

```
from google.colab import drive
drive.mount('/content/drive')
```

↗ Drive already mounted at /content/drive; to attempt to forcibly remount, call drive.mount("/content/drive", force\_remount=True).

## ✓ Importing Libraries and Data

```
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
import seaborn as sns
import plotly.express as px
import plotly.graph_objs as go
import plotly.offline as py
import folium
import warnings
warnings.filterwarnings('ignore')
```

There are multiple csv files available. Lets open the measurement summary

```
pol_data = pd.read_csv("/content/drive/MyDrive/한국분석/air_pollution_in_seoul/AirPollutionSeoul/Measurement_summary.csv")
pol_data.head()
```

↗

	Measurement date	Station code	Address	Latitude	Longitude	S02	N02	O3	CO	PM10	PM2.5
0	2017-01-01 00:00	101	19, Jong-ro 35ga-gil, Jongno-gu, Seoul, Republ...	37.572016	127.005008	0.004	0.059	0.002	1.2	73.0	57.0
1	2017-01-01 01:00	101	19, Jong-ro 35ga-gil, Jongno-gu, Seoul, Republ...	37.572016	127.005008	0.004	0.058	0.002	1.2	71.0	59.0
2	2017-01-01 02:00	101	19, Jong-ro 35ga-gil, Jongno-gu, Seoul, Republ...	37.572016	127.005008	0.004	0.056	0.002	1.2	70.0	59.0

📊

```
pol_data.shape
```

↗ (647511, 11)

There are 11 columns and 647511 rows in the dataset

```
pol_data.isnull().sum()
```

↗

	0
<b>Measurement date</b>	0
<b>Station code</b>	0
<b>Address</b>	0
<b>Latitude</b>	0
<b>Longitude</b>	0
<b>S02</b>	0
<b>N02</b>	0
<b>O3</b>	0
<b>CO</b>	0
<b>PM10</b>	0
<b>PM2.5</b>	0

**dtype:** int64

There are no null values in the data. Lets see the distribution.

```
pol_data[['S02', 'N02', 'O3', 'CO', 'PM10', 'PM2.5']].describe()
```

	S02	N02	O3	CO	PM10	PM2.5	
<b>count</b>	647511.000000	647511.000000	647511.000000	647511.000000	647511.000000	647511.000000	
<b>mean</b>	-0.001795	0.022519	0.017979	0.509197	43.708051	25.411995	
<b>std</b>	0.078832	0.115153	0.099308	0.405319	71.137342	43.924595	
<b>min</b>	-1.000000	-1.000000	-1.000000	-1.000000	-1.000000	-1.000000	
<b>25%</b>	0.003000	0.016000	0.008000	0.300000	22.000000	11.000000	
<b>50%</b>	0.004000	0.025000	0.021000	0.500000	35.000000	19.000000	
<b>75%</b>	0.005000	0.038000	0.034000	0.600000	53.000000	31.000000	
<b>max</b>	3.736000	38.445000	33.600000	71.700000	3586.000000	6256.000000	

Here we can see that the minimum value was -1 in some cases. That is not an acceptable value as there is nothing like negative pollution. This could be a measurement error. Lets count the number of occurrences of this.

```
print("We have", pol_data['S02'].loc[(pol_data['S02']<0)].count(),"negative values for S02")
print("We have", pol_data['N02'].loc[(pol_data['N02']<0)].count(),"negative values for N02")
print("We have", pol_data['O3'].loc[(pol_data['O3']<0)].count(),"negative values for O3")
print("We have", pol_data['CO'].loc[(pol_data['CO']<0)].count(),"negative values for CO")
print("We have", pol_data['PM10'].loc[(pol_data['PM10']<0)].count(),"negative values for PM10")
print("We have", pol_data['PM2.5'].loc[(pol_data['PM2.5']<0)].count(),"negative values for PM2.5")
```

```
We have 3976 negative values for S02
We have 3834 negative values for N02
We have 4059 negative values for O3
We have 4036 negative values for CO
We have 3962 negative values for PM10
We have 3973 negative values for PM2.5
```

```
data = [go.Scatter(x=pol_data['Measurement date'],
                  y=pol_data['S02'], name='S02'),
        go.Scatter(x=pol_data['Measurement date'],
                  y=pol_data['N02'], name='N02'),
        go.Scatter(x=pol_data['Measurement date'],
                  y=pol_data['CO'], name='CO'),
        go.Scatter(x=pol_data['Measurement date'],
                  y=pol_data['O3'], name='O3')]
```

```
##layout object
layout = go.Layout(title='Gases Levels',
                  yaxis={'title':'Level (ppm)'},
                  xaxis={'title':'Date'})
```

```
## Figure object
```

```
fig = go.Figure(data=data, layout=layout)
```

```
## Plotting
py.iplot(fig)
```

```
data = pol_data[pol_data['S02']<0]
```

```
data[['S02', 'N02', 'O3', 'CO', 'PM10', 'PM2.5']].describe()
```

It looks like most of this are occurring in same date, as we can see that the count is same and the mean is almost near to -1 in most of the columns. We can use imputation to replace these values with the mean.

```
from sklearn.impute import SimpleImputer
imp = SimpleImputer(missing_values=-1, strategy='mean')
df_imputed = pd.DataFrame(imp.fit_transform(pol_data[['S02', 'N02', 'O3', 'CO', 'PM10', 'PM2.5']]))
df_imputed.columns = pol_data[['S02', 'N02', 'O3', 'CO', 'PM10', 'PM2.5']].columns
df_imputed.index = pol_data.index
remain_df = pol_data[pol_data.columns.difference(['S02', 'N02', 'O3', 'CO', 'PM10', 'PM2.5'])]
df = pd.concat([remain_df, df_imputed], axis=1)
df.head()
```

	Address	Latitude	Longitude	Measurement date	Station code	S02	N02	O3	CO	PM10	PM2.5
0	19, Jong-ro 35ga-gil, Jongno-gu, Seoul, Republ...	37.572016	127.005008	2017-01-01 00:00	101	0.004	0.059	0.002	1.2	73.0	57.0
1	19, Jong-ro 35ga-gil, Jongno-gu, Seoul, Republ...	37.572016	127.005008	2017-01-01 01:00	101	0.004	0.058	0.002	1.2	71.0	59.0
2	19, Jong-ro 35ga-gil, Jongno-gu, Seoul, Republ...	37.572016	127.005008	2017-01-01 02:00	101	0.004	0.056	0.002	1.2	70.0	59.0

```
#TODO : Implement the time series with folium
last_entry = df.groupby('Station code').max() #here max is used just to get all type of pointers in the maps
# # last_entry.apply(lambda x: x.sample())
last_entry
```

	Address	Latitude	Longitude	Measurement date	S02	N02	O3	CO	PM10	PM2.5
Station code										
101	19, Jong-ro 35ga-gil, Jongno-gu, Seoul, Republ...	37.572016	127.005008	2019-12-31 23:00	0.406	0.109	0.325	40.0	516.0	513.0
102	15, Deoksugung-gil, Jung-gu, Seoul, Republic o...	37.564263	126.974676	2019-12-31 23:00	0.082	0.248	0.178	8.7	296.0	276.0
103	136, Hannam-daero, Yongsan-gu, Seoul, Republic...	37.540033	127.004850	2019-12-31 23:00	0.016	0.106	0.164	1.9	330.0	6256.0
104	215, Jinheung-ro, Eunpyeong-gu, Seoul, Republi...	37.609823	126.934848	2019-12-31 23:00	0.080	0.121	0.186	8.0	985.0	985.0
105	32, Segeomjeong-ro 4-gil, Seodaemun-gu, Seoul,...	37.593742	126.949679	2019-12-31 23:00	0.027	0.092	0.175	10.9	985.0	985.0
106	10, Poeun-ro 6-gil, Mapo-gu, Seoul, Republic o...	37.555580	126.905597	2019-12-31 23:00	0.332	0.097	0.368	31.3	985.0	985.0
107	18, Ttukseom-ro 3-gil, Seongdong-gu, Seoul, Re...	37.541864	127.049659	2019-12-31 23:00	0.144	0.113	0.189	15.3	985.0	985.0
108	571, Gwangnaru-ro, Gwangjin-gu, Seoul, Republi...	37.547180	127.092493	2019-12-31 23:00	0.342	0.135	0.336	37.5	1661.0	985.0
109	43, Cheonho-daero 13-gil, Dongdaemun-gu, Seoul...	37.575743	127.028885	2019-12-31 23:00	0.312	0.121	0.277	30.9	329.0	323.0
110	369, Yongmasan-ro, Jungnang-gu, Seoul, Republi...	37.584848	127.094023	2019-12-31 23:00	0.223	0.087	0.271	22.3	693.0	660.0
111	70, Samyang-ro 2-gil, Seongbuk-gu, Seoul, Repu...	37.606719	127.027279	2019-12-31 23:00	0.195	0.104	5.297	19.2	3403.0	995.0
112	49, Samyang-ro 139-gil, Gangbuk-gu, Seoul, Rep...	37.647930	127.011952	2019-12-31 23:00	0.097	0.108	0.156	4.1	333.0	572.0
113	34, Sirubong-ro 2-gil, Dobong-gu, Seoul, Repub...	37.654192	127.029088	2019-12-31 23:00	0.045	0.096	0.170	71.7	985.0	985.0
114	17, Sanggye-ro 23-gil, Nowon-gu, Seoul, Republ...	37.658774	127.068505	2019-12-31 23:00	0.120	0.116	0.163	12.0	389.0	387.0
115	56, Jungang-ro 52-gil, Yangcheon-gu, Seoul, Re...	37.525939	126.856603	2019-12-31 23:00	0.111	0.116	0.138	4.4	357.0	350.0
116	71, Gangseo-ro 45da-gil, Gangseo-gu, Seoul, Re...	37.544640	126.835151	2019-12-31 23:00	0.233	0.111	1.009	2.5	3586.0	175.0

다음 단계: [last\\_entry 변수로 코드 생성](#) [추천 차트 보기](#) [New interactive sheet](#)

Now we need to know about the levels of the above chemicals that are good and bad.

```
safe_limit = pd.read_csv('/content/drive/MyDrive/한국분석/air_pollution_in_seoul/AirPollutionSeoul/Original Data/Measurement_item_info.csv')
safe_limit
```



	Item code	Item name	Unit of measurement	Good(Blue)	Normal(Green)	Bad(Yellow)	Very bad(Red)
0	1	SO2	ppm	0.02	0.05	0.15	1.0
1	3	NO2	ppm	0.03	0.06	0.20	2.0
2	5	CO	ppm	2.00	9.00	15.00	50.0
3	6	O3	ppm	0.03	0.09	0.15	0.5
4	8	PM10	Mircrogram/m3	30.00	80.00	150.00	600.0
5	9	PM2.5	Mircrogram/m3	15.00	35.00	75.00	500.0



다음 단계:

[safe\\_limit 변수로 코드 생성](#)

[추천 차트 보기](#)

[New interactive sheet](#)


The get\_color function return a color based on the level of polution of each chemical

#<https://stackoverflow.com/a/16729808>

```
def get_colors(data, safe_limit, item):
    item_row = safe_limit.loc[safe_limit['Item name'] == item]
    if (data > item_row.iloc[0]['Very bad(Red)']):
        return 'red'
    elif (data > item_row.iloc[0]['Bad(Yellow)']):
        return 'yellow'
    elif (data > item_row.iloc[0]['Normal(Green)']):
        return 'green'
    else:
        return 'blue'
```

We are adding additional columns in the last\_entry dataframe for representation purpose.

```
last_entry['SO2 Color'] = last_entry['SO2'].apply(get_colors, args=(safe_limit, 'SO2' ))
last_entry['NO2 Color'] = last_entry['NO2'].apply(get_colors, args=(safe_limit, 'NO2' ))
last_entry['O3 Color'] = last_entry['O3'].apply(get_colors, args=(safe_limit, 'O3' ))
last_entry['CO Color'] = last_entry['CO'].apply(get_colors, args=(safe_limit, 'CO' ))
last_entry['PM10 Color'] = last_entry['PM10'].apply(get_colors, args=(safe_limit, 'PM10' ))
last_entry['PM2.5 Color'] = last_entry['PM2.5'].apply(get_colors, args=(safe_limit, 'PM2.5' ))
last_entry
```



	Address	Latitude	Longitude	Measurement date	S02	N02	O3	CO	PM10	PM2.5	S02 Color	N02 Color	O3 Color	CO Color	F Cc
Station code															
101	19, Jong-ro 35ga-gil, Jongno-gu, Seoul, Republ...	37.572016	127.005008	2019-12-31 23:00	0.406	0.109	0.325	40.0	516.0	513.0	yellow	green	yellow	yellow	ye
102	15, Deoksugung-gil, Jung-gu, Seoul, Republic o...	37.564263	126.974676	2019-12-31 23:00	0.082	0.248	0.178	8.7	296.0	276.0	green	yellow	yellow	blue	ye
103	136, Hannam-daero, Yongsan-gu, Seoul, Republic...	37.540033	127.004850	2019-12-31 23:00	0.016	0.106	0.164	1.9	330.0	6256.0	blue	green	yellow	blue	ye
104	215, Jinheung-ro, Eunpyeong-gu, Seoul, Republi...	37.609823	126.934848	2019-12-31 23:00	0.080	0.121	0.186	8.0	985.0	985.0	green	green	yellow	blue	
105	32, Segeomjeong-ro 4-gil, Seodaemun-gu, Seoul,...	37.593742	126.949679	2019-12-31 23:00	0.027	0.092	0.175	10.9	985.0	985.0	blue	green	yellow	green	
106	10, Poeun-ro 6-gil, Mapo-gu, Seoul, Republic o...	37.555580	126.905597	2019-12-31 23:00	0.332	0.097	0.368	31.3	985.0	985.0	yellow	green	yellow	yellow	
107	18, Ttukseom-ro 3-gil, Seongdong-gu, Seoul, Re...	37.541864	127.049659	2019-12-31 23:00	0.144	0.113	0.189	15.3	985.0	985.0	green	green	yellow	yellow	
108	571, Gwangnaru-ro, Gwangjin-gu, Seoul, Republi...	37.547180	127.092493	2019-12-31 23:00	0.342	0.135	0.336	37.5	1661.0	985.0	yellow	green	yellow	yellow	
109	43, Cheonho-daero 13-gil, Dongdaemun-	37.575743	127.028885	2019-12-31 23:00	0.312	0.121	0.277	30.9	329.0	323.0	yellow	green	yellow	yellow	ye

다음 단계:

last\_entry변수로 코드 생성

추천 차트 보기

New interactive sheet

Lets plot the map showing the level of S02

Pollution of S02

```

# This creates the map object
m = folium.Map(
    location=[37.541, 126.981], # center of where the map initializes
    #tiles='Stamen Toner', # the style used for the map (defaults to OSM)
    zoom_start=11, # the initial zoom level
    title = "Pollution level of S02")
for ind in last_entry.index:
    #print(row[1][0])
    folium.Marker([last_entry['Latitude'][ind], last_entry['Longitude'][ind]], popup=ind, icon=folium.Icon(color=last_entry['S02 Color'][ind], icon='

# Diplay the map
m
  
```

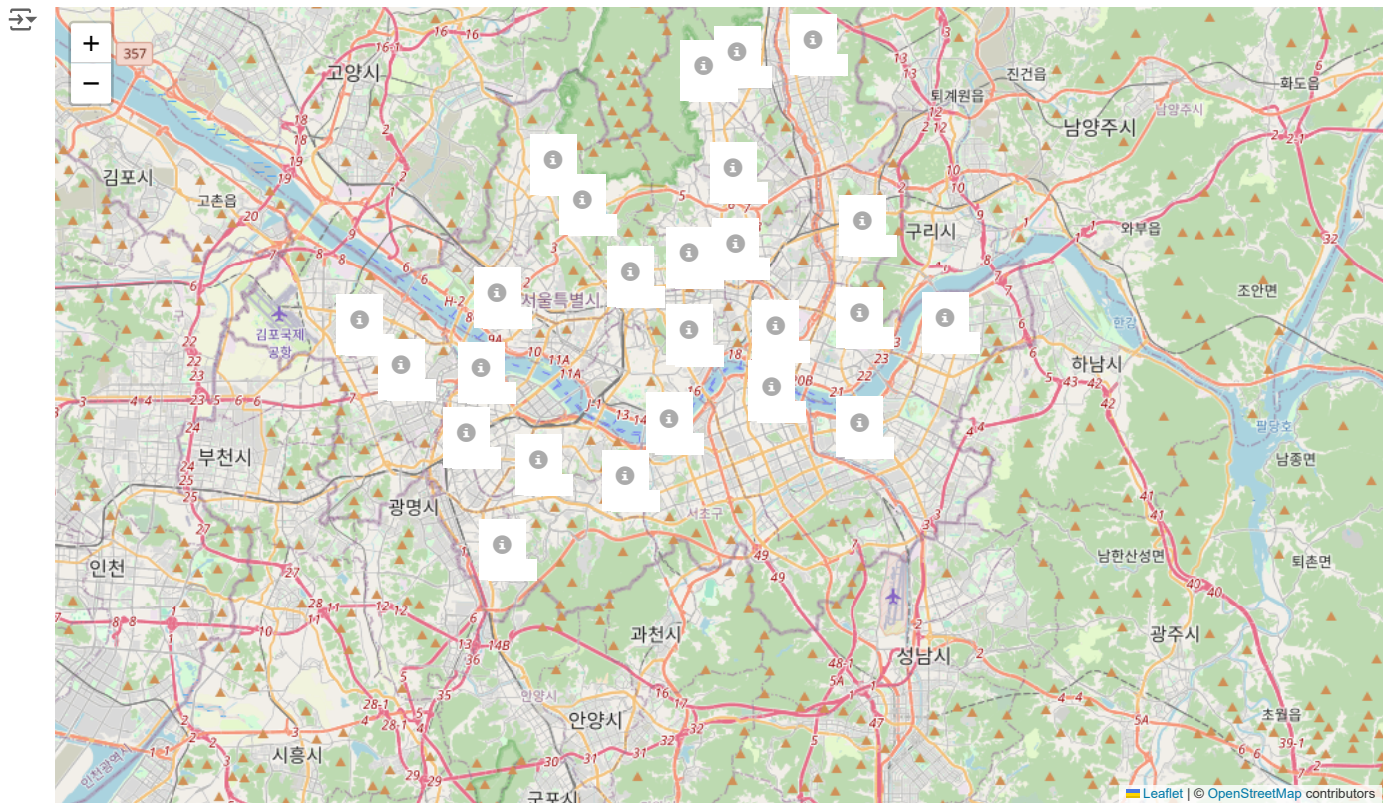


## ✎ Pollution of NO2

```

# This creates the map object
m = folium.Map(
    location=[37.541, 126.981], # center of where the map initializes
    #tiles='Stamen Toner', # the style used for the map (defaults to OSM)
    zoom_start=11, # the initial zoom level
    title = "Pollution level of NO2")
for ind in last_entry.index:
    #print(row[1][0])
    folium.Marker([last_entry['Latitude'][ind], last_entry['Longitude'][ind]], popup=ind, icon=folium.Icon(color=last_entry['NO2 Color'][ind], icon='
# Display the map
m
  
```



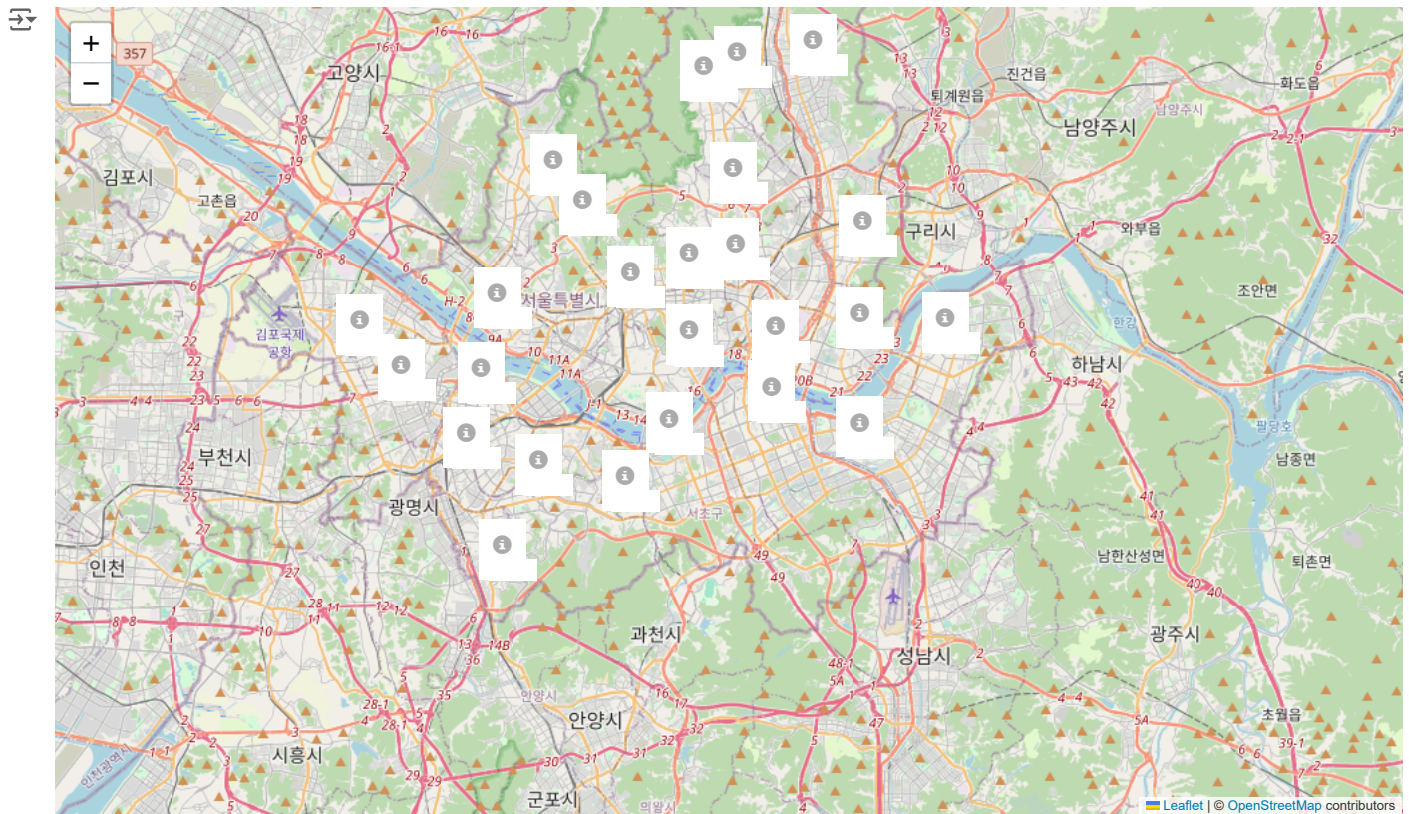


## ✎ Pollution of O3

```

# This creates the map object
m = folium.Map(
    location=[37.541, 126.981], # center of where the map initializes
    #tiles='Stamen Toner', # the style used for the map (defaults to OSM)
    zoom_start=11, # the initial zoom level
    title = "Pollution level of O3")
for ind in last_entry.index:
    #print(row[1][0])
    folium.Marker([last_entry['Latitude'][ind], last_entry['Longitude'][ind]], popup=ind, icon=folium.Icon(color=last_entry['O3 Color'][ind], icon='i

# Display the map
m
  
```

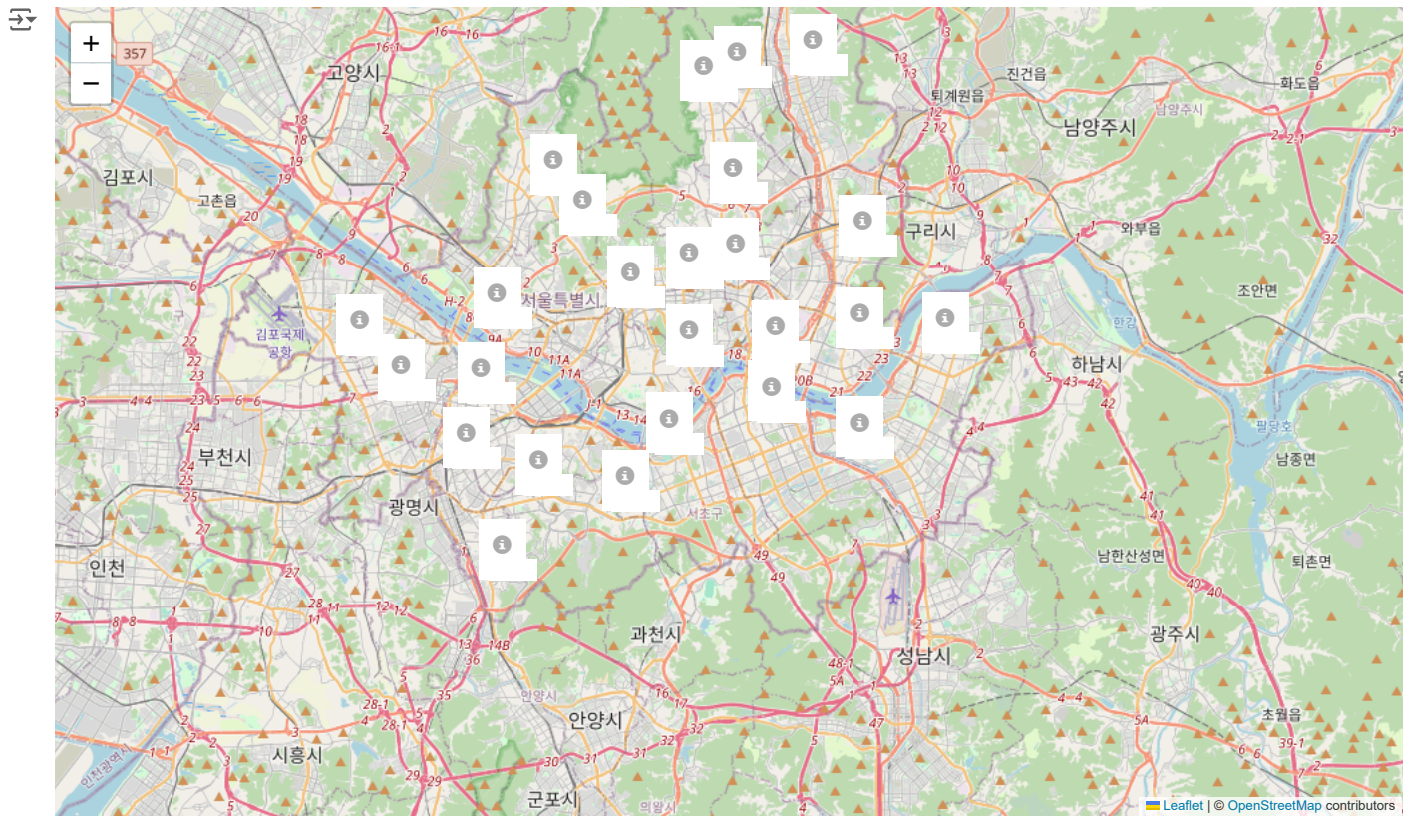


- ✓ Pollution of CO

```
# This creates the map object
m = folium.Map(
    location=[37.541, 126.981], # center of where the map initializes
    #tiles='Stamen Toner', # the style used for the map (defaults to OSM)
    zoom_start=11, # the initial zoom level
    title = "Pollution level of CO")
for ind in last_entry.index:
    #print(row[1][0])
    folium.Marker([last_entry['Latitude'][ind], last_entry['Longitude'][ind]], popup=ind, icon=folium.Icon(color=last_entry['CO Color'][ind], icon='i

# Display the map
m
```





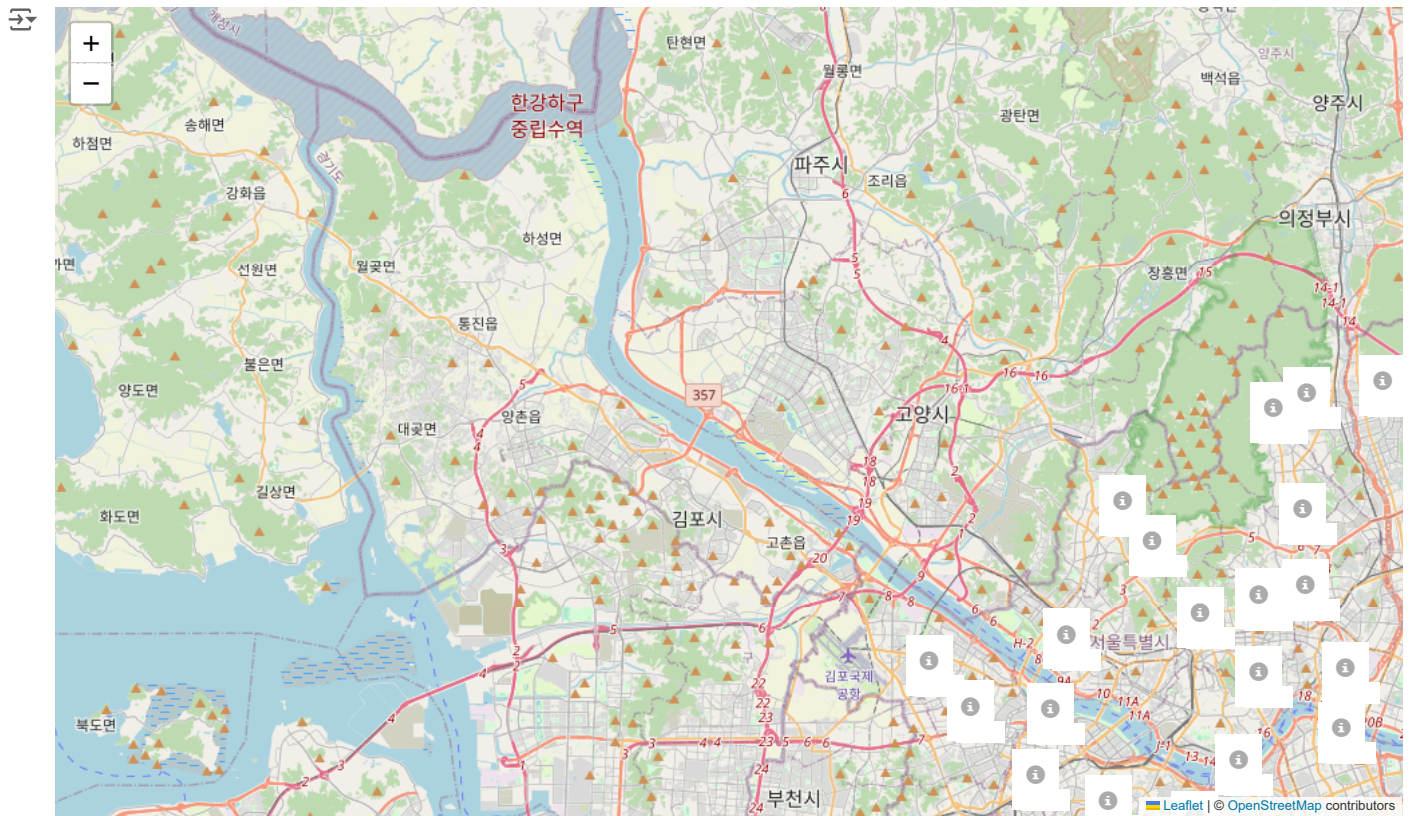
## ✎ Pollution of PM10

```

# This creates the map object
m = folium.Map(
    location=[37.541, 126.981], # center of where the map initializes
    #tiles='Stamen Toner', # the style used for the map (defaults to OSM)
    zoom_start=11, # the initial zoom level
    title = "Pollution level of PM10")
for ind in last_entry.index:
    #print(row[1][0])
    folium.Marker([last_entry['Latitude'][ind], last_entry['Longitude'][ind]], popup=ind, icon=folium.Icon(color=last_entry['PM10 Color'][ind], icon=

# Display the map
m

```



- ✧ Pollution of PM<sub>2.5</sub>

```
# This creates the map object
m = Map(x, y, z)
```