지수와 지도의 시각화

이번 장에서는 지금까지 살펴본 시각화외에도 plotly에서 추가적으로 제공하는 시각화 중 지수(index)의 시가화와 지도의 시각화를 살펴보도록 하겠다.

지수(Index)는 데이터 들을 특징일 종합하여 하나의 수치로 표현한 것을 말한다. 평균, 표준편차와 같은 수치도 가장 흔하게 만날 수 있는 지수 중에 하나이고 매일 매일 주식 시장 전체의 가격 변동을 종합한 KOPIS나 KOSDAQ와 같은 주가 종합 지수도 있다. 이렇게 하나의 수치를 시각화로 표현하는 방법을 plotly에서 제공한다.

지도(Map)의 시각화는 매우 흔하게 사용하는 시각화로 보통 행정 구역과 같은 공간 단위별로 수집된 데이터에 대해 무늬나 색상의 단계로 데이터를 표현하는 단계 구분도(Choropleth map)나 지도위에 위경도 좌표를 지닌 객체들을 표현하는 분산 맵(Scatter map) 등이 사용된다.

# 인디케이터(indicator)

인디케이터의 사전적 의미는 ‘가리키는 것’이다. 특정한 지시, 징후, 징표를 가리키는 것이라는 의미가 적절할 것이다. plotly에서 인디케이터의 의미는 ’value’ 속성으로 지정된 단일 값을 시각화하는 것이다. plotly에서는 인디케이터를 시각화하기 위해 indicator 트레이스를 제공한다. indicator 트레이스는 인디케이터로 표현할 수치를 수치, 델타, 게이지의 세 가지 방법을 사용하여 시각화하는 것이다. 이 세가지 방법은 ‘mode’ 속성을 통해 설정이 가능한데, 다음은 indicator 트레이스에서 사용하는 주요 속성이다.



## 수치(number) 인디케이터

수치 인디케이터는 단일 수치값을 시각화하는 방법이다. 이 수치 인디케이터는 indicator 트레이스 속성 중에 ‘number’ 속성에 의해 설정이 가능하다. 를 구성하는 각각의 구성 요소는 다음과 같다.



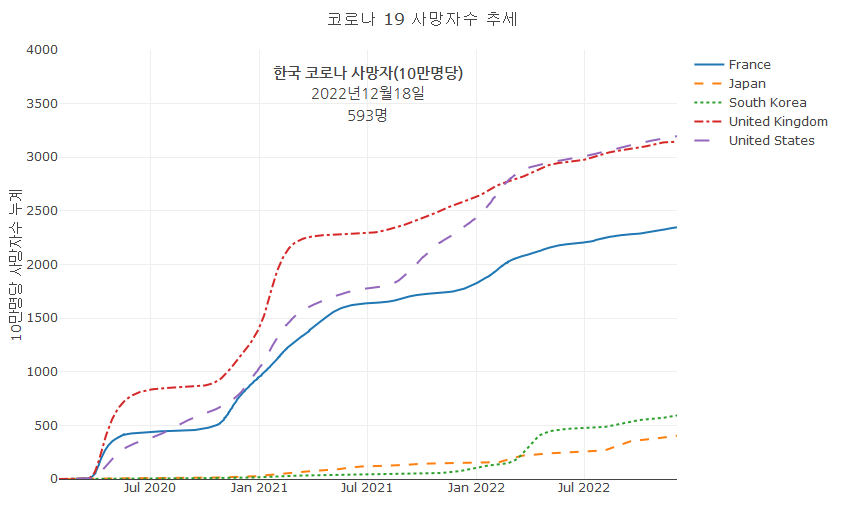
수치 인디케이터는 전체 인디케이터에서 표현되는 부분 중에 수치값을 표현하는 부분으로 수치 인디케이터가 단독으로 사용되기 보다는 뒤에 설명할 ’delta’나 ’guage’와 함께 전체 인디케이터를 구성하는 일부분으로 사용되는 경우가 일반적이다. 하지만 단순한 현재 수치를 표현할 때는 단독으로 사용될 수 있다.

다음은 주요 5개국의 코로나 19 사망자수 선 그래프에 한국의 코로나 사망자수를 indicator 트레이스로 추가하는 R과 python 코드이다. 한국의 코로나 사망자수를 그리고 나서 그래프 영역의 특정 부분(X축의 0.4~0.6, Y축의 0,85~0.95)에 수치 속성만을 가지는 indicator 트레이스를 추가하였다. indicator 트레이스의 ‘value’를 설정하기 위해 앞서 코로나 사망자수를 그리는데 사용했던 total\_deaths\_5\_nations\_by\_day 데이터프레임에서 한국의 10만명당 사망자수 데이터를 산출하고 이 데이터를 사용하여 indicator 트레이스를 생성하였다. indicator 트레이스의 위치를 설정하는데 ’domain’ 속성을 사용한다.

* R

R에서 indicator 트레이스를 만들기 위해서는 add\_trace(type = 'indicator', ...)를 사용한다. 수치 인디케이터는 ‘mode’ 속성을 ‘number’로 설정하고 ’number’ 속성을 설정하여 만들 수 있다.

number\_KOR <- total\_deaths\_5\_nations\_by\_day |>  
 filter(date == max(date), iso\_code == 'KOR') |>  
 select(total\_deaths\_per\_million) |> pull()  
  
fig1 <- total\_deaths\_5\_nations\_by\_day |>  
 plot\_ly() |>  
 ## scatter 트레이스 생성  
 add\_trace(type = 'scatter', mode = 'lines',   
 x = ~date, y = ~total\_deaths\_per\_million ,   
 linetype = ~location, connectgaps = T) |>  
 ## layout의 제목, 축제목, 여백 속성 설정  
 layout(title = list(text = '코로나 19 사망자수 추세', pad = list(b = 5)),   
 xaxis = list(title = ''),   
 yaxis = list(title = '10만명당 사망자수 누계', range = c(0, 4000)),   
 margin = margins\_R)  
  
fig1 |>  
 add\_trace(type = 'indicator', mode = 'number',   
 value = number\_KOR,   
 title = list(text = paste0('<b>한국 코로나 사망자(10만명당)</b>\n', year(today()), '년', month(today()), '월', day(today()), '일'),  
 font = list(family = '나눔고딕', size = 15)  
 ),  
 number = list(font = list(family = '나눔고딕',   
 size = 15),   
 suffix = '명'),  
 domain = list(x = c(0.4, 0.6), y = c(0.8, 0.9)))



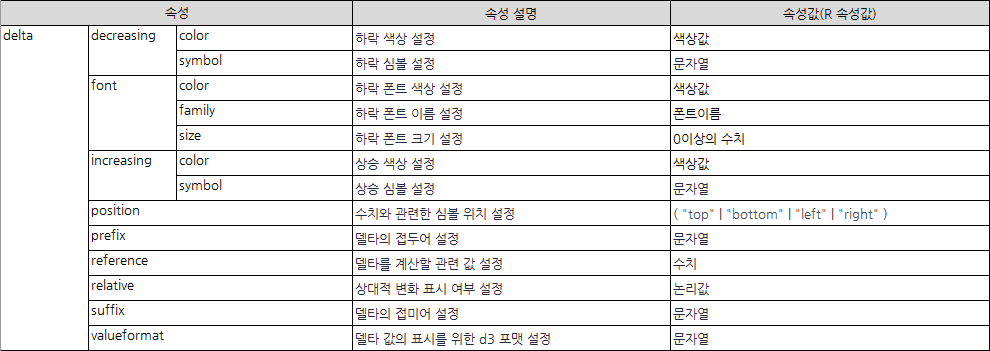
* python

python에서 indicator 트레이스를 만들기 위해서는 add\_trace()에 plotly.graph\_objects.Indicator()를 사용한다. 수치 인디케이터는 ‘number’ 속성을 설정하여 만들 수 있다.

from datetime import datetime, timedelta  
  
today = datetime.today()  
  
number\_KOR = total\_deaths\_5\_nations\_by\_day.loc[(total\_deaths\_5\_nations\_by\_day['date']==total\_deaths\_5\_nations\_by\_day['date'].max()) & (total\_deaths\_5\_nations\_by\_day['iso\_code']=='KOR')]  
number\_KOR = number\_KOR.loc[:,'total\_deaths\_per\_million'].values[0]  
  
fig = go.Figure()  
for location, group in total\_deaths\_5\_nations\_by\_day.groupby('location'):  
 fig.add\_trace(go.Scatter(  
 mode = 'lines',   
 x = group['date'],   
 y = group['total\_deaths\_per\_million'],   
 line = dict(dash = nations[location]),   
 name = location,  
 connectgaps = True  
 ))  
  
fig.update\_layout(title = dict(text = '코로나 19 사망자수 추세', x = 0.5),   
 xaxis = dict(title = ''),   
 yaxis = dict(title = '10만명당 사망자수 누계', range = (0, 4000)))   
  
fig.add\_trace(go.Indicator(  
 mode = 'number', value = number\_KOR,  
 title = dict(text = '<b>한국 코로나 사망자(10만명당)</b><br>' + str(today.year) + '년' + str(today.month) + '월' + str(today.day) + '일\n',  
 font = dict(family = '나눔고딕', size = 15)),  
 number = dict(font = dict(family = '나눔고딕', size = 15),  
 suffix = '명'),  
 domain = dict(x = (0.4, 0.6), y = (0.8, 0.9))  
))  
  
fig.show()

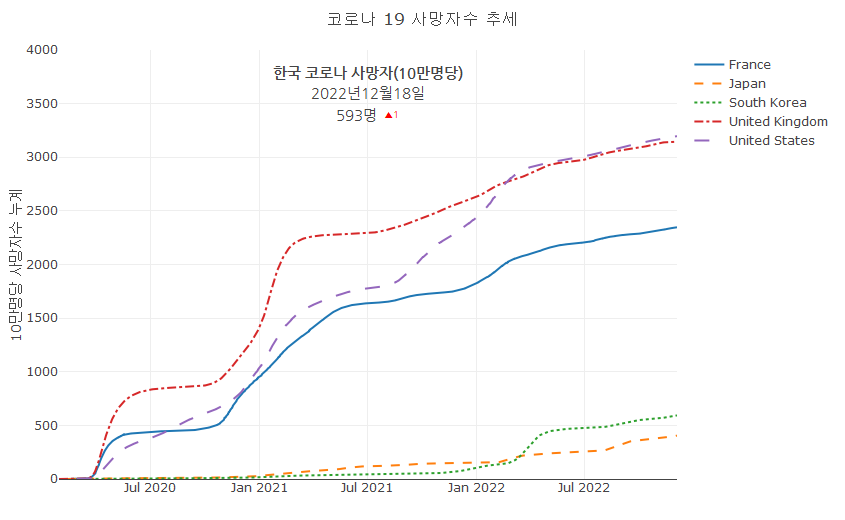
## 델타(delta) 인디케이터

델타 인디케이터는



* R

number1\_KOR <- total\_deaths\_5\_nations\_by\_day |>  
 filter(date == max(date)-1, iso\_code == 'KOR') |>  
 select(total\_deaths\_per\_million) |> pull()  
  
fig1 <- total\_deaths\_5\_nations\_by\_day |>  
 plot\_ly() |>  
 ## scatter 트레이스 생성  
 add\_trace(type = 'scatter', mode = 'lines',   
 x = ~date, y = ~total\_deaths\_per\_million ,   
 linetype = ~location, connectgaps = T) |>  
 ## layout의 제목, 축제목, 여백 속성 설정  
 layout(title = list(text = '코로나 19 사망자수 추세', pad = list(b = 5)),   
 xaxis = list(title = ''),   
 yaxis = list(title = '10만명당 사망자수 누계', range = c(0, 4000)),   
 margin = margins\_R)  
  
fig1 |>  
 add\_trace(type = 'indicator', mode = 'number+delta',   
 value = number\_KOR,   
 title = list(text = paste0('<b>한국 코로나 사망자(10만명당)</b>\n', year(today()), '년', month(today()), '월', day(today()), '일'),  
 font = list(family = '나눔고딕', size = 15)  
 ),  
 number = list(font = list(family = '나눔고딕',   
 size = 15),   
 suffix = '명'),  
 delta = list(reference = number1\_KOR, position = 'right',   
 increasing = list(color = 'red'),  
 decreasing = list(color = 'blue'),  
 font = list(family = '나눔고딕', size = 10)),  
 domain = list(x = c(0.4, 0.6), y = c(0.8, 0.9)))



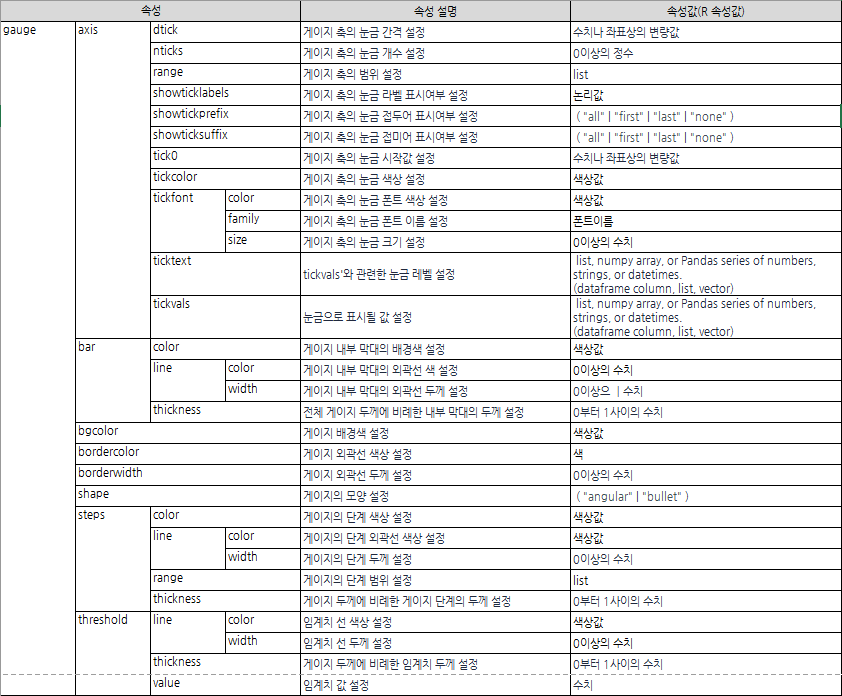
* python

number1\_KOR = total\_deaths\_5\_nations\_by\_day.loc[(total\_deaths\_5\_nations\_by\_day['date']==total\_deaths\_5\_nations\_by\_day['date'].max()- timedelta(days = 1)) & (total\_deaths\_5\_nations\_by\_day['iso\_code']=='KOR')]  
number1\_KOR = number1\_KOR.loc[:,'total\_deaths\_per\_million'].values[0]  
  
fig = go.Figure()  
for location, group in total\_deaths\_5\_nations\_by\_day.groupby('location'):  
 fig.add\_trace(go.Scatter(  
 mode = 'lines',   
 x = group['date'],   
 y = group['total\_deaths\_per\_million'],   
 line = dict(dash = nations[location]),   
 name = location,  
 connectgaps = True  
 ))  
  
fig.update\_layout(title = dict(text = '코로나 19 사망자수 추세', x = 0.5),   
 xaxis = dict(title = ''),   
 yaxis = dict(title = '10만명당 사망자수 누계', range = (0, 4000)))   
  
fig.add\_trace(go.Indicator(  
 mode = 'number+delta', value = number\_KOR,  
 title = dict(text = '<b>한국 코로나 사망자(10만명당)</b><br>' + str(today.year) + '년' + str(today.month) + '월' + str(today.day) + '일\n',  
 font = dict(family = '나눔고딕', size = 15)),  
 number = dict(font = dict(family = '나눔고딕', size = 15),  
 suffix = '명'),  
 delta = dict(reference = number1\_KOR, position = 'right',   
 increasing = dict(color = 'red'),  
 decreasing = dict(color = 'blue'),  
 font = dict(family = '나눔고딕', size = 10)),  
 domain = dict(x = (0.4, 0.6), y = (0.8, 0.9))  
))  
  
fig.show()

## 게이지(gauge) 인디케이터

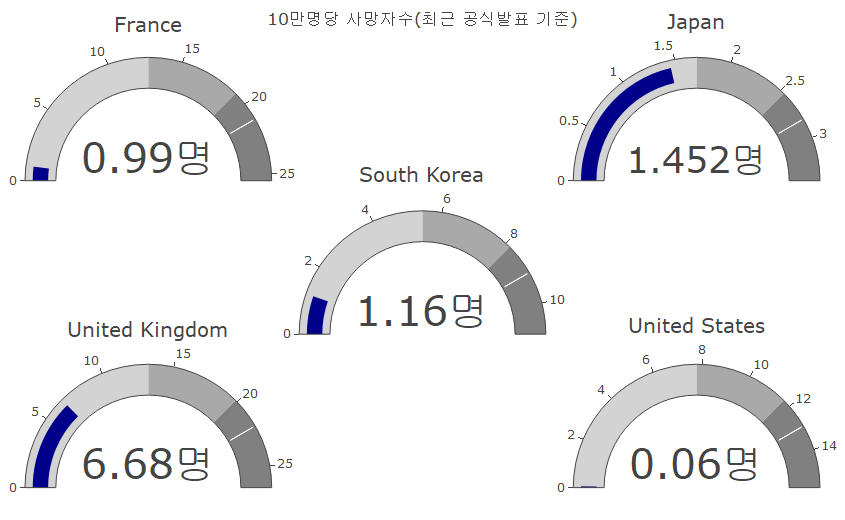
게이지 인디케이터는 둥근 반원의 형태의 틀에 둥글게 그려지는 막대의 각도에 의해 최소값과 최대값 사이에서의 현재의 지수를 표현할 때 사용하는 시각화이다. 이 시각화를 구성하는데에는 주 표현값인 ‘value’ 값 이외에 ‘steps’, ‘threshold’, ‘delta’의 속성을 사용한다. 게이지 인디케이터는 indicator 트레이스에 ’mode’ 속성에 “gauge”를 포함하여 만들 수 있다. “gauge”만을 사용하거나, “number+gauge”를 사용하여 수치와 함꼐 게이지를 사용하거나, “number+gauge+delta”를 사용하여 수치, 게이지, 증감 기호를 모두 사용할 수도 있다.

다음은 indicator 트레이스 중 ’gauge’와 관련된 주요 속성이다.



* R

max\_deaths\_per\_million\_by\_day <- total\_deaths\_5\_nations\_by\_day |> group\_by(location) |>  
 summarise(최대사망자 = max(new\_deaths\_per\_million, na.rm = TRUE))  
  
deaths\_per\_million\_in\_lateast <- total\_deaths\_5\_nations\_by\_day |> group\_by(location) |>  
 filter(is.na(new\_deaths\_per\_million) == FALSE) |>  
 filter(date == max(date)) |>  
 select(iso\_code, date, new\_deaths\_per\_million)  
  
  
df\_gauge <- left\_join(max\_deaths\_per\_million\_by\_day, deaths\_per\_million\_in\_lateast, by = 'location') |> arrange(location)  
  
fig\_gauge <- df\_gauge |> plot\_ly() |>  
 add\_trace(type = 'indicator', mode = "gauge+number", title = pull(df\_gauge[3, 1]),  
 domain = list(row = 1, column = 1), value = pull(df\_gauge[3, 5]),   
 gauge = list(axis = list(  
 range = list(NULL, pull(df\_gauge[3, 2])\*1.2)),  
 steps = list(  
 list(range = c(0, pull(df\_gauge[3, 2])\*1.2\*0.5), color = "lightgray"),  
 list(range = c(pull(df\_gauge[3, 2])\*1.2\*0.5, pull(df\_gauge[3, 2])\*1.2\*0.75), color = "darkgray"),  
 list(range = c(pull(df\_gauge[3, 2])\*1.2\*0.75, pull(df\_gauge[3, 2])\*1.2), color = "gray")),  
 threshold = list(line = list(color = 'white'),  
 value = pull(df\_gauge[3, 2])),   
 bar = list(color = "darkblue")),   
 number = list(suffix = '명'))  
  
fig\_gauge <- fig\_gauge |>  
 add\_trace(type = 'indicator', mode = "gauge+number", title = pull(df\_gauge[1, 1]),  
 domain = list(row = 0, column = 0), value = pull(df\_gauge[1, 5]),   
 gauge = list(axis = list(  
 range = list(NULL, pull(df\_gauge[1, 2])\*1.2)),  
 steps = list(  
 list(range = c(0, pull(df\_gauge[1, 2])\*1.2\*0.5), color = "lightgray"),  
 list(range = c(pull(df\_gauge[1, 2])\*1.2\*0.5, pull(df\_gauge[1, 2])\*1.2\*0.75), color = "darkgray"),  
 list(range = c(pull(df\_gauge[1, 2])\*1.2\*0.75, pull(df\_gauge[1, 2])\*1.2), color = "gray")),  
 threshold = list(line = list(color = 'white'),  
 value = pull(df\_gauge[1, 2])),   
 bar = list(color = "darkblue")),   
 number = list(suffix = '명'))  
  
fig\_gauge <- fig\_gauge |>  
 add\_trace(type = 'indicator', mode = "gauge+number", title = pull(df\_gauge[2, 1]),  
 domain = list(row = 0, column = 2), value = pull(df\_gauge[2, 5]),   
 gauge = list(axis = list(  
 range = list(NULL, pull(df\_gauge[2, 2])\*1.2)),  
 steps = list(  
 list(range = c(0, pull(df\_gauge[2, 2])\*1.2\*0.5), color = "lightgray"),  
 list(range = c(pull(df\_gauge[2, 2])\*1.2\*0.5, pull(df\_gauge[2, 2])\*1.2\*0.75), color = "darkgray"),  
 list(range = c(pull(df\_gauge[2, 2])\*1.2\*0.75, pull(df\_gauge[2, 2])\*1.2), color = "gray")),  
 threshold = list(line = list(color = 'white'),  
 value = pull(df\_gauge[2, 2])),   
 bar = list(color = "darkblue")),   
 number = list(suffix = '명'))  
  
fig\_gauge <- fig\_gauge |>  
 add\_trace(type = 'indicator', mode = "gauge+number", title = pull(df\_gauge[4, 1]),  
 domain = list(row = 2, column = 0), value = pull(df\_gauge[4, 5]),   
 gauge = list(axis = list(  
 range = list(NULL, pull(df\_gauge[4, 2])\*1.2)),  
 steps = list(  
 list(range = c(0, pull(df\_gauge[4, 2])\*1.2\*0.5), color = "lightgray"),  
 list(range = c(pull(df\_gauge[4, 2])\*1.2\*0.5, pull(df\_gauge[4, 2])\*1.2\*0.75), color = "darkgray"),  
 list(range = c(pull(df\_gauge[4, 2])\*1.2\*0.75, pull(df\_gauge[4, 2])\*1.2), color = "gray")),  
 threshold = list(line = list(color = 'white'),  
 value = pull(df\_gauge[4, 2])),   
 bar = list(color = "darkblue")),   
 number = list(suffix = '명'))  
  
fig\_gauge <- fig\_gauge |>  
 add\_trace(type = 'indicator', mode = "gauge+number", title = pull(df\_gauge[5, 1]),  
 domain = list(row = 2, column = 2), value = pull(df\_gauge[5, 5]),   
 gauge = list(axis = list(  
 range = list(NULL, pull(df\_gauge[5, 2])\*1.2)),  
 steps = list(  
 list(range = c(0, pull(df\_gauge[5, 2])\*1.2\*0.5), color = "lightgray"),  
 list(range = c(pull(df\_gauge[5, 2])\*1.2\*0.5, pull(df\_gauge[5, 2])\*1.2\*0.75), color = "darkgray"),  
 list(range = c(pull(df\_gauge[5, 2])\*1.2\*0.75, pull(df\_gauge[5, 2])\*1.2), color = "gray")),  
 threshold = list(line = list(color = 'white'),  
 value = pull(df\_gauge[5, 2])),   
 bar = list(color = "darkblue")),   
 number = list(suffix = '명'))  
  
fig\_gauge |> layout(grid=list(rows=3, columns=3),   
 margin = margins\_R,   
 title = '10만명당 사망자수(최근 공식발표 기준)')



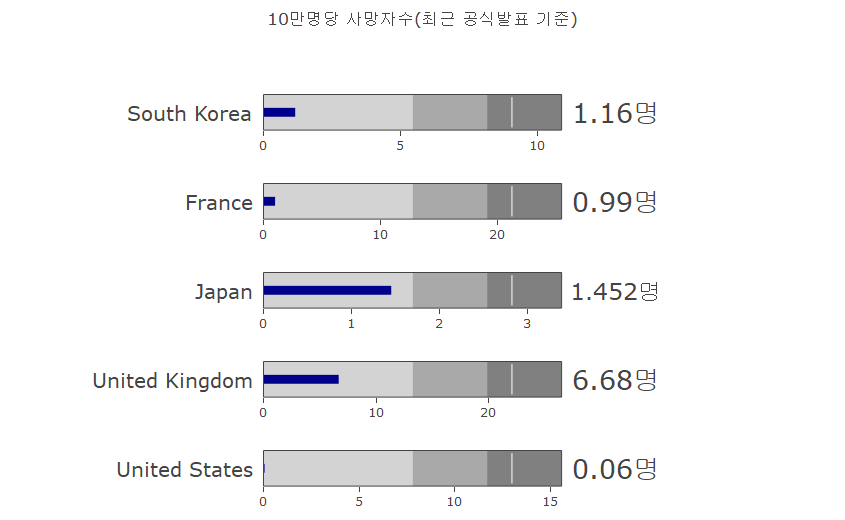
* python

deaths\_per\_million\_in\_lateast = total\_deaths\_5\_nations\_by\_day[total\_deaths\_5\_nations\_by\_day['new\_deaths\_per\_million'].isna() == False]  
deaths\_per\_million\_in\_lateast = pd.merge(deaths\_per\_million\_in\_lateast.groupby('location')['date'].max(), deaths\_per\_million\_in\_lateast, on = ("location", 'date'))[['iso\_code', 'location', 'date', 'new\_deaths\_per\_million']]  
df\_gauge = pd.merge(deaths\_per\_million\_in\_lateast, total\_deaths\_5\_nations\_by\_day.groupby('location')['new\_deaths\_per\_million'].max().reset\_index(), on = 'location').sort\_values('location')  
df\_gauge.columns = ('iso\_code', 'location', 'date', '최근사망자', '최대사망자')  
  
fig = go.Figure()  
fig.add\_trace(go.Indicator(  
 type = 'indicator', mode = "gauge+number", title = df\_gauge.iloc[2, 1],  
 domain = dict(row = 1, column = 1), value = df\_gauge.iloc[2, 3],   
 gauge = dict(axis = dict(  
 range = (0, df\_gauge.iloc[2, 4]\*1.2)),  
 steps = [  
 dict(range = (0, (df\_gauge.iloc[2, 4])\*1.2\*0.5), color = "lightgray"),  
 dict(range = ((df\_gauge.iloc[2, 4])\*1.2\*0.5, (df\_gauge.iloc[2, 4])\*1.2\*0.75), color = "darkgray"),   
 dict(range = ((df\_gauge.iloc[2, 4])\*1.2\*0.75, (df\_gauge.iloc[2, 4])\*1.2), color = "gray")],  
 threshold = dict(  
 line = dict(color = 'white'),  
 value = df\_gauge.iloc[2, 4]),   
 bar = dict(color = "darkblue")),   
 number = dict(suffix = '명')  
))  
  
fig.add\_trace(go.Indicator(  
 type = 'indicator', mode = "gauge+number", title = df\_gauge.iloc[0, 1],  
 domain = dict(row = 0, column = 0), value = df\_gauge.iloc[0, 3],   
 gauge = dict(axis = dict(  
 range = (0, df\_gauge.iloc[0, 4]\*1.2)),  
 steps = [  
 dict(range = (0, (df\_gauge.iloc[0, 4])\*1.2\*0.5), color = "lightgray"),  
 dict(range = ((df\_gauge.iloc[0, 4])\*1.2\*0.5, (df\_gauge.iloc[0, 4])\*1.2\*0.75), color = "darkgray"),  
 dict(range = ((df\_gauge.iloc[0, 4])\*1.2\*0.75, (df\_gauge.iloc[0, 4])\*1.2), color = "gray")],  
 threshold = dict(  
 line = dict(color = 'white'),  
 value = df\_gauge.iloc[0, 4]),   
 bar = dict(color = "darkblue")),   
 number = dict(suffix = '명')))  
  
fig.add\_trace(go.Indicator(  
 type = 'indicator', mode = "gauge+number", title = df\_gauge.iloc[1, 1],  
 domain = dict(row = 0, column = 2), value = df\_gauge.iloc[1, 3],   
 gauge = dict(axis = dict(  
 range = (0, df\_gauge.iloc[1, 4]\*1.2)),  
 steps = [  
 dict(range = (0, (df\_gauge.iloc[1, 4])\*1.2\*0.5), color = "lightgray"),  
 dict(range = ((df\_gauge.iloc[1, 4])\*1.2\*0.5, (df\_gauge.iloc[1, 4])\*1.2\*0.75), color = "darkgray"),  
 dict(range = ((df\_gauge.iloc[1, 4])\*1.2\*0.75, (df\_gauge.iloc[1, 4])\*1.2), color = "gray")],  
 threshold = dict(  
 line = dict(color = 'white'),  
 value = df\_gauge.iloc[1, 4]),   
 bar = dict(color = "darkblue")),   
 number = dict(suffix = '명')))  
  
fig.add\_trace(go.Indicator(  
 type = 'indicator', mode = "gauge+number", title = df\_gauge.iloc[3, 1],  
 domain = dict(row = 2, column = 0), value = df\_gauge.iloc[3, 3],   
 gauge = dict(axis = dict(  
 range = (0, df\_gauge.iloc[3, 4]\*1.2)),  
 steps = [  
 dict(range = (0, (df\_gauge.iloc[3, 4])\*1.2\*0.5), color = "lightgray"),  
 dict(range = ((df\_gauge.iloc[3, 4])\*1.2\*0.5, (df\_gauge.iloc[3, 4])\*1.2\*0.75), color = "darkgray"),  
 dict(range = ((df\_gauge.iloc[3, 4])\*1.2\*0.75, (df\_gauge.iloc[3, 4])\*1.2), color = "gray")],  
 threshold = dict(  
 line = dict(color = 'white'),  
 value = df\_gauge.iloc[3, 4]),   
 bar = dict(color = "darkblue")),   
 number = dict(suffix = '명')))  
  
fig.add\_trace(go.Indicator(  
 type = 'indicator', mode = "gauge+number", title = df\_gauge.iloc[4, 1],  
 domain = dict(row = 2, column = 2), value = df\_gauge.iloc[4, 3],   
 gauge = dict(axis = dict(  
 range = (0, df\_gauge.iloc[4, 4]\*1.2)),  
 steps = [  
 dict(range = (0, (df\_gauge.iloc[4, 4])\*1.2\*0.5), color = "lightgray"),  
 dict(range = ((df\_gauge.iloc[4, 4])\*1.2\*0.5, (df\_gauge.iloc[4, 4])\*1.2\*0.75), color = "darkgray"),  
 dict(range = ((df\_gauge.iloc[4, 4])\*1.2\*0.75, (df\_gauge.iloc[4, 4])\*1.2), color = "gray")],  
 threshold = dict(  
 line = dict(color = 'white'),  
 value = df\_gauge.iloc[4, 4]),   
 bar = dict(color = "darkblue")),   
 number = dict(suffix = '명')))  
  
fig.update\_layout(grid=dict(rows=3, columns=3),  
 title = dict(text = '10만명당 사망자수(최근 공식발표 기준)', x = 0.5))  
  
fig.show()

### bullet

* R

fig\_gauge <- df\_gauge |> plot\_ly() |>  
 add\_trace(type = 'indicator', mode = "gauge+number", title = pull(df\_gauge[3, 1]),  
 domain = list(x = c(0.3,0.8), y = c(0.82, 0.9)),  
 value = pull(df\_gauge[3, 5]),   
 gauge = list(axis = list(  
 range = list(NULL, pull(df\_gauge[3, 2])\*1.2)),  
 steps = list(  
 list(range = c(0, pull(df\_gauge[3, 2])\*1.2\*0.5), color = "lightgray"),  
 list(range = c(pull(df\_gauge[3, 2])\*1.2\*0.5, pull(df\_gauge[3, 2])\*1.2\*0.75), color = "darkgray"),  
 list(range = c(pull(df\_gauge[3, 2])\*1.2\*0.75, pull(df\_gauge[3, 2])\*1.2), color = "gray")),  
 shape = "bullet",  
 threshold = list(  
 line = list(color = 'white'),  
 value = pull(df\_gauge[3, 2])),   
 bar = list(color = "darkblue")),   
 number = list(suffix = '명'))  
  
fig\_gauge <- fig\_gauge |>  
 add\_trace(type = 'indicator', mode = "gauge+number", title = pull(df\_gauge[1, 1]),  
 domain = list(x = c(0.3,0.8), y = c(0.62, 0.7)),  
 value = pull(df\_gauge[1, 5]),   
 gauge = list(axis = list(  
 shape = "bullet",  
 range = list(NULL, pull(df\_gauge[1, 2])\*1.2)),  
 steps = list(  
 list(range = c(0, pull(df\_gauge[1, 2])\*1.2\*0.5), color = "lightgray"),  
 list(range = c(pull(df\_gauge[1, 2])\*1.2\*0.5, pull(df\_gauge[1, 2])\*1.2\*0.75), color = "darkgray"),  
 list(range = c(pull(df\_gauge[1, 2])\*1.2\*0.75, pull(df\_gauge[1, 2])\*1.2), color = "gray")),  
 shape = "bullet",  
 threshold = list(  
 line = list(color = 'white'),  
 value = pull(df\_gauge[1, 2])),   
 bar = list(color = "darkblue")),   
 number = list(suffix = '명'))  
  
fig\_gauge <- fig\_gauge |>  
 add\_trace(type = 'indicator', mode = "gauge+number", title = pull(df\_gauge[2, 1]),  
 domain = list(x = c(0.3,0.8), y = c(0.42, 0.5)),  
 value = pull(df\_gauge[2, 5]),   
 gauge = list(axis = list(  
 shape = "bullet",  
 range = list(NULL, pull(df\_gauge[2, 2])\*1.2)),  
 steps = list(  
 list(range = c(0, pull(df\_gauge[2, 2])\*1.2\*0.5), color = "lightgray"),  
 list(range = c(pull(df\_gauge[2, 2])\*1.2\*0.5, pull(df\_gauge[2, 2])\*1.2\*0.75), color = "darkgray"),  
 list(range = c(pull(df\_gauge[2, 2])\*1.2\*0.75, pull(df\_gauge[2, 2])\*1.2), color = "gray")),  
 shape = "bullet",  
 threshold = list(  
 line = list(color = 'white'),  
 value = pull(df\_gauge[2, 2])),   
 bar = list(color = "darkblue")),   
 number = list(suffix = '명'))  
  
fig\_gauge <- fig\_gauge |>  
 add\_trace(type = 'indicator', mode = "gauge+number", title = pull(df\_gauge[4, 1]),  
 domain = list(x = c(0.3,0.8), y = c(0.22, 0.3)),  
 value = pull(df\_gauge[4, 5]),   
 gauge = list(axis = list(  
 shape = "bullet",  
 range = list(NULL, pull(df\_gauge[4, 2])\*1.2)),  
 steps = list(  
 list(range = c(0, pull(df\_gauge[4, 2])\*1.2\*0.5), color = "lightgray"),  
 list(range = c(pull(df\_gauge[4, 2])\*1.2\*0.5, pull(df\_gauge[4, 2])\*1.2\*0.75), color = "darkgray"),  
 list(range = c(pull(df\_gauge[4, 2])\*1.2\*0.75, pull(df\_gauge[4, 2])\*1.2), color = "gray")),  
 shape = "bullet",  
 threshold = list(  
 line = list(color = 'white'),  
 value = pull(df\_gauge[4, 2])),   
 bar = list(color = "darkblue")),   
 number = list(suffix = '명'))  
  
fig\_gauge <- fig\_gauge |>  
 add\_trace(type = 'indicator', mode = "gauge+number", title = pull(df\_gauge[5, 1]),  
 domain = list(x = c(0.3,0.8), y = c(0.02, 0.1)),  
 value = pull(df\_gauge[5, 5]),   
 gauge = list(axis = list(  
 shape = "bullet",  
 range = list(NULL, pull(df\_gauge[5, 2])\*1.2)),  
 steps = list(  
 list(range = c(0, pull(df\_gauge[5, 2])\*1.2\*0.5), color = "lightgray"),  
 list(range = c(pull(df\_gauge[5, 2])\*1.2\*0.5, pull(df\_gauge[5, 2])\*1.2\*0.75), color = "darkgray"),  
 list(range = c(pull(df\_gauge[5, 2])\*1.2\*0.75, pull(df\_gauge[5, 2])\*1.2), color = "gray")),  
 shape = "bullet",  
 threshold = list(  
 line = list(color = 'white'),  
 value = pull(df\_gauge[5, 2])),   
 bar = list(color = "darkblue")),   
 number = list(suffix = '명'))  
  
fig\_gauge |> layout(margin = margins\_R,   
 title = '10만명당 사망자수(최근 공식발표 기준)')



* python

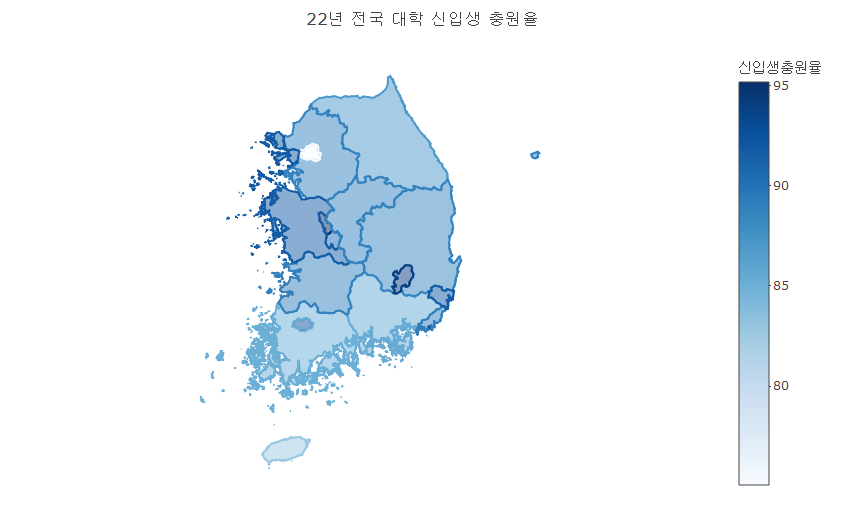
fig = go.Figure()  
fig.add\_trace(go.Indicator(  
 type = 'indicator', mode = "gauge+number", title = df\_gauge.iloc[2, 1],  
 domain = dict(x = (0.3,0.8), y = (0.82, 0.9)),  
 value = df\_gauge.iloc[2, 3],   
 gauge = dict(axis = dict(  
 range = (0, df\_gauge.iloc[2, 4]\*1.2)),  
 shape = "bullet",  
 steps = [  
 dict(range = (0, (df\_gauge.iloc[2, 4])\*1.2\*0.5), color = "lightgray"),  
 dict(range = ((df\_gauge.iloc[2, 4])\*1.2\*0.5, (df\_gauge.iloc[2, 4])\*1.2\*0.75), color = "darkgray"),  
 dict(range = ((df\_gauge.iloc[2, 4])\*1.2\*0.75, (df\_gauge.iloc[2, 4])\*1.2), color = "gray")],  
 threshold = dict(  
 line = dict(color = 'white'),  
 value = df\_gauge.iloc[2, 4]),   
 bar = dict(color = "darkblue")),   
 number = dict(suffix = '명')))  
  
fig.add\_trace(go.Indicator(  
 type = 'indicator', mode = "gauge+number", title = df\_gauge.iloc[0, 1],  
 domain = dict(x = (0.3,0.8), y = (0.62, 0.7)),  
 value = df\_gauge.iloc[0, 3],   
 gauge = dict(axis = dict(  
 range = (0, df\_gauge.iloc[0, 4]\*1.2)),  
 shape = "bullet",  
 steps = [  
 dict(range = (0, (df\_gauge.iloc[0, 4])\*1.2\*0.5), color = "lightgray"),  
 dict(range = ((df\_gauge.iloc[0, 4])\*1.2\*0.5, (df\_gauge.iloc[0, 4])\*1.2\*0.75), color = "darkgray"),  
 dict(range = ((df\_gauge.iloc[0, 4])\*1.2\*0.75, (df\_gauge.iloc[0, 4])\*1.2), color = "gray")],  
 threshold = dict(  
 line = dict(color = 'white'),  
 value = df\_gauge.iloc[0, 4]),   
 bar = dict(color = "darkblue")),   
 number = dict(suffix = '명')))  
  
fig.add\_trace(go.Indicator(  
 type = 'indicator', mode = "gauge+number", title = df\_gauge.iloc[1, 1],  
 domain = dict(x = (0.3,0.8), y = (0.42, 0.5)),  
 value = df\_gauge.iloc[1, 3],   
 gauge = dict(axis = dict(  
 range = (0, df\_gauge.iloc[1, 4]\*1.2)),  
 shape = "bullet",  
 steps = [  
 dict(range = (0, (df\_gauge.iloc[1, 4])\*1.2\*0.5), color = "lightgray"),  
 dict(range = ((df\_gauge.iloc[1, 4])\*1.2\*0.5, (df\_gauge.iloc[1, 4])\*1.2\*0.75), color = "darkgray"),  
 dict(range = ((df\_gauge.iloc[1, 4])\*1.2\*0.75, (df\_gauge.iloc[1, 4])\*1.2), color = "gray")],  
 threshold = dict(  
 line = dict(color = 'white'),  
 value = df\_gauge.iloc[1, 4]),   
 bar = dict(color = "darkblue")),   
 number = dict(suffix = '명')))  
  
fig.add\_trace(go.Indicator(  
 type = 'indicator', mode = "gauge+number", title = df\_gauge.iloc[3, 1],  
 domain = dict(x = (0.3,0.8), y = (0.22, 0.3)),  
 value = df\_gauge.iloc[3, 3],   
 gauge = dict(axis = dict(  
 range = (0, df\_gauge.iloc[3, 4]\*1.2)),  
 shape = "bullet",  
 steps = [  
 dict(range = (0, (df\_gauge.iloc[3, 4])\*1.2\*0.5), color = "lightgray"),  
 dict(range = ((df\_gauge.iloc[3, 4])\*1.2\*0.5, (df\_gauge.iloc[3, 4])\*1.2\*0.75), color = "darkgray"),  
 dict(range = ((df\_gauge.iloc[3, 4])\*1.2\*0.75, (df\_gauge.iloc[3, 4])\*1.2), color = "gray")],  
 threshold = dict(  
 line = dict(color = 'white'),  
 value = df\_gauge.iloc[3, 4]),   
 bar = dict(color = "darkblue")),   
 number = dict(suffix = '명')))  
  
fig.add\_trace(go.Indicator(  
 type = 'indicator', mode = "gauge+number", title = df\_gauge.iloc[4, 1],  
 domain = dict(x = (0.3,0.8), y = (0.02, 0.1)),  
 value = df\_gauge.iloc[4, 3],   
 gauge = dict(axis = dict(  
 range = (0, df\_gauge.iloc[4, 4]\*1.2)),  
 shape = "bullet",  
 steps = [  
 dict(range = (0, (df\_gauge.iloc[4, 4])\*1.2\*0.5), color = "lightgray"),  
 dict(range = ((df\_gauge.iloc[4, 4])\*1.2\*0.5, (df\_gauge.iloc[4, 4])\*1.2\*0.75), color = "darkgray"),  
 dict(range = ((df\_gauge.iloc[4, 4])\*1.2\*0.75, (df\_gauge.iloc[4, 4])\*1.2), color = "gray")],  
 threshold = dict(  
 line = dict(color = 'white'),  
 value = df\_gauge.iloc[4, 4]),   
 bar = dict(color = "darkblue")),   
 number = dict(suffix = '명')))  
  
fig.update\_layout(grid=dict(rows=3, columns=3),  
 title = dict(text = '10만명당 사망자수(최근 공식발표 기준)', x = 0.5))  
  
fig.show()

# 지도

## choropleth

* R

library(raster)  
library(sf)  
  
# Get the map data in sf format  
map\_data <- getData("GADM", country = "KOR", level = 1, type = "sf")  
st\_crs(map\_data) <- st\_crs(map\_data)  
  
province <- map\_data$HASC\_1  
  
#### 충원율 데이터  
df\_충원율 <- read\_excel('D:/R/git/datavisualization/plotly/RnPy/chap6/고등 주요 01-시도별 신입생 충원율(2010-2022)\_220825y.xlsx',   
 ## 'data' 시트의 데이터를 불러오는데,  
 sheet = 'Sheet1',  
 ## 앞의 10행을 제외하고  
 skip = 7,   
 ## 첫번째 행은 열 이름을 설정  
 col\_names = FALSE,   
 ## 열의 타입을 설정, 처음 8개는 문자형으로 다음 56개는 수치형으로 설정  
 col\_types = c(rep('text', 2), rep('numeric', 12)))  
  
df\_충원율 <- df\_충원율 |> dplyr::select(1, 2, 3, 4, 5)  
  
## df\_입학자의 열이름을 적절한 이름으로 설정  
colnames(df\_충원율) <- c('연도', '지역', '정원내모집인원', '정원내입학생수', '신입생충원율')  
  
  
df\_충원율 <- df\_충원율 |> filter(연도 == '2022', 지역 != '전국') |>   
 mutate(id = case\_when(  
 지역 == '강원' ~ 'KR.KW',   
 지역 == '경기' ~ 'KR.KG',  
 지역 == '경남' ~ 'KR.KN',  
 지역 == '경북' ~ 'KR.KB',  
 지역 == '광주' ~ 'KR.KJ',  
 지역 == '대구' ~ 'KR.TG',  
 지역 == '대전' ~ 'KR.TJ',  
 지역 == '부산' ~ 'KR.PU',  
 지역 == '서울' ~ 'KR.SO',  
 지역 == '세종' ~ 'KR.SJ',  
 지역 == '울산' ~ 'KR.UL',  
 지역 == '인천' ~ 'KR.IN',  
 지역 == '전남' ~ 'KR.CN',  
 지역 == '전북' ~ 'KR.CB',  
 지역 == '제주' ~ 'KR.CJ',  
 지역 == '충남' ~ 'KR.GN',  
 지역 == '충북' ~ 'KR.GB'  
 ))  
  
plot\_dat <- left\_join(map\_data, df\_충원율, by = c("HASC\_1" = "id")) %>%  
 st\_as\_sf()  
  
  
plot\_ly(plot\_dat) %>%  
 add\_sf(type = "scatter",   
 split = ~지역,  
 color = ~신입생충원율,  
 showlegend = F, # don't show a legend for each region  
 colors = "Blues",  
 text = ~paste0(지역, "\n", round(신입생충원율, 2), '%'),  
 hoveron = "fills",  
 hoverinfo = "text") %>%  
 layout(title = '22년 전국 대학 신입생 충원율',   
 margin = margins\_R)



* python

import json  
  
def cat(row):  
 key = row['지역']  
 value = {'강원' : '42','경기' : '41','경남' : '48','경북' : '47','광주' : '29','대구' : '27','대전' : '30','부산' : '26','서울' : '11','세종' : '36','울산' : '31','인천' : '28','전남' : '46','전북' : '45','제주' : '50','충남' : '44','충북' : '43'}.get(key)  
 return value  
  
geometry = json.load(open('D:/R/git/datavisualization/plotly/RnPy/chap6/TL\_SCCO\_CTPRVN.json', encoding='utf-8'))  
  
df\_충원율 = pd.read\_excel("D:/R/git/datavisualization/plotly/RnPy/chap6/고등 주요 01-시도별 신입생 충원율(2010-2022)\_220825y.xlsx",   
 sheet\_name = 'Sheet1',  
 skiprows=(6),   
 header = 0)  
df\_충원율 = df\_충원율.iloc[:, 0:5]  
df\_충원율.columns = ('연도', '지역', '정원내모집인원', '정원내입학생수', '신입생충원율')  
df\_충원율['지역코드']= df\_충원율.apply(cat, axis=1)  
df\_충원율 = df\_충원율[(df\_충원율['지역'] != '전국')&(df\_충원율['연도'] == 2022)]  
  
fig = go.Figure()  
  
fig.add\_trace(go.Choropleth(  
 geojson=geometry,   
 featureidkey='properties.CTPRVN\_CD',   
 locations = df\_충원율['지역코드'],   
 z = df\_충원율['신입생충원율'],   
 colorscale="Blues",   
# text = df\_충원율['신입생충원율'],  
 marker=dict(line=dict(width=1), opacity = 0.5),  
 colorbar = dict(y = 0.5, yanchor = 'middle'),  
 text = df\_충원율['지역'] + '\n' + round(df\_충원율['신입생충원율'], 2).astype(str) + '%',  
# hoveron = "fills",  
 hoverinfo = "text"  
))  
  
fig.update\_layout(  
 title = dict(text = '한국지도', x = 0.5),  
 geo=dict(  
 showframe = True,  
 fitbounds = "locations",  
 visible = False,  
 center=dict(lon=126.98, lat=37.56)  
 ),  
 autosize=False,  
 margin = dict(t = 50, b = 25, l = 25, r = 25,  
 pad=4,  
 autoexpand=True  
 ))

## scatter

* R

df\_univ <- read\_excel("D:/R/git/datavisualization/plotly/RnPy/chap6/university.xlsx",   
 col\_types = c('text', 'numeric', 'numeric'))  
   
map\_data\_lev2 <- getData("GADM", country = "KOR", level = 2, type = "sf")  
  
plot\_dat\_seoul <- plot\_dat |> filter(GID\_1 == 'KOR.16\_1')  
  
plot\_mapbox(plot\_dat\_seoul) |>  
 add\_sf() |>  
 add\_markers(data = df\_univ,   
 x = ~lon, y = ~lat,   
 marker = list(color = 'white'),  
 text = ~학교명) |>  
 layout(title = '서울지역 주요 대학',   
 margin = margins\_R,   
 showlegend = FALSE)

* python

mapbox\_access\_token = open(".mapbox\_token").read()  
  
fig = go.Figure(go.Scattermapbox(  
 lat=df\_univ['lat'],  
 lon=df\_univ['lon'],  
 mode='markers',  
 marker=go.scattermapbox.Marker(  
 size=9  
 ),  
 text=df\_univ['학교명'],  
 ))  
  
fig.update\_layout(  
 autosize=True,  
 hovermode='closest',  
 mapbox=dict(  
 accesstoken= mapbox\_access\_token,  
 bearing=0,  
 center=dict(lon=126.98, lat=37.56),  
 pitch=0,  
 zoom=10  
 ),  
)  
  
fig.show()