151865	34.403811	104.765094	60.824158
min	1.000000	1.000000	1.000000	1.000000
25%	459.750000	5.000000	21.000000	12.000000
50%	918.500000	14.000000	63.000000	31.000000
75%	1377.250000	39.000000	124.000000	70.000000
max	1836.000000	217.000000	500.000000	314.000000

In [12]:

```
df.memory_usage()
```

Out[12]:

```
Index              128
id               14688
country          14688
state            14688
city             14688
station          14688
```

```
pollutant_id      14688
last_update       14688
pollutant_min     14688
pollutant_max     14688
pollutant_avg     14688
dtype: int64
```

In [13]:

```
df.corr()
```

Out[13]:

	id	pollutant_min	pollutant_max	pollutant_avg
id	1.000000	-0.038355	-0.034367	-0.023175
pollutant_min	-0.038355	1.000000	0.788666	0.892249
pollutant_max	-0.034367	0.788666	1.000000	0.935664
pollutant_avg	-0.023175	0.892249	0.935664	1.000000

In [14]:

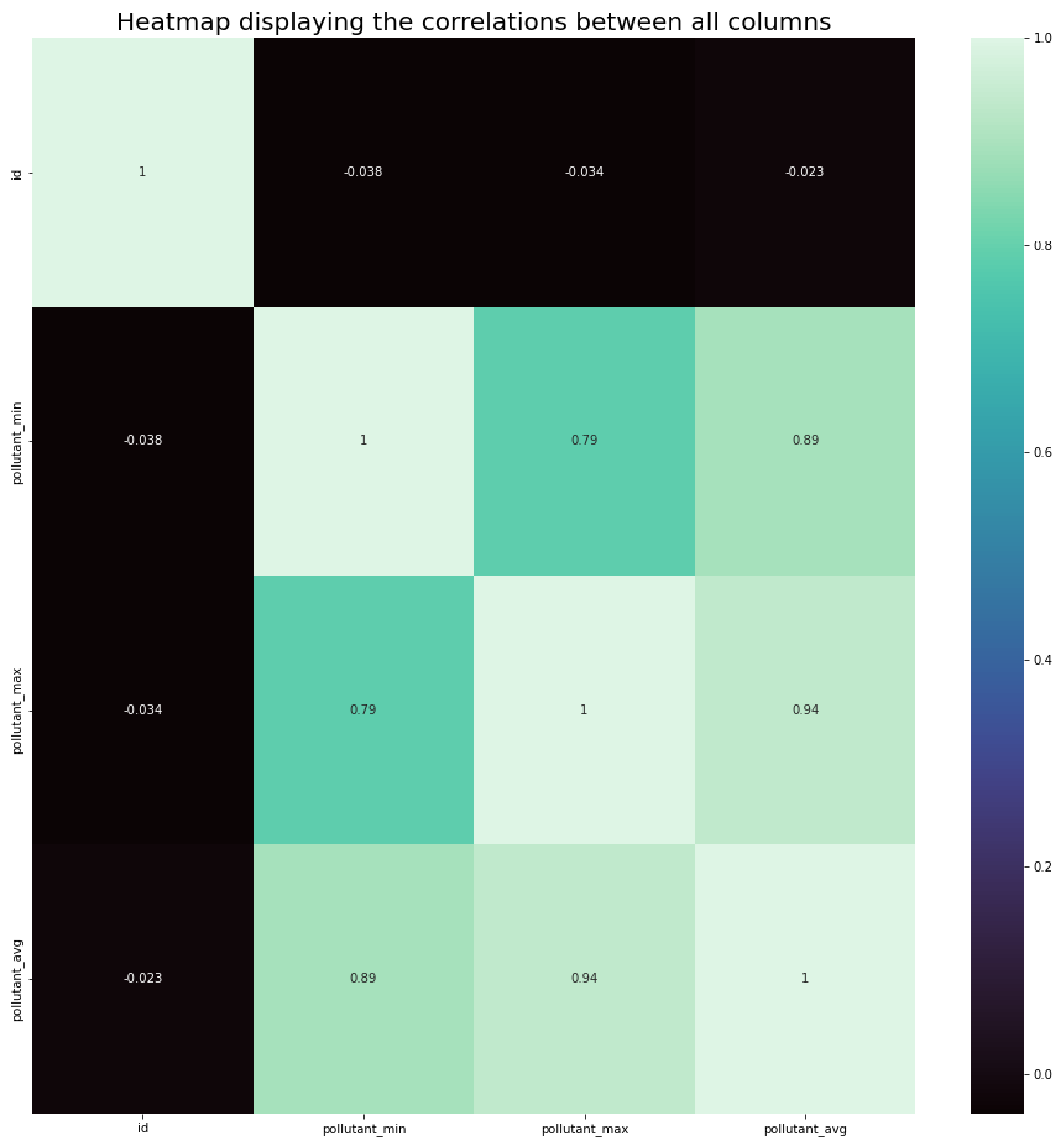
```
import seaborn as sns
import matplotlib.pyplot as plt

correlations = df.corr()

plt.figure(figsize = (16, 16))
plt.title("Heatmap displaying the correlations between all columns", fontsize = 20)
sns.heatmap(correlations, annot = True, cmap = "mako")
```

Out[14]:

```
<AxesSubplot:title={'center':'Heatmap displaying the correlations between all columns'}>
```



In [15]:
```
poll = df.corr()["pollutant_avg"]
poll = pd.DataFrame(poll)
poll
```

Out[15]:

	pollutant_avg
id	-0.023175
pollutant_min	0.892249
pollutant_max	0.935664
pollutant_avg	1.000000

In [16]:
```
prices = df.value_counts(["pollutant_avg"])
```

```
prices
```

Out[16]:

```
pollutant_avg
5.0            52
4.0            50
6.0            46
2.0            46
12.0           40
               ..
110.0           1
163.0           1
216.0           1
218.0           1
314.0           1
Length: 237, dtype: int64
```

In [29]:

```
import pandas_profiling
from pandas_profiling import ProfileReport

profile = ProfileReport(df, title = "Pandas Profiling Report", explorative = True)

profile.to_file("your_report.html")
```

In [19]:

```
df.head()
```

Out[19]:

	id	country	state	city	station	pollutant_id	last_update	pollutant_min	pollutan
0	1	India	Andhra_Pradesh	Amaravati	Secretariat, Amaravati - APPCB	PM2.5	21-10-2021 01:00:00	69.0	
1	2	India	Andhra_Pradesh	Amaravati	Secretariat, Amaravati - APPCB	PM10	21-10-2021 01:00:00	82.0	
2	3	India	Andhra_Pradesh	Amaravati	Secretariat, Amaravati - APPCB	NO2	21-10-2021 01:00:00	10.0	
3	4	India	Andhra_Pradesh	Amaravati	Secretariat, Amaravati - APPCB	NH3	21-10-2021 01:00:00	4.0	
4	5	India	Andhra_Pradesh	Amaravati	Secretariat, Amaravati - APPCB	SO2	21-10-2021 01:00:00	16.0	

In [20]:

```
del df["last_update"]

df["country"] = df["country"].astype(str)
df["state"] = df["state"].astype(str)
df["city"] = df["city"].astype(str)
df["station"] = df["station"].astype(str)
```

```python
from sklearn import preprocessing

number = preprocessing.LabelEncoder()

df["country"] = number.fit_transform(df["country"])
df["state"] = number.fit_transform(df["state"])
df["city"] = number.fit_transform(df["city"])
df["station"] = number.fit_transform(df["station"])

df.head()
```

Out[20]:

	id	country	state	city	station	pollutant_id	pollutant_min	pollutant_max	pollutant_avg
0	1	0	0	6	215	PM2.5	69.0	109.0	86.0
1	2	0	0	6	215	PM10	82.0	138.0	105.0
2	3	0	0	6	215	NO2	10.0	42.0	19.0
3	4	0	0	6	215	NH3	4.0	5.0	4.0
4	5	0	0	6	215	SO2	16.0	42.0	27.0

In [21]:

```python
df.shape
```

Out[21]:

```
(1836, 9)
```

In [22]:

```python
df = df.dropna()

df.shape
```

Out[22]:

```
(1738, 9)
```

In [23]:

```python
df = pd.get_dummies(df)
df.head()
```

Out[23]:

	id	country	state	city	station	pollutant_min	pollutant_max	pollutant_avg	pollutant_id_CO	pollu
0	1	0	0	6	215	69.0	109.0	86.0	0	
1	2	0	0	6	215	82.0	138.0	105.0	0	
2	3	0	0	6	215	10.0	42.0	19.0	0	
3	4	0	0	6	215	4.0	5.0	4.0	0	
4	5	0	0	6	215	16.0	42.0	27.0	0	

In [24]:

```python
df.shape
```

Out[24]:

```
(1738, 15)
```

In [25]:

```
X = df.drop(["pollutant_avg"], axis = 1).values
y = df["pollutant_avg"].values
```

In [26]:
```
from sklearn.model_selection import train_test_split

X_train, X_test, y_train, y_test = train_test_split(X, y, random_state = 0, test_size =
```

In [27]:
```
from sklearn.linear_model import LinearRegression

model = LinearRegression()

model.fit(X_train, y_train)
```

Out[27]:
```
LinearRegression()
```

In [28]:
```
model.score(X_test, y_test)
```

Out[28]:
```
0.9340928594638
```

In [ ]: