Out[24]:

Click here to toggle on/off the raw code.

🐍 [One Point Tutorial] Visualization III - `folium`

Python을 활용한 데이터 시각화

December, 2019

---

3. *folium* 을 배워 보자 !

- **folium이란?**

  1. 파이썬의 가장 유명한 지도 시각화 패키지
  2. 분석 프레임워크로 많이 쓰이는 pandas와 쉽게 연동되어 간편하게 시각화 가능
  3. leaflet.js를 기반으로 지도를 시각화

> **leaflet.js가 뭐야?**
> 가볍고 간단한 Mapping을 할 수 있는 오픈 소스 자바 스크립트 라이브러리

학습 목표

`folium` 의 핵심적인 시각화 기법을 이해하고 활용함

목차

import module (모듈 설치 후 불러오기)

folium 모듈이 설치 안 되어 있다면, 설치부터 하기 !

<window의 경우>
cmd 설치 방법 : pip install folium
쥬피터 설치 방법 : !pip install folium

```
Current Working Directory is changed.
```

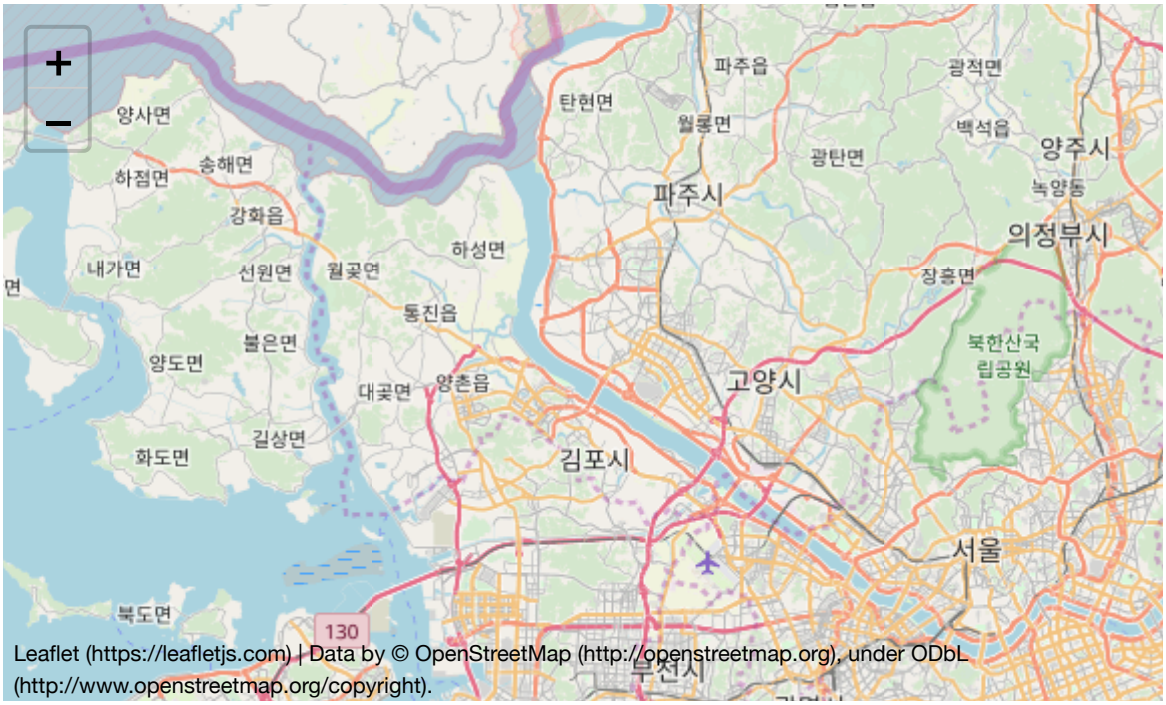
3.1. folium을 사용해보자, Basic

basic_위도와 경도를 알고 있을 때, *Map* 함수 사용

1. **Map 함수**를 이용하기 위해 필요한 정보는 **위도, 경도**
2. folium Map 함수 사용 방법(코드) :
   객체 이름 = folium.Map(location=[위도값, 경도값])
3. 여기서 location으로 지정된 **위도와 경도가 나타내는 곳을 중심**으로 하여 지도가 그려짐

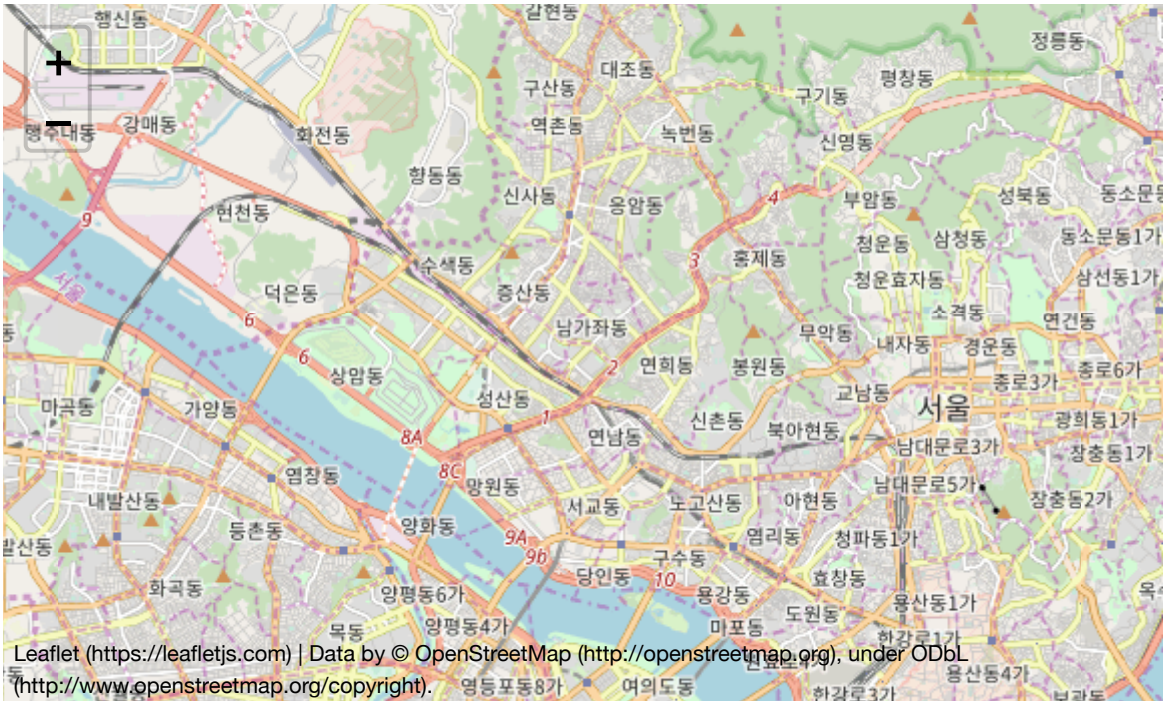
서울의 위도, 경도 (37.541, 126.986)를 사용하여 서울을 중심으로 하는 지도를 그려 보자! by *Map*

Out[7]:



초기 지도 범위를 조정하고 싶다면, *zoom_start*를 활용하자!

Out[8]:



지도 스타일도 바꿀 수 있어! by *tiles*

1. Map 함수의 default 스타일은 `Open Street Map` 을 기반으로 동작
2. `Stamen Terrain` , `Stamen Toner` , `Mapbox Bright` , `Mapbox Control room tiles` 등을 **tiles** 옵션을 통하여 적용 가능

Stamen Terrain을 적용시켰을 때,

Out[9]:

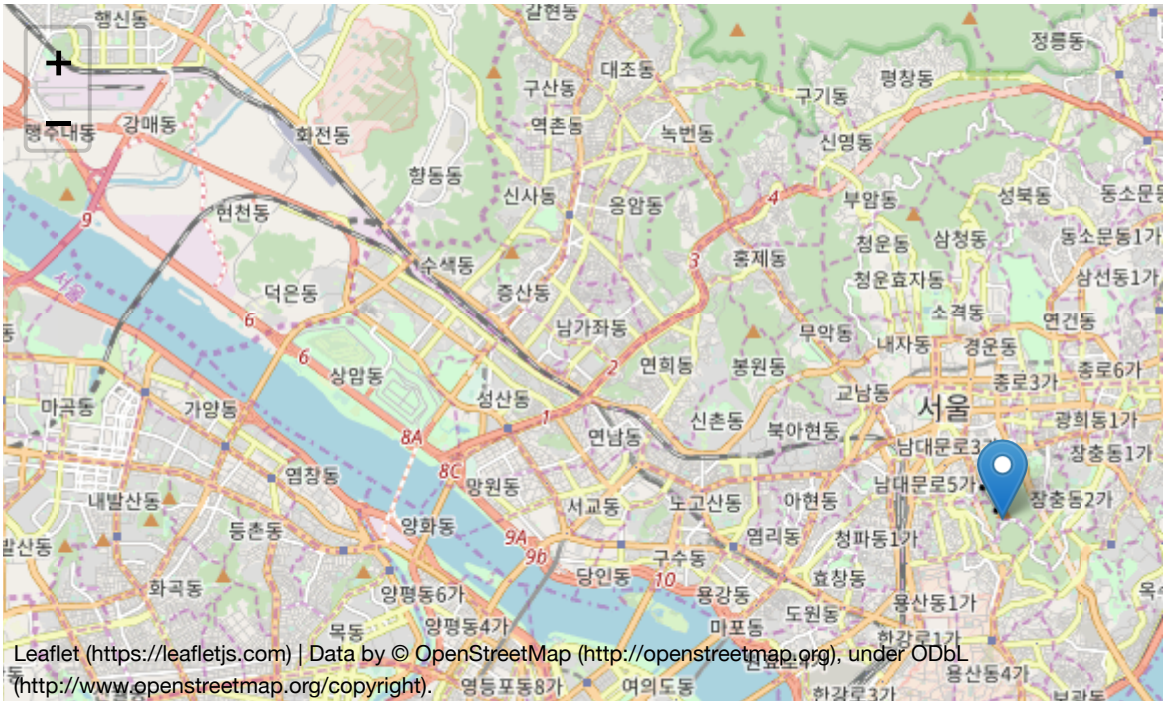
Out[11]:

원하는 곳에 *marker*와 *popup*을 설정해보자!

1. **Marker** = 특정 위치를 표시하는 표식
2. **Popup** = 마커를 클릭하였을 때 나타나는 정보

> **사용 방법:**
>
> 1. 먼저 위에서 언급한 방법으로 지도를 생성한다.
> 2. folium.Marker([위도 , 경도], popup= 마커 클릭 시 보여주고 싶은 정보).add_to(위에서 생성한 지도 객체)
>
> 여기서 `Marker` 는 마커를 그려주는 역할, `add_to` 는 만들어진 마커를 지도에 추가해주는 역할을 한다.

Out[12]:



마커 아이콘과 색깔도 변경할 수 있어 ! by *icon* 옵션

***icon* 옵션 추가**

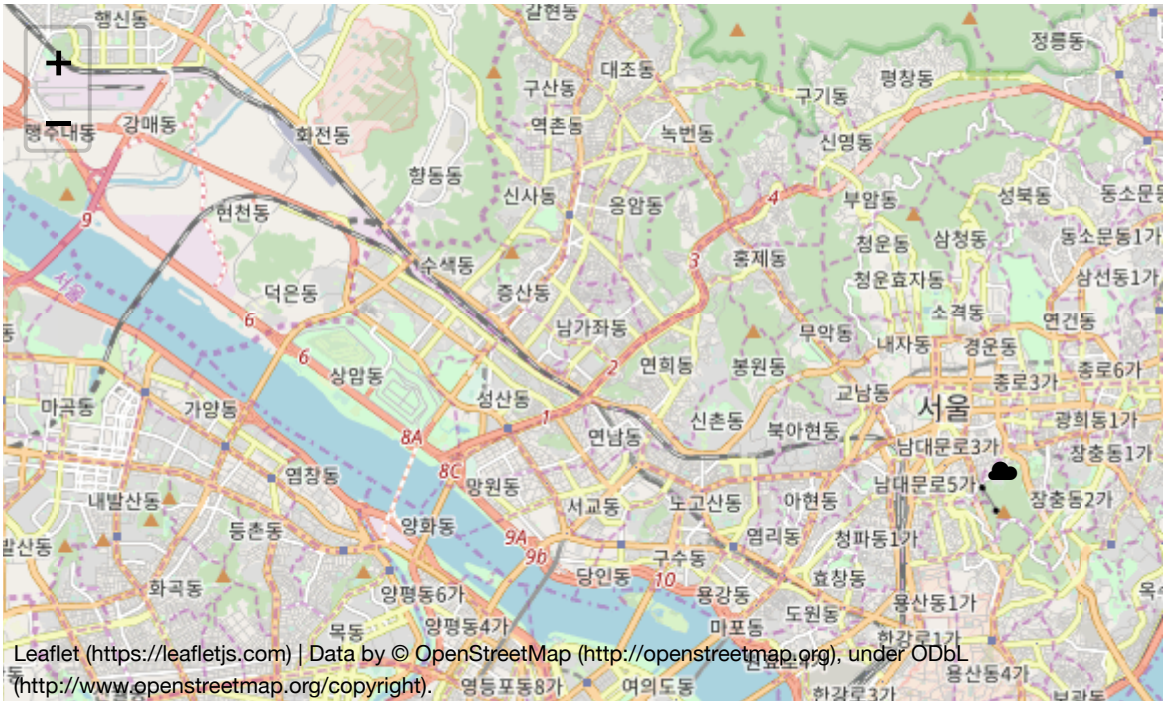
사용 방법 :

> Marker 안에서 `icon = folium.Icon(icon='원하는 옵션'))`

여기서 옵션은 `color` , `icon` 등이 있다.

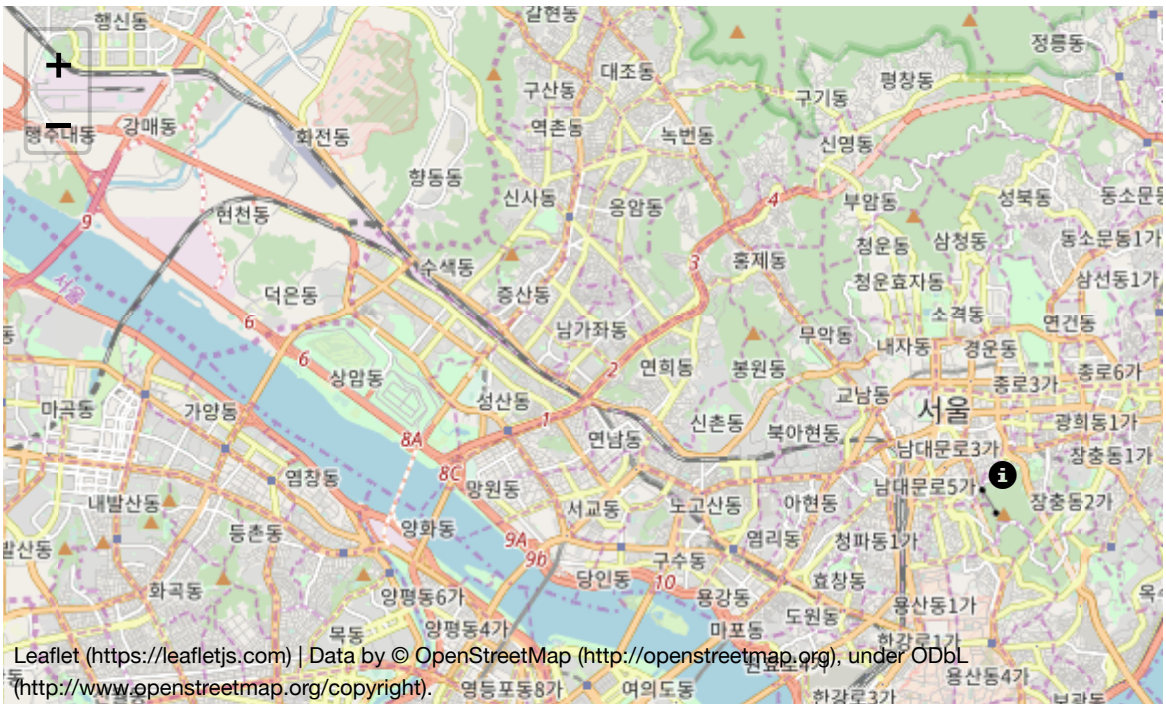
마커 아이콘을 구름으로 설정해볼까? (모양 변경은 `icon` 사용)

Out[13]:



이번에는 색깔도 바꿔볼까? (색깔 변경은 `color` 사용)

Out[14]:



원마커와 범위를 가진 마커도 설정할 수 있어 ! by *Circle & CircleMarker*

1. **Circle**
   사용 방법 :

   ```
   folium.Circle([ 위도 , 경도 ], popup= '원하는 이름' , radius=원하는 값 , color='원하는 색' ).add_to( 지도 객체 )
   ```

2. **CircleMarker**
   사용 방법 :

   ```
   folium.CircleMarker([ 위도 , 경도 ], popup= '원하는 이름' , radius=원하는 값 ,
    color='원하는 색' ,
    fill_color="원하는 색" ).add_to( 지도 객체 )
   ```

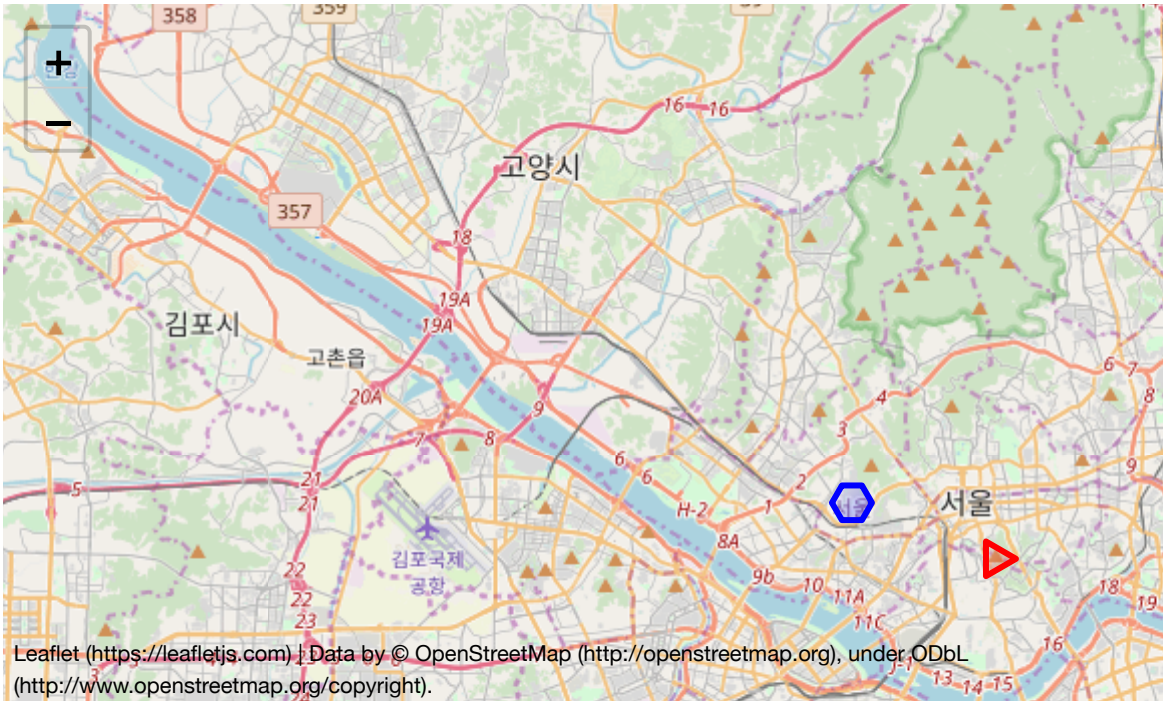
CircleMarker에서 color는 원의 테두리색, fill_color는 원 안의 채워지는 색을 의미한다.

Out[15]:

다각형 마커도 설정해볼까 ! by *RegularPolygonMarker & number_of_sides*

사용 방법:
folium.RegularPolygonMarker([위도 , 경도], popup= 원하는 이름 , fill_color= 채워질 색상 , color= 테두리 색상 , number_of_sides= 변의 수 , radius= 반경).add_to(지도 객체)

Out[16]:



3.2. folium을 사용해보자, Advanced

GeoJSON/TopoJSON Overlays_위도, 경도 직접 지정 안해도 데이터 불러와서 작업할 수 있다!

`GeoJSON` 형식 또는 `topoJSON` 형식으로 데이터를 지정하면, 오버레이를 통해 마커의 형태로 위치 정보를 지도상에 표현 !

여러 개의 레이어들을 하나의 지도에 나타낼 수 있다 !

`GeoJSON` 와 `topoJSON` 모두 다양한 지리적 데이터 구조를 표현하기 위해 인코딩된 JSON

1. `GeoJSON` 은 다양한 지리 데이터 구조를 인코딩하기 위한 형식을 제공

2. `topoJSON` 은 `GeoJSON` 의 확장형식, 각 영역을 아크(arcs)들의 영역으로 구분하여 표시하는 기능을 제공해 연산량을 적게 해주는 장점

> `topoJSON` 은 중복이 되지 않고 위상(topology)에 따라 인코딩되기 때문에 `GeoJSON` 보다 약 10배정도 가벼움 -> 파일 사이즈가 작음

- #### 3.2.1. geojson과 topojson 데이터를 활용하여 지도 그려 보기!

import module

json 설치 안되신 분은 설치부터 해주세요 !

3.2.1.1. import data (데이터 불러오기)

`json` 형식의 데이터들을 불러오자 !

3.2.1.2. 지도 객체 및 레이어 설정 후 지도 그려보기!

1. 지도 객체는 위에서 언급한 folium.Map을 통해 지정
2. `folium.GeoJson` 이용하여 `GeoJSON` 데이터 활용
3. `folium.TopoJson` 이용하여 `topoJSON` 데이터 활용

Out[19]:

- #### geojson과 csv 데이터를 활용하여 지도 그려 보기!

import module (csv 파일 불러오는 데에 필요한 pandas 모듈 불러오기)

3.2.2.1. csv 데이터와 geojson 데이터를 mapping 시키자 ! by *Choropleth* maps

1. csv 데이터와 geojson 데이터는 각자 다른 파일에 있으므로, 지도에 얹으려면 두 데이터를 mapping 해야 한다!
2. 그 때 사용하는 것이 `folium.Choropleth` 함수

여기서 먼저 알고 가야 할 것! **<geojson의 생김새>**

```
{
“type”: “Feature”,
“geometry”: {
“type”: “Point”,
“coordinates”: [125.6, 10.1]
},
“properties”: { “name”: “Dinagat Islands”
}
}
```

이렇게 dictionary 형식 비슷하게 존재한다 !

사용 방법:

```
folium.Choropleth (
geo_data = "지도 데이터 파일 경로 (.geojson, geopandas.DataFrame)",
data = "시각화 하고자 하는 데이터파일. (pandas.DataFrame)",
columns = (지도 데이터와 매핑할 값, 시각화 하고자하는 변수),
key_on = "feature.데이터 파일과 매핑할 값",
fill_color = "시각화에 쓰일 색상",
legend_name = "범주 이름"
).add_to( 지도  객체 )
```

이렇게만 보면 잘 모르겠으니 실습을 통해 알아보도록 하자!

import data (데이터 불러오기)

`geojson` 데이터와 `csv` 데이터를 불러옵니다 !

데이터 설명 : 2012년도 미국 주별 실업률을 나타냅니다!

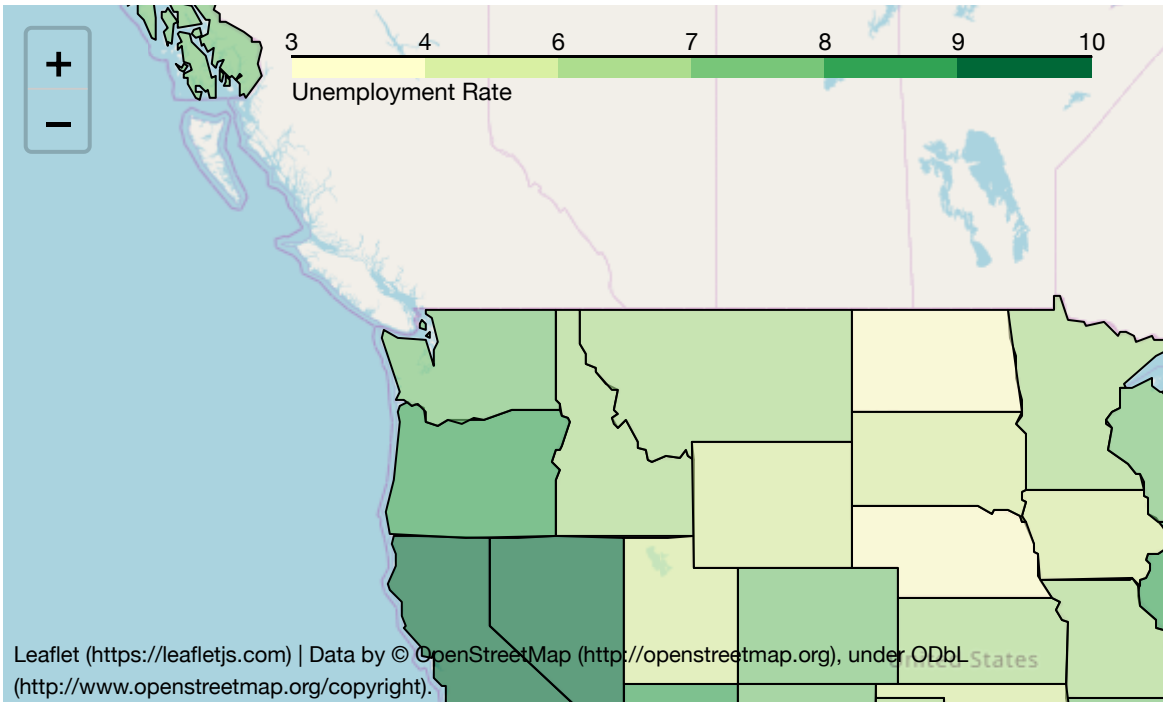
Out[21]:

	State	Unemployment
0	AL	7.1
1	AK	6.8
2	AZ	8.1
3	AR	7.2
4	CA	10.1

state_data 는 State와 Unemployment 두 가지 변수를 가진다.

choropleth 이용하여 지도에 나타내기 !

Out[23]:



우리가 시각화하고 싶은 *csv 파일*이 존재할 때, 지리에 대한 정보를 가지고 있는 *json 파일*이 있어야 지도 표현 가능 !