Technical University of Denmark

42186 Model Based Machine Learning

Project Assignment

Environmental Data Analysis

Group 8

Alex Lukas Hirsig - s230703
Jiaxin Wang - s230031
Yanfen Chen - s222420
Tonglei Liu - s233213

April 5, 2024

Research description:

This project aims to understand better air quality and how it will effect people in the future. Air quality harms the environment and human health, causing for example asthma and lung cancer [1][2][3][5]. The air quality in Hong Kong is notoriously bad and to prevent future damage reducing air pollution and improving the overall air quality are important. To do so effectively it is useful to have a good understanding of the trend of the air quality in the city for the future.
This project aims to gain such insight. In this project, the air quality of Hong Kong is predicted using a Bayesian network model from historical data on the 7 parameters (CO-, PM2.5-, PM10-, NO_2-, NOX-, O_3- and SO_2-concentration). The research question of the project is thus as follows:

What is the future trend of the air quality in Hong Kong?

Data- & Feature description:

The data used in this project contains the hourly measurement of 7 different air quality parameters from 18 measuring stations in Hong Kong in the period from January 1990 to November 2023. The parameters used to get the Air Quality Index (AQI) are listed here:

1. Carbon Monoxide **CO**
2. Fine Suspended Particulates **PM2.5**
3. Respirable Suspended Particulates **PM10**
4. Nitrogen Dioxide $\mathbf{NO_2}$
5. Nitrogen Oxides **NOX**
6. Ozone $\mathbf{O_3}$
7. Sulphur Dioxide $\mathbf{SO_2}$

The locations of the stations around the city of Hong Kong can be seen in Figure 2 in Appendix A. Not all stations measure all parameters. Fine Suspended Particulates (PM2.5) measurements are only available from 1999 on. Some stations also have data only for some time within the period. The amount of time-stamps and of stations with data are noted in Appendix A. The data is obtained from the Environmental Protection Department in Hong Kong [4].

Model description:

To predict the air quality index (AQI) in Hong Kong, supervised learning is employed. We use the Bayesian network (BN) to infer the value of a target variable(AQI) based on observed air quality parameters values of variables such as SO2, CO, NOX, etc..The Bayesian network has been trained using historical data. These data provide known outcomes and enable the network to learn the conditional probabilities required for predicting AQI from new observations.

To structure this BN, each parameter and the AQI are represented as nodes in the network, additional nodes are represented as the time and monitoring stations. The graphical model illustrates how the various parameters, along with potential temporal(year) and spatial(stations) factors, are considered to influence the AQI. Furthermore, the directed edges between nodes indicate conditional dependencies. For example, an edge from O_3 to AQI indicates that the AQI is conditionally dependent on the O_3 levels, given the other 6 pollutants. When a node like AQI depends on multiple other nodes(parameters), it acts as a common factor of other nodes, suggesting that the parameters independently contribute to the AQI.
In our model, pollutants are parent nodes indicating the assumption that 7 air quality parameters influence the air quality while AQI is a child node, suggesting AQI is impacted by the levels of all pollutants. Each node has associated a conditional probability table which maps all possible values of its incoming set of edges. Through chain rules for BNs, the conditional probability distribution is shown by given station and pollutant. The CPD is extended for better interpretation once more feature data are accessible such as population in each station.

For predication part, the dataset has been split into two subsets, with 80% of the data allocated for training the model and the remaining 20% of the data for testing its performance. If the prediction result equal to actual value, then the prediction is proved to be successful and the prediction accuracy is calculated

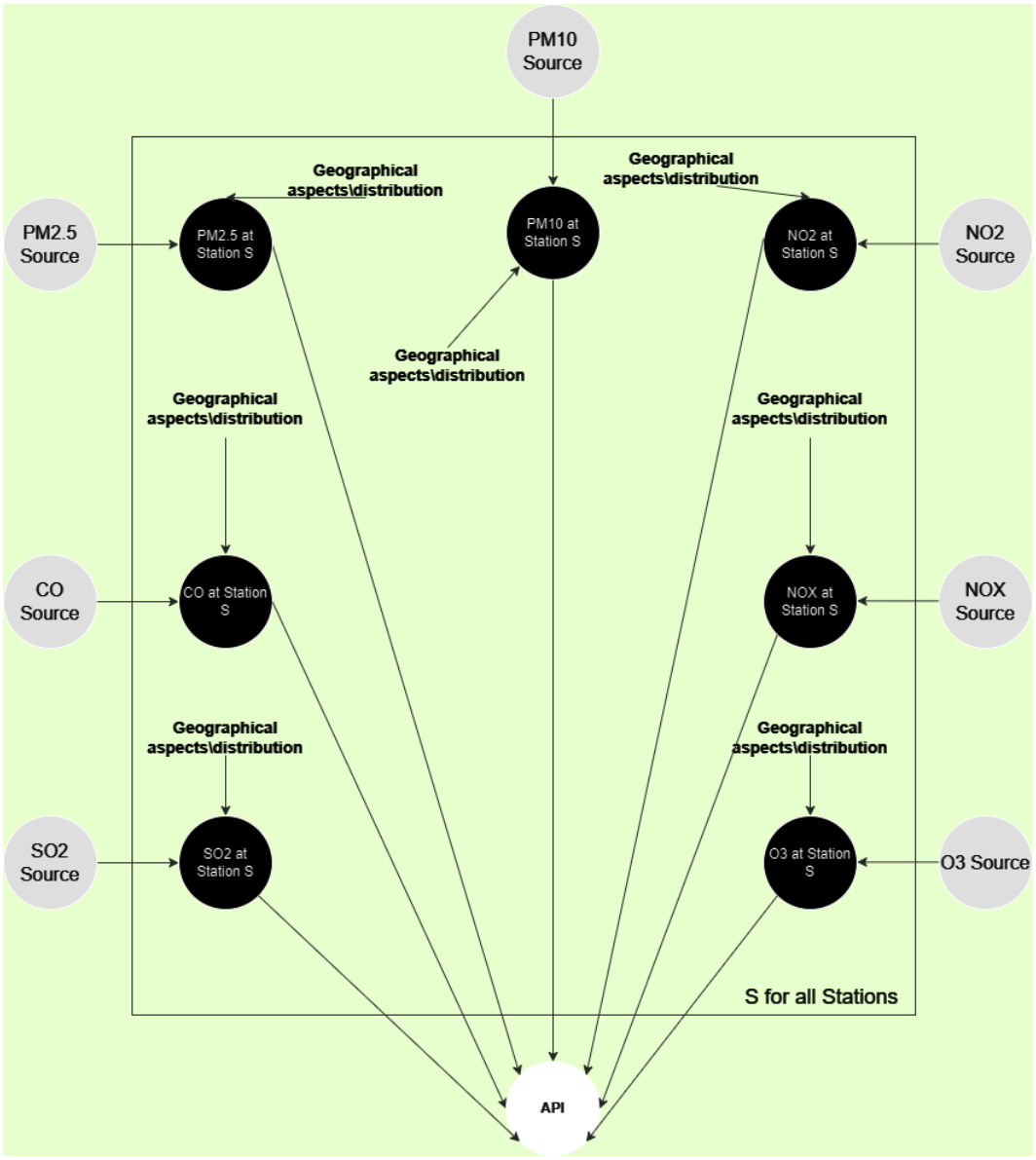


Figure 1: The PGM of the model for a given time step. The sources are time-dependent and thus should have an edge from the previous time to the current time. The black nodes are observed while the white and gray are not observed. The white node is the API and thus what we want to predict in this project.

References

[1] Bert Brunekreef. "Air Pollution and Human Health: From Local to Global Issues". In: (2009). DOI: 10.1016/j.sbspro.2010.05.010. URL: www.sciencedirect.com.

[2] Ethel Eljarrat et al. "Environmental and Health Impacts of Air Pollution: A Review". In: *Frontiers in Public Health — www.frontiersin.org* 8 (2020), p. 14. DOI: 10.3389/fpubh.2020.00014. URL: www.frontiersin.org.

[3] Adel Ghorani-Azam, Bamdad Riahi-Zanjani, and Mahdi Balali-Mood. "Effects of air pollution on human health and practical measures for prevention in Iran". In: *Journal of Research in Medical Sciences* 21.5 (2016). ISSN: 17357136. DOI: 10.4103/1735-1995.189646. URL: https://journals.lww.com/jrms/fulltext/2016/21000/effects_of_air_pollution_on_human_health_and.65.aspx.

[4] *Home — Environmental Protection Department*. URL: https://www.epd.gov.hk/epd/english/top.html.

[5] Ioannis Manisalidis et al. "Environmental and Health Impacts of Air Pollution: A Review". In: *Frontiers in Public Health* 8 (Feb. 2020), p. 505570. ISSN: 22962565. DOI: 10.3389/FPUBH.2020.00014/BIBTEX. URL: www.frontiersin.org.

A Appendix:

Data samples per parameter:

Parameter	Stations	time stamps
CO	9	286368
PM2.5	18	218400
PM10	18	270984
NO_2	18	297288
NOX	17	297288
O_3	18	297288
SO_2	18	297288

Table 1: Number of stations and time-stamps (1 time-stamp per hour) available for each parameter in the data.

Statoins in Hong Kong:

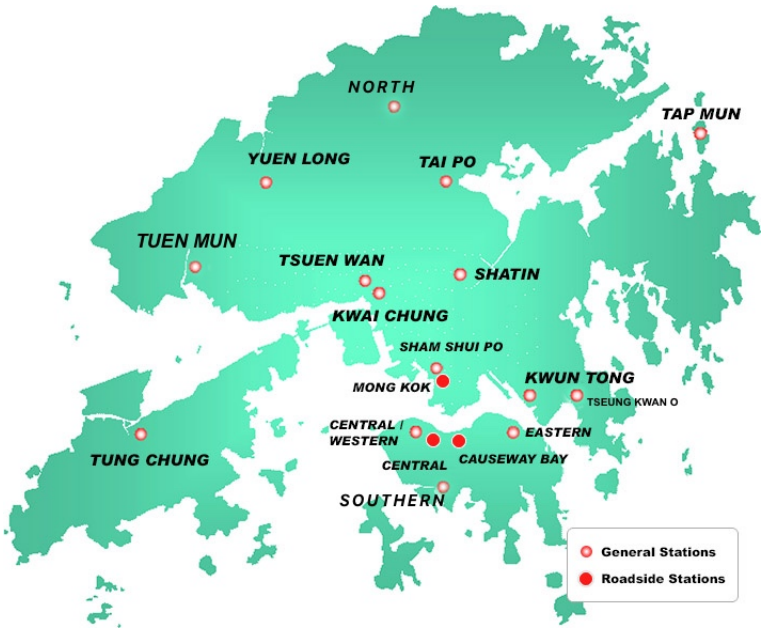


Figure 2: Location of the 18 stations around the city of Hong Kong [4].

Data Preparation:

Initial notebook cleaning the data and plotting the average time series of the parameters.

Data Preparation

1. Data Preprocessing

In [ ]:
```
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt

# Load the provided datasets
file_paths = {
    'CO': 'Data/CO.csv',
    'NO2': 'Data/NO2.csv',
    'NOX': 'Data/NOX.csv',
    'O3': 'Data/O3.csv',
    'PM25': 'Data/PM25.csv',
    'PM10': 'Data/PM10.csv',
    'SO2': 'Data/SO2.csv'
}

dataframes = {pollutant: pd.read_csv(filepath) for pollutant, filepath in file_paths.i

for pollutant, df in dataframes.items():
    print(f"First few rows of the {pollutant} dataset:")
    display(df.head())
    print("\n")
```

```
First few rows of the CO dataset:
```

	DATE	HOUR	POLLUTANT	SHATIN	TSUEN WAN	CENTRAL	EASTERN	KWUN TONG	TUEN MUN	TUNG CHUNG	...	YU LOI
0	1991-01-04	1	Carbon Monoxide	NaN	NaN	NaN	NaN	NaN	NaN	NaN	...	N
1	1991-01-04	23	Carbon Monoxide	NaN	NaN	NaN	NaN	NaN	NaN	NaN	...	N
2	1991-01-04	22	Carbon Monoxide	NaN	NaN	NaN	NaN	NaN	NaN	NaN	...	N
3	1991-01-04	21	Carbon Monoxide	NaN	NaN	NaN	NaN	NaN	NaN	NaN	...	N
4	1991-01-04	20	Carbon Monoxide	NaN	NaN	NaN	NaN	NaN	NaN	NaN	...	N

5 rows × 22 columns

```
First few rows of the NO2 dataset:
```

	DATE	HOUR	POLLUTANT	SHATIN	TSUEN WAN	CENTRAL	EASTERN	KWUN TONG	TUEN MUN	TUNG CHUNG	...	YU LOI
0	1990-01-01	1	Nitrogen Dioxide	NaN	NaN	NaN	NaN	NaN	NaN	NaN	...	N
1	1990-01-01	24	Nitrogen Dioxide	NaN	22.0	NaN	NaN	41.0	NaN	NaN	...	N
2	1990-01-01	23	Nitrogen Dioxide	NaN	22.0	NaN	NaN	43.0	NaN	NaN	...	N
3	1990-01-01	22	Nitrogen Dioxide	NaN	26.0	NaN	NaN	42.0	NaN	NaN	...	N
4	1990-01-01	21	Nitrogen Dioxide	NaN	29.0	NaN	NaN	42.0	NaN	NaN	...	N

5 rows × 22 columns

```
First few rows of the NOX dataset:
```

	DATE	HOUR	POLLUTANT	SHATIN	TSUEN WAN	CENTRAL	EASTERN	KWUN TONG	TUEN MUN	TUNG CHUNG	...	YU LOI
0	1990-01-01	1	Nitrogen Oxides	NaN	NaN	NaN	NaN	NaN	NaN	NaN	...	N
1	1990-01-01	24	Nitrogen Oxides	NaN	74.0	NaN	NaN	151.0	NaN	NaN	...	N
2	1990-01-01	23	Nitrogen Oxides	NaN	98.0	NaN	NaN	182.0	NaN	NaN	...	N
3	1990-01-01	22	Nitrogen Oxides	NaN	60.0	NaN	NaN	264.0	NaN	NaN	...	N
4	1990-01-01	21	Nitrogen Oxides	NaN	50.0	NaN	NaN	346.0	NaN	NaN	...	N

5 rows × 22 columns

```
First few rows of the O3 dataset:
```

	DATE	HOUR	POLLUTANT	SHATIN	TSUEN WAN	CENTRAL	EASTERN	KWUN TONG	TUEN MUN	TUNG CHUNG	...	YU LOI
0	1990-01-01	1	Ozone	NaN	18.0	NaN	NaN	NaN	NaN	NaN	...	N
1	1990-01-01	24	Ozone	NaN	5.0	NaN	NaN	NaN	NaN	NaN	...	N
2	1990-01-01	23	Ozone	NaN	8.0	NaN	NaN	NaN	NaN	NaN	...	N
3	1990-01-01	22	Ozone	NaN	8.0	NaN	NaN	NaN	NaN	NaN	...	N
4	1990-01-01	21	Ozone	NaN	9.0	NaN	NaN	NaN	NaN	NaN	...	N

5 rows × 22 columns

```
First few rows of the PM25 dataset:
```

	DATE	HOUR	POLLUTANT	SHATIN	TSUEN WAN	CENTRAL	EASTERN	KWUN TONG	TUEN MUN	TUNG CHUNG	...	YU LOI
0	1999-01-01	1	Fine Suspended Particulates	NaN	73.0	72.0	NaN	NaN	NaN	NaN	...	N
1	1999-01-01	23	Fine Suspended Particulates	NaN	80.0	67.0	NaN	NaN	NaN	NaN	...	N
2	1999-01-01	22	Fine Suspended Particulates	NaN	95.0	77.0	NaN	NaN	NaN	NaN	...	N
3	1999-01-01	21	Fine Suspended Particulates	NaN	109.0	91.0	NaN	NaN	NaN	NaN	...	N
4	1999-01-01	20	Fine Suspended Particulates	NaN	115.0	96.0	NaN	NaN	NaN	NaN	...	N

5 rows × 22 columns

```
First few rows of the PM10 dataset:
```

	DATE	HOUR	POLLUTANT	SHATIN	TSUEN WAN	CENTRAL	EASTERN	KWUN TONG	TUEN MUN	TUNG CHUNG	...	YU LOI
0	1993-01-01	1	Respirable Suspended Particulates	56.0	37.0	NaN	NaN	59.0	NaN	NaN	...	N
1	1993-01-01	12	Respirable Suspended Particulates	37.0	49.0	NaN	NaN	38.0	NaN	NaN	...	N
2	1993-01-01	2	Respirable Suspended Particulates	61.0	40.0	NaN	NaN	44.0	NaN	NaN	...	N
3	1993-01-01	3	Respirable Suspended Particulates	60.0	38.0	NaN	NaN	37.0	NaN	NaN	...	N
4	1993-01-01	4	Respirable Suspended Particulates	40.0	41.0	NaN	NaN	39.0	NaN	NaN	...	N

5 rows × 22 columns

```
First few rows of the SO2 dataset:
```

	DATE	HOUR	POLLUTANT	SHATIN	TSUEN WAN	CENTRAL	EASTERN	KWUN TONG	TUEN MUN	TUNG CHUNG	...	YU LOI
0	1990-01-01	1	Sulphur Dioxide	NaN	NaN	NaN	NaN	NaN	NaN	NaN	...	N
1	1990-01-01	12	Sulphur Dioxide	NaN	19.0	NaN	NaN	21.0	NaN	NaN	...	N
2	1990-01-01	2	Sulphur Dioxide	NaN	12.0	NaN	NaN	6.0	NaN	NaN	...	N
3	1990-01-01	3	Sulphur Dioxide	NaN	9.0	NaN	NaN	8.0	NaN	NaN	...	N
4	1990-01-01	4	Sulphur Dioxide	NaN	8.0	NaN	NaN	8.0	NaN	NaN	...	N

5 rows × 22 columns

1.1 Data Cleaning

In [ ]:
```python
def clean_and_reformat(data_path):
    data = pd.read_csv(data_path)
    data = data.replace('N.A.', pd.NA).apply(pd.to_numeric, errors='ignore')
    data['DATE'] = pd.to_datetime(data['DATE'], dayfirst=False, errors='coerce')
    data['DATETIME'] = data['DATE'] + pd.to_timedelta(data['HOUR'] - 1, unit='h')
    data = data.drop(['DATE', 'HOUR', 'POLLUTANT'], axis=1).set_index('DATETIME')
    return data
```

```
# clean and reformat each dataset
datasets = {pollutant: clean_and_reformat(path) for pollutant, path in file_paths.item

# combine all datasets
all_data = pd.concat(datasets.values(), axis=1, keys=datasets.keys())

all_data.head()
```

Out[ ]:

	CO									
	SHATIN	TSUEN WAN	CENTRAL	EASTERN	KWUN TONG	TUEN MUN	TUNG CHUNG	SHAM SHUI PO	SOUTHERN	YUEN LONG
DATETIME										
1990-01-01 00:00:00	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN
1990-01-01 01:00:00	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN
1990-01-01 02:00:00	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN
1990-01-01 03:00:00	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN
1990-01-01 04:00:00	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN

5 rows × 133 columns

In [ ]:
```
# generate statistical summary for each pollutant
statistical_summary = all_data.describe()
```

In [ ]:
```
# Check for missing data
missing_data_summary = all_data.isnull().mean().unstack(level=0).mul(100).round(2)
# display the percentage of missing data for each pollutant in each location
missing_data_summary
```

Out[ ]:

	CO	NO2	NOX	O3	PM25	PM10	SO2
SHATIN	100.00	16.26	16.26	25.19	66.55	14.45	7.89
TSUEN WAN	25.72	7.47	7.46	18.67	30.51	15.02	6.68
CENTRAL	28.38	28.17	28.17	62.52	30.12	29.74	28.18
EASTERN	100.00	30.69	100.00	30.69	63.87	28.74	30.69
KWUN TONG	100.00	6.70	6.69	27.11	63.68	16.38	5.95
TUEN MUN	71.56	71.78	71.78	71.72	72.08	72.08	71.68
TUNG CHUNG	29.71	29.96	29.97	29.87	29.48	29.17	30.40
SHAM SHUI PO	100.00	6.07	6.07	26.59	64.16	25.91	4.90
SOUTHERN	90.11	90.15	90.15	90.10	90.28	90.27	90.15
YUEN LONG	43.07	20.54	33.83	20.38	45.80	19.94	20.44
CENTRAL/WESTERN	100.00	6.55	6.53	6.11	64.60	14.70	5.32
NORTH	90.10	90.14	90.14	90.09	90.48	90.47	90.12
KWAI CHUNG	96.29	4.52	4.52	4.09	63.05	13.11	3.59
TAP MUN	29.49	29.51	29.26	29.81	30.64	27.66	28.92
TSEUNG KWAN O	77.79	77.80	77.80	77.78	77.79	77.76	77.86
TAI PO	100.00	7.61	50.00	19.61	65.36	27.38	18.55
MONG KOK	8.72	9.16	9.16	63.13	63.22	22.77	8.43
CAUSEWAY BAY	27.30	26.53	26.53	63.05	64.55	26.62	26.40
Average	5.31	0.87	0.85	0.72	26.54	8.88	0.98

CO

In [ ]:
```
# show  statistical summary
statistical_summary['CO']
```

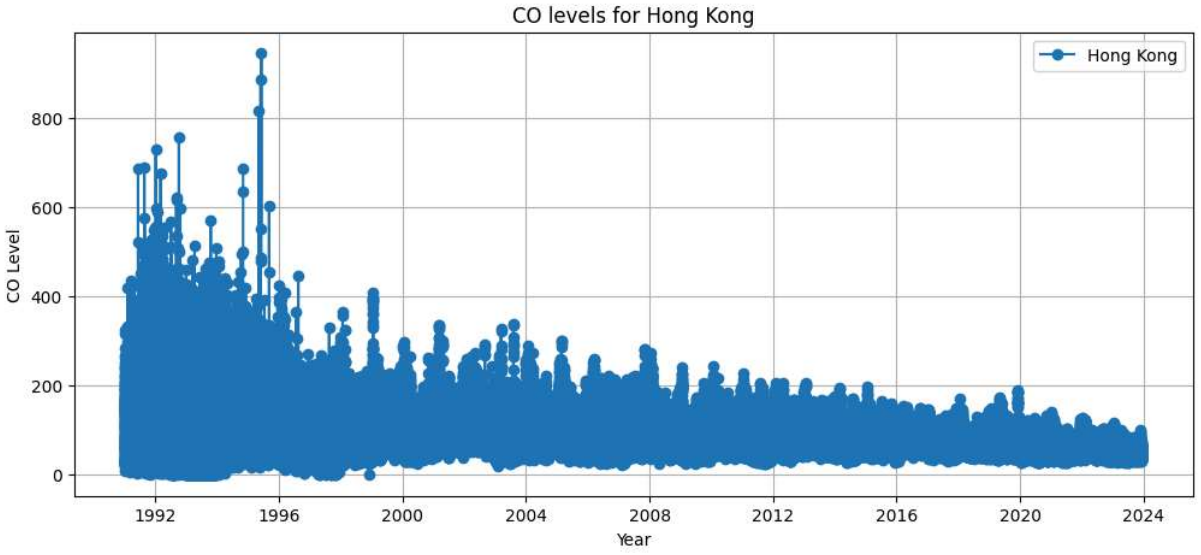
Out[ ]:

	SHATIN	TSUEN WAN	CENTRAL	EASTERN	KWUN TONG	TUEN MUN	TUNG CHUNG	SHAM SHUI PO
count	0.0	220821.000000	212909.000000	0.0	0.0	84554.000000	208964.000000	0.0
mean	NaN	71.981342	91.662325	NaN	NaN	67.557478	63.483763	NaN
std	NaN	33.606701	45.396913	NaN	NaN	21.801282	35.420134	NaN
min	NaN	0.000000	0.000000	NaN	NaN	9.000000	0.000000	NaN
25%	NaN	49.000000	59.000000	NaN	NaN	53.000000	40.000000	NaN
50%	NaN	69.000000	85.000000	NaN	NaN	65.000000	57.000000	NaN
75%	NaN	91.000000	115.000000	NaN	NaN	80.000000	80.000000	NaN
max	NaN	529.000000	518.000000	NaN	NaN	261.000000	573.000000	NaN

In [ ]:
```
co_data = all_data['CO']

# plot the 'CO' data for the average level of all locations
def plot_co(data):
    plt.figure(figsize=(12, 5))
    plt.plot(data.index, data['Average'], marker='o', linestyle='-', label='Hong Kong'
    plt.title('CO levels for Hong Kong')
    plt.xlabel('Year')
    plt.ylabel('CO Level')
    plt.legend()
    plt.grid(True)
    plt.show()


plot_co(co_data)
```



NO_2

In [ ]:
```
# show  statistical summary
statistical_summary['NO2']
```

Out[ ]:

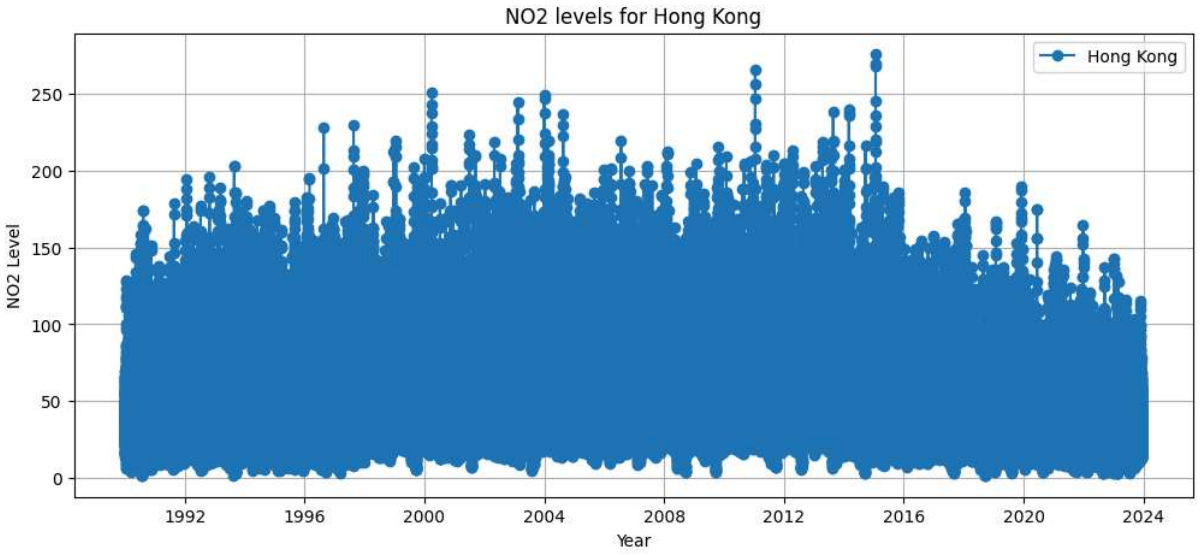
	SHATIN	TSUEN WAN	CENTRAL	EASTERN	KWUN TONG	TUEN MUN	
count	248936.000000	275078.000000	213552.000000	206047.000000	277382.000000	83888.000000	208
mean	41.394121	57.942140	93.694262	49.868122	59.148438	45.268394	
std	28.160858	31.032418	48.473515	26.334906	32.419963	27.530433	
min	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000	
25%	22.000000	36.000000	58.000000	30.000000	35.000000	26.000000	
50%	35.000000	53.000000	86.000000	46.000000	54.000000	39.000000	
75%	53.000000	74.000000	121.000000	65.000000	77.000000	59.000000	
max	282.000000	383.000000	528.000000	287.000000	428.000000	367.000000	

In [ ]:

```
no2_data = all_data['NO2']

# plot the 'NO2' data for the average level of all locations
def plot_no2(data):
    plt.figure(figsize=(12, 5))
    plt.plot(data.index, data['Average'], marker='o', linestyle='-', label='Hong Kong'
    plt.title('NO2 levels for Hong Kong')
    plt.xlabel('Year')
    plt.ylabel('NO2 Level')
    plt.legend()
    plt.grid(True)
    plt.show()


plot_no2(no2_data)
```



NOX

In [ ]:

```
# show  statistical summary
statistical_summary['NOX']
```

Out[ ]:

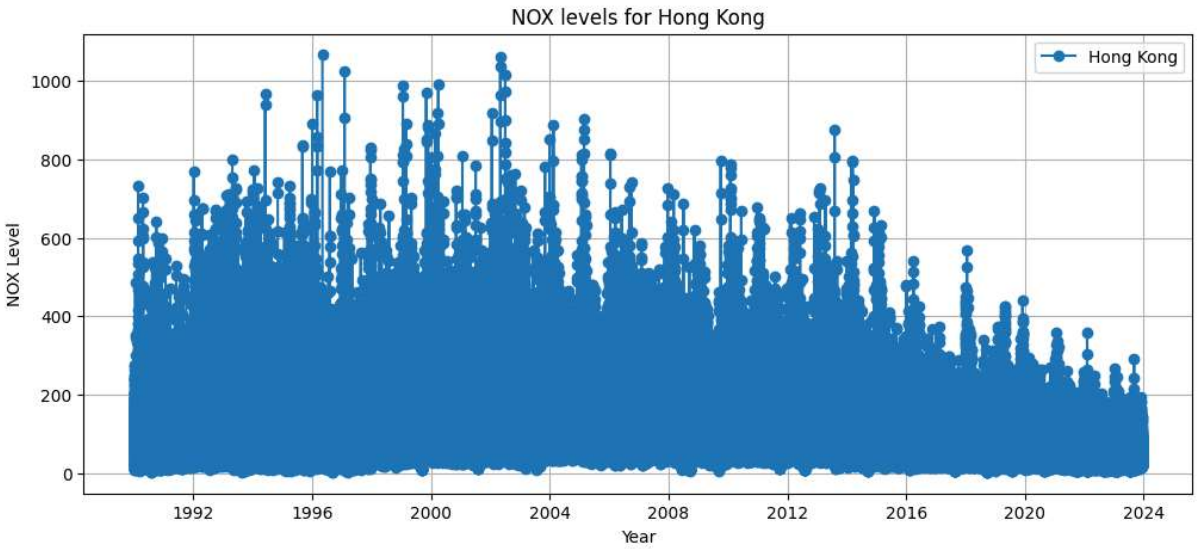
	SHATIN	TSUEN WAN	CENTRAL	EASTERN	KWUN TONG	TUEN MUN	CH
count	248944.000000	275112.000000	213553.000000	0.0	277398.000000	83888.000000	208191.0(
mean	71.781288	112.098313	276.700252	NaN	131.708235	67.002158	61.0
std	75.717525	84.591464	196.174339	NaN	102.380547	51.584106	53.3
min	0.000000	0.000000	0.000000	NaN	0.000000	0.000000	0.0(
25%	26.000000	55.000000	128.000000	NaN	55.000000	33.000000	23.0(
50%	47.000000	96.000000	231.000000	NaN	108.000000	54.000000	45.0(
75%	87.000000	144.000000	379.000000	NaN	182.000000	85.000000	83.0(
max	1368.000000	1611.000000	2453.000000	NaN	1716.000000	766.000000	782.0(

In [ ]:

```
nox_data = all_data['NOX']

def plot_nox(data):
    plt.figure(figsize=(12, 5))
    plt.plot(data.index, data['Average'], marker='o', linestyle='-', label='Hong Kong'
    plt.title('NOX levels for Hong Kong')
    plt.xlabel('Year')
    plt.ylabel('NOX Level')
    plt.legend()
    plt.grid(True)
    plt.show()


plot_nox(nox_data)
```



O_3

In [ ]:

```
# show  statistical summary
statistical_summary['O3']
```

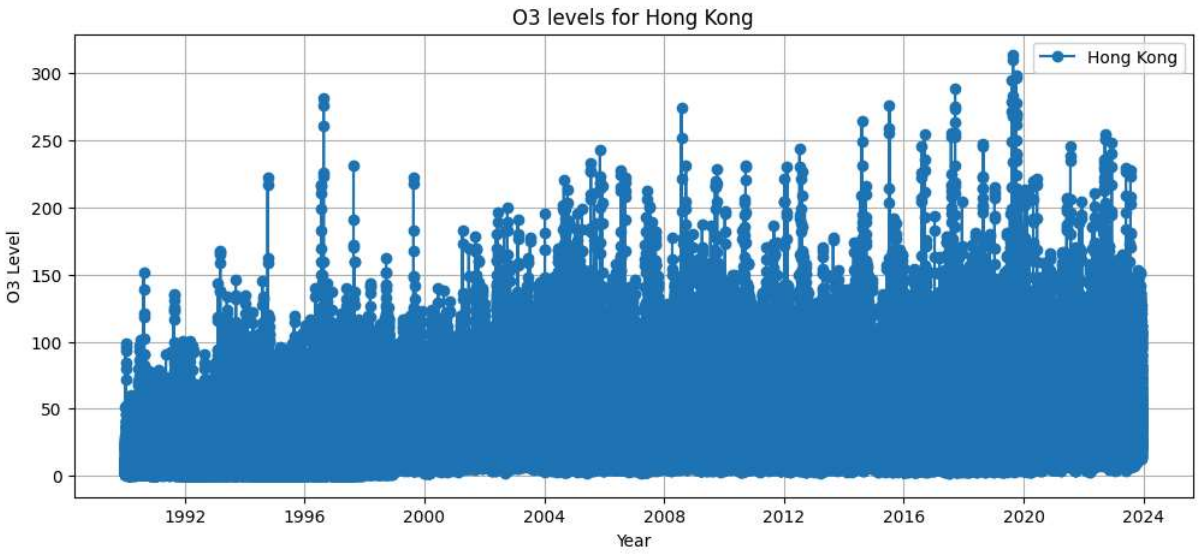
Out[ ]:

	SHATIN	TSUEN WAN	CENTRAL	EASTERN	KWUN TONG	TUEN MUN	208
count	222408.000000	241788.000000	111410.000000	206037.000000	216705.000000	84085.000000	
mean	45.822174	31.423830	26.660291	46.616263	38.003461	45.330499	
std	40.139073	29.777157	26.256881	30.464911	31.509565	39.026232	
min	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000	
25%	12.000000	9.000000	7.000000	25.000000	12.000000	17.000000	
50%	35.000000	22.000000	17.000000	40.000000	29.000000	35.000000	
75%	72.000000	46.000000	39.000000	62.000000	57.000000	64.000000	
max	368.000000	414.000000	298.000000	380.000000	317.000000	434.000000	

In [ ]:
```
o3_data = all_data['O3']

def plot_o3(data):
    plt.figure(figsize=(12, 5))
    plt.plot(data.index, data['Average'], marker='o', linestyle='-', label='Hong Kong'
    plt.title('O3 levels for Hong Kong')
    plt.xlabel('Year')
    plt.ylabel('O3 Level')
    plt.legend()
    plt.grid(True)
    plt.show()


plot_o3(o3_data)
```



$PM25$

In [ ]:
```
# show  statistical summary
statistical_summary['PM25']
```

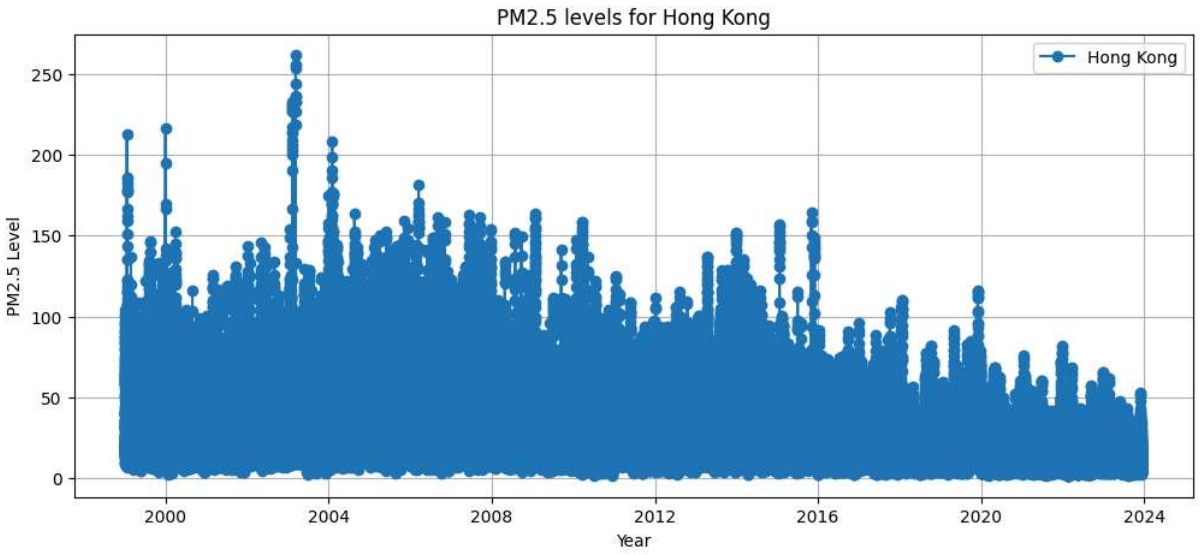
Out[ ]:

	SHATIN	TSUEN WAN	CENTRAL	EASTERN	KWUN TONG	TUEN MUN	
count	99438.000000	206596.000000	207758.000000	107401.000000	107973.000000	82996.000000	20966
mean	19.884732	29.591778	34.474133	20.353451	22.878618	23.845065	2
std	15.980296	21.908670	23.788795	14.963840	16.044267	17.251763	2
min	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000	
25%	9.000000	14.000000	17.000000	9.000000	12.000000	11.000000	1
50%	16.000000	24.000000	29.000000	17.000000	19.000000	20.000000	2
75%	26.000000	39.000000	47.000000	28.000000	30.000000	32.000000	3
max	172.000000	299.000000	353.000000	159.000000	205.000000	235.000000	43

In [ ]:
```
pm25_data = all_data['PM25']

def plot_pm25(data):
    plt.figure(figsize=(12, 5))
    plt.plot(data.index, data['Average'], marker='o', linestyle='-', label='Hong Kong'
    plt.title('PM2.5 levels for Hong Kong')
    plt.xlabel('Year')
    plt.ylabel('PM2.5 Level')
    plt.legend()
    plt.grid(True)
    plt.show()


plot_pm25(pm25_data)
```



$PM10$

In [ ]:
```
# show  statistical summary
statistical_summary['PM10']
```

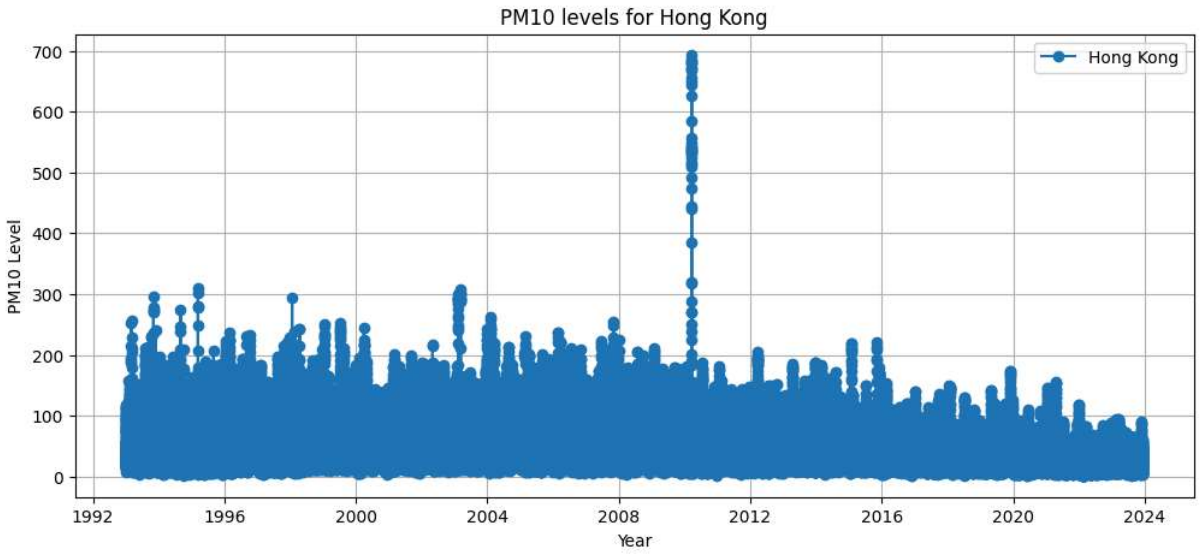
Out[ ]:

	SHATIN	TSUEN WAN	CENTRAL	EASTERN	KWUN TONG	TUEN MUN	
count	254317.000000	252649.000000	208885.000000	211841.000000	248587.000000	82992.000000	210
mean	41.752974	45.221046	53.112425	39.450612	47.490231	39.668631	
std	29.438181	31.195749	35.364419	27.832659	30.478020	26.316733	
min	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000	
25%	21.000000	24.000000	28.000000	19.000000	26.000000	20.000000	
50%	34.000000	38.000000	46.000000	33.000000	41.000000	34.000000	
75%	55.000000	59.000000	71.000000	53.000000	62.000000	52.000000	
max	783.000000	731.000000	716.000000	775.000000	785.000000	386.000000	

In [ ]:
```
pm10_data = all_data['PM10']

def plot_pm10(data):
    plt.figure(figsize=(12, 5))
    plt.plot(data.index, data['Average'], marker='o', linestyle='-', label='Hong Kong'
    plt.title('PM10 levels for Hong Kong')
    plt.xlabel('Year')
    plt.ylabel('PM10 Level')
    plt.legend()
    plt.grid(True)
    plt.show()


plot_pm10(pm10_data)
```



SO_2

In [ ]:
```
# show  statistical summary
statistical_summary['SO2']
```

Out[ ]:

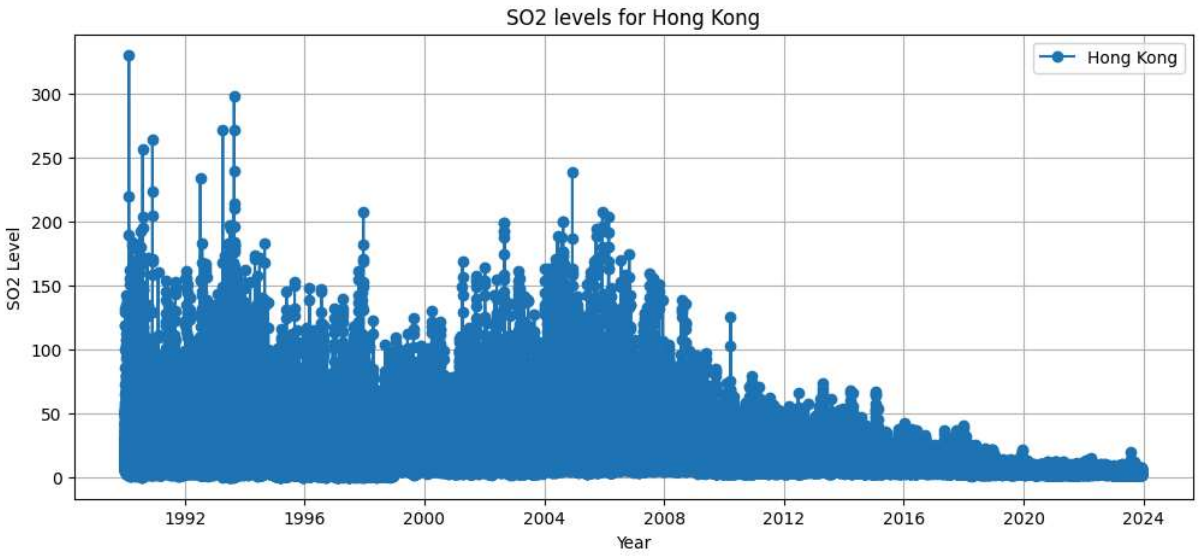
	SHATIN	TSUEN WAN	CENTRAL	EASTERN	KWUN TONG	TUEN MUN	
count	273822.000000	277422.000000	213503.000000	206041.000000	279609.000000	84196.000000	206
mean	11.546782	19.282833	14.298417	8.669590	14.343290	7.306772	
std	14.387693	26.917502	18.258044	12.920725	22.454732	6.394185	
min	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000	
25%	5.000000	7.000000	5.000000	2.000000	5.000000	3.000000	
50%	8.000000	12.000000	9.000000	5.000000	9.000000	5.000000	
75%	13.000000	22.000000	17.000000	10.000000	15.000000	9.000000	
max	458.000000	959.000000	507.000000	466.000000	1110.000000	89.000000	

In [ ]:

```
# Continuing from your existing code
so2_data = all_data['SO2']

def plot_so2(data):
    plt.figure(figsize=(12, 5))  # Set a larger figure size for better visibility
    plt.plot(data.index, data['Average'], marker='o', linestyle='-', label='Hong Kong'
    plt.title('SO2 levels for Hong Kong')
    plt.xlabel('Year')
    plt.ylabel('SO2 Level')
    plt.legend()
    plt.grid(True)
    plt.show()


plot_so2(so2_data)
```



2. Exploratory Data Analysis