빅데이터 실습

11주차 3차시

Folium을 활용한 지도 시각화 [2]

Folium 활용한 지도 시각화

학습개요

1/ 서울시 유동 인구 조사 지점 지도 시각화
(feat. 좌표계 변환)

2/ GeoJson으로 경계 표현하기

3/ Choropleth로 데이터와 GeoJson 결합하기

01

서울시 유동 인구 조사 지점 지도 시각화 (feat. 좌표계 변환)

① 서울시 유동 인구 조사 지점 지도 시각화 (feat. 좌표계 변환)

서울시 유동 인구 조사 지점 지도 시각화

서울시 유동 인구 조사 지점 지도 시각화

◎ 서울시 유동 인구 데이터 MarkerCluster로 추가하기

✔ 데이터 읽기(data/서울시유동인구/4_유동인구_조사지점정보_2015.xlsx)

- data/서울시유동인구/4_유동인구_조사지점정보_2015.xlsx 활용
- 조사구분이 '본조사'인 데이터만 대상으로 함

In [38]:
```
import pandas as pd
```

In [*]:
```
# 데이터 읽기
측정위치 = pd.read_excel('data/서울시유동인구/4_유동인구_조사지점정보_2015.xlsx',
                         skiprows = [0,1,3])
측정위치.head()
```

In [ ]:

In [ ]:

서울시 유동 인구 조사 지점 지도 시각화

◎ 서울시 유동 인구 데이터 MarkerCluster로 추가하기

✔ 본조사 데이터만 선택

구/4_유동인구_조사지점정보_2015.xlsx 활용
'인 데이터만 대상으로 함

```
In [38]: import pandas as pd

In [39]: # 데이터 읽기
         측정위치 = pd.read_excel('data/서울시유동인구/4_유동인구_조사지점정보_2015.xlsx',
                                  skiprows = [0,1,3])
         측정위치.head()

In [41]: # 본조사 데이터만 선택
         측정위치측정위치[측정위치.조사구분 == '본조사']
```

Out[41]:

	조사지점코드	조사지점명	구코드	동코드	주번지	부번지	도로명	보도너비	차선수	버스차로유무	...	지구중심상세내용	도심부도심지역명	용도구분	거주유형구분	입지유형명	X좌표	Y좌표
0	01-003	신흥모피명품전문크리	11010.0	1101055.0	127	11	NaN	3.0	8.0	유	...	NaN	NaN	2종주거	NaN	NaN	196423.97707	455511.52968

서울시 유동 인구 조사 지점 지도 시각화

◎ 서울시 유동 인구 데이터 MarkerCluster로 추가하기

✔ 필요한 컬럼만 선택

동인구/4_유동인구_조사지점정보_2015.xlsx 활용
조사'인 데이터만 대상으로 함

In [38]:
```
import pandas as pd
```

In [39]:
```
# 데이터 읽기
측정위치 = pd.read_excel('data/서울시유동인구/4_유동인구_조사지점정보_2015.xlsx',
                         skiprows = [0,1,3])
측정위치.head()
```

...

In [42]:
```
# 본조사 데이터만 선택
측정위치 = 측정위치[측정위치.조사구분 == '본조사']
```

In [44]:
```
# 필요한 컬럼 선택
측정위치 = 측정위치[['조사지점명','X좌표','Y좌표']]
```

In [ ]:

In [ ]:

In [ ]:

서울시 유동 인구 조사 지점 지도 시각화

◎ 서울시 유동 인구 데이터 MarkerCluster로 추가하기

✔ 필요한 컬럼만 선택

동인구/4_유동인구_조사지점정보_2015.xlsx 활용
조사'인 데이터만 대상으로 함

In [38]:
```
import pandas as pd
```

In [39]:
```
# 데이터 읽기
측정위치 = pd.read_excel('data/서울시유동인구/4_유동인구_조사지점정보_2015.xlsx',
                         skiprows = [0,1,3])
측정위치.head()
```

...

In [42]:
```
# 본조사 데이터만 선택
측정위치 = 측정위치[측정위치.조사구분 == '본조사']
```

In [44]:
```
# 필요한 컬럼 선택
측정위치 = 측정위치[['조사지점명','X좌표','Y좌표']]
```

In [45]:
```
len(측정위치)
```

Out[45]: 1227

In [ ]:

In [ ]:

서울시 유동 인구 조사 지점 지도 시각화

◎ 서울시 유동 인구 데이터 MarkerCluster로 추가하기

✔ 필요한 컬럼만 선택

동인구/4_유동인구_조사지점정보_2015.xlsx 활용
조사'인 데이터만 대상으로 함

```
In [38]: import pandas as pd

In [39]: # 데이터 읽기
         측정위치 = pd.read_excel('data/서울시유동인구/4_유동인구_조사지점정보_2015.xlsx',
                                skiprows = [0,1,3])
         측정위치.head()

         ...

In [42]: # 본조사 데이터만 선택
         측정위치 = 측정위치[측정위치.조사구분 == '본조사']

In [44]: # 필요한 컬럼 선택
         측정위치 = 측정위치[['조사지점명','X좌표','Y좌표']]

In [45]: # 맨 앞에 있는 100개만 선택
         측정위치[]

Out[45]: 1227
```

시간이 오래 걸릴 수 있기 때문에 맨 앞에 있는 100개만 Marker로 추가합니다.

서울시 유동 인구 조사 지점 지도 시각화

◎ 서울시 유동 인구 데이터 MarkerCluster로 추가하기

✓ 좌표계 변환

of the control (one of the map corners), can be 'topleft', 'topright',
default: 'topright'
– If true the control will be collapsed into an icon and expanded on
e) – If true the control assigns zIndexes in increasing order to all of its
served when switching them on/off.

- ****kwargs** – Additional (possibly inherited) options. See https://leafletjs.com/reference-1.6.0.html#control-layers

render (***kwargs*)

Renders the HTML representation of the element.

reset ()

class folium.map.**Marker** (*location=None, popup=None, tooltip=None, icon=None, draggable=False, **kwargs*)

Bases: **branca.element.MacroElement**

Create a simple stock Leaflet marker on the map, with optional popup text or Vincent visualization.

Parameters:
- **location** (*tuple or list*) – Latitude and Longitude of Marker (Northing, Easting)
- **popup** (*string or folium.Popup, default None*) – Label for the Marker; either an escaped HTML string to initialize folium.Popup or a folium.Popup instance.
- **tooltip** (*str or folium.Tooltip, default None*) – Display a text when hovering over the object.
- **icon** (*Icon plugin*) – the Icon plugin to use to render the marker.
- **draggable** (*bool, default False*) – Set to True to be able to drag the marker around the map.

Returns:

Return type: Marker names and HTML in obj.template_vars

Examples

```
>>> Marker(location=[45.5, -122.3], popup='Portland, OR')
>>> Marker(location=[45.5, -122.3], popup=Popup('Portland, OR'))
# If the popup label has characters that need to be escaped in HTML
>>> Marker(location=[45.5, -122.3],
...        popup=Popup('Mom & Pop Arrow Shop >>', parse_html=True))
```

render ()

Renders the HTML representation of the element.

Marker에는 CRS(좌표계) 지정 인자가 없기 때문에 직접 변환해 주어야 합니다.

Create a Popup instance that can be linked to a Layer.

서울시 유동 인구 조사 지점 지도 시각화

◎ 서울시 유동 인구 데이터 MarkerCluster로 추가하기

✔ 좌표계 변환

```
In [ ]: # 조사 데이터만 선택
        측정위치 = 측정위치[측정위치.조사구분 == '본조사']

In [44]: # 필요한 컬럼 선택
         측정위치 = 측정위치[['조사지점명','X좌표','Y좌표']]

In [46]: # 맨 앞에 있는 100개만 선택
         측정위치 = 측정위치[:100]

In [47]: 측정위치

In [ ]: # 좌표계 변환
        from pyproj import Proj, t

In [ ]:
In [ ]:
In [ ]:
In [ ]:
In [ ]:
In [ ]:
```

pyproj 좌표계를 변환해 주는 라이브러리

서울시 유동 인구 조사 지점 지도 시각화

◎ 서울시 유동 인구 데이터 MarkerCluster로 추가하기

✔ 좌표계 변환

```
Collecting branca>=0.3.0
  Downloading branca-0.4.2-py3-none-any.whl (24 kB)
Requirement already satisfied: jinja2>=2.9 in c:\programdata\anaconda3\lib\site-packages (from folium) (2.11.2)
Requirement already satisfied: numpy in c:\programdata\anaconda3\lib\site-packages (from folium) (1.19.2)
Requirement already satisfied: requests in c:\programdata\anaconda3\lib\site-packages (from folium) (2.24.0)
Requirement already satisfied: MarkupSafe>=0.23 in c:\programdata\anaconda3\lib\site-packages (from jinja2>=2.9->folium
) (1.1.1)
Requirement already satisfied: certifi>=2017.4.17 in c:\programdata\anaconda3\lib\site-packages (from requests->folium)
(2020.6.20)
Requirement already satisfied: urllib3!=1.25.0,!=1.25.1,<1.26,>=1.21.1 in c:\programdata\anaconda3\lib\site-packages (f
rom requests->folium) (1.25.11)
Requirement already satisfied: chardet<4,>=3.0.2 in c:\programdata\anaconda3\lib\site-packages (from requests->folium)
(3.0.4)
Requirement already satisfied: idna<3,>=2.5 in c:\programdata\anaconda3\lib\site-packages (from requests->folium) (2.10
)
Installing collected packages: branca, folium
Successfully installed branca-0.4.2 folium-0.12.1

(base) C:\Windows\system32>pip install pyproj
Collecting pyproj
  Downloading pyproj-3.0.0.post1-cp38-cp38-win_amd64.whl (14.4 MB)
     |████████████████████████████████| 14.4 MB 6.8 MB/s
Requirement already satisfied: certifi in c:\programdata\anaconda3\lib\site-packages (from pyproj) (2020.6.20)
Installing collected packages: pyproj
Successfully installed pyproj-3.0.0.post1

(base) C:\Windows\system32>
```

In [44]:
```
# 필요
측정위
```

In [46]:
```
# 맨 윗
측정위
```

In [47]:
```
측정위
```

In [48]:
```
# 좌표계 변환
from pyproj import Proj, transform
```

```
---------------------------------------------------------------------------
ModuleNotFoundError                       Traceback (most recent call last)
<ipython-input-48-0e2db2347226> in <module>
      1 # 좌표계 변환
----> 2 from pyproj import Proj, transform

ModuleNotFoundError: No module named 'pyproj'
```

Anaconda 기본 설치에 포함되어 있지 않기 때문에 pip install pyproj를 실행하여 설치합니다.

서울시 유동 인구 조사 지점 지도 시각화

◎ 서울시 유동 인구 데이터 MarkerCluster로 추가하기

✔ 좌표계 변환

```
조사 데이터만 선택
치 = 측정위치[측정위치.조사구분 == '본조사']
```

In [44]:
```
# 필요한 컬럼 선택
측정위치 = 측정위치[['조사지점명','X좌표','Y좌표']]
```

In [46]:
```
# 맨 앞에 있는 100개만 선택
측정위치 = 측정위치[:100]
```

In [47]:
```
측정위치
```

convert( ) (x, y) 좌표값 받아 위경도 정보로 변환

In [49]:
```
# 좌표계 변환
from pyproj import Proj, transform
```

In [ ]:
```
def convert(x, y):
    inProj = Proj(init = 'epsg:5181')
    outProj = Proj(init = 'epsg:4326')
    x2, y2 = tranform(inProj, outProj, x, y)
```

transform( ) pyproj에서 제공하는 좌표 정보를 변환하는 함수

In [ ]:

In [ ]:

서울시 유동 인구 조사 지점 지도 시각화

◎ 서울시 유동 인구 데이터 MarkerCluster로 추가하기

✔ 측정 위치에 위경도 컬럼 추가

```
7707  455511.52968
0243  455621.38262
```

In [49]:
```
# 좌표계 변환
from pyproj import Proj, transform
```

In [53]:
```
def convert(x, y):
    inProj = Proj(init = 'epsg:5181')
    outProj = Proj(init = 'epsg:4326')
    x2, y2 = transform(inProj, outProj, x, y)
    return [y2, x2]
```

In [54]:
```
convert(196423.97707, 455511.52968)
```

...

In [55]:
```
측정위치.apply(lambda 지점: convert(지점.X좌표, 지점.Y좌표), axis = 1)
```

Out[55]:

	조사지점명	X좌표	Y좌표
0	신흥모피명품전문크리닝.	196423.97707	455511.52968
1	GS25	196315.80243	455621.38262
2	세검정정류장	196357.17125	455680.82580
3	안성타워內 굿모닝파워공인중개사.	197904.19277	456718.34996

서울시 유동 인구 조사 지점 지도 시각화

◎ 서울시 유동 인구 데이터 MarkerCluster로 추가하기

✔ MarkerCluster에 Marker 추가

Run Code

19277	456718.34996	[37.6100475136769, 126.97626371692706]
44943	456405.89296	[37.60722749795384, 126.9587814478111]

In [59]:
```
map3 = folium.Map(location = [37.566535, 126.9779691999996], zoom_start = 10,
                  zoom_control = False, control_scale = True,
                  tiles = 'https://tiles.stadiamaps.com/tiles/outdoors/{z}/{x}/{y}{r}.png',
                  attr = '&copy; <a href="https://stadiamaps.com/">Stadia Maps</a>, &copy; <a href="https:
map3
```

In [ ]:
```
marker_cluster = MarkerCluster().add_to(map3)
for val in 측정위치
```

In [ ]:

In [ ]:

In [ ]:

4. GeoJson을 활용한 다각형 표현
   - 행정구역과 같은 경계선이나 경로를 표현하는데 효율적

서울시 유동 인구 조사 지점 지도 시각화

◎ 서울시 유동 인구 데이터 MarkerCluster로 추가하기

✔ MarkerCluster에 Marker 추가

19277	456718.34996	[37.6100475136769, 126.97626371692706]
44943	456405.89296	[37.60722749795384, 126.9587814478111]

```
In [59]: map3 = folium.Map(location = [37.566535, 126.9779691999996], zoom_start = 10,
                           zoom_control = False, control_scale
                           tiles = 'https://tiles.stadiamaps.com/t                    y}{r}.png',
                           attr = '&copy: <a href="https://stadian                    a>, &copy: <a href="https:
         map3
```

...

```
In [60]: type(측정위치)
Out[60]: pandas.core.frame.DataFrame
```

iterrows( )

행(row) 단위로 반복

인덱스의 번호와 행(row) 리턴

```
In [ ]: marker_cluster = MarkerCluster().add_to(map3)
        for val in 측정위치.iterrows():
```

```
In [ ]:
In [ ]:
In [ ]:
```

서울시 유동 인구 조사 지점 지도 시각화

◎ 서울시 유동 인구 데이터 MarkerCluster로 추가하기

✔ MarkerCluster에 Marker 추가

Run Code

19277 456718.34996 [37.6100475136769, 126.97626371692706]

44943 456405.89296 [37.60722749795384, 126.9587814478111]

In [59]:
```
map3 = folium.Map(location = [37.566535, 126.9779691999996], zoom_start = 10,
                  zoom_control = False, control_scale = True,
                  tiles = 'https://tiles.stadiamaps.com/tiles/outdoors/{z}/{x}/{y}{r}.png',
                  attr = '&copy; <a href="https://stadiamaps.com/">Stadia Maps</a>, &copy; <a href="https:
map3
```

...

In [60]:
```
type(측정위치)
```

Out[60]: pandas.core.frame.DataFrame

In [61]:
```
marker_cluster = MarkerCluster().add_to(map3)
for idx, val in 측정위치.iterrows():
    folium.Marker(location = val.위경도, popup = folium.Popup(val.조사지점명,).add_to(marker_cluster)
```

0 조사지점명 신흥모피명품전문크리닝.

X좌표 196424

Y좌표 455512

객체는 항상 대문자로 시작합니다.

X좌표 196316

Y좌표 455621

서울시 유동 인구 조사 지점 지도 시각화

◎ 서울시 유동 인구 데이터 MarkerCluster로 추가하기

✔ MarkerCluster에 Marker 추가

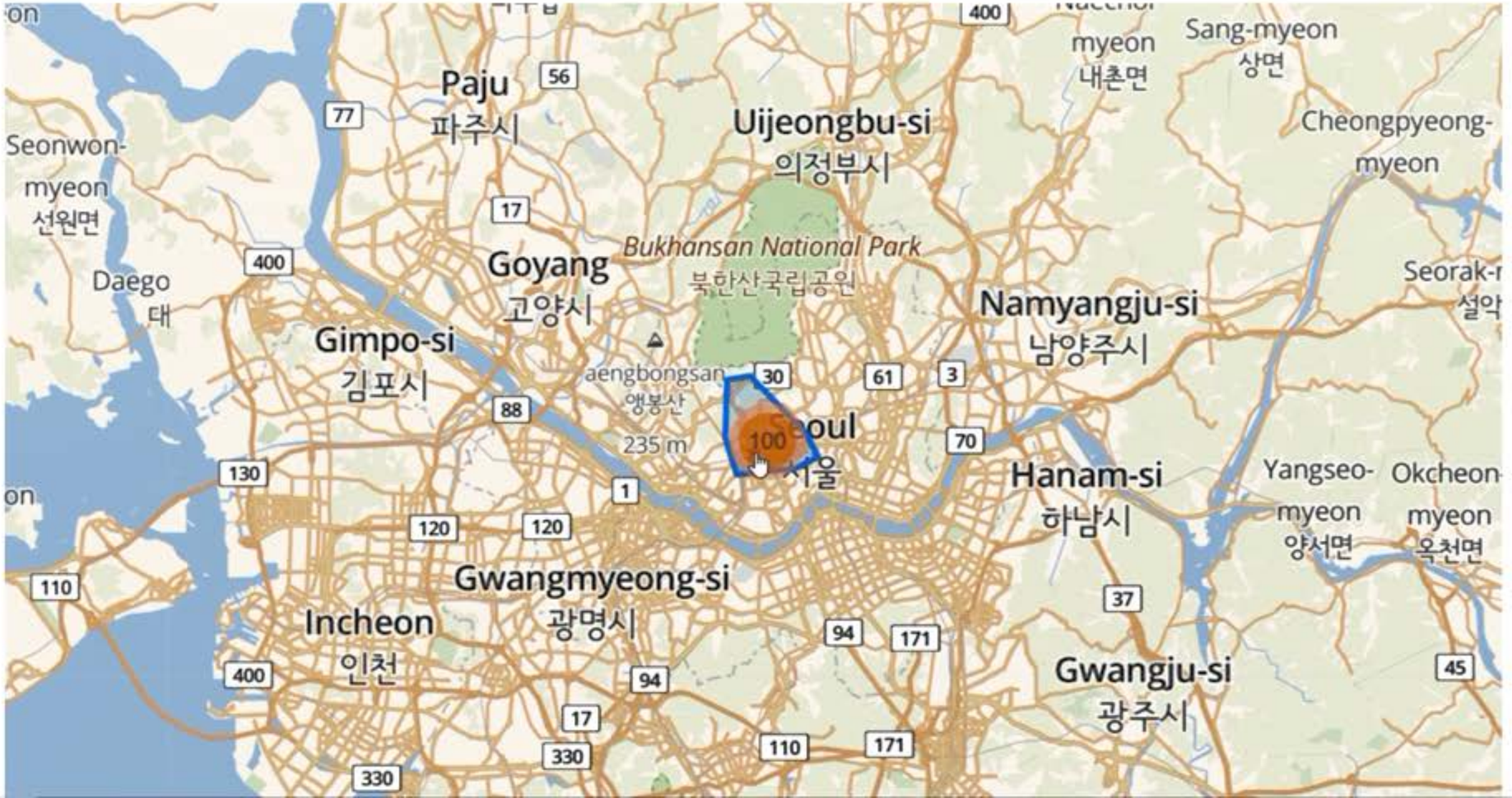
```
                                                    19277  456718.34996  [37.6100475136769, 126.97626371692706]
                                                    44943  456405.89296  [37.60722749795384, 126.9587814478111]

In [59]: map3 = folium.Map(location = [37.566535, 126.9779691999996], zoom_start = 10,
                           zoom_control = False, control_scale = True,
                         tiles = 'https://tiles.stadiamaps.com/tiles/outdoors/{z}/{x}/{y}{r}.png',
                         attr = '&copy; <a href="https://stadiamaps.com/">Stadia Maps</a>, &copy; <a href="https:
         map3

In [60]: type(측정위치)

Out[60]: pandas.core.frame.DataFrame

In [62]: marker_cluster = MarkerCluster().add_to(map3)
         for idx, val in 측정위치.iterrows():
             folium.Marker(location = val.위경도, popup = folium.Popup(val.조사지점명, max_width = 300)
                           .add_to(marker_cluster)

           File "<ipython-input-62-5845bb097217>", line 4
             .add_to(marker_cluster)

           ^

         SyntaxError: unexpected EOF while parsing
```

서울시 유동 인구 조사 지점 지도 시각화

◎ 서울시 유동 인구 데이터 MarkerCluster로 추가하기

✔ MarkerCluster에 Marker 추가

Out[64]:



서울시 유동 인구 조사 지점 지도 시각화

◎ 서울시 유동 인구 데이터 MarkerCluster로 추가하기

Warning

```
    x2, y2 = transform(inProj, outProj, x, y)
    return [y2, x2]

In [54]: convert(196423.97707, 455511.52968)

...

In [57]: 측정위치['위경도'] = 측정위치.apply(lambda 지점: convert(지점.X좌표, 지점.Y좌표), axis = 1)
```

```
er-changes-in-proj-6
  projstring = _prepare_from_string(" ".join((projstring, projkwargs)))
<ipython-input-53-42793f5a76b6>:4: DeprecationWarning: This function is deprecated. See: https://pyproj
4.github.io/pyproj/stable/gotchas.html#upgrading-to-pyproj-2-from-pyproj-1
  x2, y2 = transform(inProj, outProj, x, y)
C:\ProgramData\Anaconda3\lib\site-packages\pyproj\crs\crs.py:53: FutureWarning: '+init=<authority>:<cod
e>' syntax is deprecated. '<authority>:<code>' is the preferred initialization method. When making the
change, be mindful of axis order changes: https://pyproj4.github.io/pyproj/stable/gotchas.html#axis-ord
er-changes-in-proj-6
  return _prepare_from_string(" ".join(pjargs))
C:\ProgramData\Anaconda3\lib\site-packages\pyproj\crs\crs.py:294: FutureWarning: '+init=<authority>:<co
de>' syntax is deprecated. '<authority>:<code>' is the preferred initialization method. When making the
change, be mindful of axis order changes: https://pyproj4.github.io/pyproj/stable/gotchas.html#axis-ord
er-changes-in-proj-6
  projstring = _prepare_from_string(" ".join((projstring, projkwargs)))
```

proj와 transform 함수를 사용하는 방식은 더 이상 지원하지 않으니
새로운 구문으로 작성하라는 의미입니다.

서울시 유동 인구 조사 지점 지도 시각화

◎ 서울시 유동 인구 데이터 MarkerCluster로 추가하기

Warning

```
                zoom_control = False, control_scale = True,
            tiles = 'https://tiles.stadiamaps.com/tiles/outdoors/{z}/{x}/{y}{r}.png',
            attr = '&copy; <a href="https://stadiamaps.com/">Stadia Maps</a>, &copy; <a href="https:
map3
```

...

In [60]:
```
type(측정위치)
```

Out[60]:
```
pandas.core.frame.DataFrame
```

In [63]:
```
marker_cluster = MarkerCluster().add_to(map3)
for idx, val in 측정위치.iterrows():
    folium.Marker(location = val.위경도, popup = folium.Popup(val.조사지점명, max_width = 300))₩
                .add_to(marker_cluster)
```

In [64]:

Transformer 객체 사용

..

In [ ]:
```
from pyproj import Transformer
transfor
```

In [ ]:

In [ ]:

서울시 유동 인구 조사 지점 지도 시각화

◎ 서울시 유동 인구 데이터 MarkerCluster로 추가하기

Warning

```
                zoom_control = False, control_scale = True,
            tiles = 'https://tiles.stadiamaps.com/tiles/outdoors/{z}/{x}/{y}{r}.png',
            attr = '&copy; <a href="https://stadiamaps.com/">Stadia Maps</a>, &copy; <a href="https:
map3
```

...

In [60]:
```
type(측정위치)
```

Out[60]:
```
pandas.core.frame.DataFrame
```

In [63]:
```
marker_cluster = MarkerCluster().add_to(map3)
for idx, val in 측정위치.iterrows():
    folium.Marker(location = val.위경도, popup = folium.Popup(val.조사지점명, max_width = 300))₩
                  .add_to(marker_cluster)
```

In [64]:
```
map3
```

...

In [ ]:
```
from pyproj import Transformer
transformer = Transformer.from_crs()
```

from crs( ) 함수 사용

In [ ]:

In [ ]:

서울시 유동 인구 조사 지점 지도 시각화

◎ 서울시 유동 인구 데이터 MarkerCluster로 추가하기

Warning

```
                zoom_control = False, control_scale = True,
               tiles = 'https://tiles.stadiamaps.com/tiles/outdoors/{z}/{x}/{y}{r}.png',
               attr = '&copy; <a href="https://stadiamaps.com/">Stadia Maps</a>, &copy; <a href="https:
map3
```

...

In [60]:
```
type(측정위치)
```

Out[60]:
```
pandas.core.frame.DataFrame
```

In [63]:
```
marker_cluster = MarkerCluster().add_to(map3)
for idx, val in 측정위치.iterrows():
    folium.Marker(location = val.위경도, popup = folium.Popup(val.조사지점명, max_width = 300))₩
                  .add_to(marker_cluster)
```

In [64]:
```
map3
```

...

In [ ]:
```
from pyproj import Transformer
transformer = Transformer.from_crs("epsg:5181", "epsg:4326")
transformer.transform()
```

In [ ]:

In [ ]:

서울시 유동 인구 조사 지점 지도 시각화

◎ 서울시 유동 인구 데이터 MarkerCluster로 추가하기

Warning

```
In [60]: type(측정위치)
Out[60]: pandas.core.frame.DataFrame

In [63]: marker_cluster = MarkerCluster().add_to(map3)
         for idx, val in 측정위치.iterrows():
             folium.Marker(location = val.위경도, popup = folium.Popup(val.조사지점명, max_width = 300))
                          .add_to(marker_cluster)

In [64]: map3

In [65]: from pyproj import Transformer
         transformer = Transformer.from_crs("epsg:5181", "epsg:4326")
         transformer.transform(196423.97707, 455511.52968)
Out[65]: (35.231807288603704, 129.80651596957898)

In [ ]:

In [ ]:
```

02

GeoJson으로 경계 표현하기

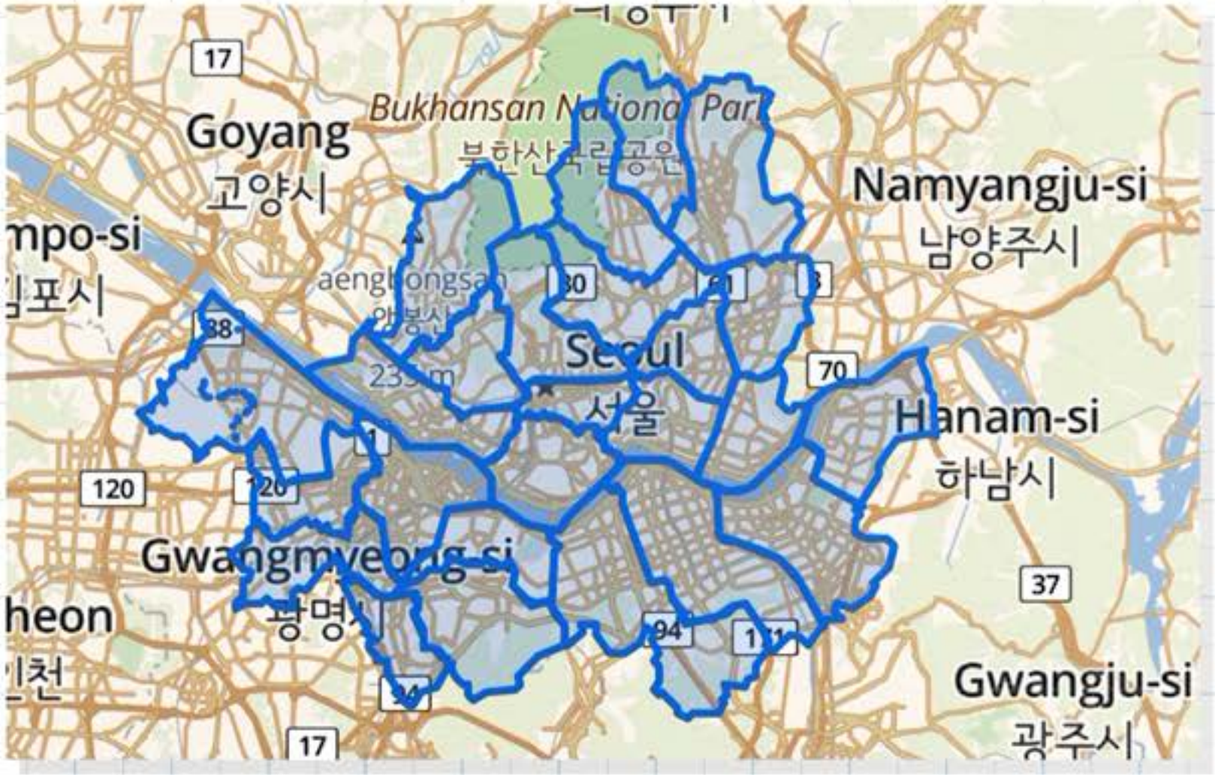
② GeoJson으로 경계 표현하기

GeoJson과 Choropleth을 활용한 지도 시각화

② GeoJson으로 경계 표현하기

ㄱ GeoJson을 활용한 다각형 표현

✓ 행정구역과 같은 경계선이나 경로를 표현하는데 효율적

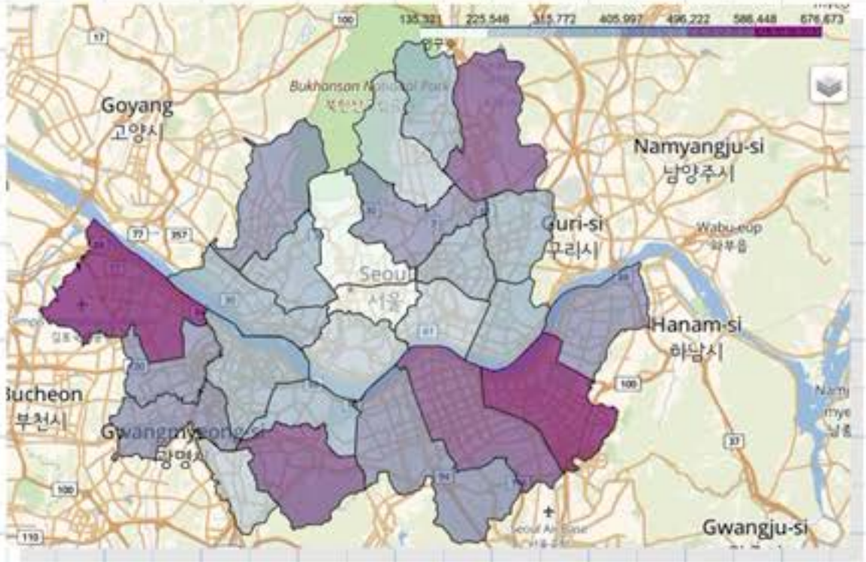


② GeoJson으로 경계 표현하기

ㄱ Choropleth을 활용하여 GeoJson과 데이터 결합

GeoJson 행정구역 경계 지정하고 표시

\+ **Choropleth** 각각의 행정구역 별로 다른 색상으로 표현

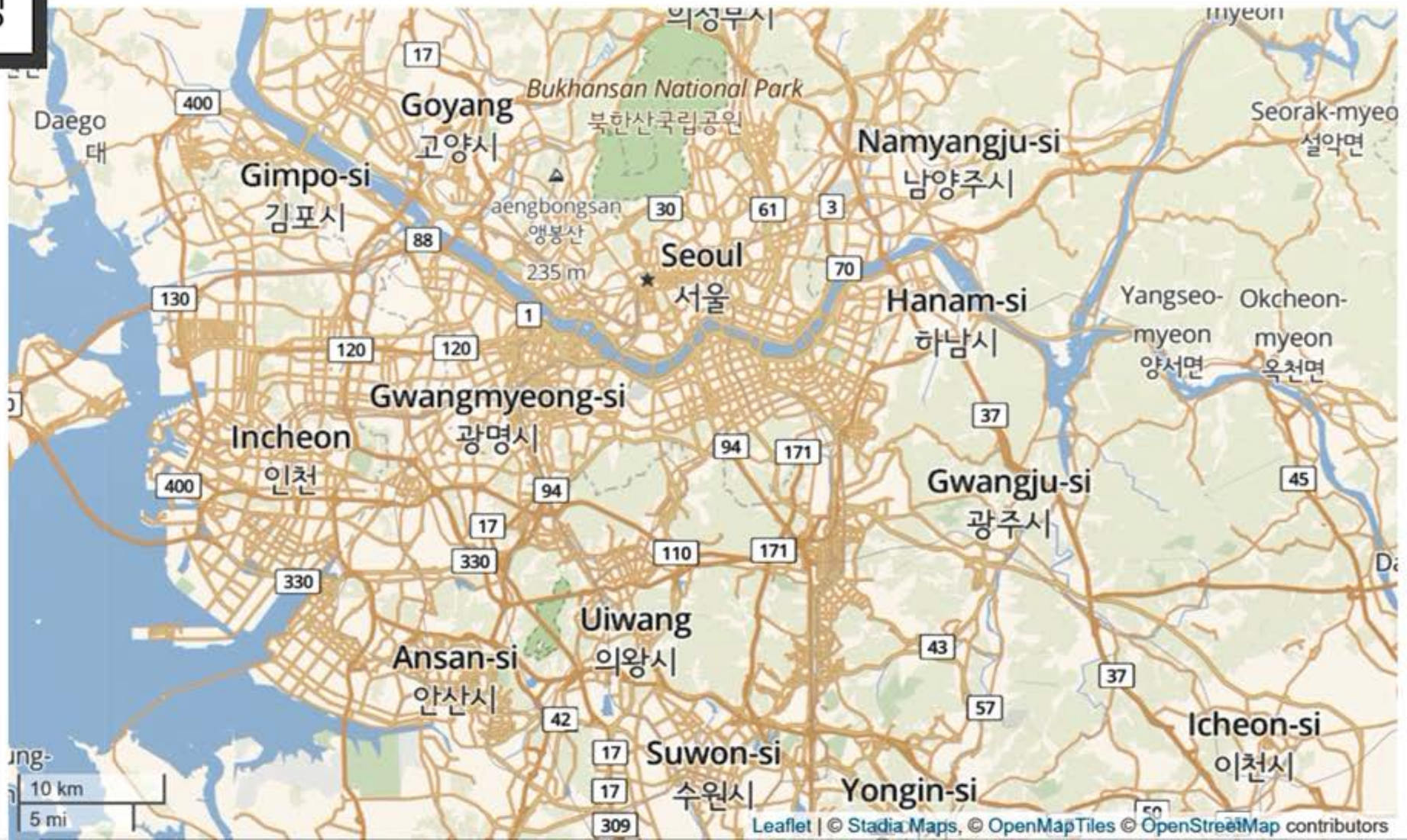


↳ 데이터 값이 높을수록 색이 짙어짐

GeoJson을 활용한 다각형 표현 실습

◎ 행정구역과 같은 경계선이나 경로 표현

✔ Map 생성



GeoJson을 활용한 다각형 표현 실습

◎ 행정구역과 같은 경계선이나 경로 표현

✔ 데이터 읽어오기

```
um.Map(location = [37.566535, 126.9779691999996], zoom_start = 10,
                    zoom_control = False, control_scale = True,
                   tiles = 'https://tiles.stadiamaps.com/tiles/outdoors/{z}/{x}/{y}{r}.png',
                   attr = '&copy; <a href="https://stadiamaps.com/">Stadia Maps</a>, &copy; <a href="https:
map4
```

...

In [68]:
```
import json
with open('data/seoul_muncipalities_geo.json', encoding = 'utf-8') as file:
    geo = json.loads(file.read())
    file.close()
```

In [ ]:

In [ ]:

In [ ]:

In [ ]:

In [ ]:

In [ ]:

01. import json 실행

02. seoul_municipalities_geo.json 파일 열기 (utf-8 인코딩)

03. 파일 내용을 읽어서 geo 변수에 저장

GeoJson을 활용한 다각형 표현 실습

◎ 행정구역과 같은 경계선이나 경로 표현

✓ 데이터 읽어오기

```
tiles = 'https://tiles.stadiamaps.com/tiles/outdoors/{z}/{x}/{y}{r}.png',
attr = '&copy; <a href="https://stadiamaps.com/">Stadia Maps</a>, &copy; <a href="https:
```

```
...
```

In [68]:
```
import json
with open('data/seoul_muncipalities_geo.json', encoding = 'utf-8') as file:
    geo = json.loads(file.read())
    file.close()
```

In [69]:
```
geo
```

Out[69]:
```
{'type': 'FeatureCollection',
 'features': [{'type': 'Feature',
   'properties': {'SIG_CD': '11320',
    'SIG_KOR_NM': '도봉구',
    'SIG_ENG_NM': 'Dobong-gu',
    'ESRI_PK': 0,
    'SHAPE_AREA': 0.00211,
    'SHAPE_LEN': 0.239901},
   'geometry': {'type': 'Polygon',
    'coordinates': [[[127.019851357, 37.7008849019999984],
      [127.02217147700003, 37.699607367999999],
```

JSON 형태의 데이터는 사전(Dict) 타입으로 데이터를 읽어 옵니다.

```
      [127.02704481599994, 37.7010115069999976],
```

GeoJson을 활용한 다각형 표현 실습

◎ 행정구역과 같은 경계선이나 경로 표현

✓ 데이터 읽어오기

```
tiles = 'https://tiles.stadiamaps.com/tiles/outdoors/{z}/{x}/{y}{r}.png',
attr = '&copy; <a href="https://stadiamaps.com/">Stadia Maps</a>, &copy; <a href="https:
```

...

In [68]:
```
import json
with open('data/seoul_muncipalities_geo.json', encoding = 'utf-8') as file:
    geo = json.loads(file.read())
    file.close()
```

In [69]:
```
geo
```

```
'SIG_KOR_NM': '도봉구',
'SIG_ENG_NM': 'Dobong-gu',
'ESRI_PK': 0,
'SHAPE_AREA': 0.00211,
'SHAPE_LEN': 0.239901},
'geometry': {'type': 'Polygon',
'coordinates': [[[127.019851357, 37.7008849019999984],
  [127.02217147700003, 37.69960736799999],
  [127.02341184299996, 37.69995983299998],
  [127.02533791899998, 37.69948105499998],
  [127.02692041600005, 37.70015310500003],
  [127.02704481599994, 37.701011506999976],
  [127.02771004600004, 37.700837447000026],
  [127.02823835000004, 37.700188083],
  [127.02833534499996, 37.700068851000026],
  [127.029013076999995, 37.699573188999998]
```

GeoJson을 활용한 다각형 표현 실습

◎ 행정구역과 같은 경계선이나 경로 표현

✔ 경계 정보를 Map에 추가

Run Code

```
'https://tiles.stadiamaps.com/tiles/outdoors/{z}/{x}/{y}{r}.png',
'&copy: <a href="https://stadiamaps.com/">Stadia Maps</a>, &copy: <a href="https:
```

...

In [68]:
```
import json
with open('data/seoul_muncipalities_geo.json', encoding = 'utf-8') as file:
    geo = json.loads(file.read())
    file.close()
```

In [69]:
```
geo
```

...

In [70]:
```
folium.GeoJson(geo).add_to(map4)
```

Out[70]: <folium.features.GeoJson at 0x188ad580dc0>

In [ ]:

In [ ]:

In [ ]:

In [ ]:

GeoJson을 활용한 다각형 표현 실습

◎ 행정구역과 같은 경계선이나 경로 표현

✔ 경계 정보를 Map에 추가

Run Markdown

n at 0x188ad590dc0>

In [71]: map4

Out[71]:

GeoJson을 활용한 다각형 표현 실습

◎ 행정구역과 같은 경계선이나 경로 표현

✔ 경계 정보를 Map에 추가

```
In [68]: import json
         with open('data/seoul_muncipalities_geo.json', encoding = 'utf-8') as file:
             geo = json.loads(file.read())
             file.close()

In [69]: geo

In [70]: folium.GeoJson(geo).add_to(map4)
Out[70]: <folium.features.GeoJson at 0x188ad590dc0>

In [71]: map4

In [ ]:

In [ ]:
```

click to expand output

geo는 각 구별 경계점에 대한 정보를 가지고 있습니다.

Choropleth을 활용하여 GeoJson과 데이터 결합 실습

◎ 행정구역 경계 표시 및 인구수에 따른 행정구역별 색상 차별화

✔ 준비된 데이터 읽어오기

5. Choropleth을 활용하여 GeoJson과 데이터 결합 실습

```
In [72]: population = pd.read_csv('data/자치구별인구수.txt', delimiter = '₩t', header = [0,1,2])
```

```
In [ ]:
```

```
In [ ]:
```

```
In [ ]:
```

```
In [ ]:
```

```
In [ ]:
```

header 컬럼명으로 사용할 데이터 지정

여러 줄을 컬럼명으로 사용하므로 "계층 색인"임

Choropleth을 활용하여 GeoJson과 데이터 결합 실습

◎ 행정구역 경계 표시 및 인구수에 따른 행정구역별 색상 차별화

✓ population.columns

하여 GeoJson과 데이터 결합 실습

```
In [72]: population = pd.read_csv('data/자치구별인구수.txt', delimiter = '₩t', header = [0,1,2])

In [73]: population.head()

...

In [74]: population.columns

Out[74]: MultiIndex([(      '기간',      '기간',      '기간'),
                     (    '자치구',    '자치구',    '자치구'),
                     (     '세대',     '세대',     '세대'),
                     (     '인구',     '합계',      '계'),
                     (     '인구',     '합계',     '남자'),
                     (     '인구',     '합계',     '여자'),
                     (     '인구',    '한국인',      '계'),
                     (     '인구',    '한국인',     '남자'),
                     (     '인구',    '한국인',     '여자'),
                     (     '인구',  '등록외국인',      '계'),
                     (     '인구',  '등록외국인',     '남자'),
```

계층 색인의 경우, 특정 컬럼을 선택할 때 튜플(Tuple)을 사용합니다.

Choropleth을 활용하여 GeoJson과 데이터 결합 실습

◎ 행정구역 경계 표시 및 인구수에 따른 행정구역별 색상 차별화

✔ 오브젝트 타입의 값을 숫자값으로 변환

5. Choropleth을 활용하여 GeoJson과 데이터 결합 실습

```
In [72]: population = pd.read_csv('data/자치구별인구수.txt', delimiter = '\t', header = [0,1,2], thousands = ',')
```

```
In [73]: population.head()
```

```
In [74]: population.columns
```

```
In [ ]:
```

```
In [77]: population.head()
```

...

```
In [76]: population.info()
```

thousands

',' → 천 단위의 콤마는 무시하고 숫자로 읽음

Choropleth을 활용하여 GeoJson과 데이터 결합 실습

◎ 행정구역 경계 표시 및 인구수에 따른 행정구역별 색상 차별화

✔ Choropleth 객체 생성 후 Map에 추가

```
                                          26 non-null      int64
                                          26 non-null      int64
                                          26 non-null      int64
                                          26 non-null      int64
 12  (세대당인구, 세대당인구, 세대당인구)          26 non-null      float64
 13  (65세이상고령자, 65세이상고령자, 65세이상고령자)  26 non-null      int64
dtypes: float64(1), int64(11), object(2)
memory usage: 3.0+ KB
```

```
In [83]: map5 = folium.Map(location = [37.566535, 126.9779691999996], zoom_start = 10,
                               zoom_control = False, control_scale = True,
                           tiles = 'https://tiles.stadiamaps.com/tiles/outdoors/{z}/{x}/{y}{r}.png',
                           attr = '&copy; <a href="https://stadiamaps.com/">Stadia Maps</a>, &copy; <a href="https:
         map5
```

```
In [ ]: folium.Choropleth(geo_data = geo,
                          )
```

folium.Choropleth 객체 생성

```
In [ ]:

In [ ]:

In [ ]:
```

Choropleth을 활용하여 GeoJson과 데이터 결합 실습

◎ 행정구역 경계 표시 및 인구수에 따른 행정구역별 색상 차별화

✔ Choropleth 객체 생성 후 Map에 추가

```
26 non-null     int64
26 non-null     int64
26 non-null     int64
26 non-null     int64
12  (세대당인구, 세대당인구, 세대당인구)                26 non-null     float64
13  (65세이상고령자, 65세이상고령자, 65세이상고령자)  26 non-null     int64
dtypes: float64(1), int64(11), object(2)
memory usage: 3.0+ KB
```

```
In [83]: map5 = folium.Map(location = [37.566535, 126.97796919999996], zoom_start = 10,
                           zoom_control = False, control_scale = True,
                           tiles = 'https://tiles.stadiamaps.com/tiles/outdoors/{z}/{x}/{y}{r}.png'
```

data 행정구역별 색상을 결정해 줄 수치화된 값을 가지고 있는 데이터

```
In [ ]: folium.Choropleth(geo_data = geo,
                          name = 'choropleth',
                          data = population)
```

```
In [ ]:
```

```
In [ ]:
```

```
In [ ]:
```

Choropleth을 활용하여 GeoJson과 데이터 결합 실습

◎ 행정구역 경계 표시 및 인구수에 따른 행정구역별 색상 차별화

✔ Choropleth 객체 생성 후 Map에 추가

```
26 non-null     int64
26 non-null     int64
26 non-null     int64
26 non-null     int64
12  (세대당인구, 세대당인구, 세대당인구)                26 non-null     float64
13  (65세이상고령자, 65세이상고령자, 65세이상고령자)  26 non-null     int64
dtypes: float64(1), int64(11), object(2)
ry usage: 3.0+ KB
```

columns

시각화에 사용할 컬럼명을 지정. 2개의 컬럼만 사용해야 함
(① geojson 데이터와의 결합을 위한 컬럼, ② 색상 지정에 사용할 컬럼)

```
In [ ]: folium.Choropleth(geo_data = geo,
                          name = 'choropleth',
                          data = population,
                          columns = [])
```

```
In [ ]:
```

```
In [ ]:
```

```
In [ ]:
```

Choropleth을 활용하여 GeoJson과 데이터 결합 실습

◎ 행정구역 경계 표시 및 인구수에 따른 행정구역별 색상 차별화

✔ Choropleth 객체 생성 후 Map에 추가

```
6.9779691999996], zoom_start = 10,
ontrol_scale = True,
diamaps.com/tiles/outdoors/{z}/{x}/{y}{r}.png',
        attr = '&copy; <a href="https://stadiamaps.com/">Stadia Maps</a>, &copy; <a href="https:
map5
```

```
In [ ]: folium.Choropleth(geo_data = geo,
                          name = 'choropleth',
                           data = population,
                           columns = [('자치구','자치구','자치구'), ('인구','합계','계')]),
                          key_on = ''
```

```
In [ ]:
```

key_on 데이터 결합을 위한 geojson에 있는 키 지정

```
In [ ]:
```

```
In [ ]:
```

Choropleth을 활용하여 GeoJson과 데이터 결합 실습

◎ 행정구역 경계 표시 및 인구수에 따른 행정구역별 색상 차별화

✔ Choropleth 객체 생성 후 Map에 추가

```
6.9779691999996], zoom_start = 10,
ontrol_scale = True,
diamaps.com/tiles/outdoors/{z}/{x}/{y}{r}.png',
        attr = '&copy; <a href="https://stadiamaps.com/">Stadia Maps</a>, &copy; <a href="https:
map5
```

...

In [84]:
```
geo
```

...

In [88]:
```
folium.Choropleth(geo_data = geo,
                  name = 'choropleth',
                   data = population,
                   columns = [('자치구','자치구','자치구'), ('인구','합계','계')],
                  key_on = 'feature.properties.SIG_KOR_NM',
                  legend_name = '인구수',
                  fill_color = 'BuPu'
                ).add_to(map5)
```

```
  File "<tokenize>", line 8
    ).add_to(map5)
    ^
IndentationError: unindent does not match any outer indentation level
```

Choropleth을 활용하여 GeoJson과 데이터 결합 실습

◎ 행정구역 경계 표시 및 인구수에 따른 행정구역별 색상 차별화

✔ Map 실행

```
5 = folium.Map(location = [37.566535, 126.9779691999996], zoom_start = 10,
               zoom_control = False, control_scale = True,
               tiles = 'https://tiles.stadiamaps.com/tiles/outdoors/{z}/{x}/{y}{r}.png',
               attr = '&copy; <a href="https://stadiamaps.com/">Stadia Maps</a>, &copy; <a href="https:
map5
```

...

In [84]:
```
geo
```

...

In [89]:
```
folium.Choropleth(geo_data = geo,
                  name = 'choropleth'
                   data = population,
                   columns = [('자치구','자치구','자치구'), ('인구','합계','계')],
                  key_on = 'feature.properties.SIG_KOR_NM',
                  legend_name = '인구수',
                  fill_color = 'BuPu'
                ).add_to(map5)
```

Out[89]:
```
<folium.features.Choropleth at 0x188add153d0>
```

In [90]:
```
map5
```

Out[90]:

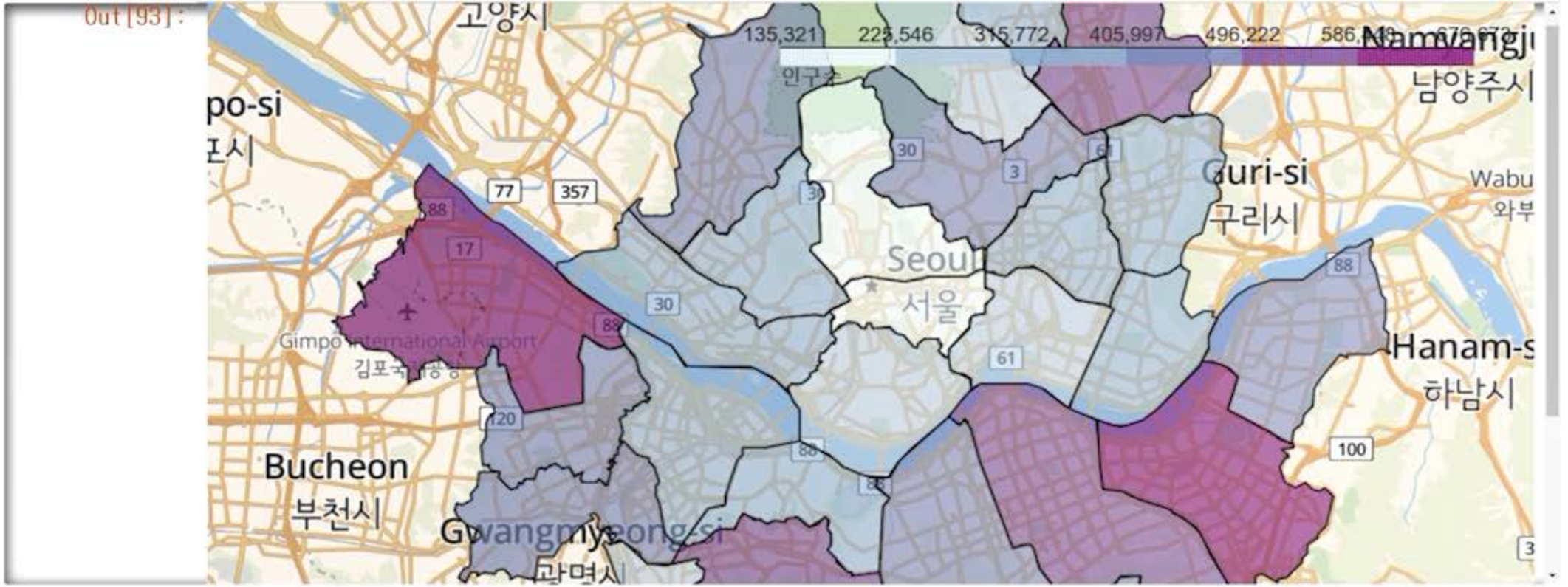
Choropleth을 활용하여 GeoJson과 데이터 결합 실습

◎ 행정구역 경계 표시 및 인구수에 따른 행정구역별 색상 차별화

Out[92]: <folium.features.Choropleth at 0x188add35eb0>

In [93]: map5

Out[93]:



In [ ]:

Folium 활용한 지도 시각화

학습완료

1/ 서울시 유동 인구 조사 지점 지도 시각화 (feat. 좌표계 변환)

2/ GeoJson으로 경계 표현하기

3/ Choropleth로 데이터와 GeoJson 결합하기

수고하셨습니다!