air quality index predication

```
[174] import numpy as np
      import matplotlib.pyplot as plt
      import pandas as pd
      import seaborn as sns
      from sklearn.metrics import classification_report
      from sklearn import metrics
      from sklearn import tree
```

```
[105] df=pd.read_csv('city_day.csv',na_values='=')
```

```
[106] df
```

```
[106] df
```

	City	Date	PM2.5	PM10	NO	NO2	NOx	NH3	CO	SO2	O3	Benzene	Toluene	Xylene	AQI	AQI_Bucket
0	Ahmedabad	2015-01-01	NaN	NaN	0.92	18.22	17.15	NaN	0.92	27.64	133.36	0.00	0.02	0.00	NaN	NaN
1	Ahmedabad	2015-01-02	NaN	NaN	0.97	15.69	16.46	NaN	0.97	24.55	34.06	3.68	5.50	3.77	NaN	NaN
2	Ahmedabad	2015-01-03	NaN	NaN	17.40	19.30	29.70	NaN	17.40	29.07	30.70	6.80	16.40	2.25	NaN	NaN
3	Ahmedabad	2015-01-04	NaN	NaN	1.70	18.48	17.97	NaN	1.70	18.59	36.08	4.43	10.14	1.00	NaN	NaN
4	Ahmedabad	2015-01-05	NaN	NaN	22.10	21.42	37.76	NaN	22.10	39.33	39.31	7.01	18.89	2.78	NaN	NaN
...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...
29526	Visakhapatnam	2020-06-27	15.02	50.94	7.68	25.06	19.54	12.47	0.47	8.55	23.30	2.24	12.07	0.73	41.0	Good
29527	Visakhapatnam	2020-06-28	24.38	74.09	3.42	26.06	16.53	11.99	0.52	12.72	30.14	0.74	2.21	0.38	70.0	Satisfactory
29528	Visakhapatnam	2020-06-29	22.91	65.73	3.45	29.53	18.33	10.71	0.48	8.42	30.96	0.01	0.01	0.00	68.0	Satisfactory
29529	Visakhapatnam	2020-06-30	16.64	49.97	4.05	29.26	18.80	10.03	0.52	9.84	28.30	0.00	0.00	0.00	54.0	Satisfactory
29530	Visakhapatnam	2020-07-01	15.00	66.00	0.40	26.85	14.05	5.20	0.59	2.10	17.05	NaN	NaN	NaN	50.0	Good

29531 rows × 16 columns

```
[107] df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 29531 entries, 0 to 29530
Data columns (total 16 columns):
 #   Column      Non-Null Count  Dtype  
---  ------      --------------  -----  
 0   City        29531 non-null  object 
 1   Date        29531 non-null  object 
 2   PM2.5       24933 non-null  float64
 3   PM10        18391 non-null  float64
 4   NO          25949 non-null  float64
 5   NO2         25946 non-null  float64
 6   NOx         25346 non-null  float64
 7   NH3         19203 non-null  float64
 8   CO          27472 non-null  float64
 9   SO2         25677 non-null  float64
 10  O3          25509 non-null  float64
 11  Benzene     23908 non-null  float64
 12  Toluene     21490 non-null  float64
 13  Xylene      11422 non-null  float64
 14  AQI         24850 non-null  float64
 15  AQI_Bucket  24850 non-null  object 
dtypes: float64(13), object(3)
memory usage: 3.6+ MB
```

```
[108] df.isnull().sum()
```

```
City             0
Date             0
PM2.5         4598
PM10         11140
NO            3582
NO2           3585
NOx           4185
NH3          10328
CO            2059
SO2           3854
O3            4022
Benzene       5623
Toluene       8041
Xylene       18109
AQI           4681
AQI_Bucket    4681
dtype: int64
```

```
df.head(6)
```

	City	Date	PM2.5	PM10	NO	NO2	NOx	NH3	CO	SO2	O3	Benzene	Toluene	Xylene	AQI	AQI_Bucket
0	Ahmedabad	2015-01-01	NaN	NaN	0.92	18.22	17.15	NaN	0.92	27.64	133.36	0.00	0.02	0.00	NaN	NaN
1	Ahmedabad	2015-01-02	NaN	NaN	0.97	15.69	16.46	NaN	0.97	24.55	34.06	3.68	5.50	3.77	NaN	NaN
2	Ahmedabad	2015-01-03	NaN	NaN	17.40	19.30	29.70	NaN	17.40	29.07	30.70	6.80	16.40	2.25	NaN	NaN
3	Ahmedabad	2015-01-04	NaN	NaN	1.70	18.48	17.97	NaN	1.70	18.59	36.08	4.43	10.14	1.00	NaN	NaN
4	Ahmedabad	2015-01-05	NaN	NaN	22.10	21.42	37.76	NaN	22.10	39.33	39.31	7.01	18.89	2.78	NaN	NaN
5	Ahmedabad	2015-01-06	NaN	NaN	45.41	38.48	81.50	NaN	45.41	45.76	46.51	5.42	10.83	1.93	NaN	NaN

[143] df.columns

```
Index(['City', 'Date', 'PM2.5', 'PM10', 'NO', 'NO2', 'NOx', 'NH3', 'CO', 'SO2',
       'O3', 'Benzene', 'Toluene', 'Xylene', 'AQI', 'AQI_Bucket'],
      dtype='object')
```

[144] data2=df.copy()

replace null values with mean

[145] data2=data2.fillna(data2.mean())

```
<ipython-input-145-d18b7429a1bb>:1: FutureWarning:

The default value of numeric_only in DataFrame.mean is deprecated. In a future version, it will default to False. In addition, specifying 'numeric_only=None' is dep
```

[146] data2.head()

	City	Date	PM2.5	PM10	NO	NO2	NOx	NH3	CO	SO2	O3	Benzene	Toluene	Xylene	AQI	AQI_Bucket
0	Ahmedabad	2015-01-01	67.450578	118.127103	0.92	18.22	17.15	23.483476	0.92	27.64	133.36	0.00	0.02	0.00	166.463581	NaN
1	Ahmedabad	2015-01-02	67.450578	118.127103	0.97	15.69	16.46	23.483476	0.97	24.55	34.06	3.68	5.50	3.77	166.463581	NaN
2	Ahmedabad	2015-01-03	67.450578	118.127103	17.40	19.30	29.70	23.483476	17.40	29.07	30.70	6.80	16.40	2.25	166.463581	NaN
3	Ahmedabad	2015-01-04	67.450578	118.127103	1.70	18.48	17.97	23.483476	1.70	18.59	36.08	4.43	10.14	1.00	166.463581	NaN
4	Ahmedabad	2015-01-05	67.450578	118.127103	22.10	21.42	37.76	23.483476	22.10	39.33	39.31	7.01	18.89	2.78	166.463581	NaN

mapping

```
[148] dist=(data2['City'])
      distset=set(dist)
      dd=list(distset)
      dictofWords={ dd[i] : i for i in range (0, len(dd))}
      data2['City']=data2['City'].map(dictofWords)
```

```
      dist=(data2['AQI_Bucket'])
      distset=set(dist)
      dd=list(distset)
      dictofWords={ dd[i] : i for i in range (0, len(dd))}
      data2['AQI_Bucket']=data2['AQI_Bucket'].map(dictofWords)
```

```
[150] data2["AQI_Bucket"]=data2["AQI_Bucket"].fillna(data2["AQI_Bucket"].mean())
```

```
[151] data2
```

	City	Date	PM2.5	PM10	NO	NO2	NOx	NH3	CO	SO2	O3	Benzene	Toluene	Xylene	AQI	AQI_Bucket
0	21	2015-01-01	67.450578	118.127103	0.92	18.22	17.15	23.483476	0.92	27.64	133.36	0.00000	0.020000	0.000000	166.463581	5
1	21	2015-01-02	67.450578	118.127103	0.97	15.69	16.46	23.483476	0.97	24.55	34.06	3.68000	5.500000	3.770000	166.463581	5
2	21	2015-01-03	67.450578	118.127103	17.40	19.30	29.70	23.483476	17.40	29.07	30.70	6.80000	16.400000	2.250000	166.463581	5
3	21	2015-01-04	67.450578	118.127103	1.70	18.48	17.97	23.483476	1.70	18.59	36.08	4.43000	10.140000	1.000000	166.463581	5
4	21	2015-01-05	67.450578	118.127103	22.10	21.42	37.76	23.483476	22.10	39.33	39.31	7.01000	18.890000	2.780000	166.463581	5
...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...
29526	3	2020-06-27	15.020000	50.940000	7.68	25.06	19.54	12.470000	0.47	8.55	23.30	2.24000	12.070000	0.730000	41.000000	2
29527	3	2020-06-28	24.380000	74.090000	3.42	26.06	16.53	11.990000	0.52	12.72	30.14	0.74000	2.210000	0.380000	70.000000	1
29528	3	2020-06-29	22.910000	65.730000	3.45	29.53	18.33	10.710000	0.48	8.42	30.96	0.01000	0.010000	0.000000	68.000000	1
29529	3	2020-06-30	16.640000	49.970000	4.05	29.26	18.80	10.030000	0.52	9.84	28.30	0.00000	0.000000	0.000000	54.000000	1

✓ 0s [152] `data2.isnull().sum()`

```
City          0
Date          0
PM2.5         0
PM10          0
NO            0
NO2           0
NOx           0
NH3           0
CO            0
SO2           0
O3            0
Benzene       0
Toluene       0
Xylene        0
AQI           0
AQI_Bucket    0
dtype: int64
```

✓ 0s [153] `data2`

	City	Date	PM2.5	PM10	NO	NO2	NOx	NH3	CO	SO2	O3	Benzene	Toluene	Xylene	AQI	AQI_Bucket
0	21	2015-01-01	67.450578	118.127103	0.92	18.22	17.15	23.483476	0.92	27.64	133.36	0.00000	0.020000	0.000000	166.463581	5
1	21	2015-01-02	67.450578	118.127103	0.97	15.69	16.46	23.483476	0.97	24.55	34.06	3.68000	5.500000	3.770000	166.463581	5
2	21	2015-01-03	67.450578	118.127103	17.40	19.30	29.70	23.483476	17.40	29.07	30.70	6.80000	16.400000	2.250000	166.463581	5
3	21	2015-01-04	67.450578	118.127103	1.70	18.48	17.97	23.483476	1.70	18.59	36.08	4.43000	10.140000	1.000000	166.463581	5
4	21	2015-01-05	67.450578	118.127103	22.10	21.42	37.76	23.483476	22.10	39.33	39.31	7.01000	18.890000	2.780000	166.463581	5
...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...

```
[154] data2=data2.drop('Date',1)
```

```
<ipython-input-154-8ab6c7f675ab>:1: FutureWarning:

In a future version of pandas all arguments of DataFrame.drop except for the argument 'labels' will be keyword-only.
```

```
[155] data2.columns
```

```
Index(['City', 'PM2.5', 'PM10', 'NO', 'NO2', 'NOx', 'NH3', 'CO', 'SO2', 'O3',
       'Benzene', 'Toluene', 'Xylene', 'AQI', 'AQI_Bucket'],
      dtype='object')
```

```
data2=data2.drop('AQI_Bucket', 1)
```

```
<ipython-input-156-72484e9c5a44>:1: FutureWarning:

In a future version of pandas all arguments of DataFrame.drop except for the argument 'labels' will be keyword-only.
```

```
[157] data2.columns
```

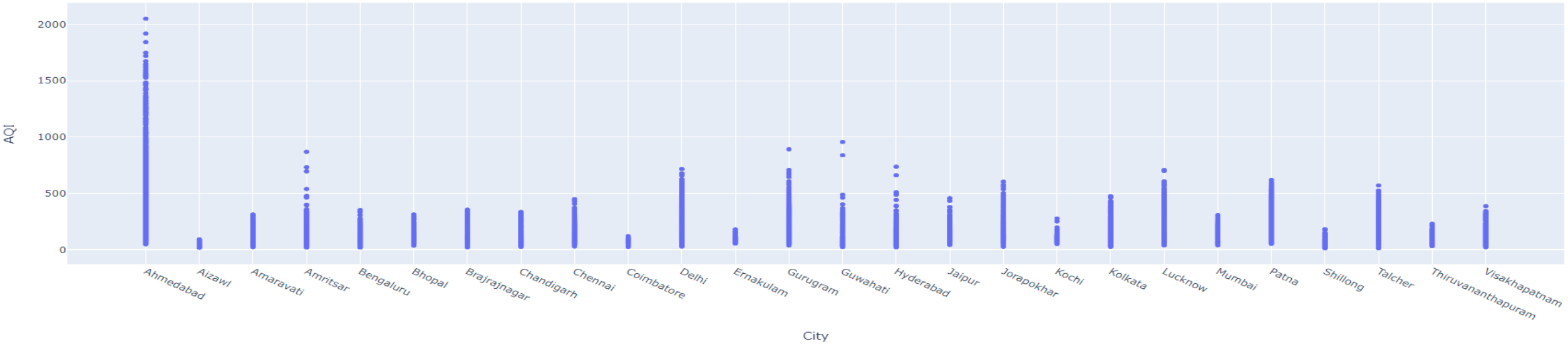
```
Index(['City', 'PM2.5', 'PM10', 'NO', 'NO2', 'NOx', 'NH3', 'CO', 'SO2', 'O3',
       'Benzene', 'Toluene', 'Xylene', 'AQI'],
      dtype='object')
```

```
#EDA(analyse the data)
```

```
[158] import plotly.express as px
```

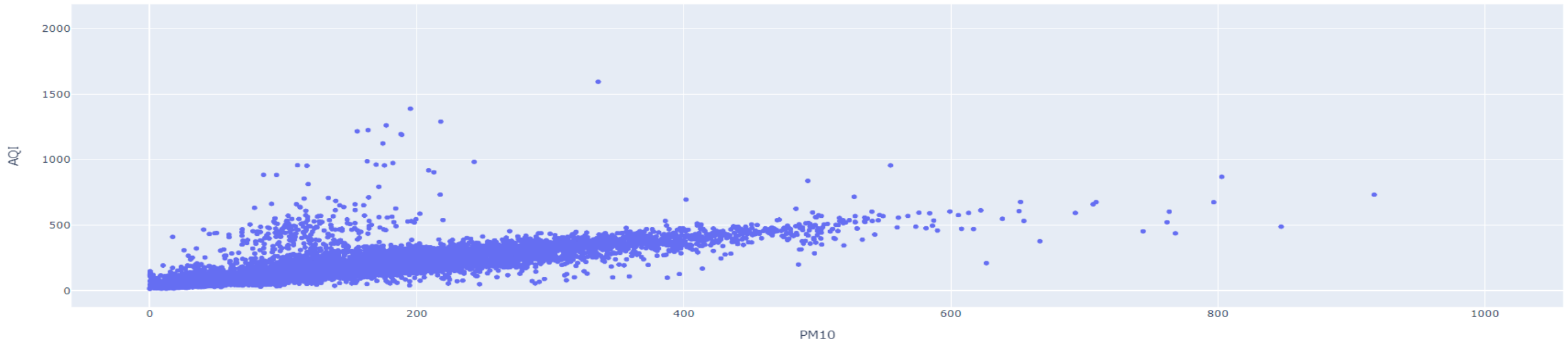
```
[159] #plotting the bubble chart
      fig = px.scatter(df, x="City",y="AQI")
```

```
[160] #Showing the plot
      fig.show()
```



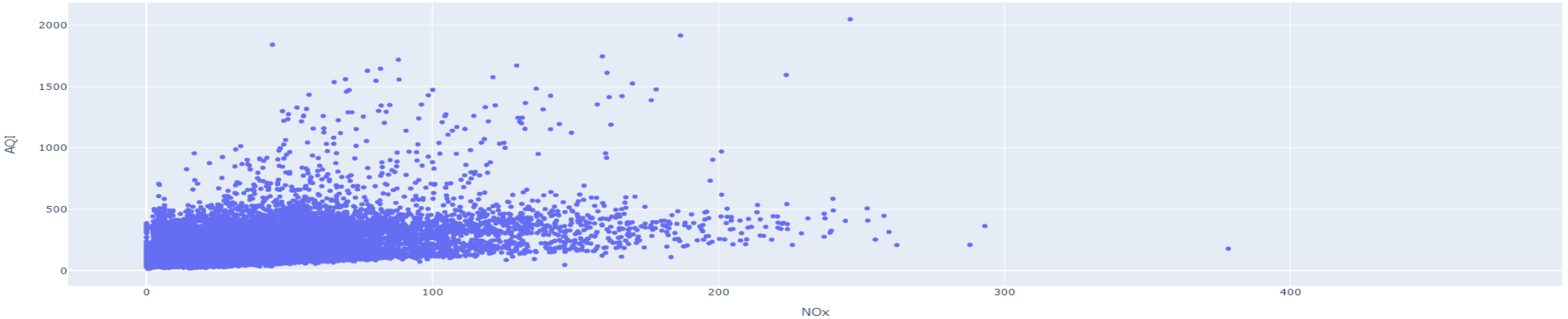
[161] #plotting the bubble chart
fig2=px.scatter(df, x="PM10" , y="AQI")

[162] #showing the plot
fig2.show()



```
fig5=px.scatter(df, x="NOx",y="AQI")
```

```
#Showing the plot
fig5.show()
```




```
[165] import matplotlib.pyplot as plt
```

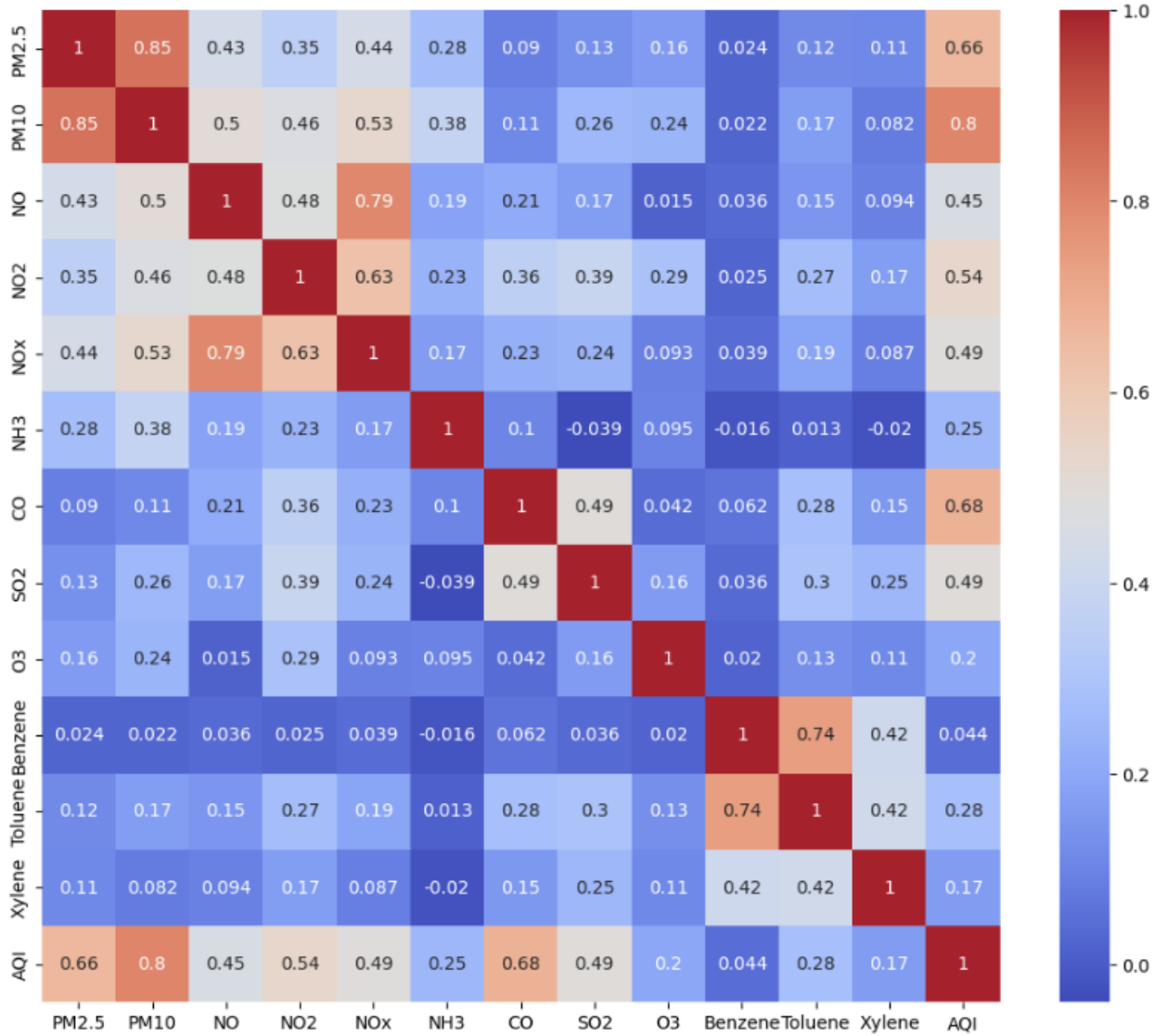
```
[166] plt.figure(figsize=(12,10))
      sns.heatmap(df.corr(),cmap='coolwarm',annot=True)
```

```
<ipython-input-166-080b192cd7cf>:2: FutureWarning:

The default value of numeric_only in DataFrame.corr is deprecated. In a future version, it will default to False. Select only valid columns or specify the value of numeric_only to silence this warning.

<Axes: >
```

<ipython-input-186-0b0b192cd7c1>:2: FutureWarning:

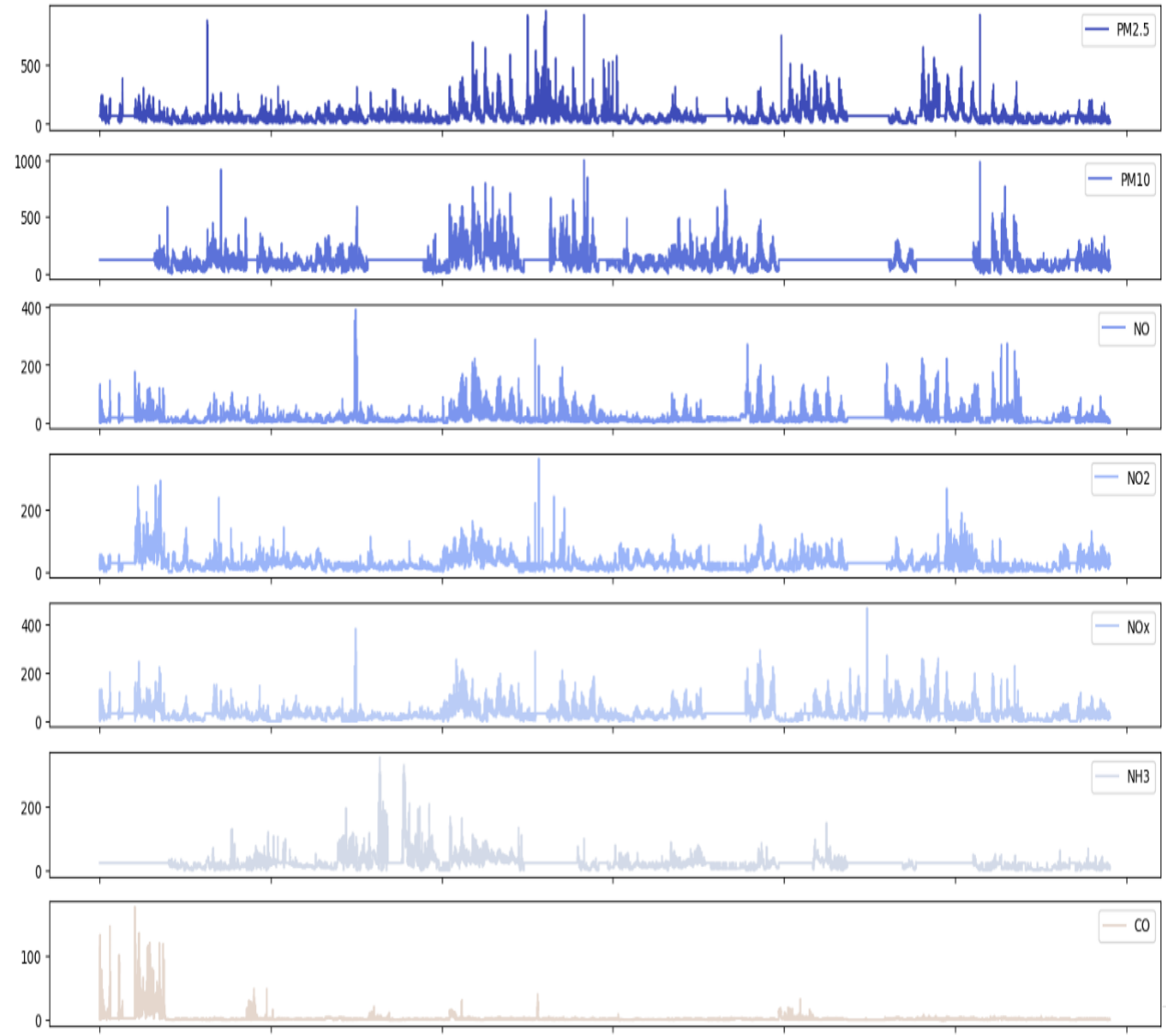
The default value of numeric_only in DataFrame.corr is deprecated. In a future version, it will default to False. Selec

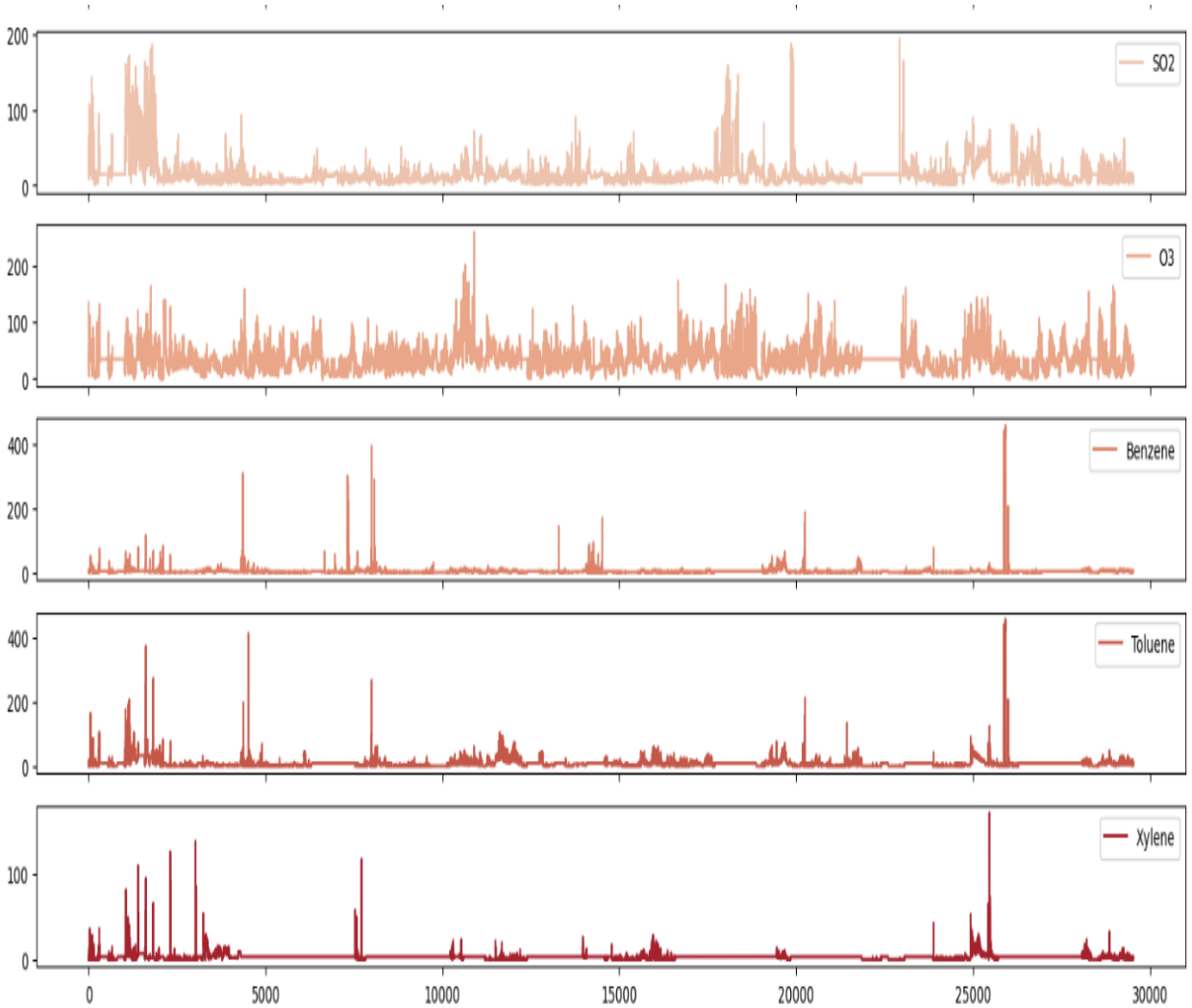
<Axes: >



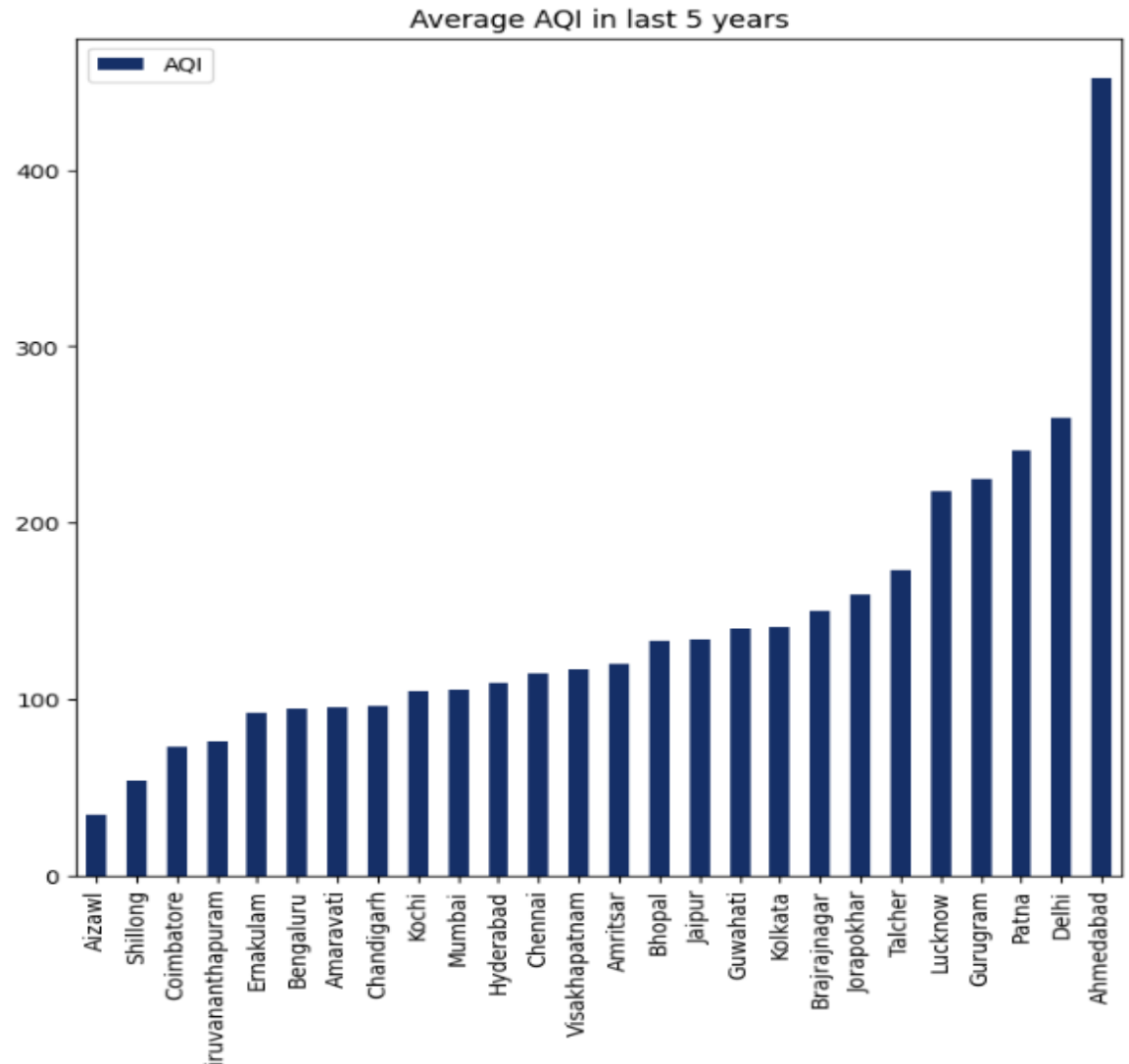
```
pollutants = ['PM2.5', 'PM10', 'NO', 'NO2', 'NOx', 'NH3', 'CO', 'SO2', 'O3', 'Benzene', 'Toluene', 'Xylene']
data2 = data2[pollutants]

print('Distribution of different pollutants in last 5 years')
data2.plot(kind='line',figsize=(18,18),cmap='coolwarm',subplots=True,fontsize=10);
```

Distribution of different pollutants in last 5 years





```
df[['City','AQI']].groupby('City').mean().sort_values('AQI').plot(kind='bar',cmap='Blues_r',figsize=(8,8))
plt.title('Average AQI in last 5 years');
```



```
[137] #preprocessing , feature selecation
```

```
[138] # train and test split data
```

```
[139] #training multiple models
```

```
[167] data2.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 29531 entries, 0 to 29530
Data columns (total 14 columns):
 #   Column   Non-Null Count  Dtype  
---  ------   --------------  -----  
 0   City     29531 non-null  int64  
 1   PM2.5    29531 non-null  float64
 2   PM10     29531 non-null  float64
 3   NO       29531 non-null  float64
 4   NO2      29531 non-null  float64
 5   NOx      29531 non-null  float64
 6   NH3      29531 non-null  float64
 7   CO       29531 non-null  float64
 8   SO2      29531 non-null  float64
 9   O3       29531 non-null  float64
 10  Benzene  29531 non-null  float64
 11  Toluene  29531 non-null  float64
 12  Xylene   29531 non-null  float64
 13  AQI      29531 non-null  float64
dtypes: float64(13), int64(1)
memory usage: 3.2 MB
```

```
[168] data2.columns
```

```
Index(['City', 'PM2.5', 'PM10', 'NO', 'NO2', 'NOx', 'NH3', 'CO', 'SO2', 'O3',
       'Benzene', 'Toluene', 'Xylene', 'AQI'],
      dtype='object')
```

```
[169] features = data2[['City', 'PM2.5', 'PM10', 'NO', 'NO2', 'NOx', 'NH3', 'CO', 'SO2', 'O3','Benzene', 'Toluene', 'Xylene']]
      labels = data2['AQI']
```

```
[170] #splitting into train and test data
```

```
[173] from sklearn.model_selection import train_test_split
      Xtrain, Xtest, Ytrain, Ytest = train_test_split(features,labels,test_size = 0.2,random_state=2)
```

```
[177] from sklearn.ensemble import RandomForestRegressor
      from sklearn.datasets import make_regression
      regr = RandomForestRegressor(max_depth=2,random_state=0)
      regr.fit(Xtrain,Ytrain)
      print(regr.predict(Xtest))
```

```
[120.44445696 120.44445696 120.44445696 ... 120.44445696 307.33721799
 307.33721799]
```

```
[178] y_pred=regr.predict(Xtest)
```

```
[181] from sklearn.metrics import r2_score
```

```
[182] r2_score(Ytest, y_pred)
```

```
0.6540144120689928
```