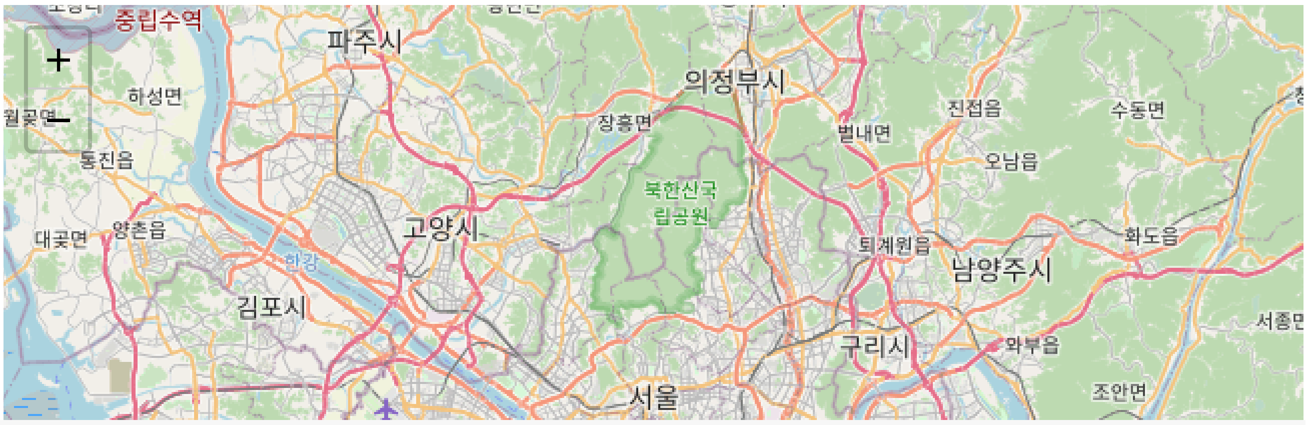
Folium 한번에 제대로 배우기

- 인터랙티브 지도 시각화 라이브러리
- Leaflet을 사용할 수 있는 leaflet.js 라이브러리를 파이썬에서 연동
- 지도 상의 마커에서 vector, raster, HTML 시각화 가능
- Choropleth 시각화
- TileSet, Image, Video, GeoJSON, TopoJSON 연동

```
import numpy as np
import pandas as pd
import json
import requests
import branca
import folium
folium.__version__
```

```
'0.8.3'
```

```
m = folium.Map(location=[37.566697, 126.978426])
m
```



```
m.save('map.html')
```

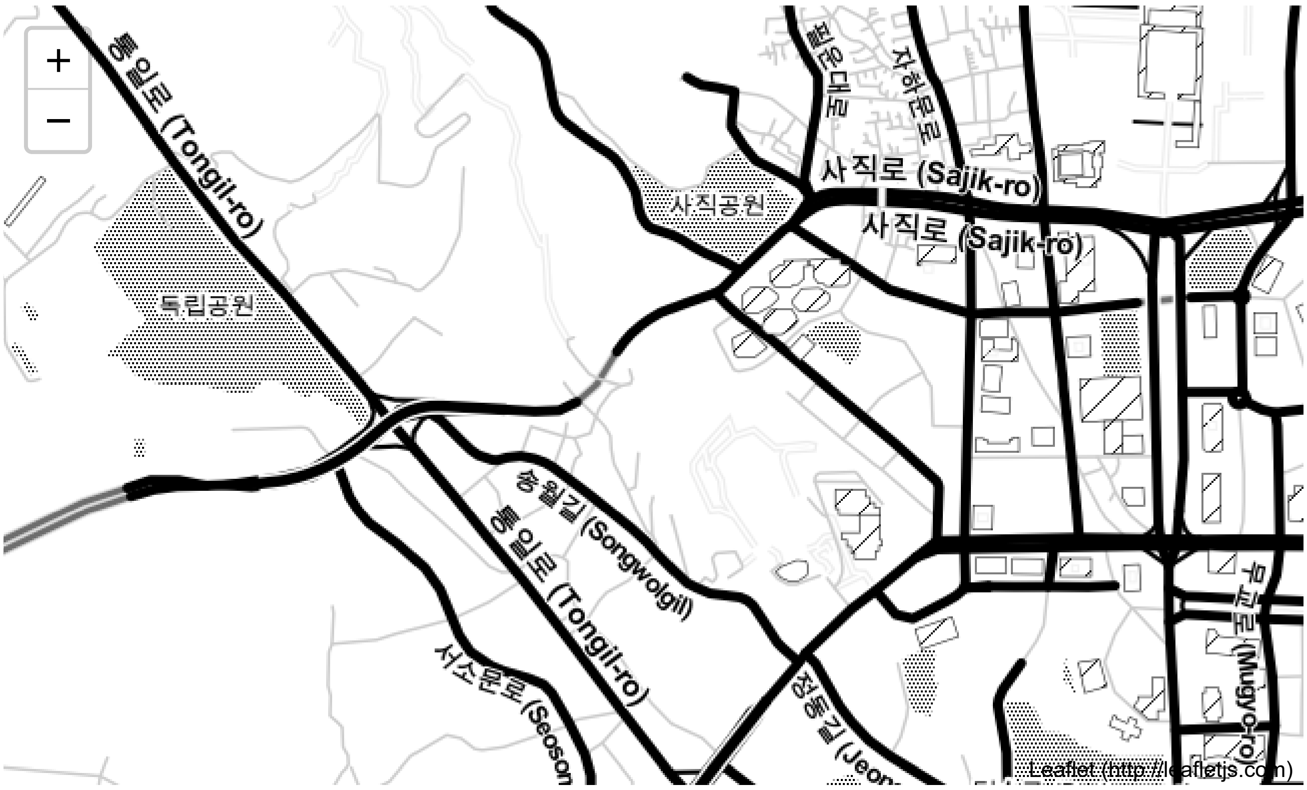
```
!ls
```

```
map.html  sample_data
```

타일(Tiles)

- OpenStreetMap
- Mapbox Bright (Limited levels of zoom for free tiles)
- Mapbox Control Room (Limited levels of zoom for free tiles)
- Stamen (Terrain, Toner, and Watercolor)
- Cloudmade (Must pass API key)
- Mapbox (Must pass API key)
- CartoDB (positron and dark_matter)

```
folium.Map(
    location=[37.566697, 126.978426],
    tiles='Stamen Toner',
    zoom_start=15
)
```



마커(Markers)

```
m = folium.Map(
    location=[37.566697, 126.978426],
    tiles='Stamen Terrain',
    zoom_start=13
)

tooltip = "Click me!"

folium.Marker(
    [37.555150, 126.970538],
    popup='<b>Seoul Station</b>',
    tooltip=tooltip
).add_to(m)

folium.Marker(
    [37.529759, 126.964642],
    popup='<b>Yongsan Station</b>',
    tooltip=tooltip
).add_to(m)

m
```

+
−
30
1
6
Leaflet (http://leafletjs.com)

아이콘(Icon)

```
m = folium.Map(
    location=[37.566697, 126.978426],
    tiles='Stamen Terrain',
    zoom_start=12
)

folium.Marker(
    [37.555150, 126.970538],
    popup='<b>Seoul Station</b>',
    icon=folium.Icon(color='red', icon='info-sign')
).add_to(m)

folium.Marker(
    [37.529759, 126.964642],
    popup='<b>Yongsan Station</b>',
    icon=folium.Icon(color='green', icon='bookmark')
).add_to(m)

folium.Marker(
    [37.560704, 127.038819],
    popup='<b>Wangsimni Station</b>',
    icon=folium.Icon(color='blue', icon='flag')
).add_to(m)

m
```

고양대로
제2자유로
자유로
Seoul
Leaflet (http://leafletjs.com)

```
from folium import plugins

m = folium.Map(
    location=[37.566697, 126.978426],
    tiles='Stamen Terrain'
    tiles='Stamen Terrain',
    zoom_start=11
)

icon_plane = plugins.BeautifyIcon(
    icon='plane',
    border_color='darkblue',
    text_color='darkblue',
    icon_shape='circle'
)

icon_flag = plugins.BeautifyIcon(
    icon='flag',
    border_color='green',
    text_color='green',
    icon_shape='triangle'
)

icon_number = plugins.BeautifyIcon(
    number=10,
    border_color='darkred',
    text_color='darkred',
    inner_icon_style='margin-top:0;'
)

folium.Marker(
    [37.558834, 126.802794],
    popup='Plane',
    icon=icon_plane
).add_to(m)

folium.Marker(
    [37.530357, 126.930722],
    popup='Number',
    icon=icon_number
).add_to(m)

folium.Marker(
    [37.551040, 126.990666],
    popup='Flag',
    icon=icon_flag
).add_to(m)

m
```

+
−
100
한강
Leaflet (http://leafletjs.com)

보트 마커(Boat Marker)

```
m = folium.Map([30, -180], zoom_start=3)

plugins.BoatMarker(
    location=(41.584185, 161.792354),
    heading=45,
    wind_heading=120,
    wind_speed=45,
    color='purple'
).add_to(m)

plugins.BoatMarker(
    location=(28.572786, 157.095985),
    heading=-20,
    wind_heading=45,
    wind_speed=15,
    color='darkblue'
).add_to(m)

plugins.BoatMarker(
    location=(39.836756, 176.479184),
    heading=-30,
    wind_heading=25,
    wind_speed=80,
    color='green'
).add_to(m)

m
```

+
−
Монгол улс
中国
日本
Leaflet (http://leafletjs.com)

클릭 마커

```
m = folium.Map(
    location=[37.566697, 126.978426],
    tiles='Stamen Terrain',
    zoom_start=13
)
```

```
)

folium.Marker(
    [37.566697, 126.978426],
    popup='Seoul Cityhall'
).add_to(m)

m.add_child(folium.ClickForMarker(popup='Marker'))

m
```

+
−
(Tongil-ro)
평창문화로
30
1
Leaflet (http://leafletjs.com)

원(Circle)

```python
m = folium.Map(
    location=[37.566697, 126.978426],
    tiles='Stamen Terrain',
    zoom_start=12
)

folium.Circle(
    [37.555150, 126.970538],
    popup='<b>Seoul Station</b>',
    radius=60,
    color='royalblue',
    fill=False
).add_to(m)

folium.CircleMarker(
    [37.529759, 126.964642],
    popup='<b>Yongsan Station</b>',
    radius=40,
    color='darkblue',
    fill=True,
    fill_color='darkblue'
).add_to(m)

folium.CircleMarker(
    [37.560704, 127.038819],
    popup='<b>Wangsimni Station</b>',
    radius=20,
    color='purple',
    fill=True,
    fill_color='purple'
).add_to(m)

m
```

- 서울: [37.566687, 126.978417]
- 부산: [35.179774, 129.075004]
- 인천: [37.455900, 126.705522]
- 대구: [35.871380, 128.601743]
- 대전: [36.350451, 127.384827]
- 광주: [35.160072, 126.851440]

```
cities = [[37.566687, 126.978417],
          [35.179774, 129.075004],
          [37.455900, 126.705522],
          [35.871380, 128.601743],
          [36.350451, 127.384827],
          [35.160072, 126.851440]]

m = folium.Map(
    location=[36.577629, 127.770135],
    tiles='Stamen Terrain',
    zoom_start=7
)

for i in range(len(cities)):
  folium.CircleMarker(
      location=cities[i],
      radius=10,
      color='red'
  ).add_to(m)

m
```

+
−
Dalian
P'YŎNGAN-NAMDO
Wŏnsan
P'YŎNGYANG
HWANGHAE-BUKTO
Haeju
INCH'ŎN-GWANGYOKSI
Cheongju-si
Leaflet (http://leafletjs.com)

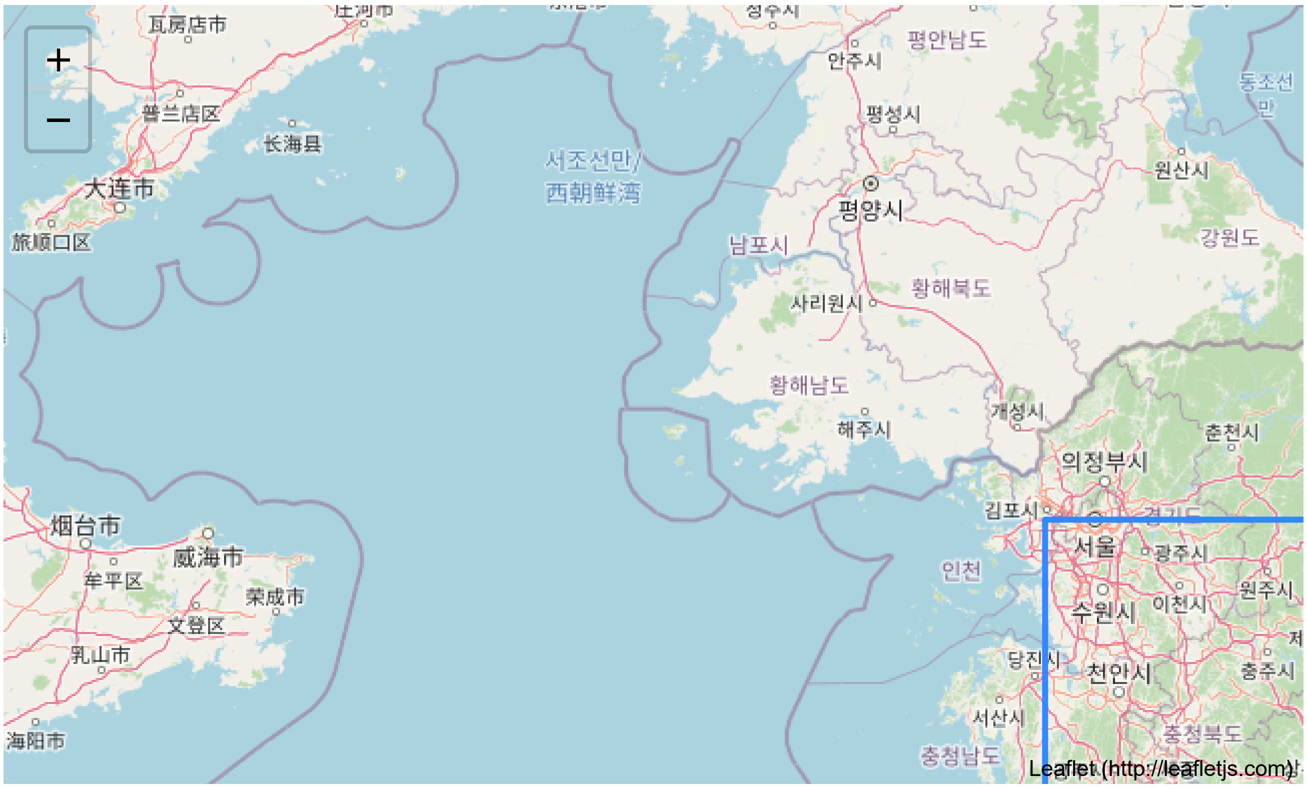
폴리 라인(Poly Line)

```
m = folium.Map(
    location=[36.577629, 127.770135],
    zoom_start=7
)

folium.PolyLine(
    locations=cities,
    tooltip='PolyLine'
).add_to(m)

m
```

+
−
瓦房店市
普兰店区
长海县
大连市
旅顺口区
서조선만/
西朝鲜湾
안주시
평안남도
평성시
평양시
원산시
강원도
동조선만
남포시
사리원시
황해북도
황해남도
해주시
개성시
춘천시
의정부시
김포시
경기도
서울
광주시
인천
수원시
이천시
원주시
당진시
천안시
충주시
서산시
충청남도
충청북도
烟台市
威海市
牟平区
荣成市
文登区
乳山市
海阳市
Leaflet (http://leafletjs.com)

사각형(Rectangle)

```
m = folium.Map(
    location=[36.577629, 127.770135],
    zoom_start=7
)

folium.Rectangle(
    bounds=cities,
    tooltip='Rectangle'
).add_to(m)

m
```

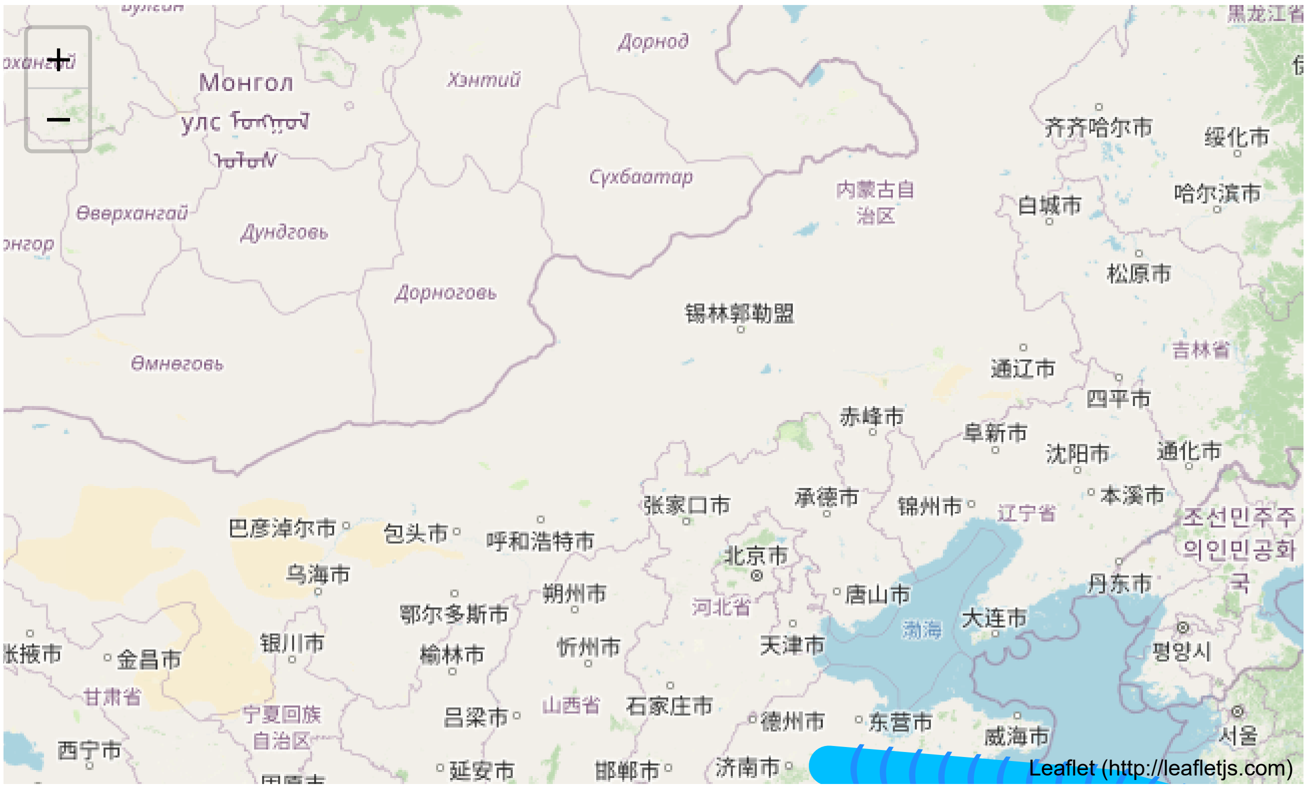


폴리곤(Polygon)

```
m = folium.Map(
    location=[36.577629, 127.770135],
    zoom_start=7
)

folium.Polygon(
    locations=cities,
    fill=True,
    tooltip='Polygon'
).add_to(m)

m
```

PolyLineTextPath

```
m = folium.Map([37, 127], zoom_start=5)

wind_positions=[[32.587385, 133.058046],
                [34.359654, 130.083293],
                [35.570943, 127.200923],
                [36.214572, 124.226170],
                [36.348622, 121.916579],
                [36.645685, 117.981036]]

wind_line=folium.PolyLine(
    wind_positions,
    weight=20,
    color='deepskyblue'
).add_to(m)

plugins.PolyLineTextPath(
    wind_line,
    ') ',
    repeat=True,
    offset=7,
    attributes={'fill': 'dodgerblue', 'font-weight': 'bold', 'font-size': '24'}
).add_to(m)

m
```



```
[illegible]
```

```
marine_currents=[[42.398026, 132.300500],
                 [41.670695, 130.674548],
                 [40.458767, 130.305013],
                 [39.595738, 128.845352],
                 [38.375014, 129.344224],
                 [37.295202, 129.880049],
                 [36.065358, 130.046339]]

m = folium.Map()

folium.plugins.AntPath(
    locations=marine_currents,
    dash_array=[20, 20]
).add_to(m)

m.fit_bounds(m.get_bounds())

m
```

팝업(Popup)

```
m = folium.Map(
    location=[37.566697, 126.978426],
    zoom_start=13
)

m.add_child(folium.LatLngPopup())

m
```

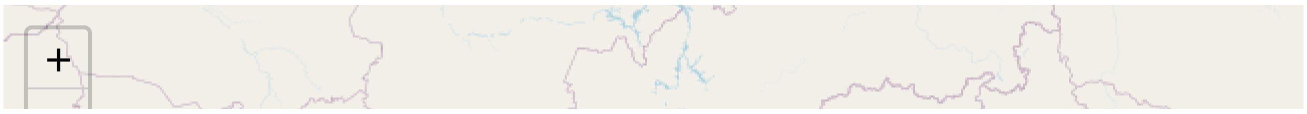
```
m = folium.Map(
    location=[37.566697, 126.978426],
    zoom_start=7
)

html="""
    <h1>Seoul</h1><br>
    <p>
    Seoul, officially the Seoul Special City, is the capital and largest metropolis of South Korea.
    </p>
    <a href="https://en.wikipedia.org/wiki/Seoul" target=_blank>wikipedia</a>
"""

folium.Marker(
    [37.566697, 126.978426],
    popup=html
).add_to(m)

m
```

```
m = folium.Map(
    location=[37.566697, 126.978426],
    zoom_start=7
)
```

```
df = pd.DataFrame(data=[[2000, 9879000],
                        [2010, 9796000],
                        [2020, 9963000]],
                  columns=['Year', 'Pop'])

html = df.to_html(classes='table table-striped table-hover table-condensed table-responsive')

folium.Marker(
    [37.566697, 126.978426],
    popup=html
).add_to(m)

m
```

营口市
盖州市
辽东湾
丹东市
평안북도
大连市
서조선만/西朝鲜湾
평양시
황해북도
황해남도
烟台市
Leaflet (http://leafletjs.com)

```
m = folium.Map(
    location=[37.566697, 126.978426],
    zoom_start=4
)

f = branca.element.Figure()
folium.Map(location=[37.566697, 126.978426], zoom_start=7).add_to(f)
iframe = branca.element.IFrame(width=500, height=300)
f.add_to(iframe)
popup = folium.Popup(iframe, max_width=2650)

folium.Marker(
    [37.566697, 126.978426],
    popup=popup
).add_to(m)

m
```



마커 클러스터(Marker Cluster)

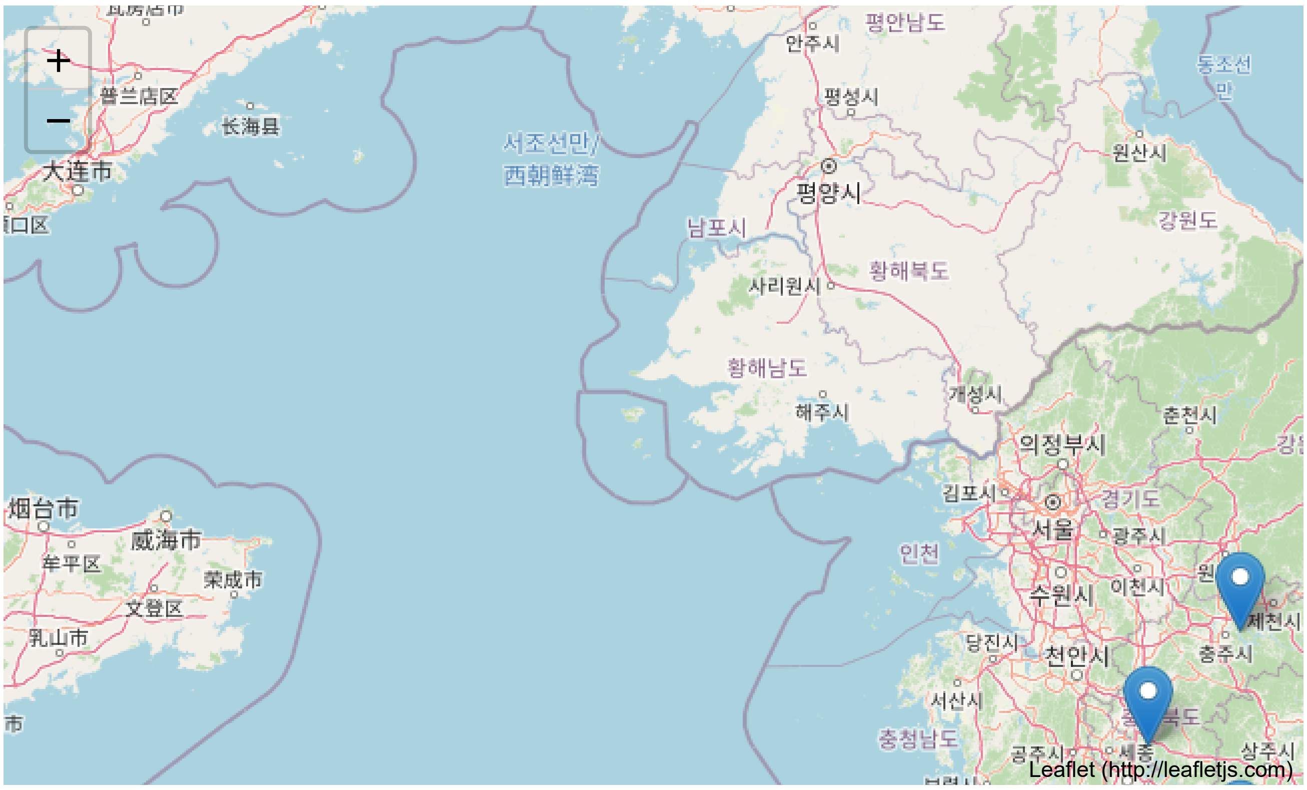
```
N = 100
data = np.array([np.random.uniform(low=35.5, high=37.5, size=N),
                 np.random.uniform(low=127, high=129, size=N),]).T
popups = [str(i) for i in range(N)]

m = folium.Map([36.5, 128], zoom_start=8)
plugins.MarkerCluster(data, popups=popups).add_to(m)

m
```

Vega(베가)

```
!pip install vincent
```

```
Collecting vincent
  Downloading https://files.pythonhosted.org/packages/11/bf/a12ecaa21a2e376a16de67e09f64a38a4
Requirement already satisfied: pandas in /usr/local/lib/python3.6/dist-packages (from vincent
Requirement already satisfied: numpy>=1.13.3 in /usr/local/lib/python3.6/dist-packages (from
Requirement already satisfied: python-dateutil>=2.6.1 in /usr/local/lib/python3.6/dist-packag
Requirement already satisfied: pytz>=2017.2 in /usr/local/lib/python3.6/dist-packages (from p
Requirement already satisfied: six>=1.5 in /usr/local/lib/python3.6/dist-packages (from pytho
Building wheels for collected packages: vincent
  Building wheel for vincent (setup.py) ... done
  Created wheel for vincent: filename=vincent-0.4.4-cp36-none-any.whl size=35172 sha256=b4f2f
  Stored in directory: /root/.cache/pip/wheels/4c/0d/8a/65f34c765c6094a71cce3e42a49a26533eef6
Successfully built vincent
Installing collected packages: vincent
Successfully installed vincent-0.4.4
```

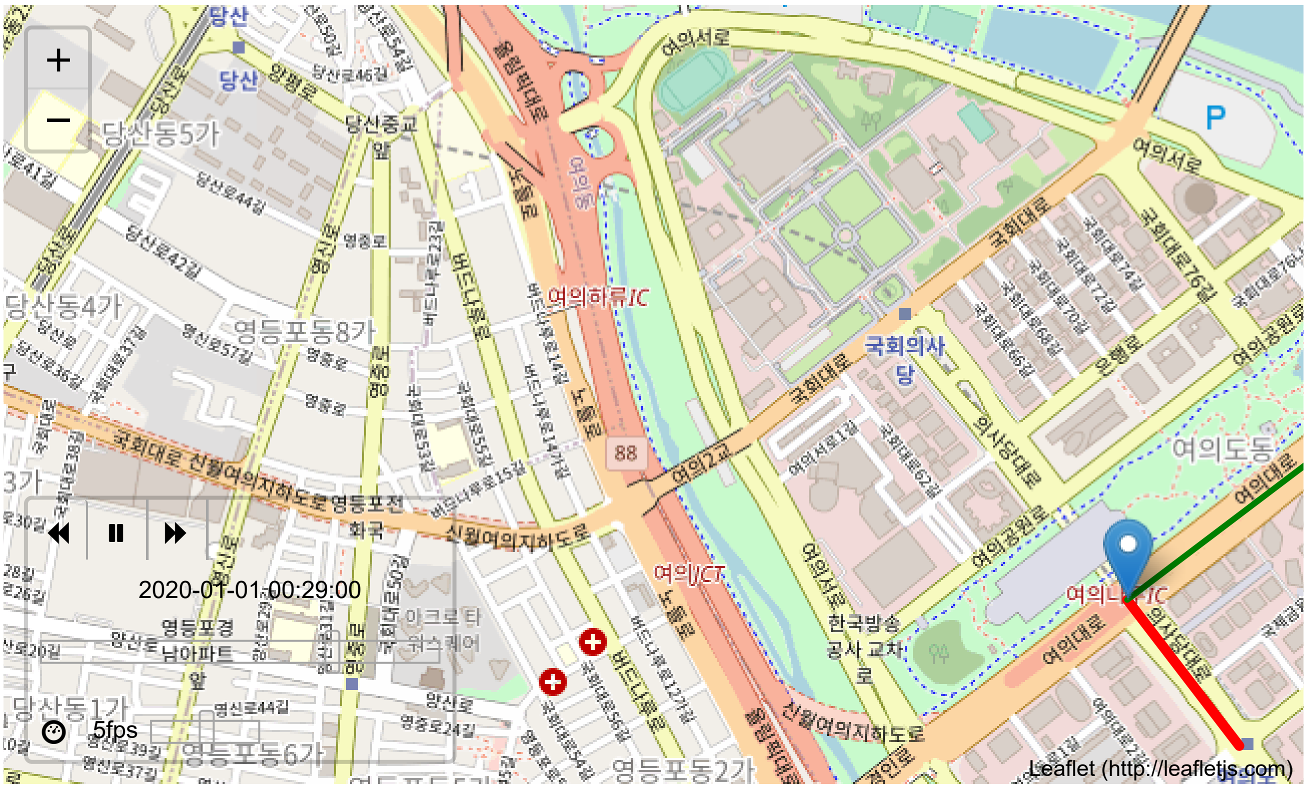
```
import vincent

scatter_points = {
    'x': np.random.randn(50).cumsum(),
    'y': np.random.randn(50).cumsum(),
}

scatter_chart = vincent.Scatter(scatter_points,
                                iter_idx='x',
                                width=400,
                                height=200)
scatter_json = scatter_chart.to_json()
scatter_dict = json.loads(scatter_json)

m = folium.Map([36.5, 128], zoom_start=7)

popup = folium.Popup()
folium.Vega(scatter_chart, height=250, width=450).add_to(popup)
folium.Marker([36, 128], popup=popup).add_to(m)

popup = folium.Popup()
folium.Vega(scatter_json, height=250, width=450).add_to(popup)
folium.Marker([37, 128], popup=popup).add_to(m)

popup = folium.Popup()
folium.Vega(scatter_dict, height=250, width=450).add_to(popup)
folium.Marker([36.5, 127.5], popup=popup).add_to(m)

m
```



- https://raw.githubusercontent.com/python-visualization/folium/master/examples/data
    - vis1.json - [47.3489, -124.708]
    - vis2.json - [44.639, -124.5339]
    - vis3.json - [46.216, -124.1280]

```
url = 'https://raw.githubusercontent.com/python-visualization/folium/master/examples/data'
vis1 = json.loads(requests.get(f'{url}/vis1.json').text)
vis2 = json.loads(requests.get(f'{url}/vis2.json').text)
vis3 = json.loads(requests.get(f'{url}/vis3.json').text)

m = folium.Map(
    location=[47.3489, -124.708],
    zoom_start=7,
    tiles='Stamen Terrain'
)

folium.Marker(
    [47.3489, -124.708],
    popup=folium.Popup(max_width=450).add_child(
        folium.Vega(vis1, width=450, height=250))
).add_to(m)

folium.Marker(
    [44.639, -124.5339],
    popup=folium.Popup(max_width=450).add_child(
        folium.Vega(vis2, width=450, height=250))
).add_to(m)

folium.Marker(
    [46.216, -124.1280],
    popup=folium.Popup(max_width=450).add_child(
        folium.Vega(vis3, width=450, height=250))
).add_to(m)

m
```

```
m = folium.Map(
    location=[37.521628, 126.924127],
    zoom_start=15
)

lines = [
        { 'coordinates': [[126.924127, 37.521628],
                          [126.921754, 37.524117],],
         'dates': ['2020-01-01T00:00:00', '2020-01-01T00:10:00'],
         'color': 'red'
        },
        { 'coordinates': [[126.921754, 37.524117],
                          [126.926168, 37.526768],],
         'dates': ['2020-01-01T00:10:00', '2020-01-01T00:20:00'],
         'color': 'green', 'weight': 3,
        },
        { 'coordinates': [[126.926168, 37.526768],
                          [126.928490, 37.524340],],
         'dates': ['2020-01-01T00:20:00', '2020-01-01T00:30:00'],
         'color': 'blue', 'weight': 10,
        },
        { 'coordinates': [[126.928490, 37.524340],
                          [126.924127, 37.521628],],
         'dates': ['2020-01-01T00:30:00', '2020-01-01T00:40:00'],
         'color': 'black'
        },
]

features = [
           {'type': 'Feature',
            'geometry': {
                'type': 'LineString',
                'coordinates': line['coordinates'],
            },
            'properties': {
                'times': line['dates'],
                'style': {
                    'color': line['color'],
                    'weight': line['weight'] if 'weight' in line else 5
                }
            }
           } for line in lines
]

plugins.TimestampedGeoJson({
    'type': 'FeatureCollection',
    'features': features,
}, period='PT1M', add_last_point=True).add_to(m)
}, period='PT1M', add_last_point=True).add_to(m)

m
```



스타일(Style)

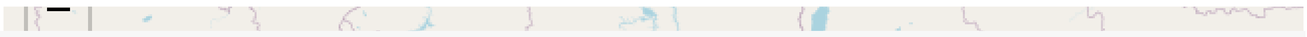
타일(Tiles)

- Map Tiles
  - OpenStreetMap

    - Stamen Terrain
    - Stamen Toner
    - Stamen Watercolor
    - CartoDB positron
    - CartoDB dark_matter

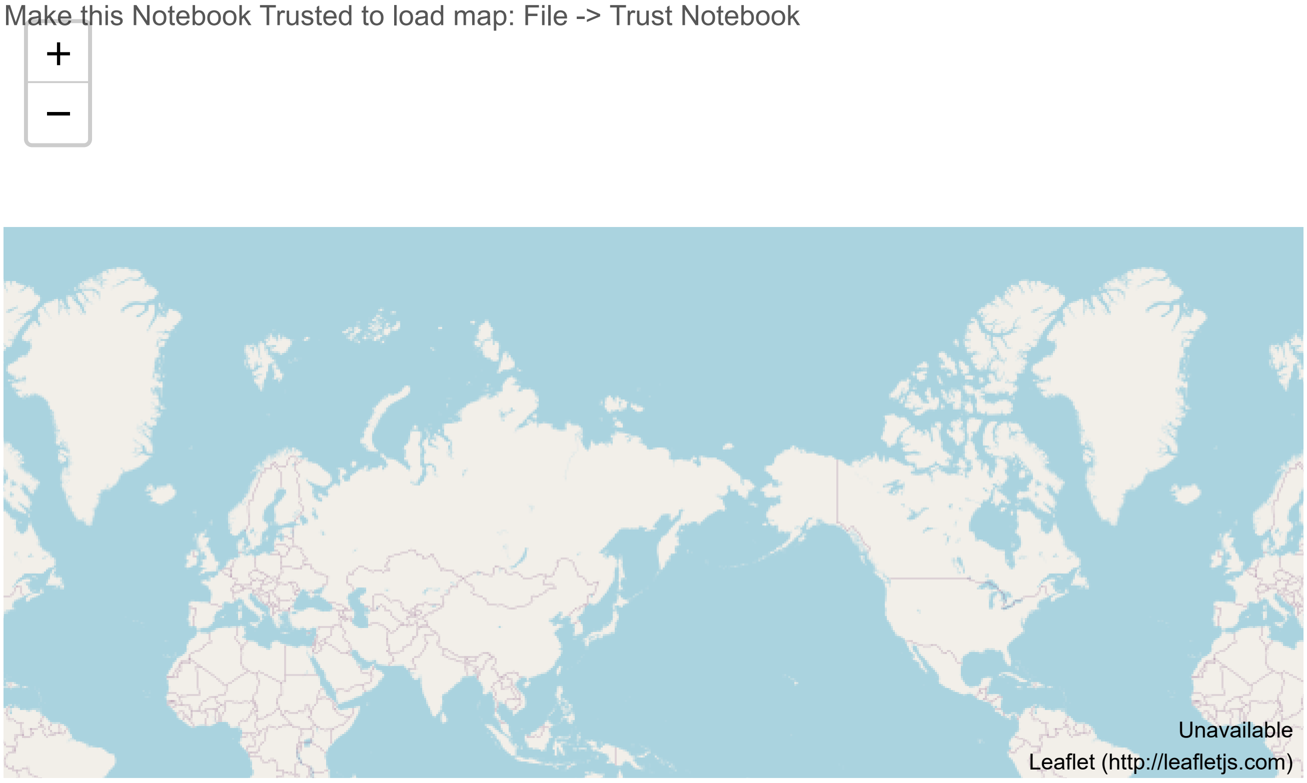
```
m = folium.Map(
    location=[37, 128],
    tiles='OpenStreetMap',
    zoom_start=4
)

m
```

위치(Position)



```
from folium.plugins import MousePosition

m = folium.Map()
MousePosition().add_to(m)

m
```

Make this Notebook Trusted to load map: File -> Trust Notebook

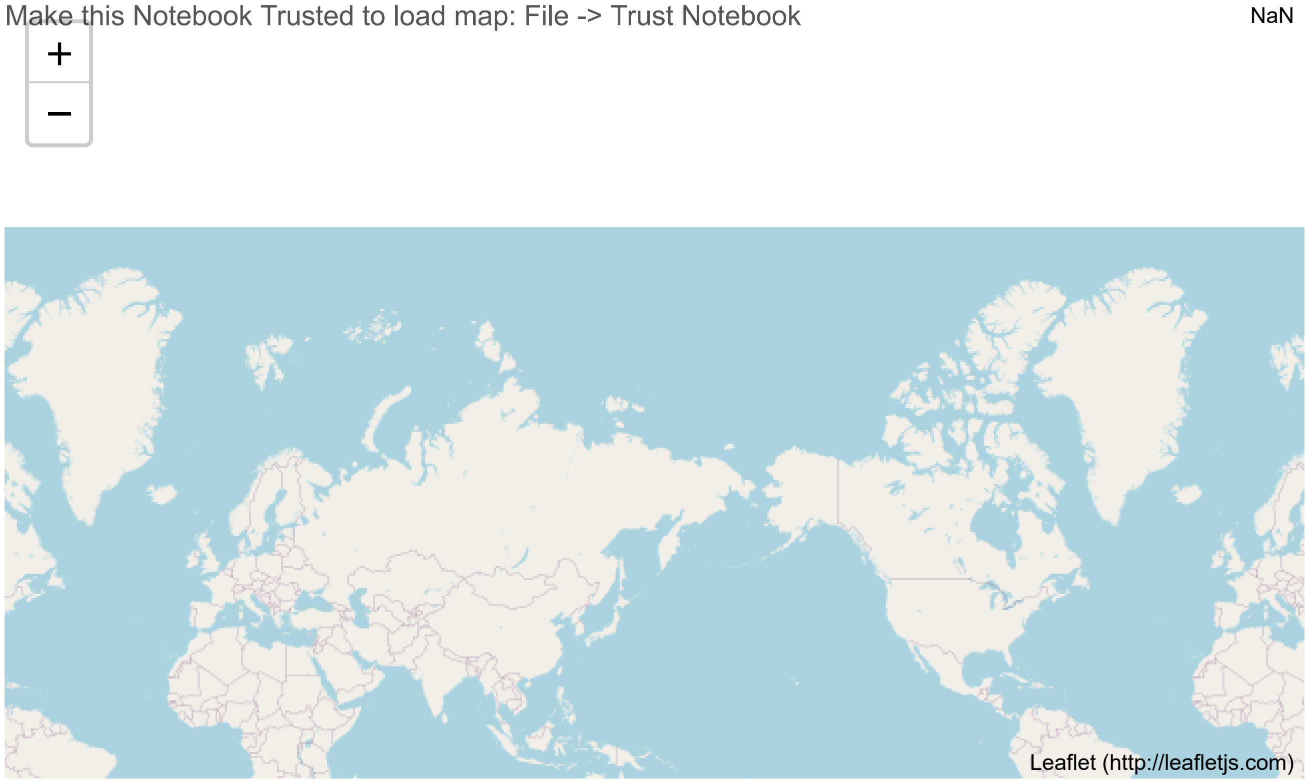


```
m = folium.Map()
formatter = "function(num) {return L.Util.formatNum(num, 3);};"
```

```
MousePosition(
    position='topright',
    separator=' | ',
    empty_string='NaN',
    lng_first=True,
    num_digits=20,
    prefix='Coordinates: ',
    lat_formatter=formatter,
    lng_formatter=formatter
).add_to(m)

m
```

Make this Notebook Trusted to load map: File -> Trust Notebook

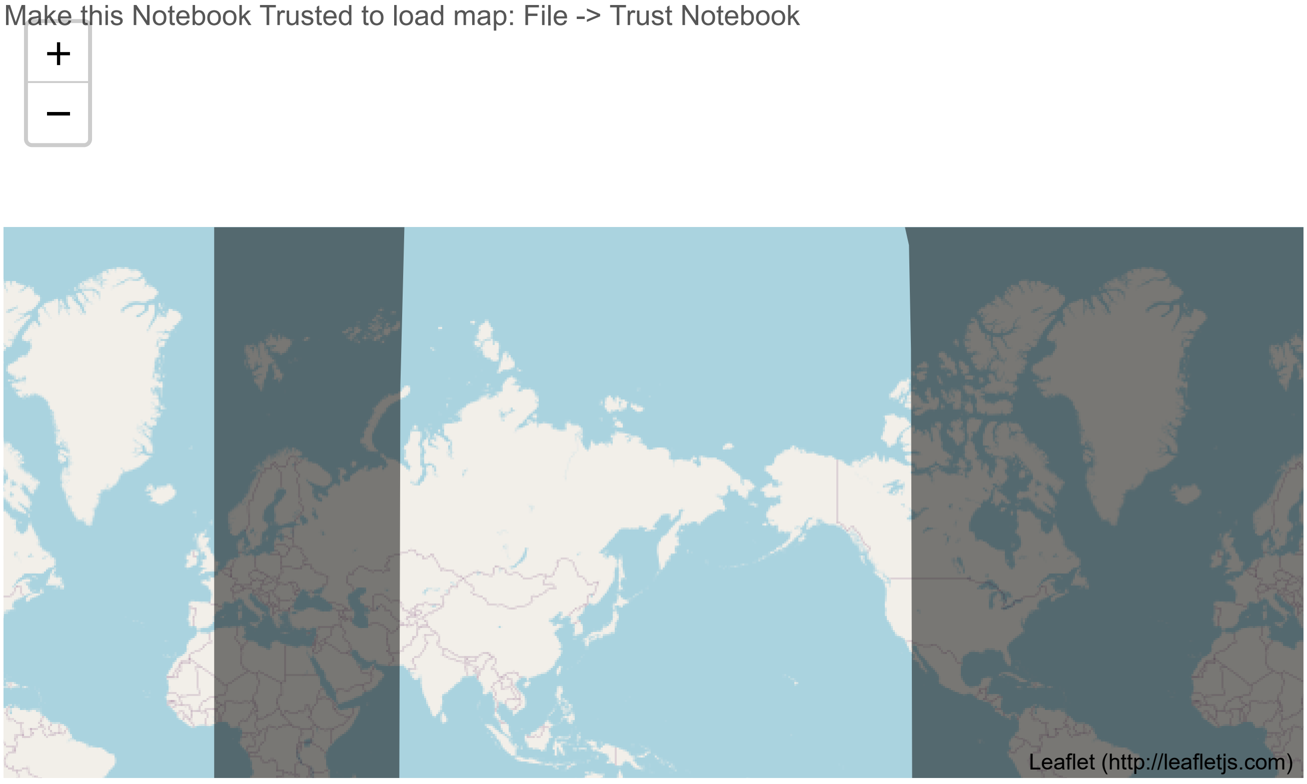
NaN



터미네이터(Terminator)

```
m = folium.Map(zoom_start=1)

plugins.Terminator().add_to(m)

m
```

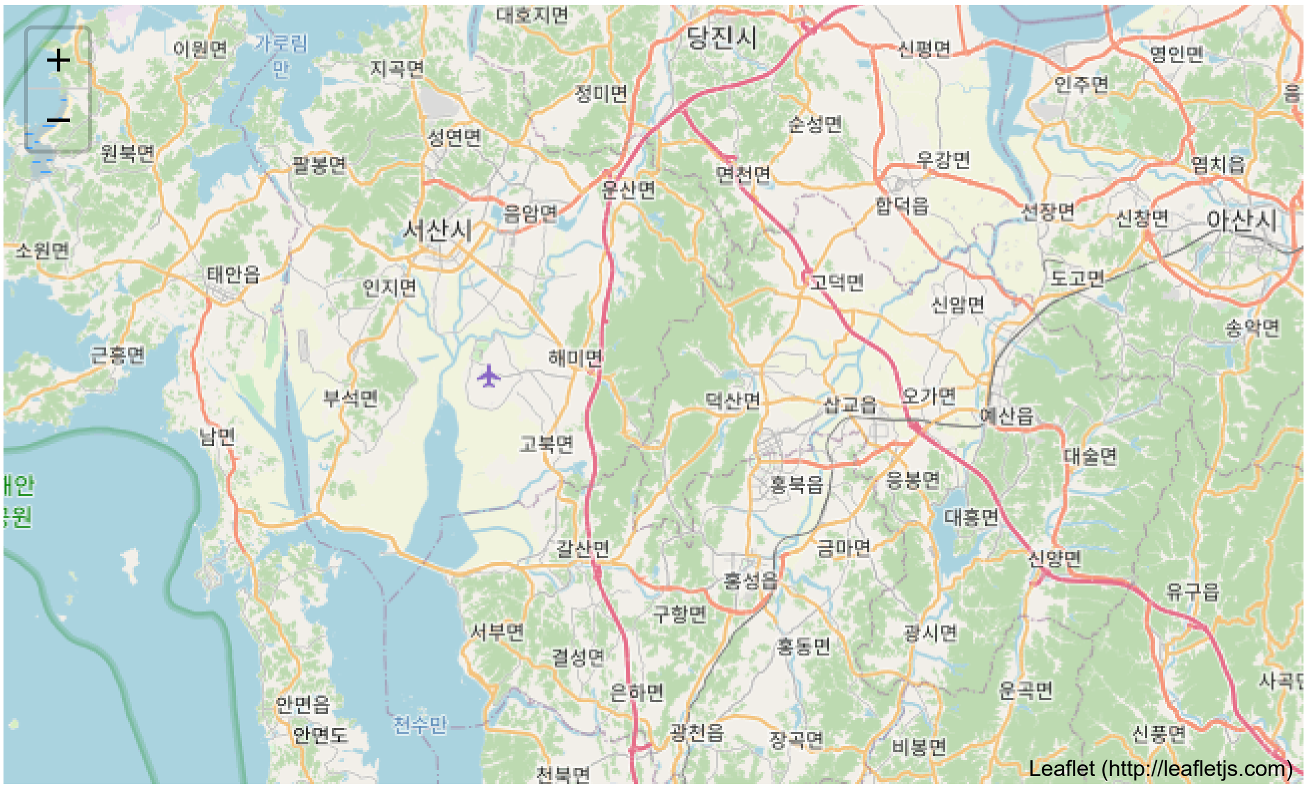
Make this Notebook Trusted to load map: File -> Trust Notebook



측정 제어(Measure Control)

```
from folium.plugins import MeasureControl

m = folium.Map([36.5, 127], zoom_start=10)
```

```
m.add_child(MeasureControl())

m
```

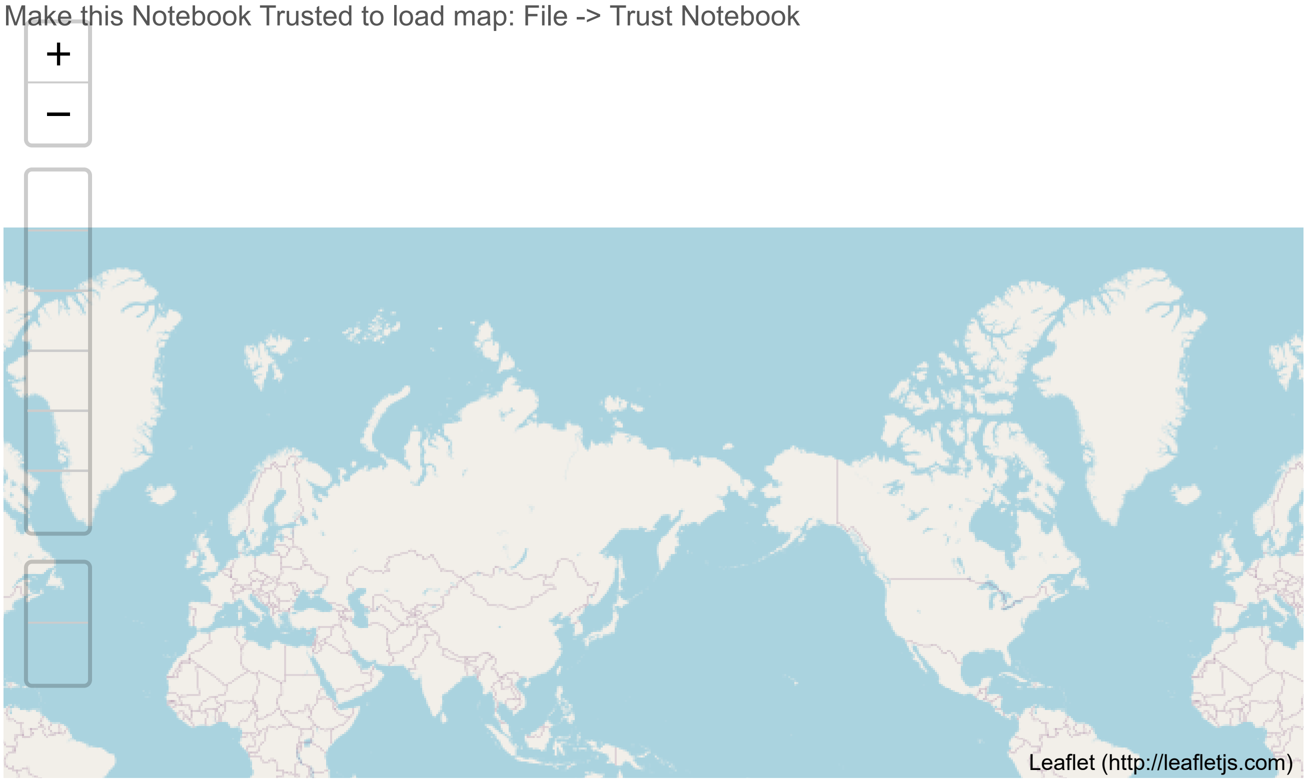


그리기(Draw)

```
from folium.plugins import Draw

m = folium.Map()
draw = Draw()
draw.add_to(m)

m
```

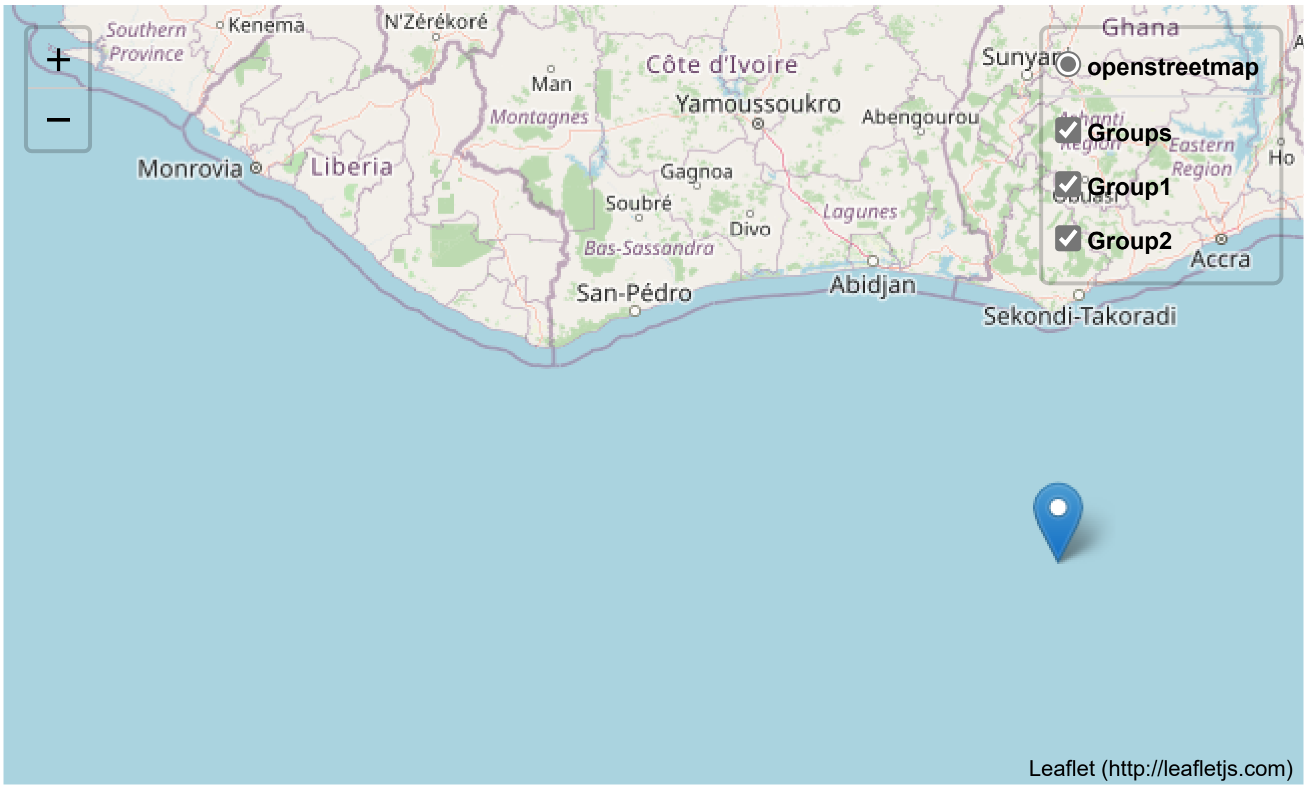
```
...
```

Make this Notebook Trusted to load map: File -> Trust Notebook



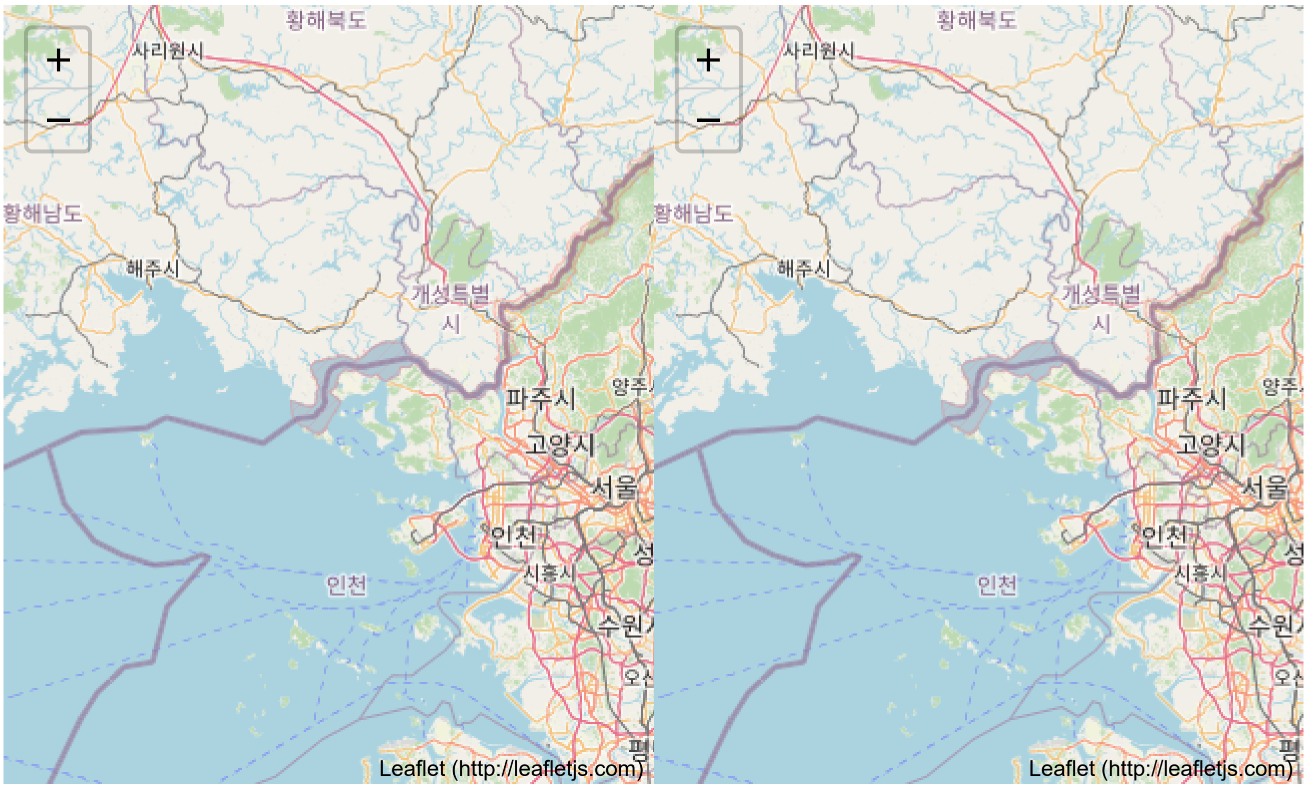
그룹(Group)

```
m = folium.Map(location=[0, 0], zoom_start=6)

gs = folium.FeatureGroup(name='Groups')
m.add_child(gs)
g1 = plugins.FeatureGroupSubGroup(gs, 'Group1')
m.add_child(g1)
g2 = plugins.FeatureGroupSubGroup(gs, 'Group2')
m.add_child(g2)

folium.Marker([-2, -2]).add_to(g1)
folium.Marker([2, 2]).add_to(g1)
folium.Marker([-2, 2]).add_to(g2)
folium.Marker([2, -2]).add_to(g2)
folium.LayerControl(collapsed=False).add_to(m)

m
```

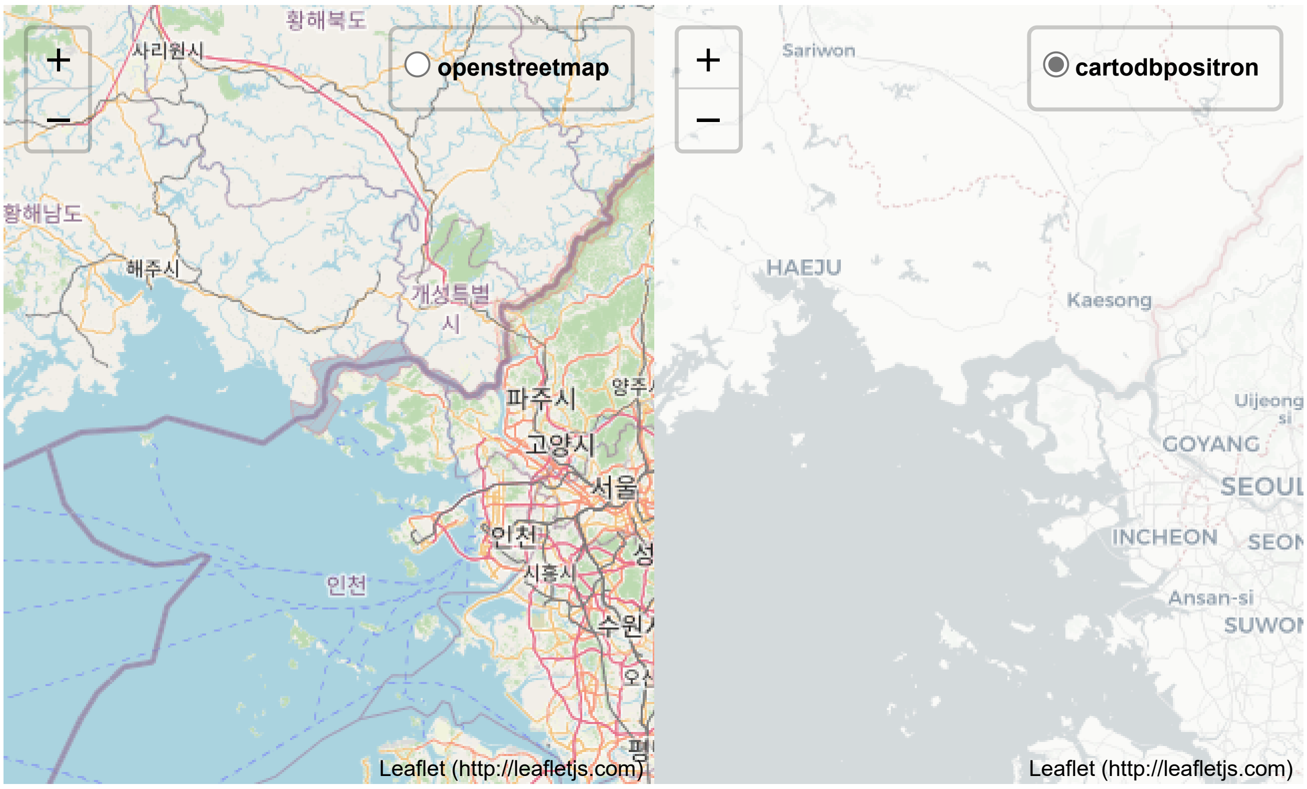


듀얼맵(Dualmap)

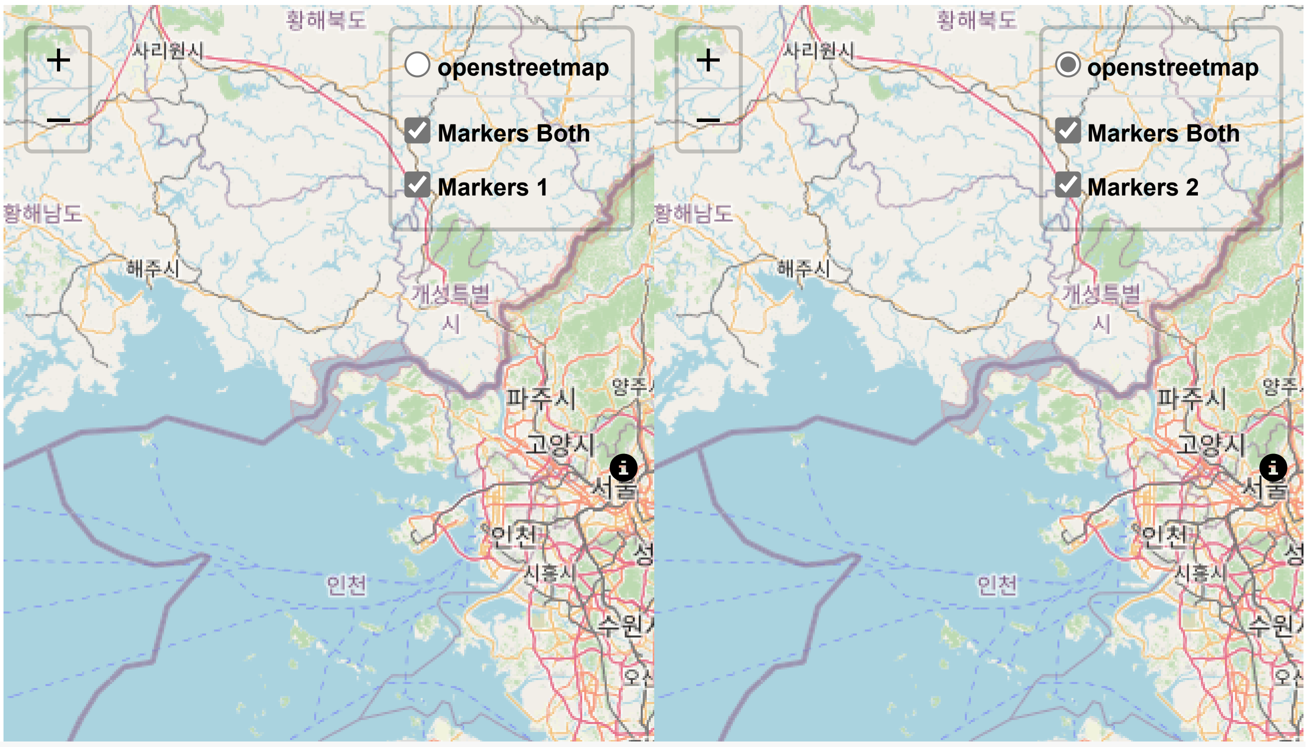
```
m = folium.plugins.DualMap(location=(37, 127), zoom_start=8)
m
```



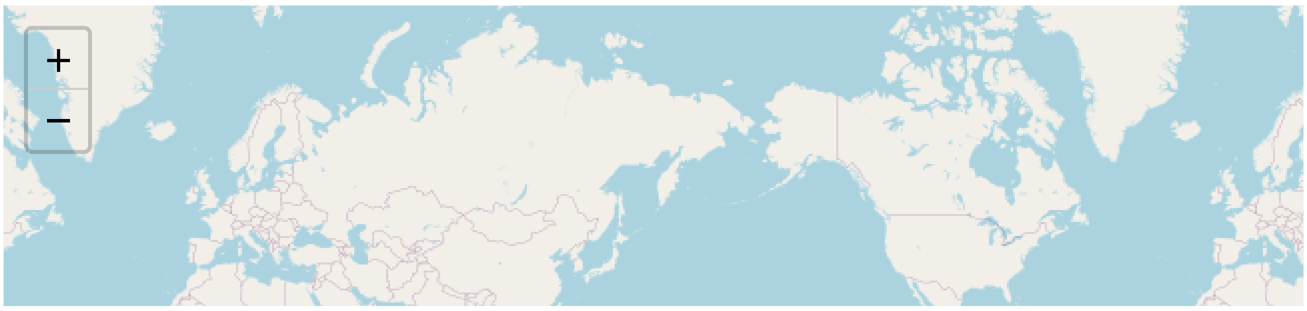
```
m = folium.plugins.DualMap(location=(37, 127), tiles=None, zoom_start=8)

folium.TileLayer('OpenStreetMap').add_to(m.m1)
folium.TileLayer('CartoDBPositron').add_to(m.m2)
folium.LayerControl(collapsed=False).add_to(m)

m
```



```
m = folium.plugins.DualMap(location=(37, 127), zoom_start=8)

fg_both = folium.FeatureGroup(name='Markers Both').add_to(m)
fg_1 = folium.FeatureGroup(name='Markers 1').add_to(m.m1)
fg_2 = folium.FeatureGroup(name='Markers 2').add_to(m.m2)
icon_red = folium.Icon(color='red')
folium.Marker((37.5, 127), tooltip='Both', icon=icon_red).add_to(fg_both)
folium.Marker((37, 127.5), tooltip='1').add_to(fg_1)
folium.Marker((36.5, 127), tooltip='2').add_to(fg_2)
folium.LayerControl(collapsed=False).add_to(m)

m
```

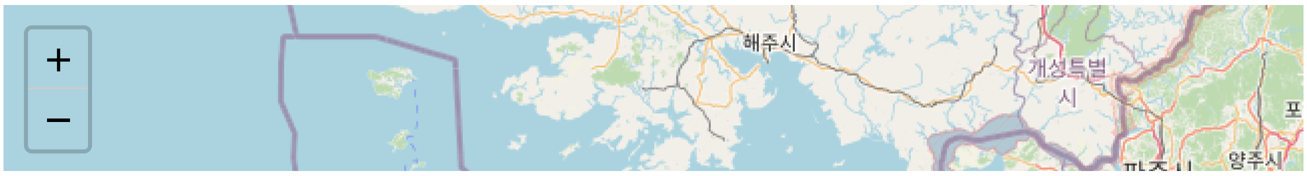


```
m = folium.plugins.DualMap(layout='vertical')
m
```

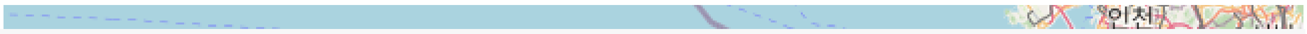


미니맵(Minimap)

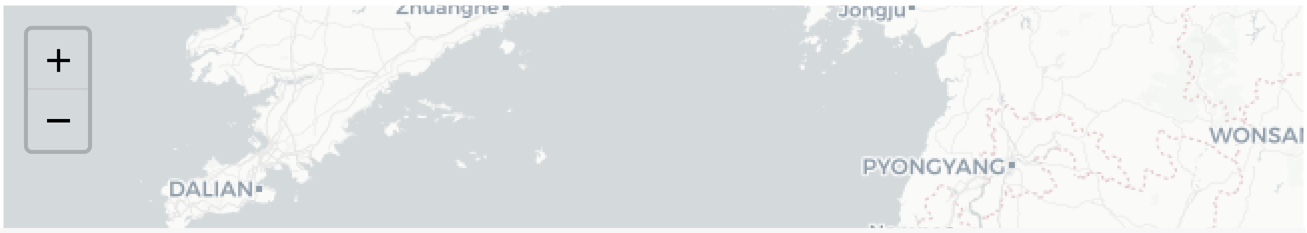
```
m = folium.Map(location=(36.5, 127), zoom_start=8)

minimap = plugins.MiniMap()
m.add_child(minimap)

m
```



히트맵(Heatmap)



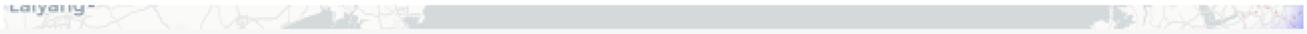
```
from folium.plugins import HeatMap

m = folium.Map(
    location=(36.5, 127),
    zoom_start=7,
    tiles = 'CartoDB Positron'
)

HeatMap(cities).add_to(m)

m
```



```
data = (
    np.random.normal(size=(100, 3)) *
    np.array([[1, 0.5, 1]]) +
    np.array([[37, 128, 1]])
)

m = folium.Map(
    location=(36.5, 127),
    zoom_start=7,
    tiles = 'CartoDB Positron'
)

HeatMap(data).add_to(m)

m
```

GeoJson, TopoJson, Choropleth maps

- https://raw.githubusercontent.com/python-visualization/folium/master/examples/data
    - us-states.json
    - US_Unemployment_Oct2012.csv

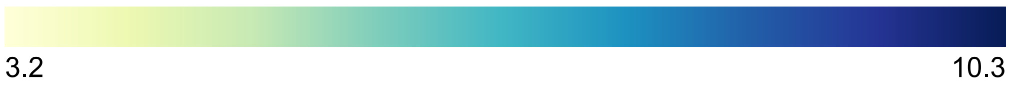


```
url = 'https://raw.githubusercontent.com/python-visualization/folium/master/examples/data'
state_geo = f'{url}/us-states.json'
state_unemployment = f'{url}/US_Unemployment_Oct2012.csv'
state_data = pd.read_csv(state_unemployment)
state_data
```

	State	Unemployment
0	AL	7.1
1	AK	6.8
2	AZ	8.1
3	AR	7.2
4	CA	10.1
5	CO	7.7
6	CT	8.4
7	DE	7.1
8	FL	8.2
9	GA	8.8
10	HI	5.4
11	ID	6.6
12	IL	8.8
13	IN	8.4
14	IA	5.1
15	KS	5.6
16	KY	8.1
17	LA	5.9
18	ME	7.2
19	MD	6.8
20	MA	6.7
21	MI	9.1
22	MN	5.6
23	MS	9.1
24	MO	6.7
25	MT	5.8
26	NE	3.9
27	NV	10.3
28	NH	5.7
29	NJ	9.6
30	NM	6.8

```
from branca.colormap import linear
```

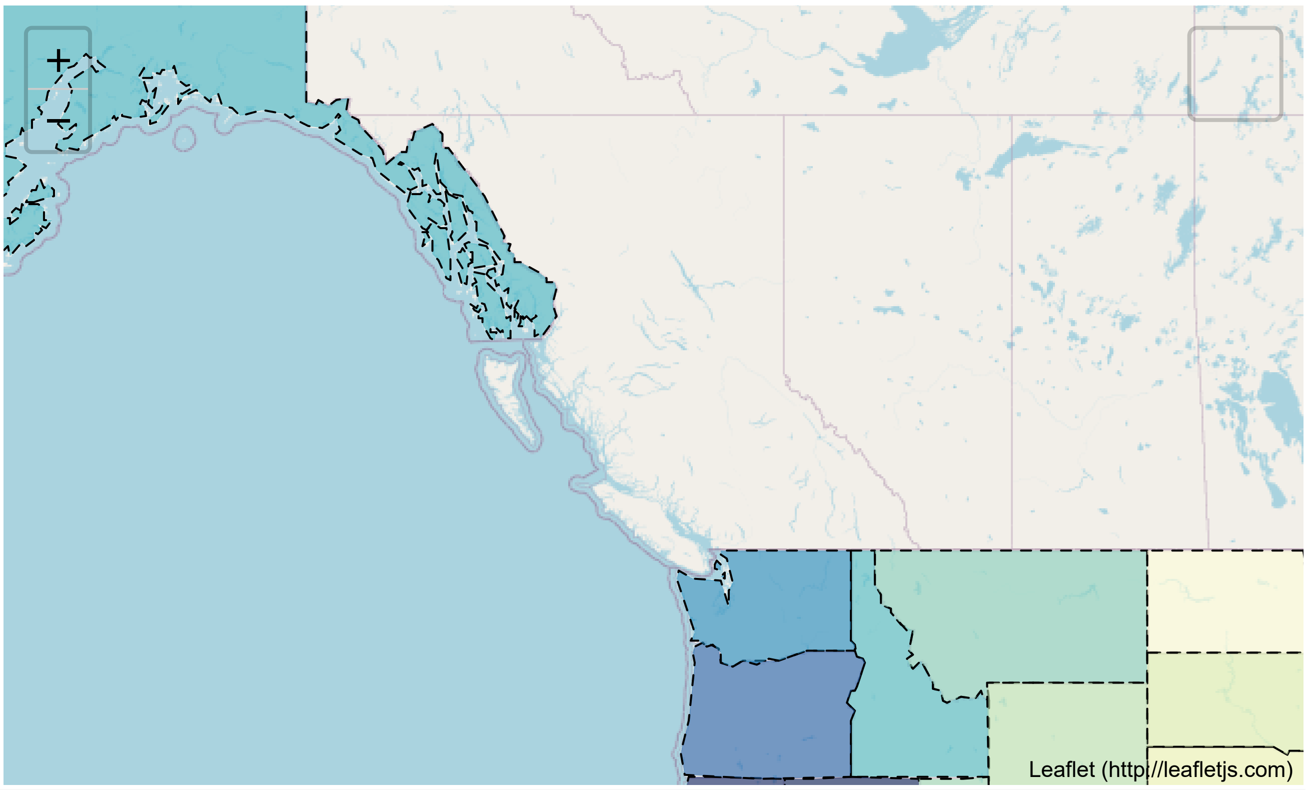
```
colormap = linear.YlGnBu_09.scale(
    state_data.Unemployment.min(),
    state_data.Unemployment.max()
)

colormap
```



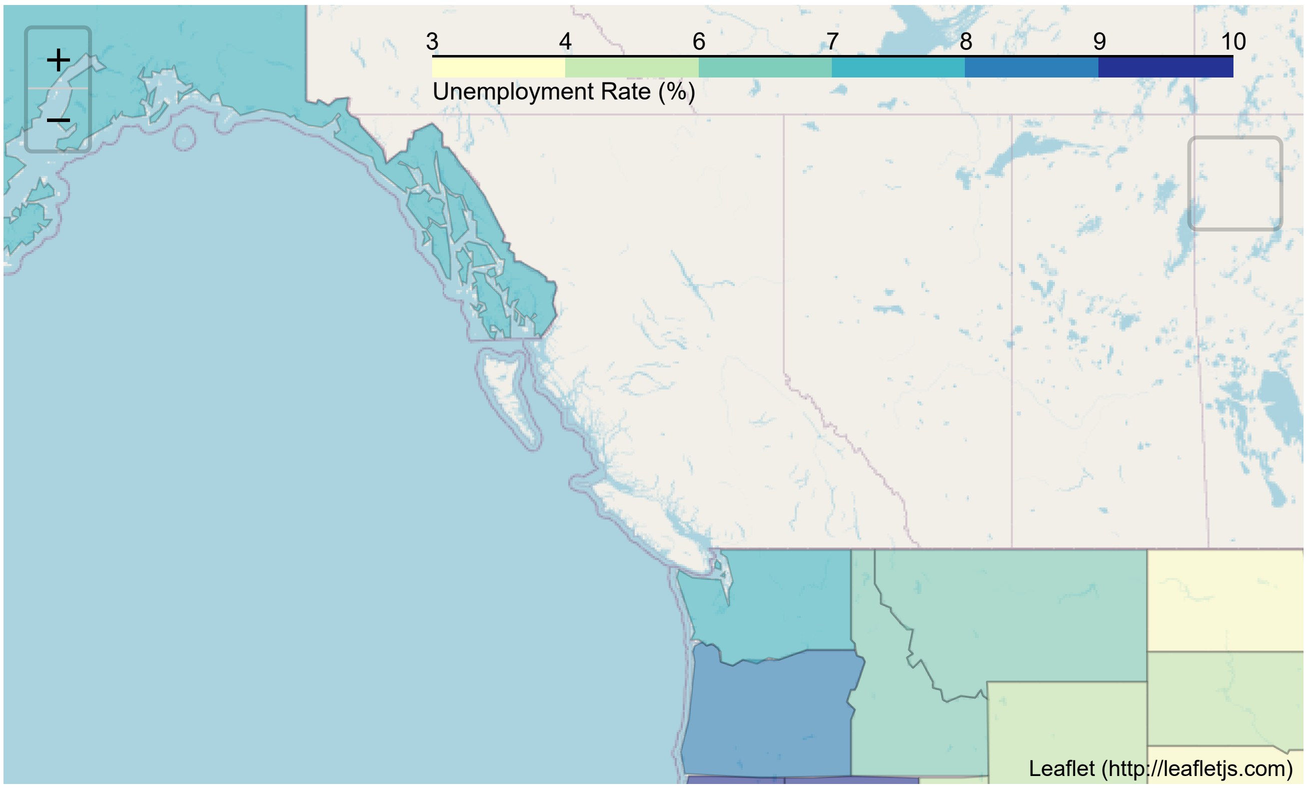
```
state_data_dict = state_data.set_index('State')['Unemployment']
state_data_dict['AL']
```

```
7.1
```

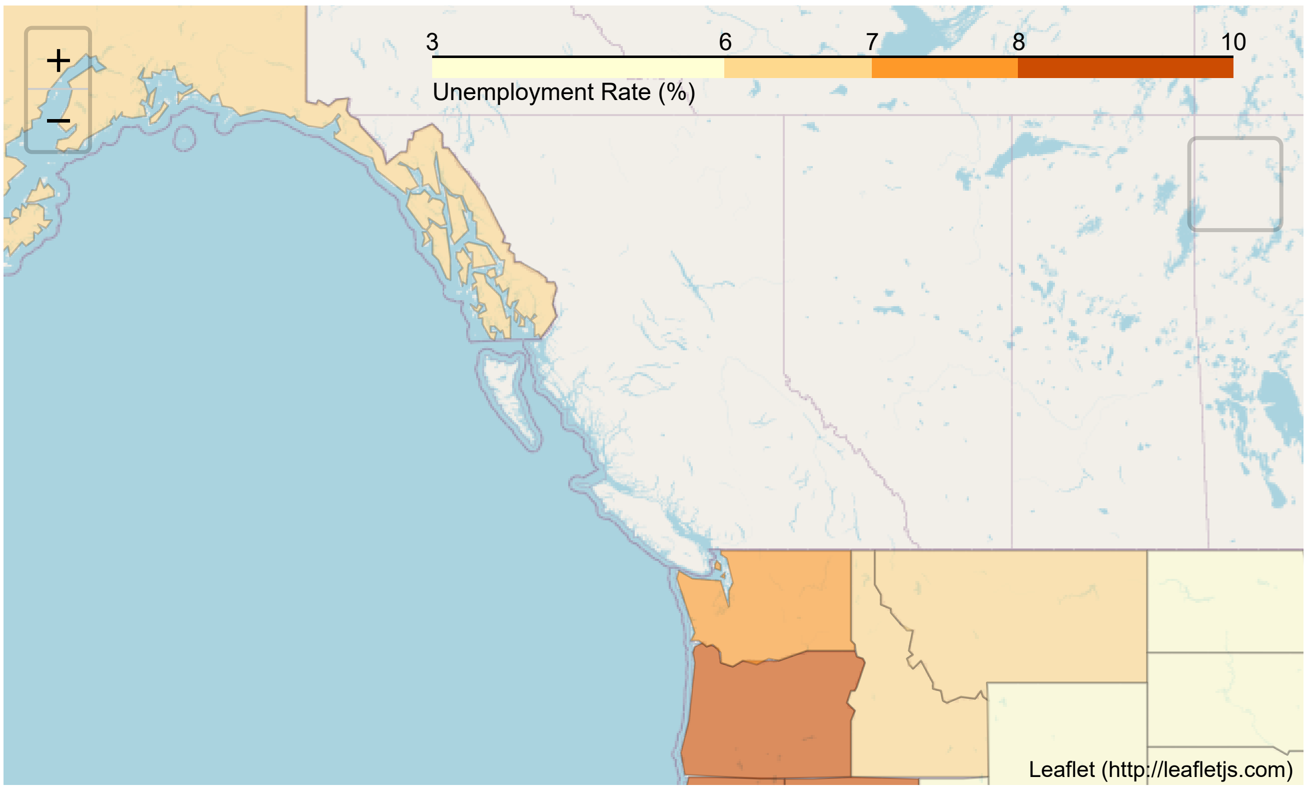
```
m = folium.Map([43, -100], zoom_start=4)

folium.GeoJson(
    state_geo,
    name='unemployment',
    style_function=lambda feature: {
        'fillColor': colormap(state_data_dict[feature['id']]),
        'color': 'black',
        'weight': 1,
        'dashArray': '5, 5',
        'fillOpacity': 0.6,
    }
).add_to(m)

folium.LayerControl().add_to(m)

m
```



```
m = folium.Map([43, -100], zoom_start=4)

folium.Choropleth(
    geo_data=state_geo,
    data=state_data,
    columns=['State', 'Unemployment'],
    key_on='feature.id',
    fill_color='YlGnBu',
    fill_opacity=0.6,
    line_opacity=0.2,
    legend_name='Unemployment Rate (%)'
).add_to(m)

folium.LayerControl().add_to(m)

m
```



```
bins = list(state_data['Unemployment'].quantile([0, 0.25, 0.5, 0.75, 1]))

m = folium.Map([43, -100], zoom_start=4)

folium.Choropleth(
    geo_data=state_geo,
    data=state_data,
    columns=['State', 'Unemployment'],
    key_on='feature.id',
    fill_color='YlOrBr',
    fill_opacity=0.6,
    line_opacity=0.2,
    legend_name='Unemployment Rate (%)',
    bins=bins,
).add_to(m)

folium.LayerControl().add_to(m)

m
```

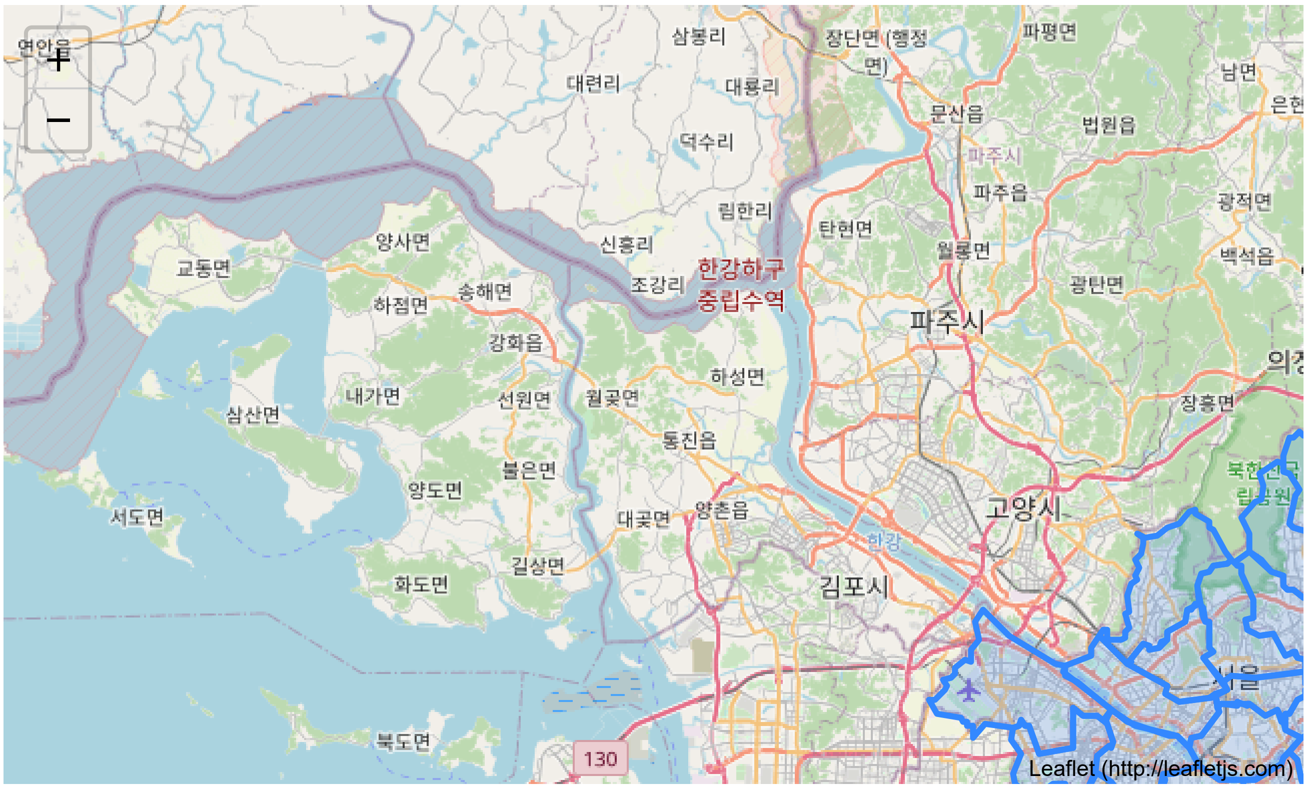


- South Korea github: https://github.com/southkorea
- 서울 열린데이터 광장: https://data.seoul.go.kr/
- 서울시 지도: https://raw.githubusercontent.com/suanlab/dataset/master/seoul_municipalities_geo_simple.json
- 서울시 인구수: https://raw.githubusercontent.com/suanlab/dataset/master/seoul_population.csv

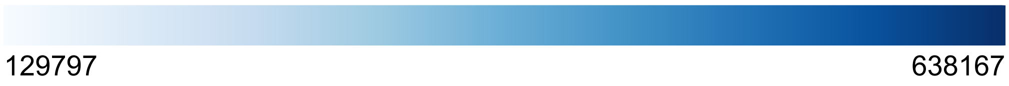
```
url = 'https://raw.githubusercontent.com/suanlab/dataset/master'
seoul_geo = f'{url}/seoul_municipalities_geo_simple.json'
seoul_population = f'{url}/seoul_population.csv'
seoul_data = pd.read_csv(seoul_population, encoding='utf-8')
seoul_data
```

	name	population
0	종로구	157967
1	중구	129797
2	용산구	226938
3	성동구	306796
4	광진구	362304
5	동대문구	358141
6	중랑구	391668
7	성북구	438734
8	강북구	309138
9	도봉구	328243
10	노원구	534096
11	은평구	462552
12	서대문구	318874
13	마포구	368181
14	양천구	445591
15	강서구	578539
16	구로구	433765
17	금천구	249344
18	영등포구	395286
19	동작구	397980
20	관악구	510303
21	서초구	409491

```
m = folium.Map(
    location=[37.528043, 126.980238],
    zoom_start=10
)

folium.GeoJson(
    json.loads(requests.get(seoul_geo).text),
    name='seoul_municipalities'
).add_to(m)

m
```



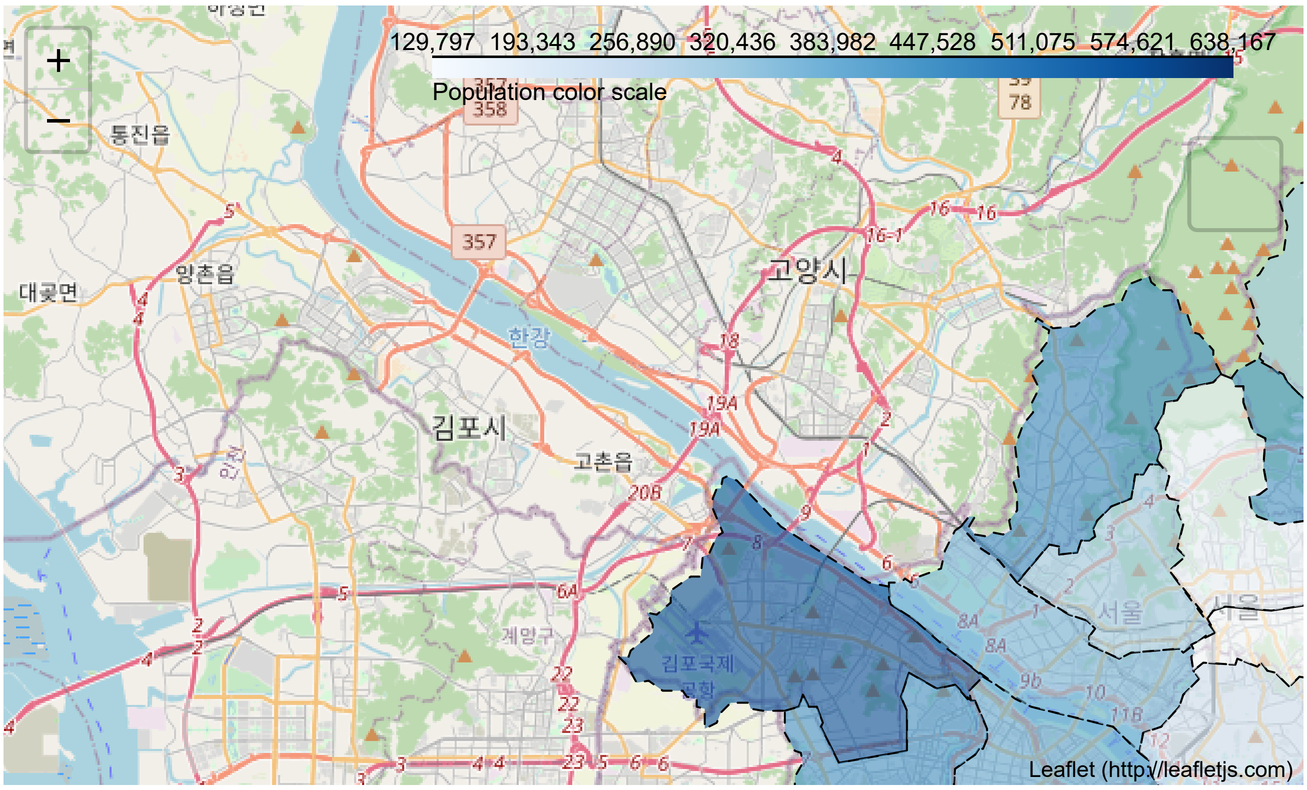
```
colormap = linear.Blues_09.scale(
    seoul_data.population.min(),
    seoul_data.population.max()
)

colormap
```



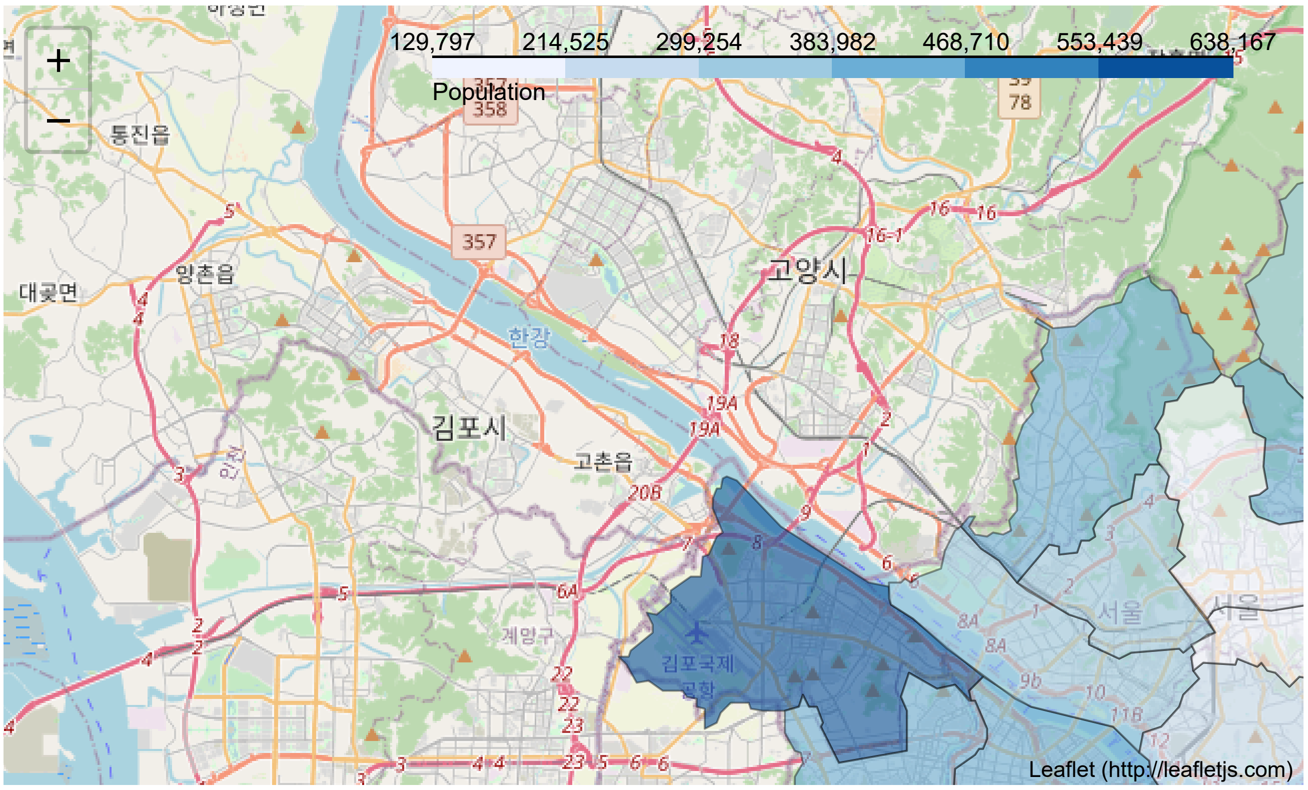
```
population_dict = seoul_data.set_index('name')['population']
color_dict = {str(key): colormap(population_dict[key]) for key in population_dict.keys()}
color_dict
```

```
{'강남구': '#2273b6ff',
 '강동구': '#57a1cfff',
 '강북구': '#a5cde4ff',
 '강서구': '#084f99ff',
 '관악구': '#2171b5ff',
 '광진구': '#7cb8daff',
 '구로구': '#4b98caff',
 '금천구': '#c9ddf0ff',
 '노원구': '#1865acff',
 '도봉구': '#98c7e0ff',
 '동대문구': '#80badbff',
 '동작구': '#62a8d3ff',
 '마포구': '#78b5d9ff',
 '서대문구': '#9fcbe2ff',
 '서초구': '#5aa3d0ff',
 '성동구': '#a7cee4ff',
 '성북구': '#4796c8ff',
 '송파구': '#08306bff',
 '양천구': '#4393c7ff',
 '영등포구': '#64a9d3ff',
 '용산구': '#d2e3f3ff',
 '은평구': '#3a8ac2ff',
 '종로구': '#ecf4fcff',
 '중구': '#f7fbffff',
 '중랑구': '#66abd4ff'}
```

```
m = folium.Map(
    location=[37.528043, 126.980238],
    zoom_start=11
)

folium.GeoJson(
    seoul_geo,
    name='population',
    style_function=lambda feature: {
        'fillColor': color_dict[feature['properties']['name']],
        'color': 'black',
        'weight': 1,
        'dashArray': '5, 5',
        'fillOpacity': 0.6,
    }
).add_to(m)

colormap.caption='Population color scale'
colormap.add_to(m)

folium.LayerControl().add_to(m)

m
```



```
m = folium.Map(
    location=[37.528043, 126.980238],
    zoom_start=11
)

folium.Choropleth(
    geo_data=seoul_geo,
    data=seoul_data,
    columns=['name', 'population'],
    key_on='properties.name',
    fill_color='Blues',
    fill_opacity=0.6,
    line_opacity=0.4,
    legend_name='Population'
).add_to(m)

m
```

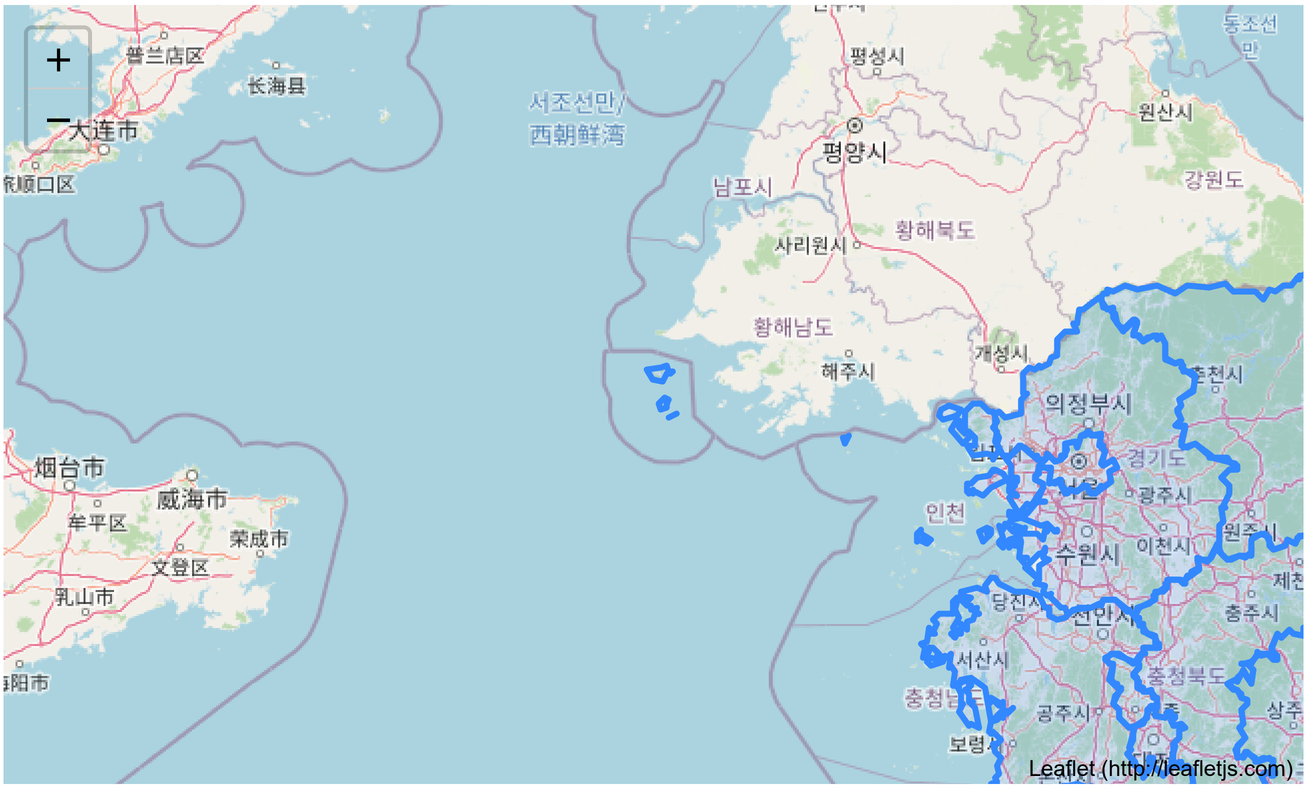


- South Korea github: https://github.com/southkorea
- 국가통계포털: http://kosis.kr
- 전국시도 지도: https://raw.githubusercontent.com/suanlab/dataset/master/skorea-provinces-2018-geo.json
- 전국시도 인구수: https://raw.githubusercontent.com/suanlab/dataset/master/skorea_provinces_population.csv

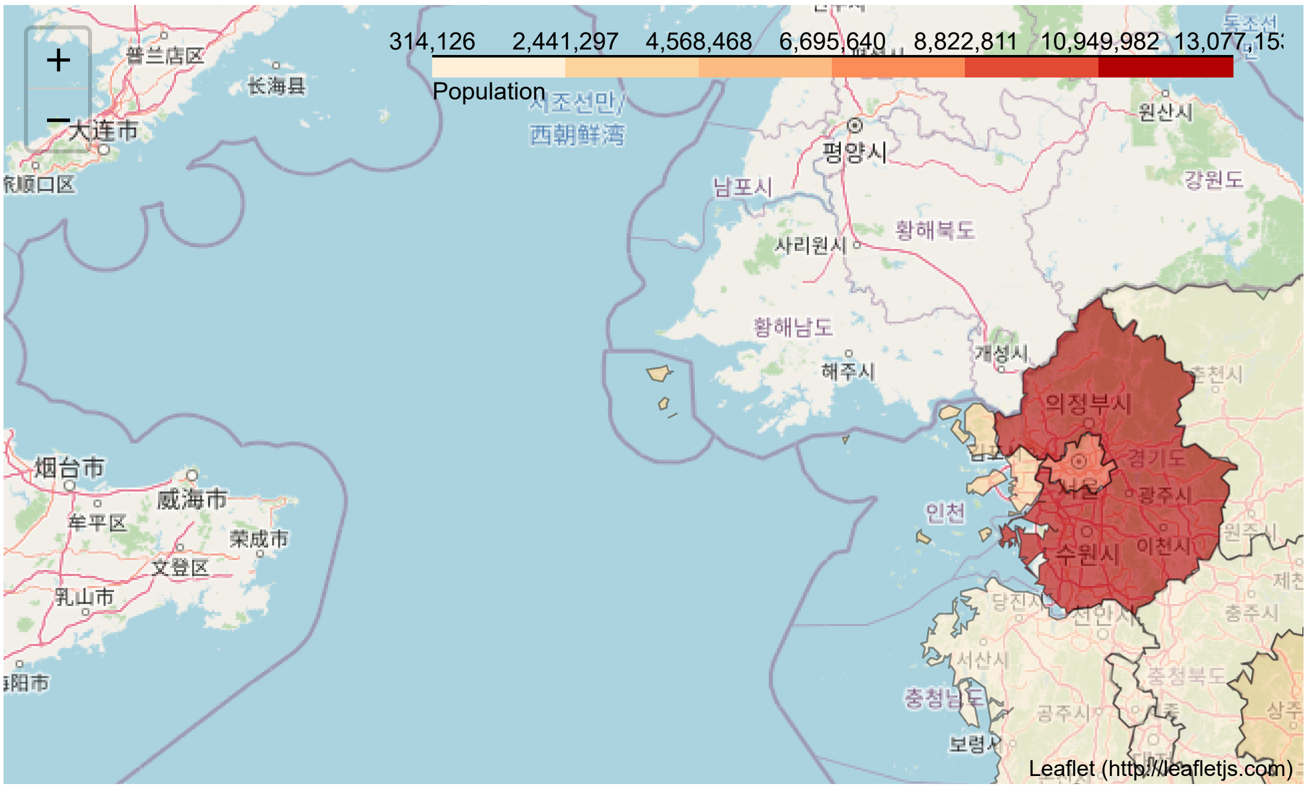
```
url = 'https://raw.githubusercontent.com/suanlab/dataset/master'
skorea_provinces_geo = f'{url}/skorea-provinces-2018-geo.json'
skorea_provinces_population = f'{url}/skorea_provinces_population.csv'
skorea_provinces_df = pd.read_csv(skorea_provinces_population, encoding='utf-8')
skorea_provinces_df
```

	name	population
0	서울특별시	9765623
1	부산광역시	3441453
2	대구광역시	2461769
3	인천광역시	2954642
4	광주광역시	1459336
5	대전광역시	1489936
6	울산광역시	1155623
7	세종특별자치시	314126
8	경기도	13077153
9	강원도	1543052
10	충청북도	1599252
11	충청남도	2126282
12	전라북도	1836832
13	전라남도	1882970

```
m = folium.Map(
    location=[36.320329, 127.861451],
    zoom_start=7
)

folium.GeoJson(
    skorea_provinces_geo,
    name='skorea-provinces'
).add_to(m)

m
```



```
m = folium.Map(
    location=[36.320329, 127.861451],
    zoom_start=7
)

folium.Choropleth(
  geo_data=skorea_provinces_geo,
  data=skorea_provinces_df,
  columns=['name', 'population'],
  key_on='feature.properties.name',
  fill_color='OrRd',
  fill_opacity=0.6,
  line_opacity=0.4,
  legend_name='Population'
).add_to(m)

m
```

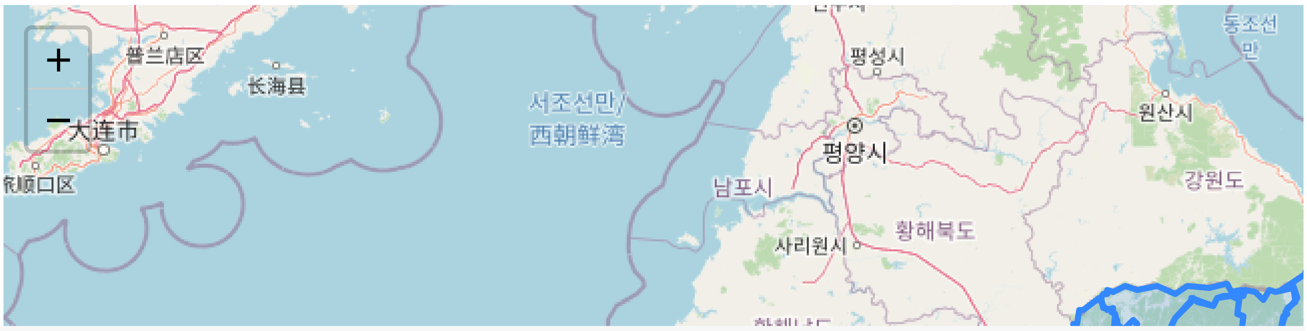


- South Korea github: https://github.com/southkorea
- 국가통계포털: http://kosis.kr
- 행정구역 지도: https://raw.githubusercontent.com/suanlab/dataset/master/skorea-municipalities-2018-geo.json
- 행정구역 인구수: https://raw.githubusercontent.com/suanlab/dataset/master/skorea_municipalities_population.csv

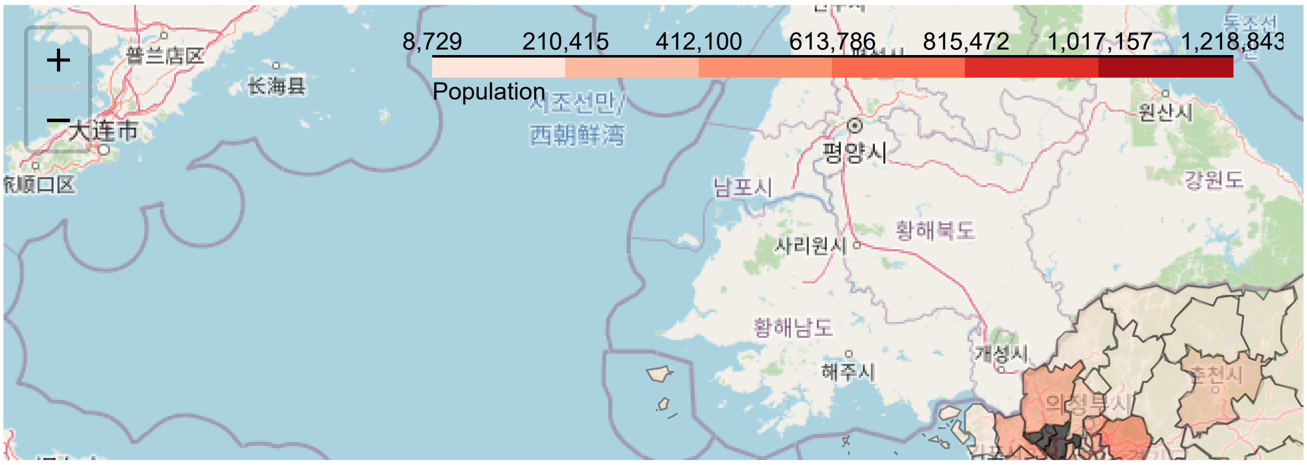
```
url = 'https://raw.githubusercontent.com/suanlab/dataset/master'
skorea_municipalities_geo = f'{url}/skorea-municipalities-2018-geo.json'
skorea_municipalities_population = f'{url}/skorea_municipalities_population.csv'
skorea_municipalities_df = pd.read_csv(skorea_municipalities_population, encoding='utf-8')
skorea_municipalities_df
```

	name	population
0	종로구	157967
1	중구	129797
2	용산구	226938
3	성동구	306796
4	광진구	362304
...	...	...

```
m = folium.Map(
    location=[36.320329, 127.861451],
    zoom_start=7
)

folium.GeoJson(
    skorea_municipalities_geo,
    name='skorea-municipalities'
).add_to(m)

m
```



```
m = folium.Map(
    location=[36.320329, 127.861451],
    zoom_start=7
)

folium.Choropleth(
  geo_data=skorea_municipalities_geo,
  data=skorea_municipalities_df,
  columns=['name', 'population'],
  key_on='feature.properties.name',
  fill_color='Reds',
  fill_opacity=0.6,
  line_opacity=0.4,
  legend_name='Population'
).add_to(m)

m
```



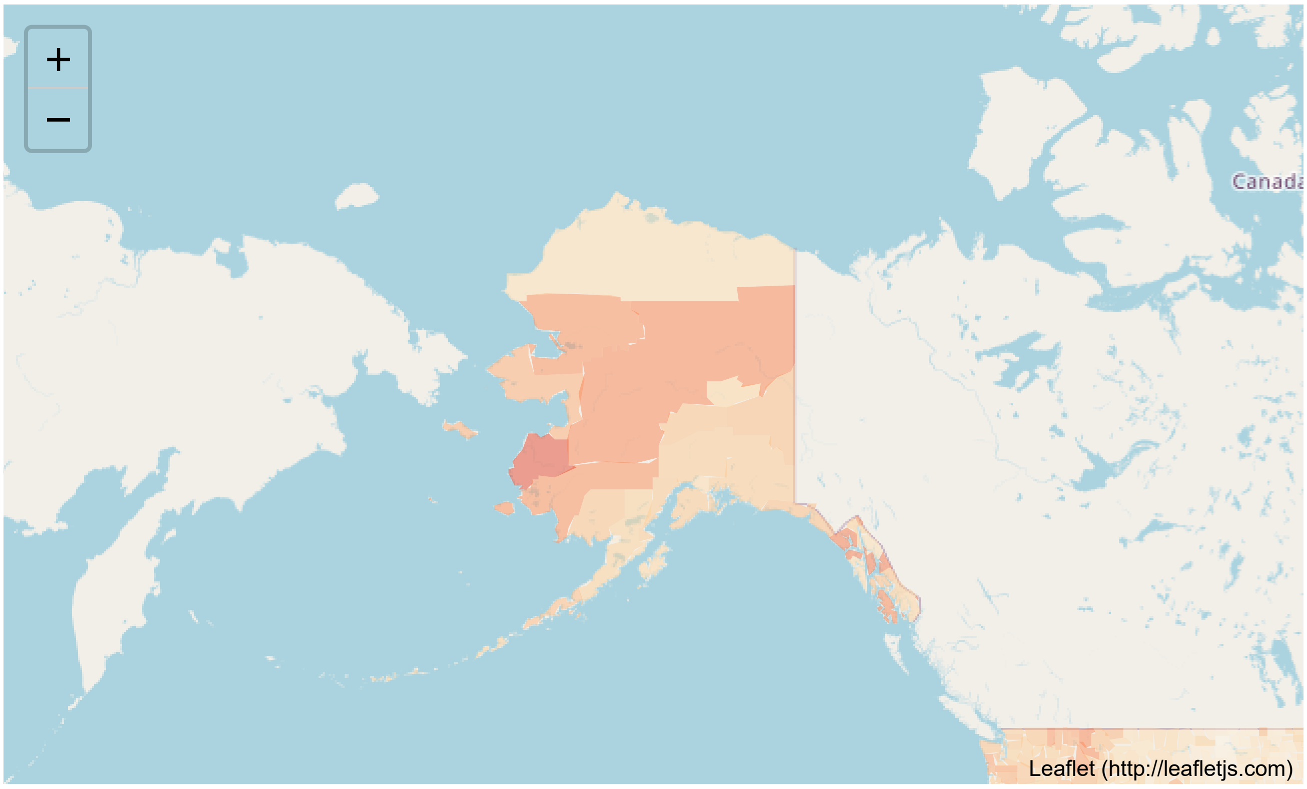
- https://raw.githubusercontent.com/python-visualization/folium/master/examples/data
  - us_county_data.csv
  - us_counties_20m_topo.json

```
url = 'https://raw.githubusercontent.com/python-visualization/folium/master/examples/data'
county_data = f'{url}/us_county_data.csv'
county_geo = f'{url}/us_counties_20m_topo.json'

df = pd.read_csv(county_data, na_values=[' '])
df
```

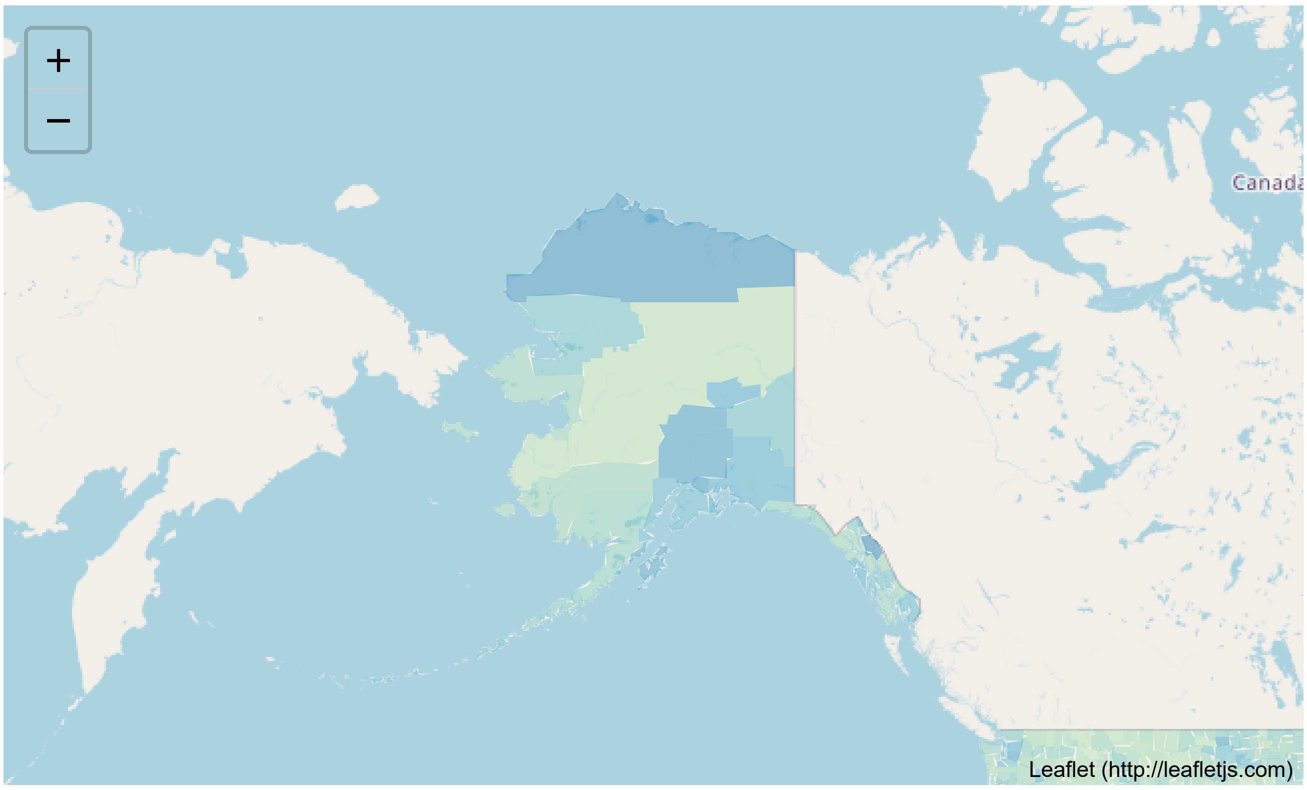
	FIPS_Code	State	Area_name	Civilian_labor_force_2011	Employed_2011	Unemplo

```
colorscale = branca.colormap.linear.PuBuGn_09.scale(0, 50e3)
employed_series = df.set_index('FIPS_Code')['Employed_2011']

def style_function(feature):
  employed = employed_series.get(int(feature['id'][-5:]), None)
  return {
      'fillOpacity': 0.5,
      'weight': 0,
      'fillColor': '#black' if employed is None else colorscale(employed)
  }

m = folium.Map(
    location=[48, -102],
    zoom_start=3
)

folium.TopoJson(
    json.loads(requests.get(county_geo).text),
    'objects.us_counties_20m',
    style_function=style_function
).add_to(m)

m
```

```
colorscale = branca.colormap.linear.OrRd_09.scale(0, 30)
unemployed_series = df.set_index('FIPS_Code')['Unemployment_rate_2011']

def style_function(feature):
  unemployed = unemployed_series.get(int(feature['id'][-5:]), None)
  return {
      'fillOpacity': 0.5,
      'weight': 0,
      'fillColor': '#black' if unemployed is None else colorscale(unemployed)
  }

m = folium.Map(
    location=[48, -102],
    zoom_start=3
)

folium.TopoJson(
    json.loads(requests.get(county_geo).text),
    'objects.us_counties_20m',
    style_function=style_function
).add_to(m)

m
```



```
colorscale = branca.colormap.linear.GnBu_09.scale(0, 100000)
income_series = df.set_index('FIPS_Code')['Median_Household_Income_2011'].dropna()

def style_function(feature):
  income = income_series.get(int(feature['id'][-5:]), None)
  return {
      'fillOpacity': 0.5,
      'weight': 0,
      'fillColor': '#black' if income is None else colorscale(income)
  }

m = folium.Map(
    location=[48, -102],
    zoom_start=3
)

folium.TopoJson(
    json.loads(requests.get(county_geo).text),
    'objects.us_counties_20m',
    style_function=style_function
).add_to(m)

m
```



대화형(Interactive)

- OpenStreetMap
- Stamen Terrain
- Stamen Toner
- Stamen Watercolor
- CartoDB positron
- CartoDB dark_matter

```
from ipywidgets import interact

tiles = [name.strip() for name in """
  OpenStreetMap
  Stamen Terrain
  Stamen Toner
  Stamen Watercolor
  CartoDB positron
  CartoDB dark_matter""".strip().split('₩n')]

@interact(latitude=(-90., 90.), longitude=(-180., 180.), tiles=tiles, zoom=(1, 18))
```

```
@interact(latitude=(-90., 90.), longitude=(-180., 180.), tiles=tiles, zoom=(1, 18))
def create_map(latitude=36, longitude=127, tiles="Stamen Toner", zoom=7):
  return folium.Map(location=(latitude, longitude), tiles=tiles, zoom_start=zoom)
```

latitude 36.00

longitude 127.00

tiles Stamen Toner

zoom 7

참고문헌

- https://python-visualization.github.io/folium/