사례연구 6

– 분석 결과 시각화 –

A3조 이순규 문태웅 남원식

목차

Ⅰ.서론

1) 사용된 패키지

- Ggplot2 패키지(R)
- ploytly 패키지(R)
- Matplotlib 패키지(Python)

2) 사용된 데이터셋

막대차트 (가로), 막대차트 (세로), 누적막대차트, 원형차트, 점차트

2018 1분기	2019 1분기	2018 2분기	2019 2분기	2018 3분기	2019 3분기	2018 4분기	2019 4분기
305	450	320	460	330	480	380	520

상자 그래프

VADeaths 데이터셋

	Rural Male	Rural Female	Urban Male	Urban Female
50-54	11.7	8.7	15.4	8.4
55-59	18.1	11.7	24.3	13.6
60-64	26.9	20.3	37.0	19.3
65-69	41.0	30.9	54.6	35.1
70-74	66.0	54.3	71.1	50.0

히스토그램, 변수 간의 비교 시각화, 3차원 산점도 그래프, 밀도그래프

Iris 데이터셋

Sepal.Length	Sepal.width	Petal.Length	Petal.Width	Species
5.1	3.5	1.4	0.2	Setosa
4.9	3.0	1.4	0.2	Setosa

*중략

중첩자료 시각화

Parent	Child
70.5	61.7
68.5	61.7

*중략

지도 시각화

지역명	LAT	LON	총인구수	남자인구수	여자인구수	세대수
서울특별시	37.56510	126.9895	9,766,288	4,772,822	4,993,466	4,271,551
부산광역시	35.16277	129.0477	3,438,259	1,690,527	1,747,732	1.481,315

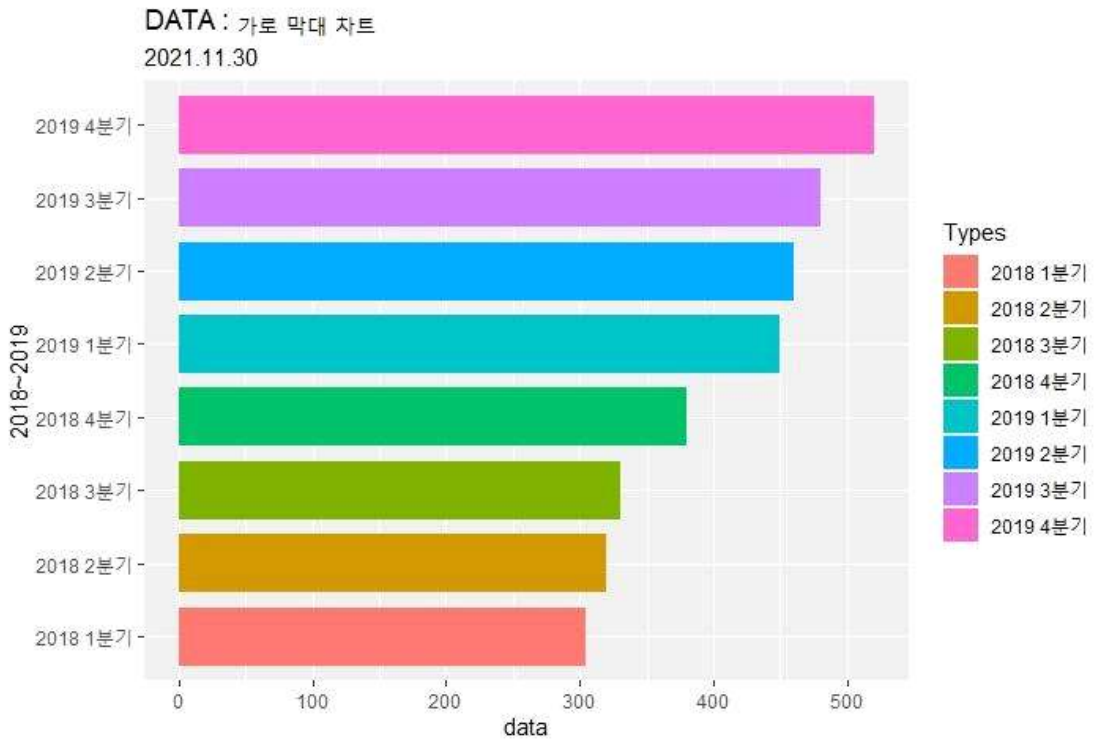
*중략

Ⅱ. 본론

1. 각 패키지별 차트 비교

1) 막대차트 (가로)

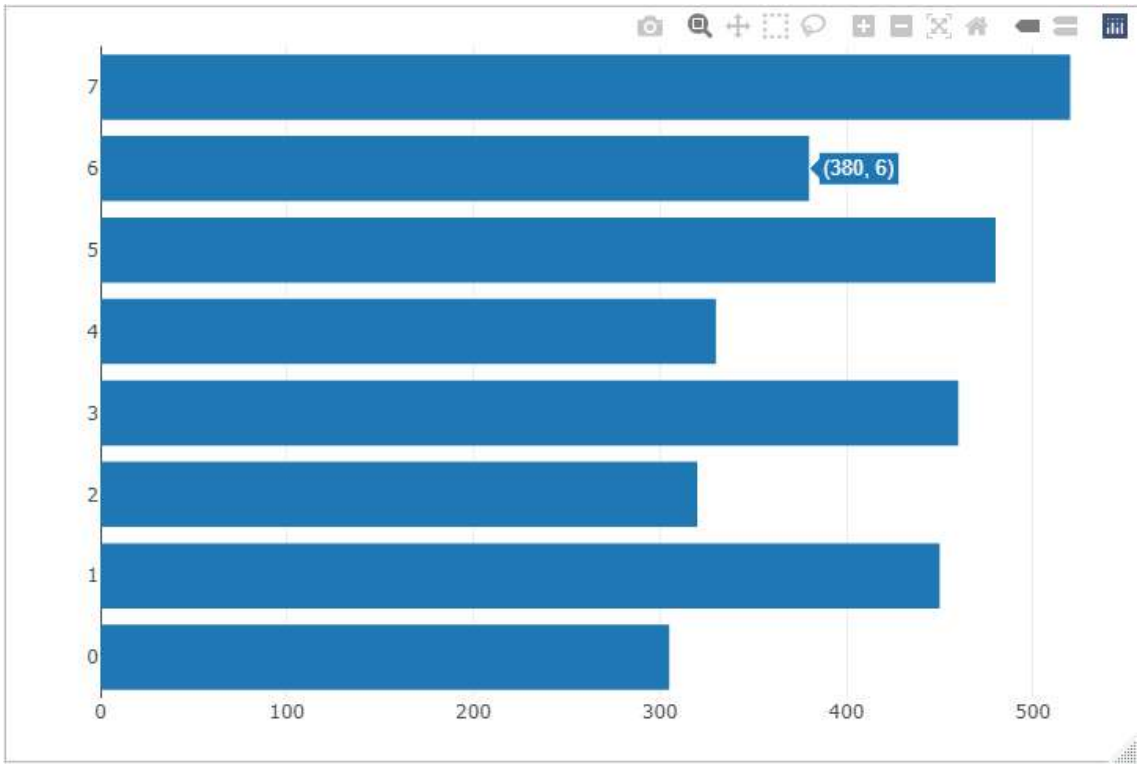
- ggplot2 패키지(R)



- 사용된 methods

ggplot + coord_flip() : 가로 그래프로 출력이 가능하다.

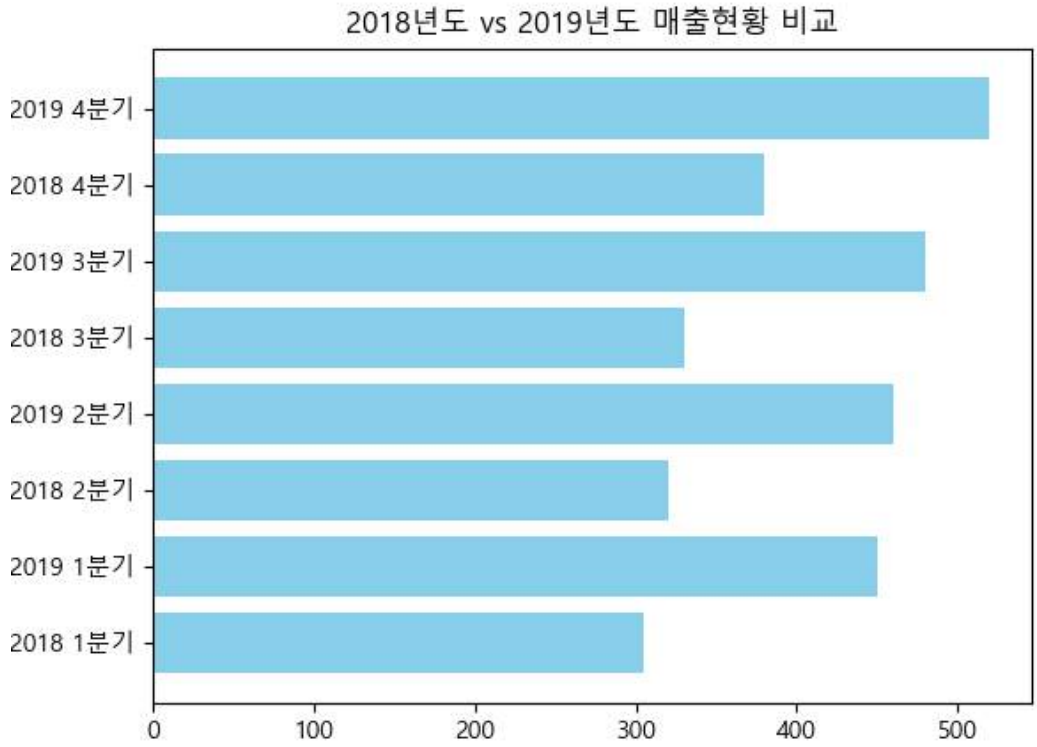
- plotly 패키지



- 사용된 methods

x축에 데이터값 입력시 가로로 출력이 가능하다.

- Matplotlib 패키지

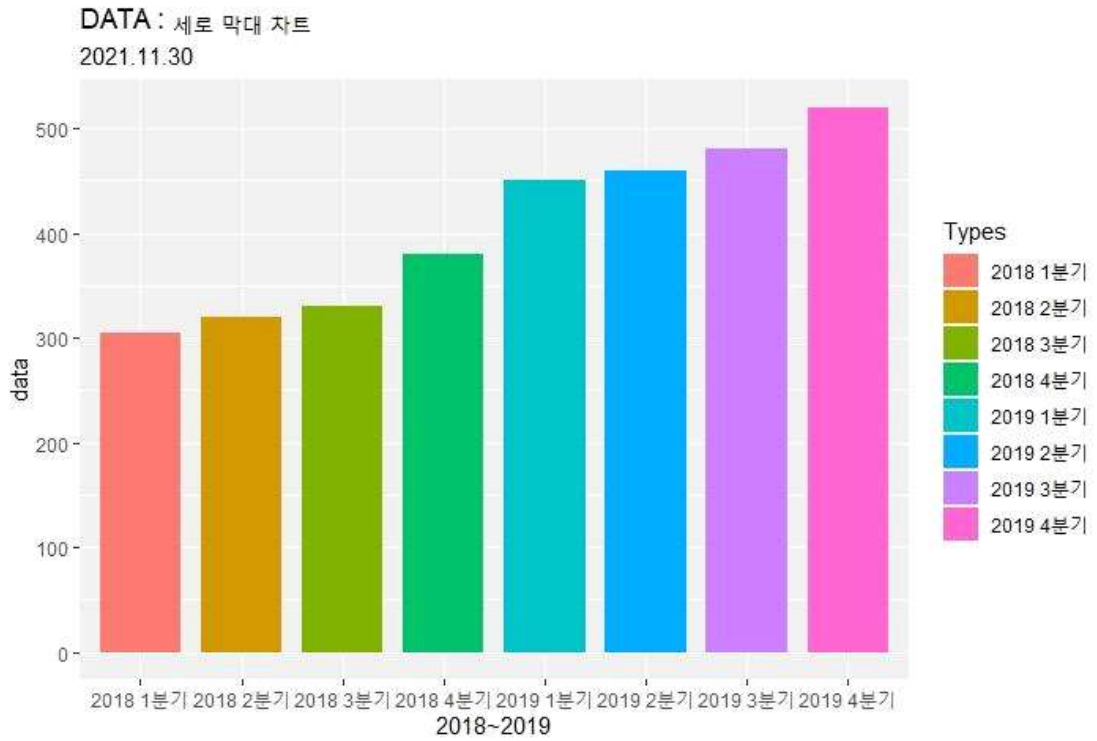


- 사용된 methods

plt.barh()를 이용하면 가로 막대 그래프 출력이 가능하다.

2) 막대차트(세로)

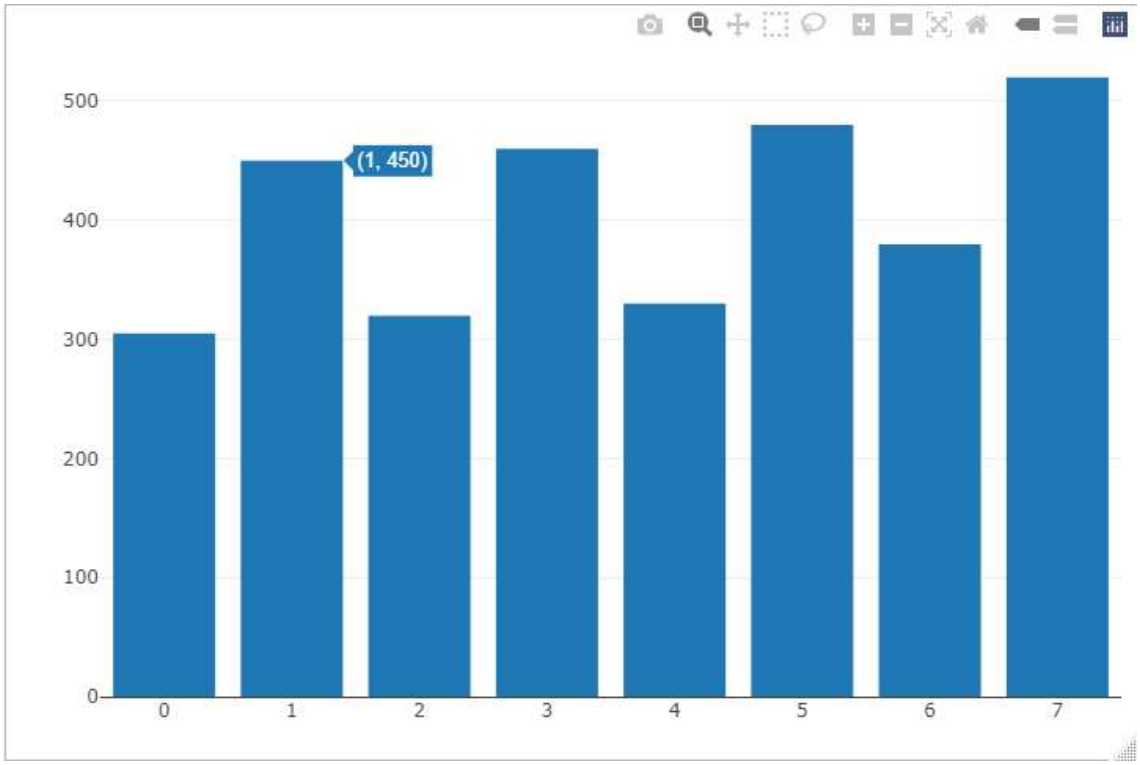
- ggplot2 패키지(R)



*사용된 methods

Default 값

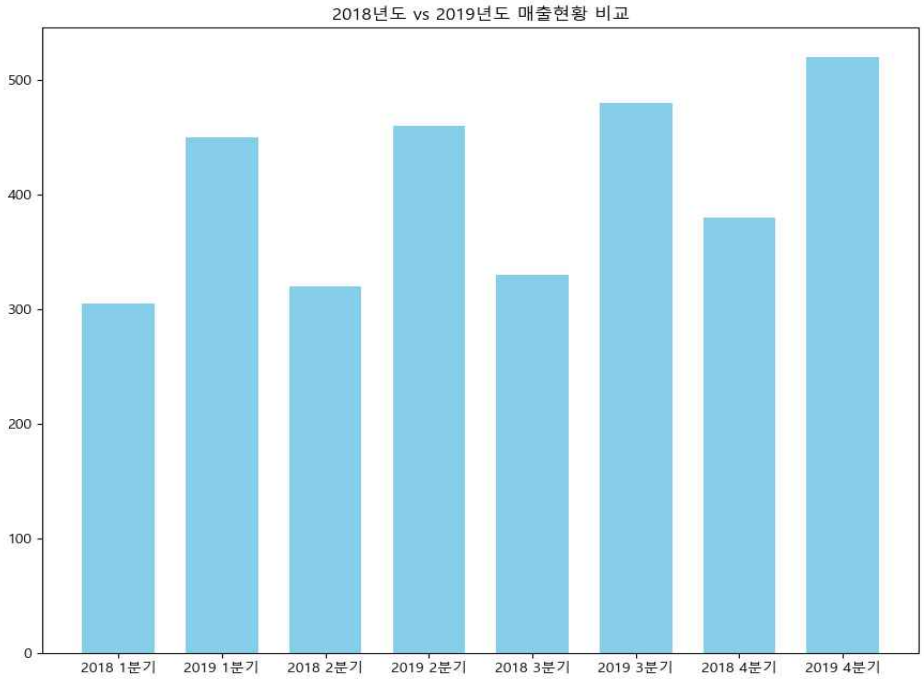
- plotly패키지(R)



*사용된 methods

Y축에 데이터값 입력시 세로로 출력이 가능하다.

- Matplotlib 패키지(Python)

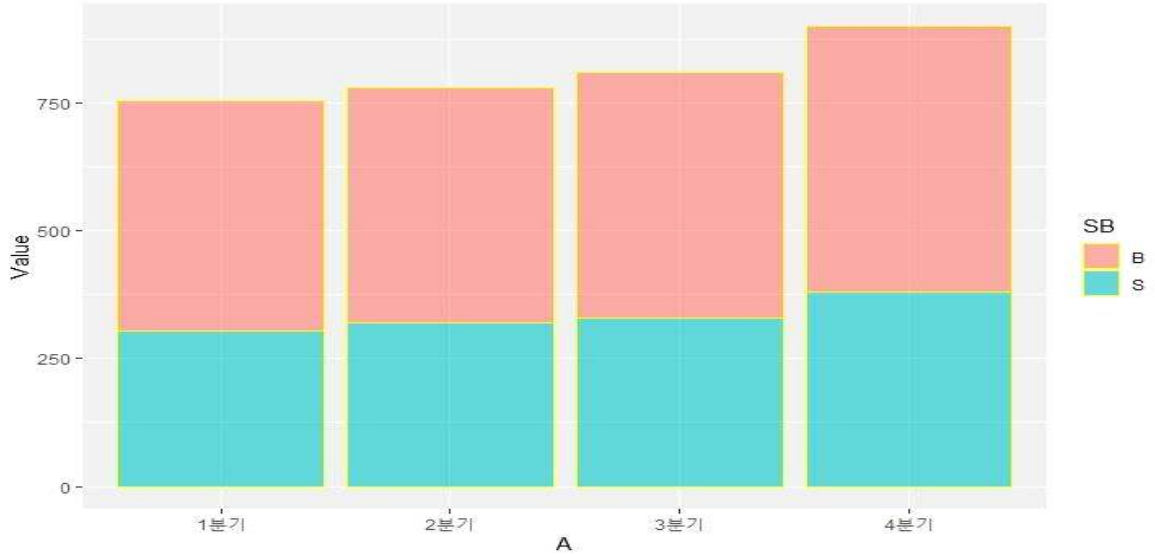


*사용된 methods

Plt.bar()를 이용하면 세로로 막대 그래프 출력이 가능하다.

2. 누적 막대 차트

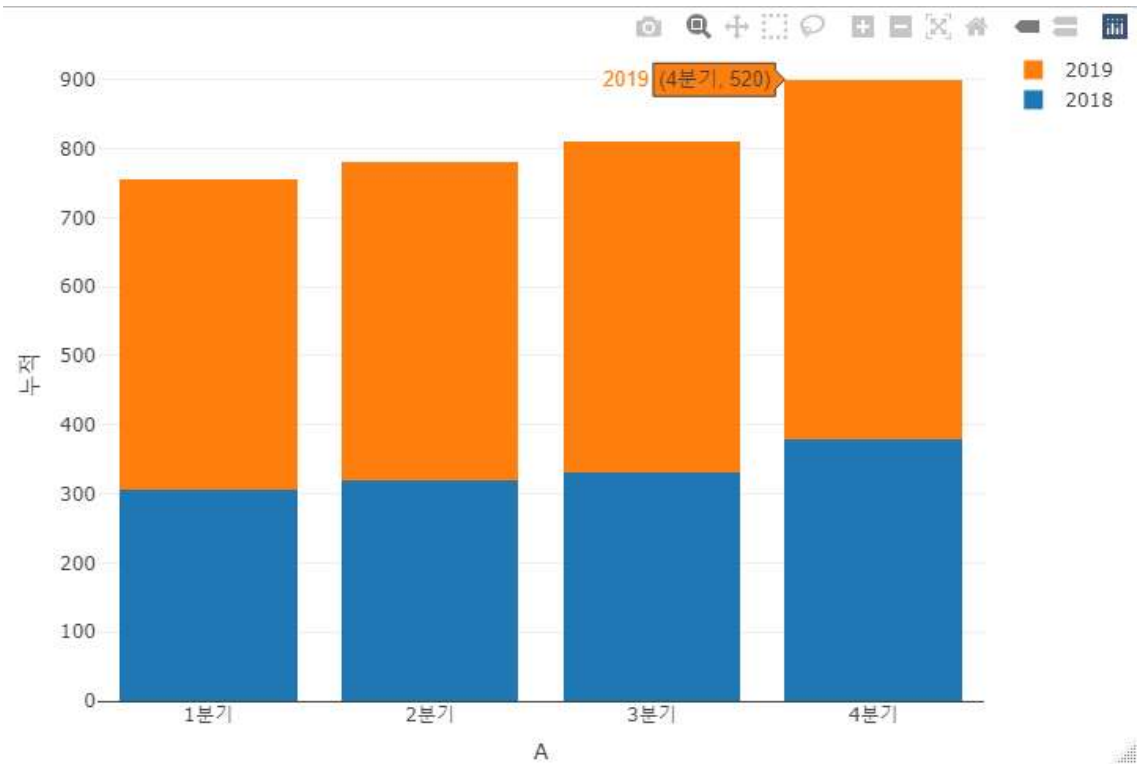
- ggplot2 패키지(R)



*사용된 methods

ggplot() + geom_bar(position = "stack")를 이용하면 누적그래프 출력이 가능하다.

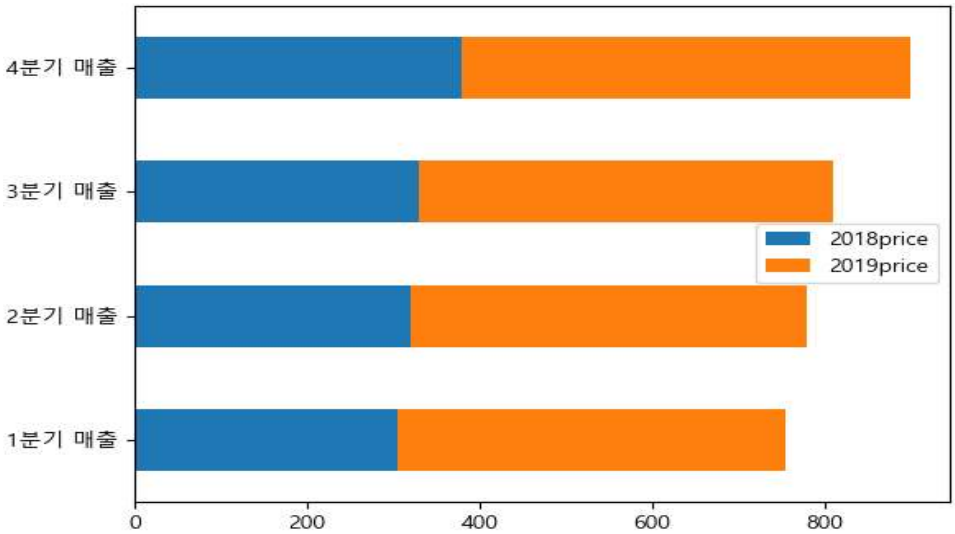
- plotly패키지(R)

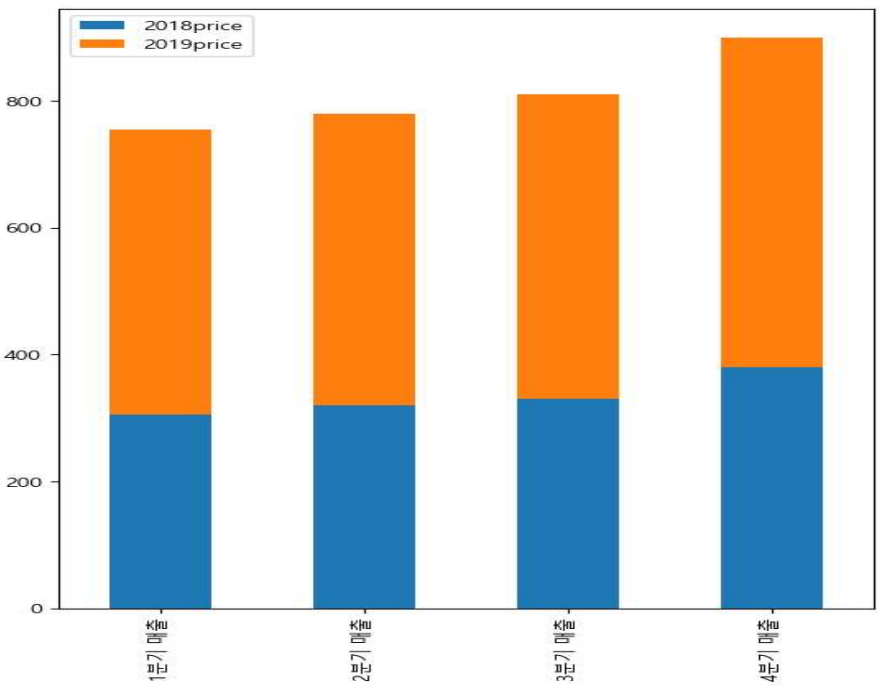


*사용된 methods

layout(barmode = "stack") 을이용해 누적그래프 출력이 가능하다.

- Matplotlib 패키지(Python)



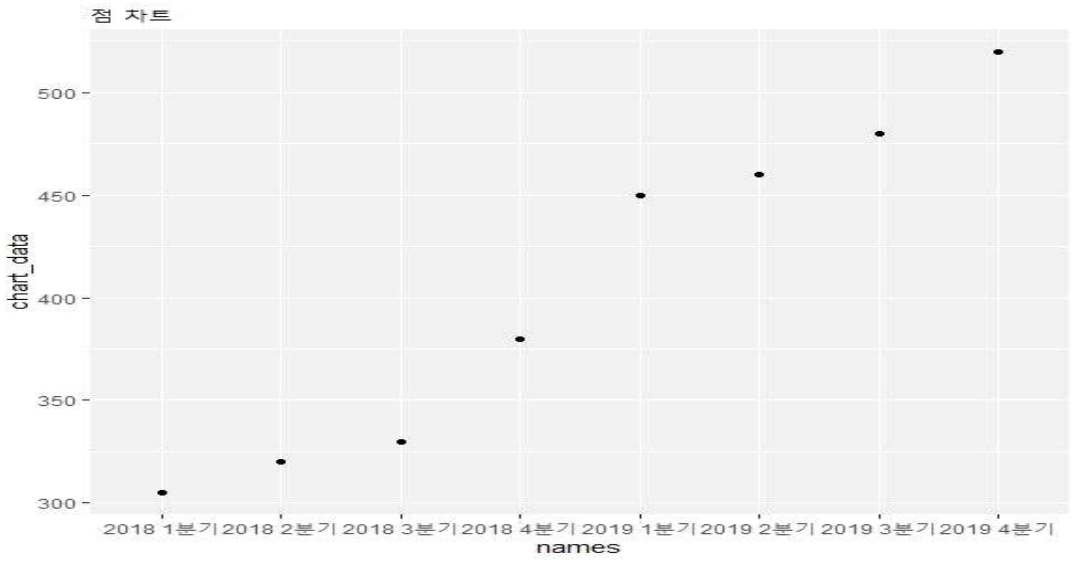


*사용된 methods

chart_data.plot(kind = 'bar' or 'barh', stacked = TRUE)와 같이 stacked methods를 이용하여 누적그래프 생성이 가능하다.

3) 점차트

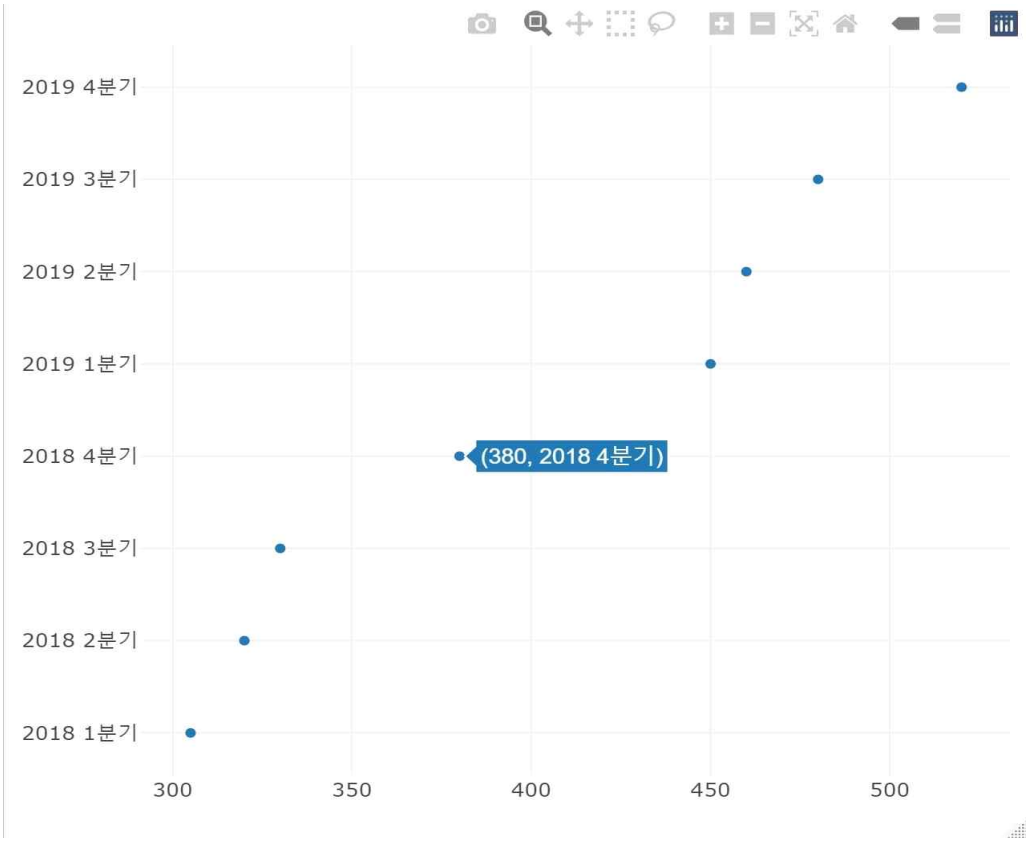
- ggplot2 패키지(R)



*사용된 methods

Ggplot() + geom.point()를 이용하여 점차트 생성이 가능하다.

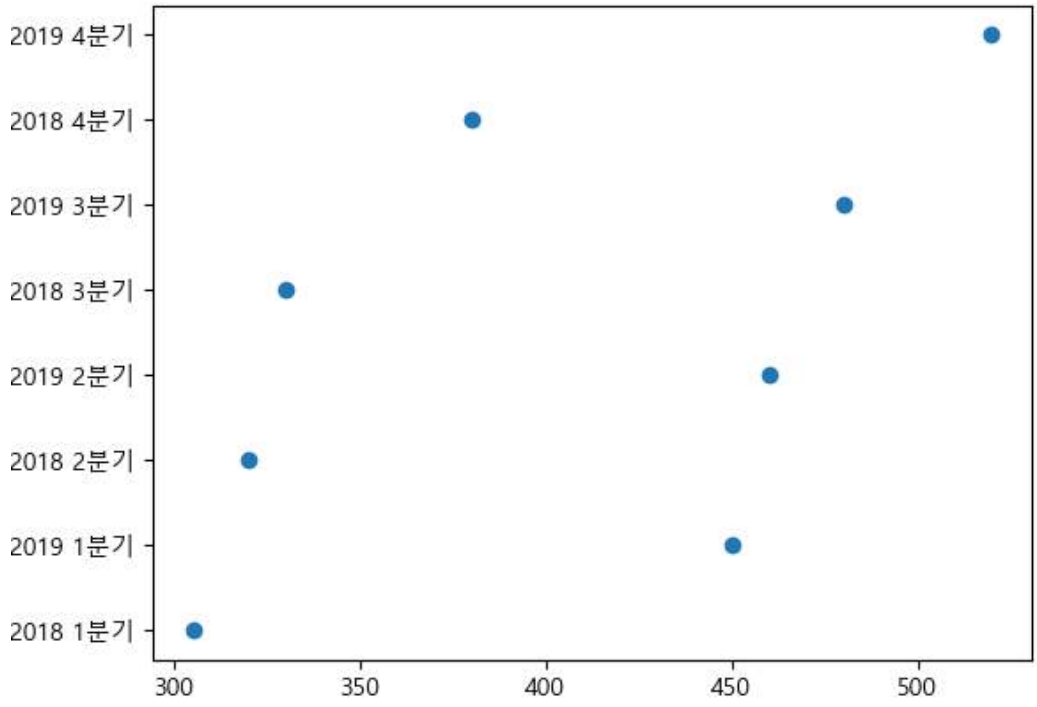
- plotly패키지(R)



*사용된 methods

Plot_ly(data,x,y)와같이 데이터와 x,y값입력시 점차트 생성이 가능하다.

- Matplotlib 패키지(Python)

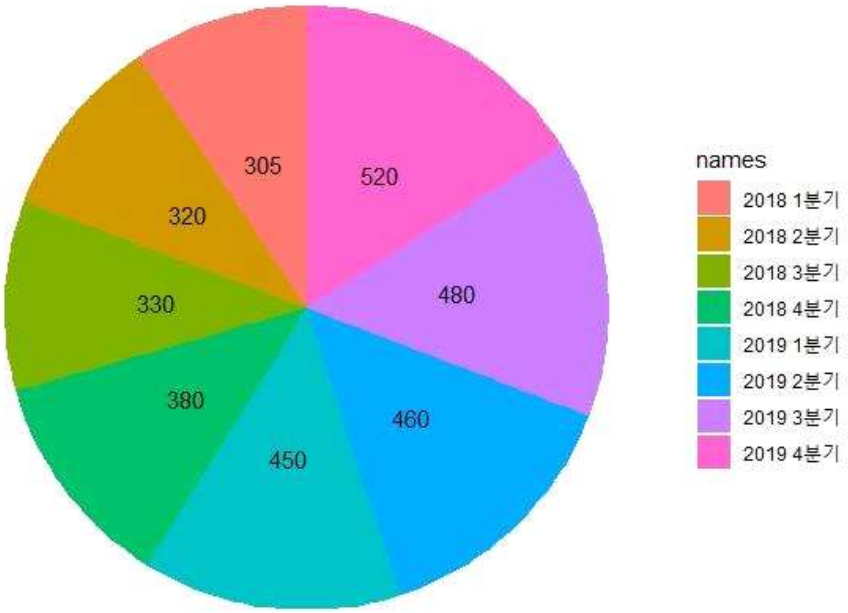


*사용된 methods

px.scatter()를 이용하여 점차트 생성이 가능하다.

4) 원차트

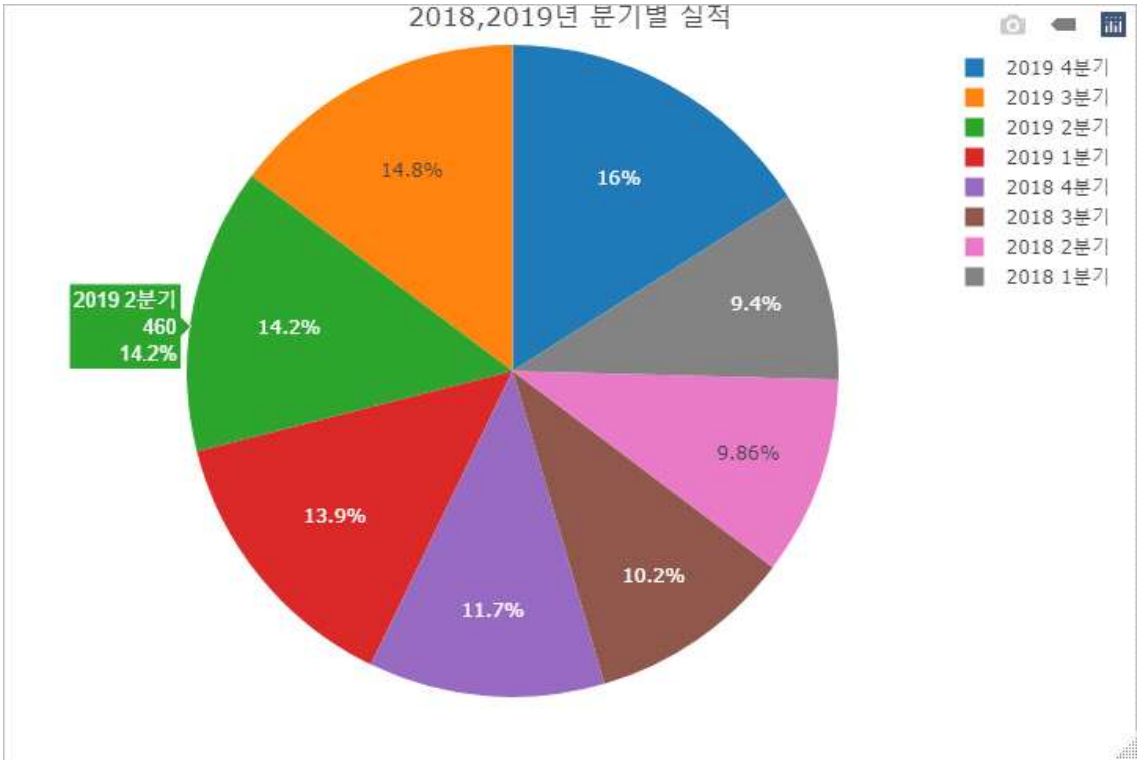
- ggplot2 패키지(R)



*사용된 methods

ggplot() + coord_polar()를 이용해 원차트 생성이가능하다.

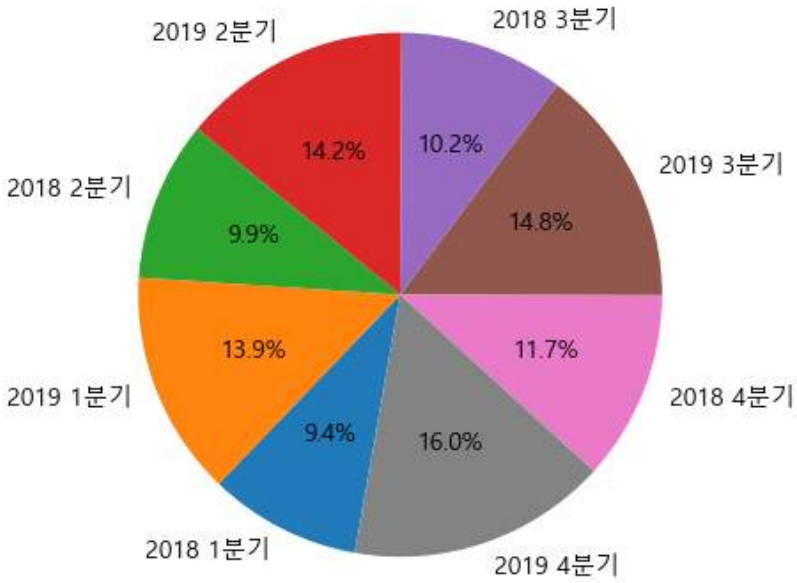
- plotly패키지(R)



*사용된 methods

plotly(data,labels = ~이름, values = ~값, type ='pie') 를 이용해 원차트 생성이가능하다.

- Matplotlib 패키지(Python)

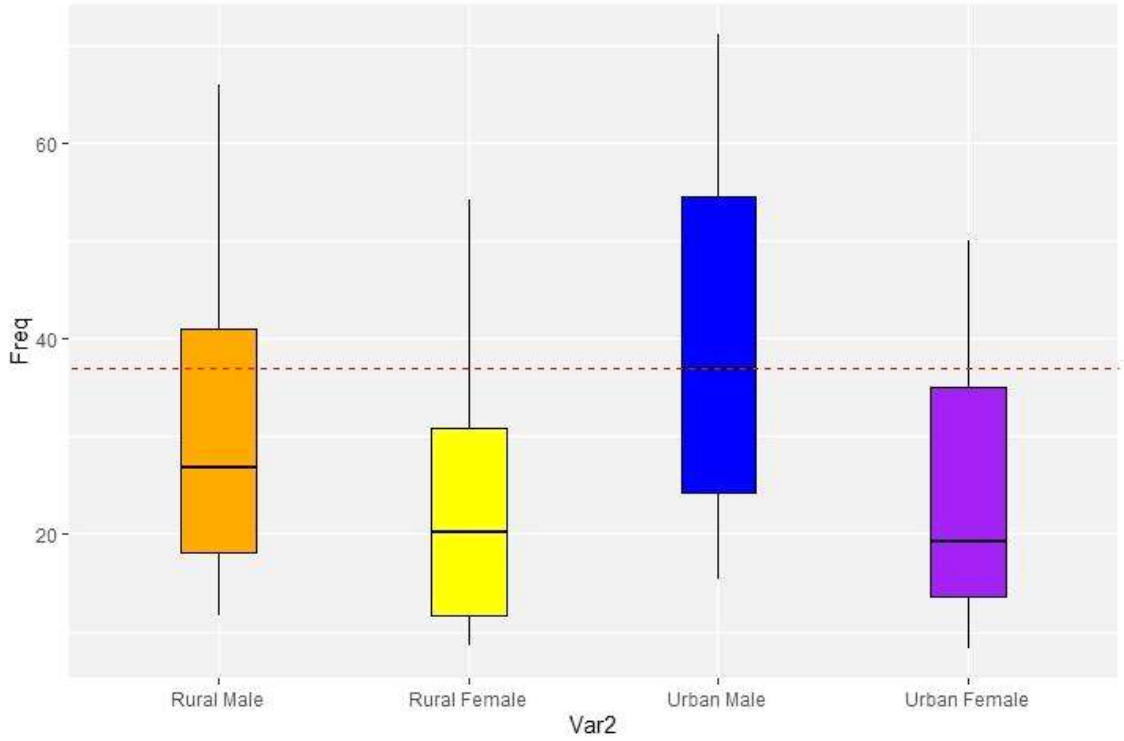


*사용된 methods

Plt.pie()를 이용해 원차트 생성이 가능하다. 사용가능 method로는 **"startangle = 그려지는 시작각도","counterclock = 반시계 방향으로 생성"** 등이 있다.

5) 상자 그래프

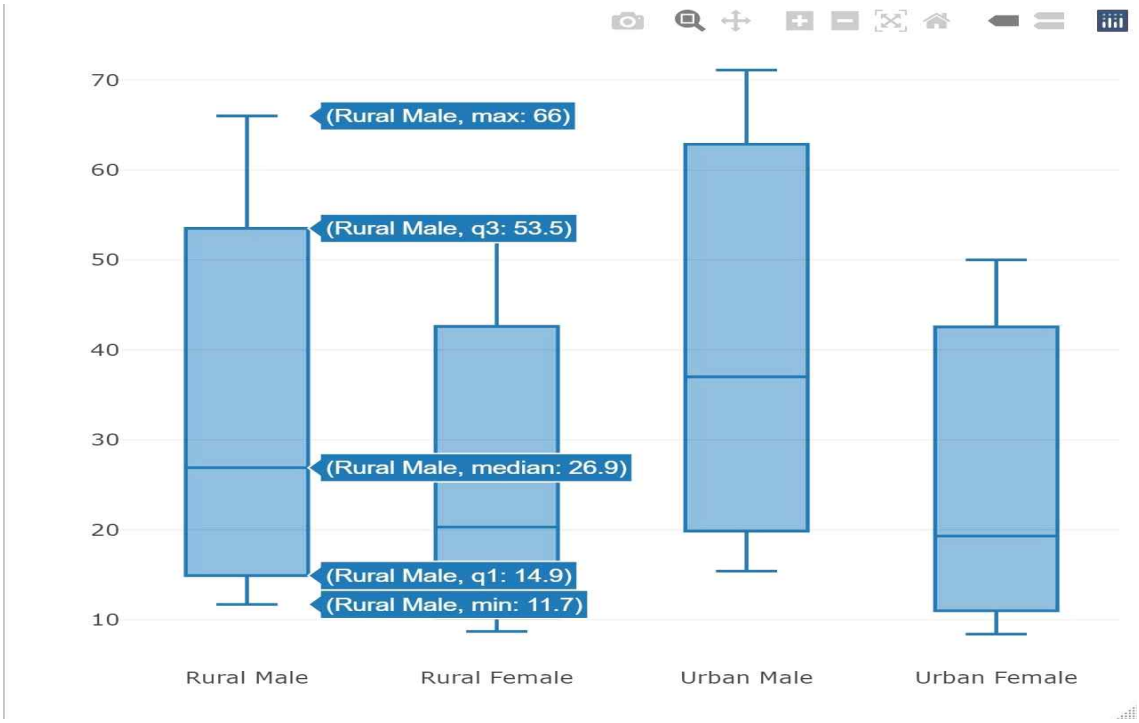
- ggplot2 패키지(R)



*사용된 methods

ggplot() + geom_boxplot()을 이용하여 상자그래프 생성이 가능하다.

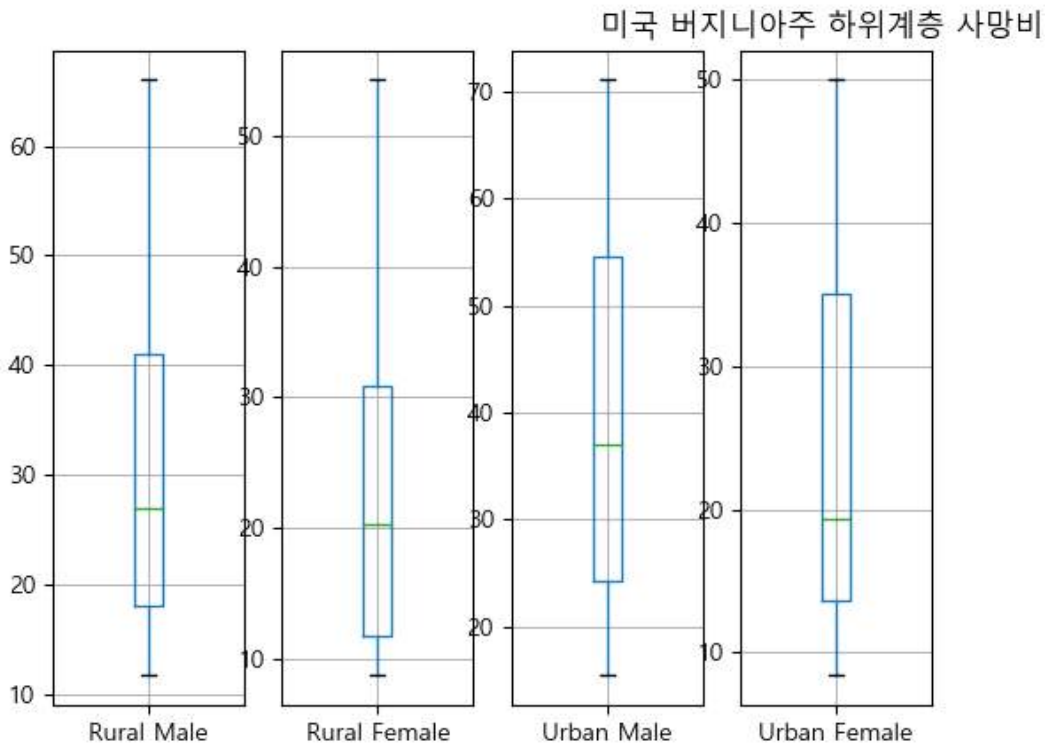
- plotly패키지(R)



*사용된 methods

plot_ly(y=data,ty type = "box")를 이용하여 상자그래프 생성이 가능하다

- Matplotlib 패키지(Python)

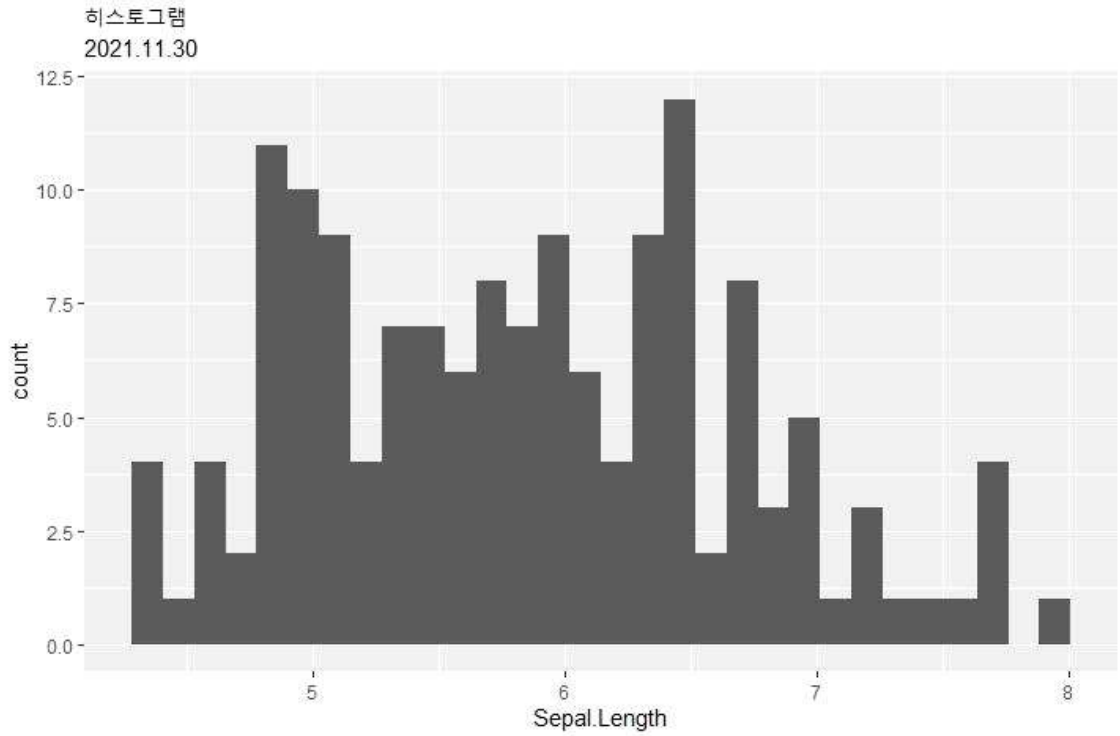


- 사용된 methods

data.botxplot()를 이용하여 상자 그래프 생성이 가능하다.

6) 히스토그램

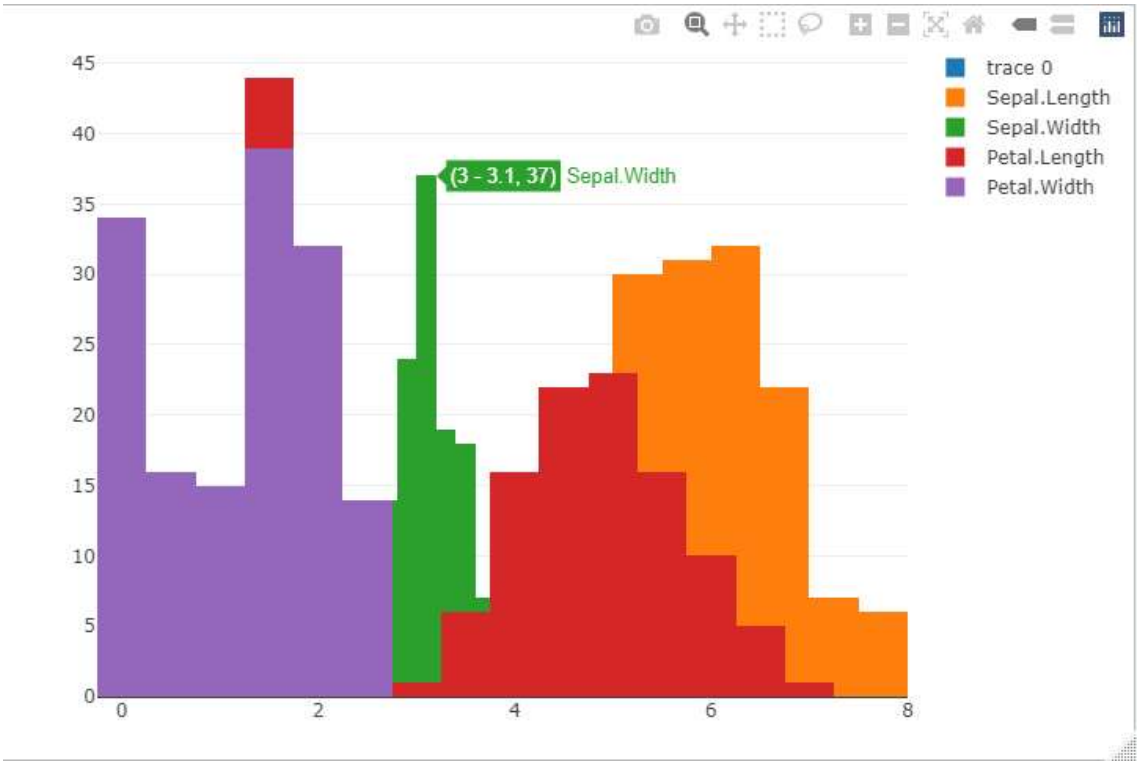
- ggplot2 패키지(R)



- 사용된 methods

Ggplot() + geom_histogram()를 이용하여 히스토그램 생성이 가능하다.

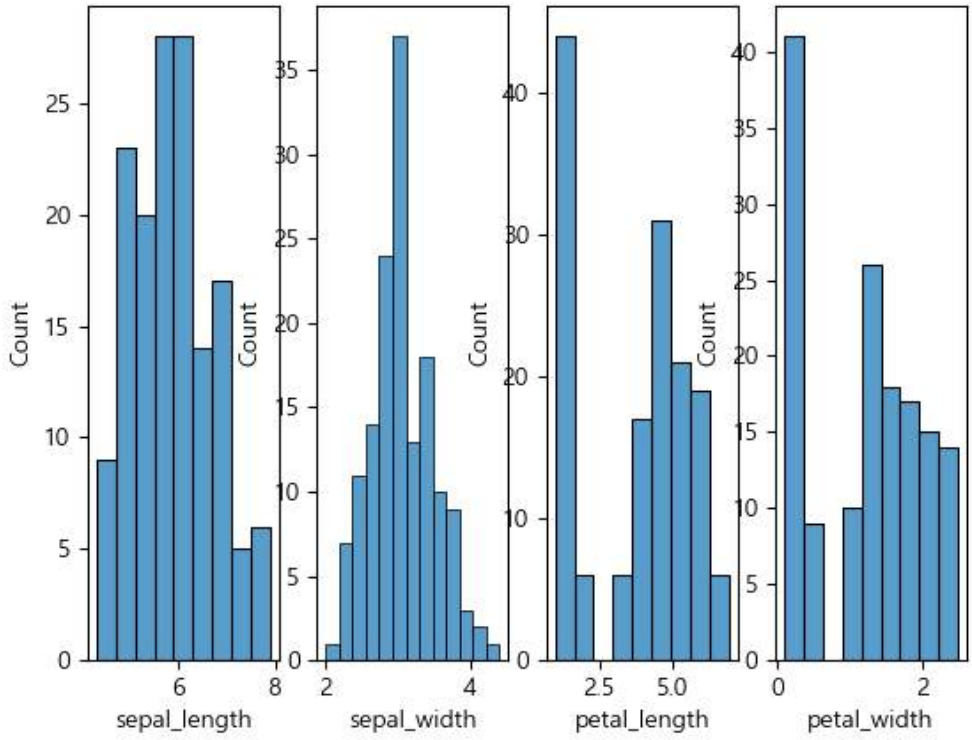
- plotly패키지(R)



- 사용된 methods

Plot_ly(type = "histogram")를 이용하여 히스토그램 생성이 가능하다.

- Matplotlib 패키지(Python)

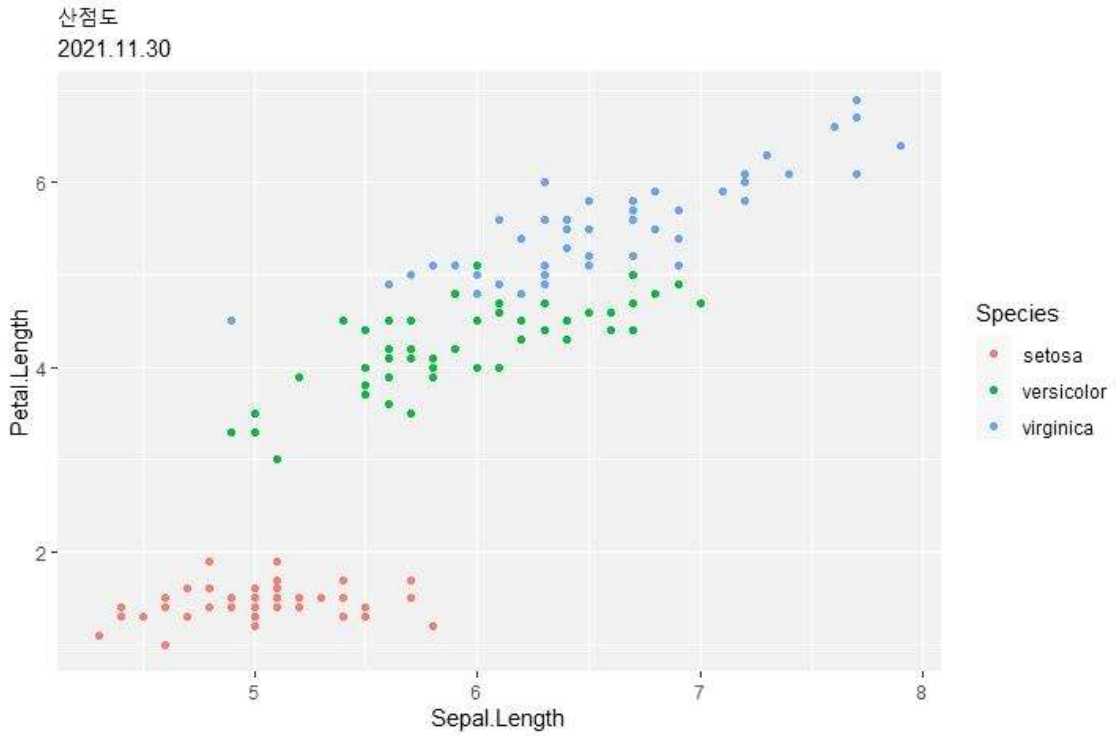


- 사용된 methods

data.histogram()를 이용하여 히스토그램 생성이 가능하다.

7) 산점도

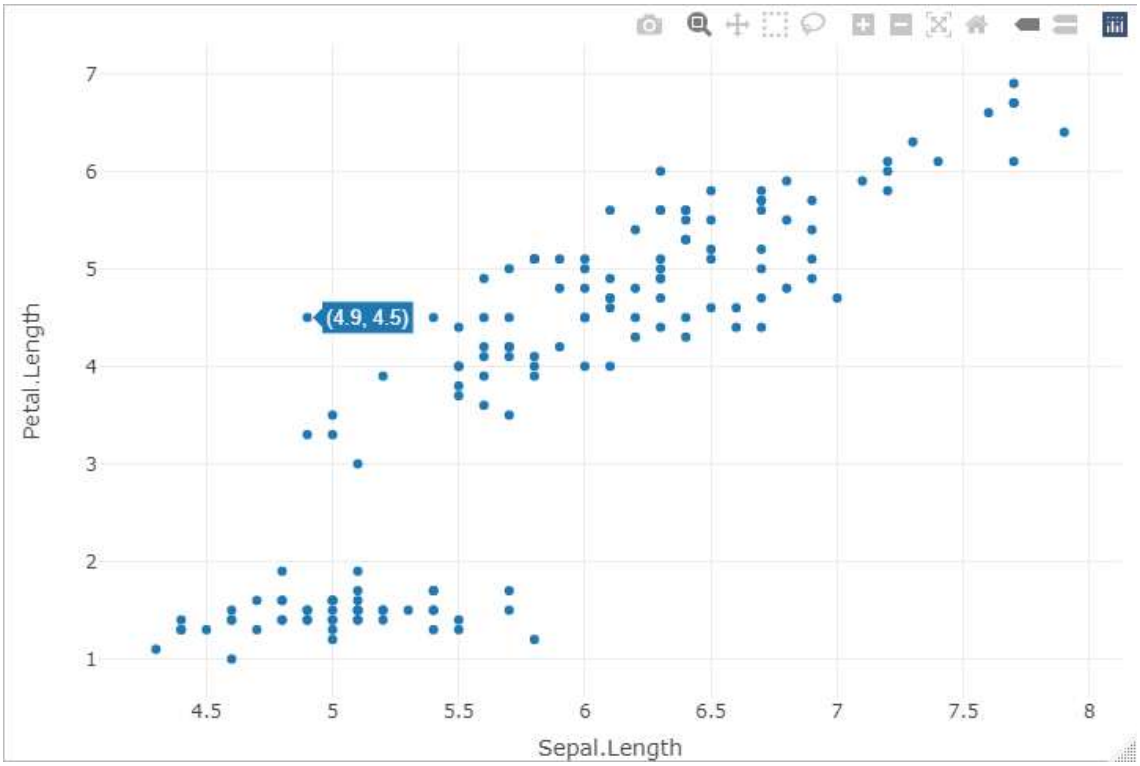
- ggplot2 패키지(R)



- 사용된 methods

Ggplot() + geom_point()를 이용하여 산점도 그래프 생성이 가능하다.

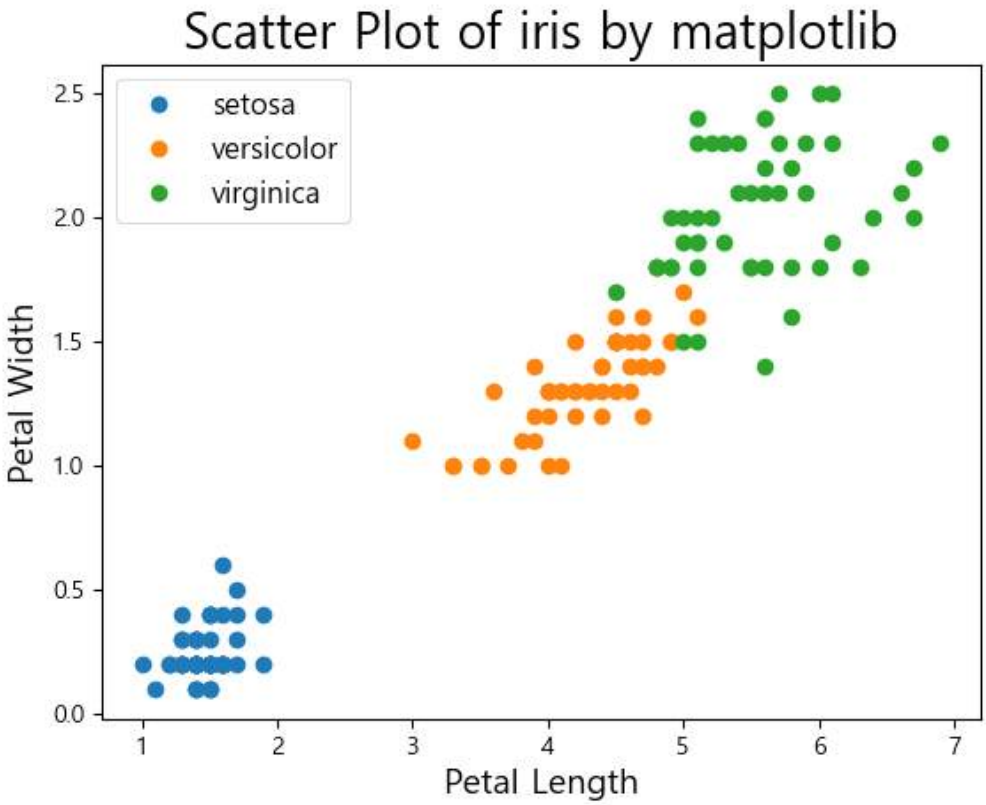
- plotly패키지(R)



- 사용된 methods

x,y값을 입력만하면 산점도 그래프 생성이 가능하다.

- Matplotlib 패키지(Python)

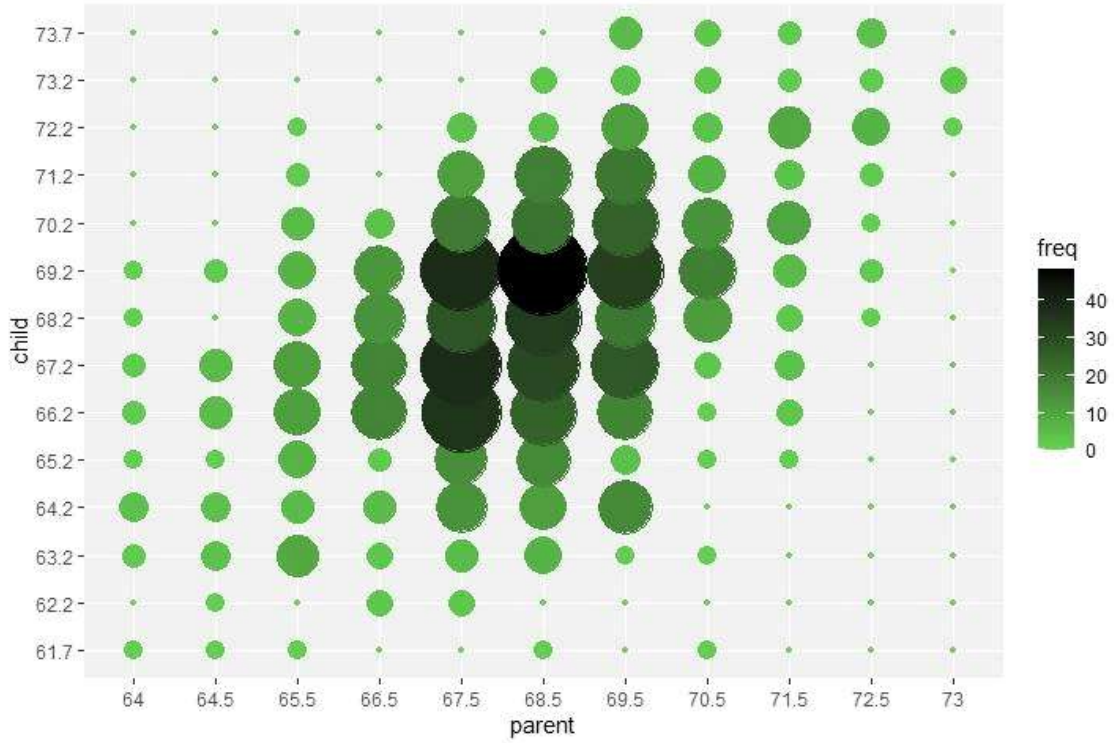


- 사용된 methods

data.plot(x,y)를 이용하여 산점도 그래프 생성이 가능하다.

8) 중첩자료

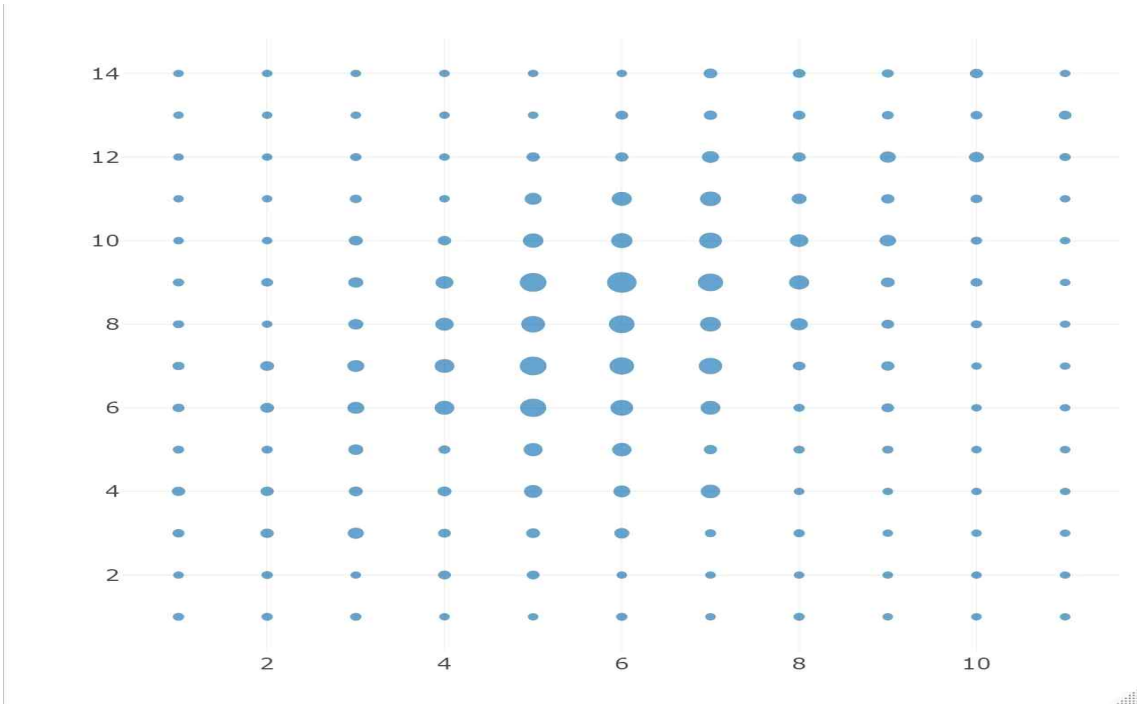
- ggplot2 패키지(R)



- 사용된 methods

Ggplot() + geom_point(ase(size = freq))를 이용하여 중복된 자료의 많을수록 표시가 커지는 그래프 생성이 가능하다.

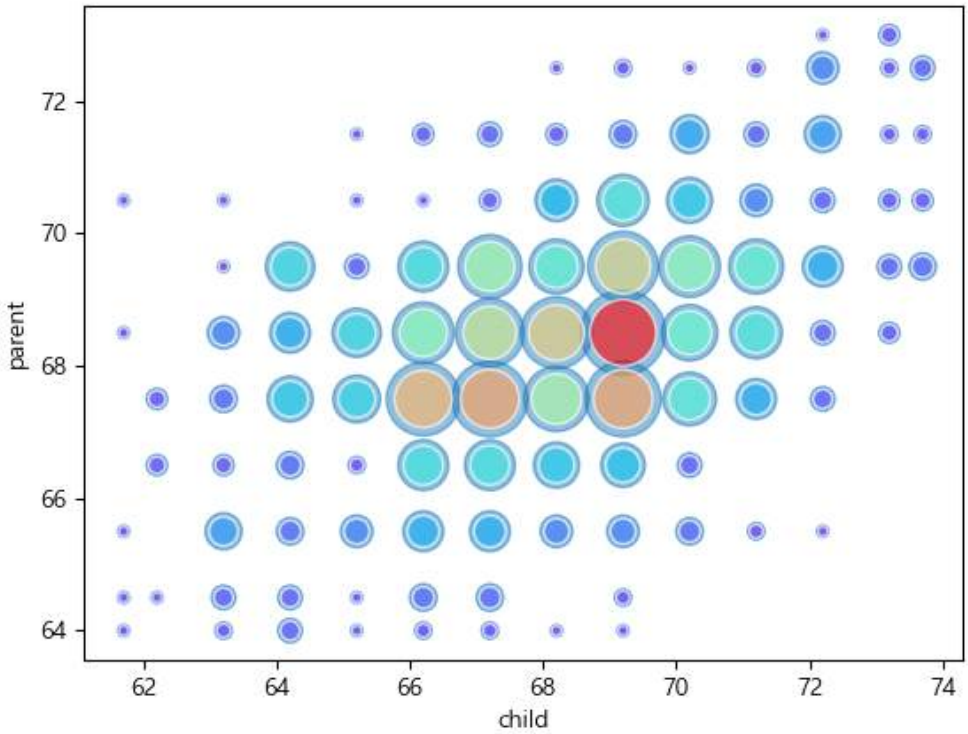
- plotly패키지(R)



- 사용된 methods

Plot_ly(data,x,y,,size = freq) 를 이용하여 중복된 자료의 많을수록 표시가 커지는 그래프 생성이 가능하다.

- Matplotlib 패키지(Python)

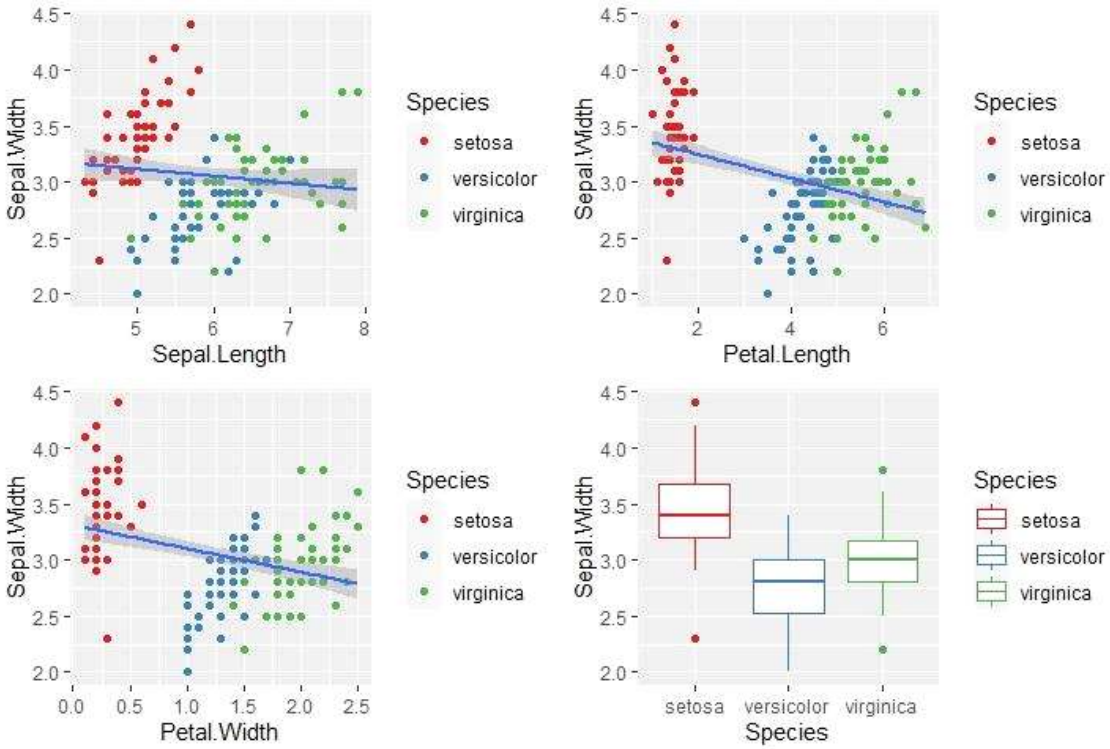


- 사용된 methods

table함수를 이용하여 빈도수를 확인후 **plot.scatter(x,y,s=table*25,alpha=0.5)**를 이용하여 중복된 자료의 많을수록 표시가 커지는 그래프 생성이 가능하다. **"s = 사이즈","alpha=투명도"**의 조정이 가능한 method이다.

9) 변수간의 비교 시각화

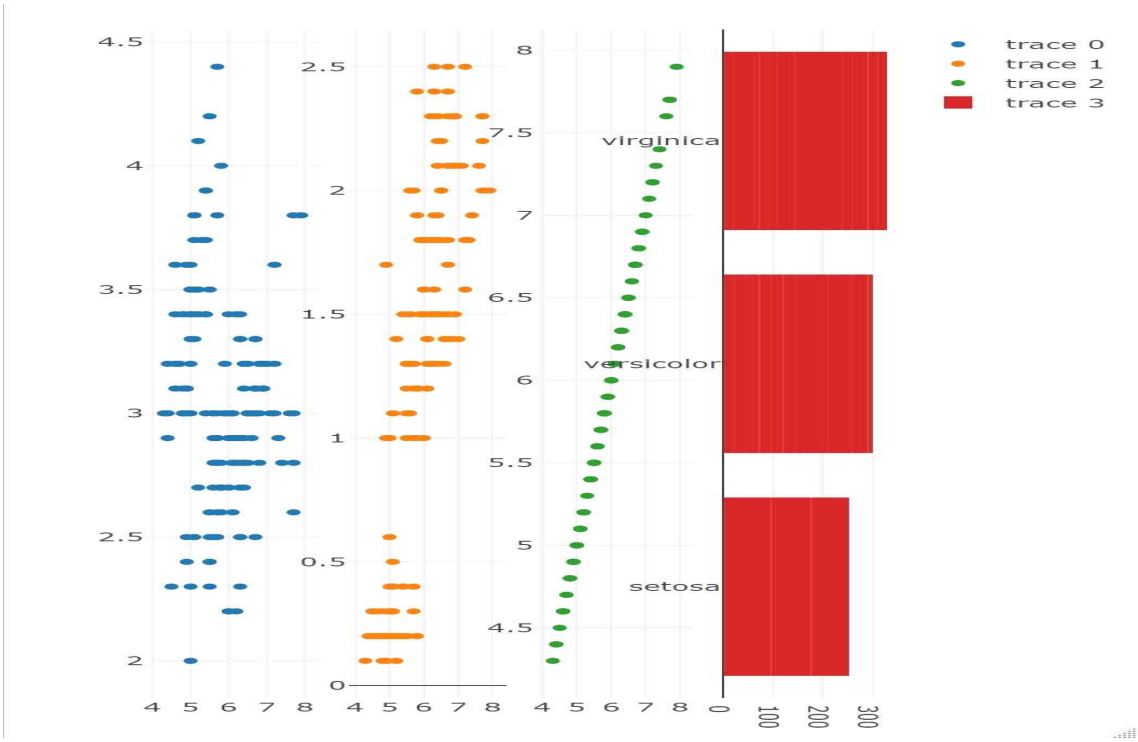
- ggplot2 패키지(R)



- 사용된 methods

Ggplot2 패키지를 사용하여 변수간 비교를 할 수 있다.

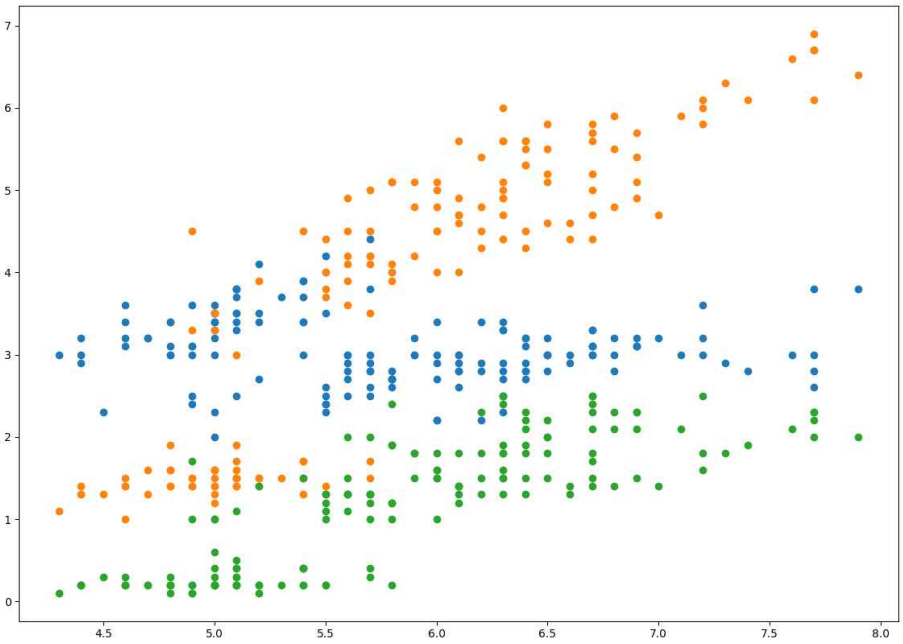
- plotly패키지(R)



- 사용된 methods

subplot(fig,fig1,fig2,fig3)을 이용하면 한번에 비교할 수 있다.

- Matplotlib 패키지(Python)

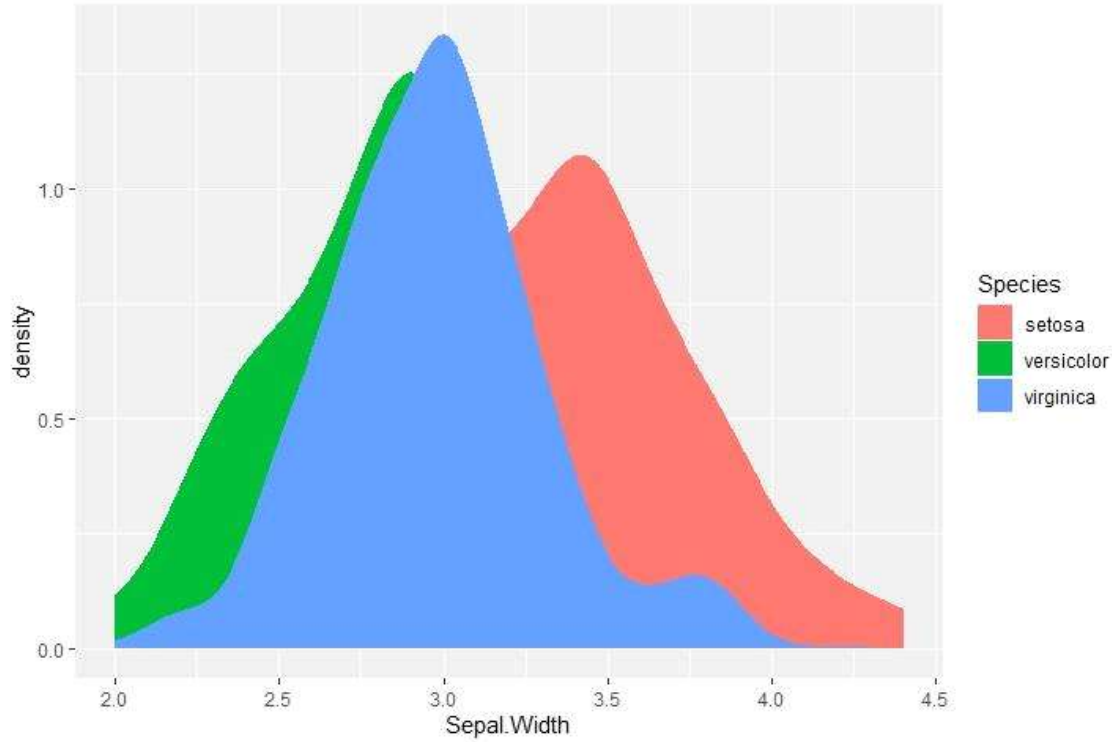


- 사용된 methods

plt.scatter(iris.data[:,0],iris.data[:,1]) 변수 지정 후 비교 할 수 있다.

10) 밀도그래프

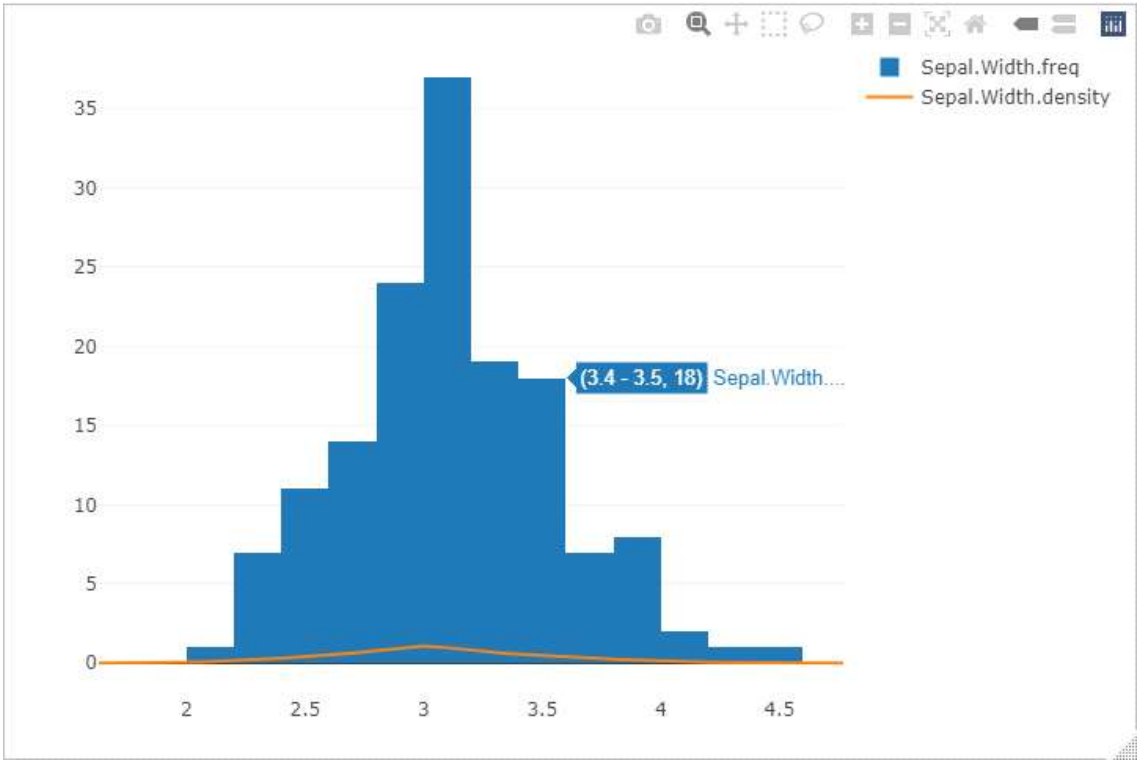
- ggplot2 패키지(R)



- 사용된 methods

Ggplot() + geom_density()를 이용하여 밀도 그래프를 생성할수있다.

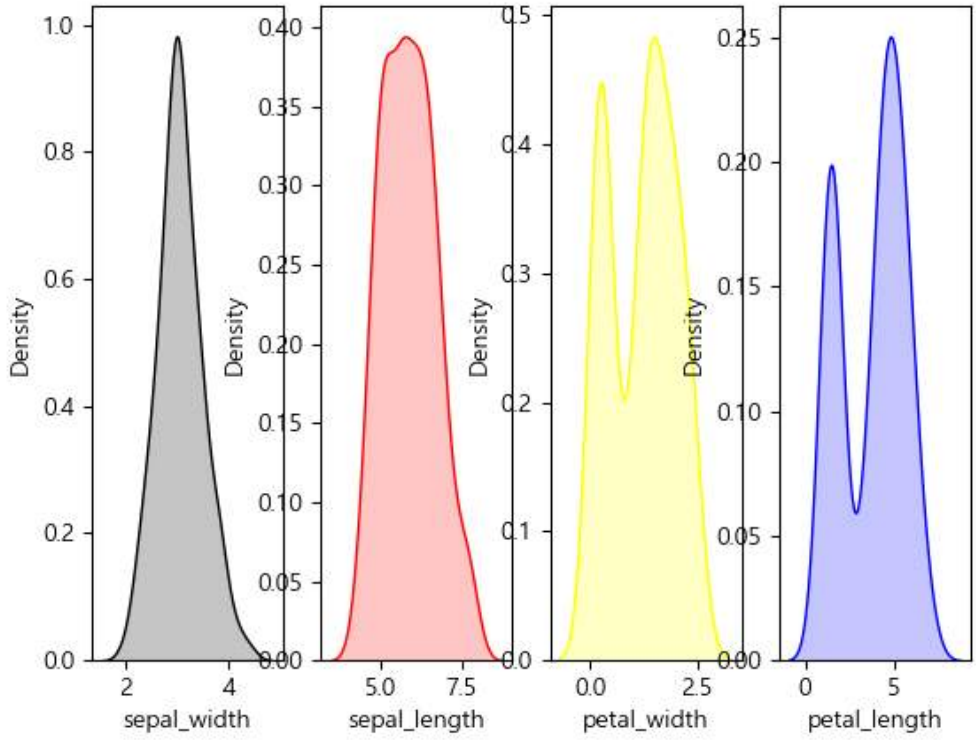
- plotly패키지(R)



- 사용된 methods

density(iris)를 이용하여 밀도를 구한뒤, **plot_ly(data, type="histogram")**를 이용하여 히스토그램을 만든뒤 **add_lines(density$x,density$y)**를 이용하여 밀도선을 추가할수 있다.

- Matplotlib 패키지(Python)



- 사용된 methods

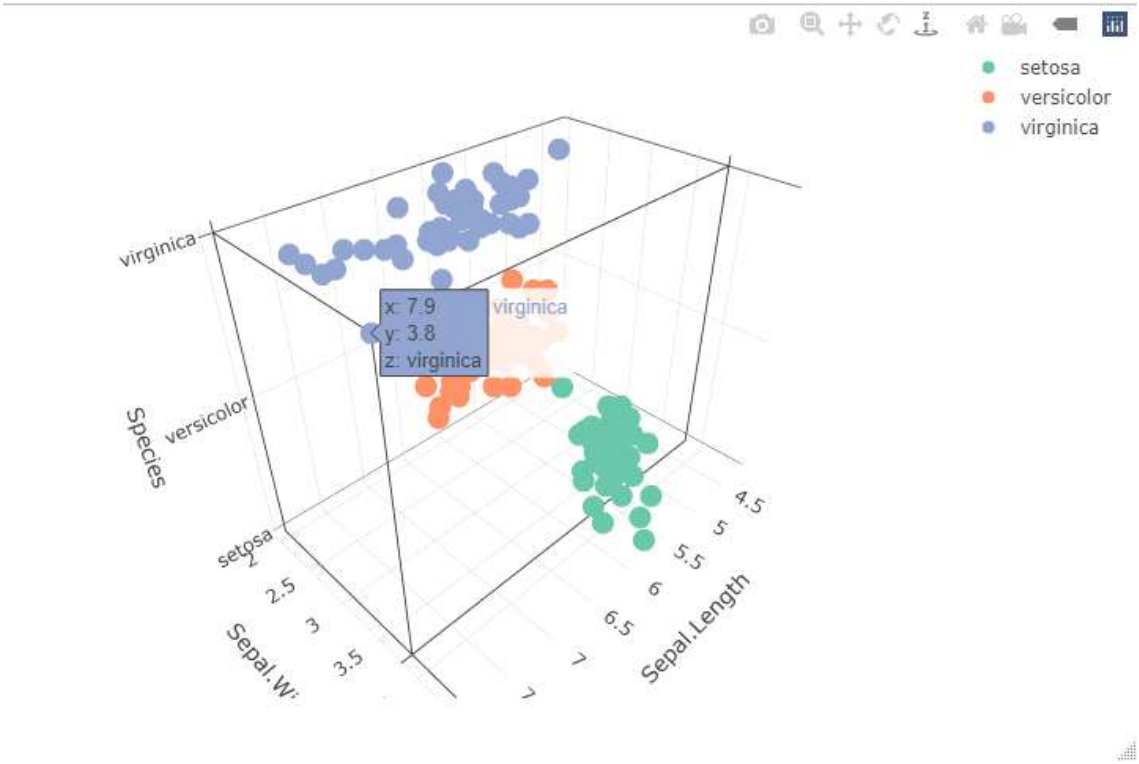
data.kdeplot()를 이용하여 밀도 그래프를 그릴 수 있다.

11) 3차원 그래프

- ggplot2 패키지(R)
- 사용된 methods

ggplot2 패키지에서는 자체적으로 3차원 그래프를 지원하지 않는다.

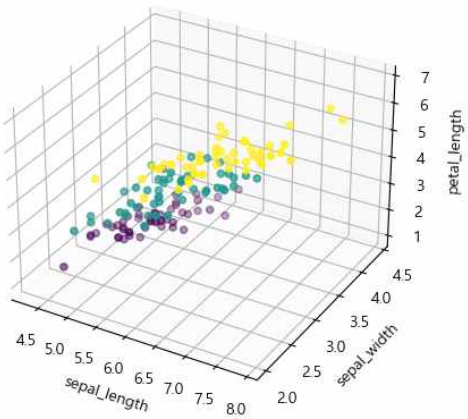
- plotly패키지(R)



- 사용된 methods

Plot_ly(x,y,z)를 이용하여 3차원 그래프를 그릴 수 있다.

- Matplotlib 패키지(Python)

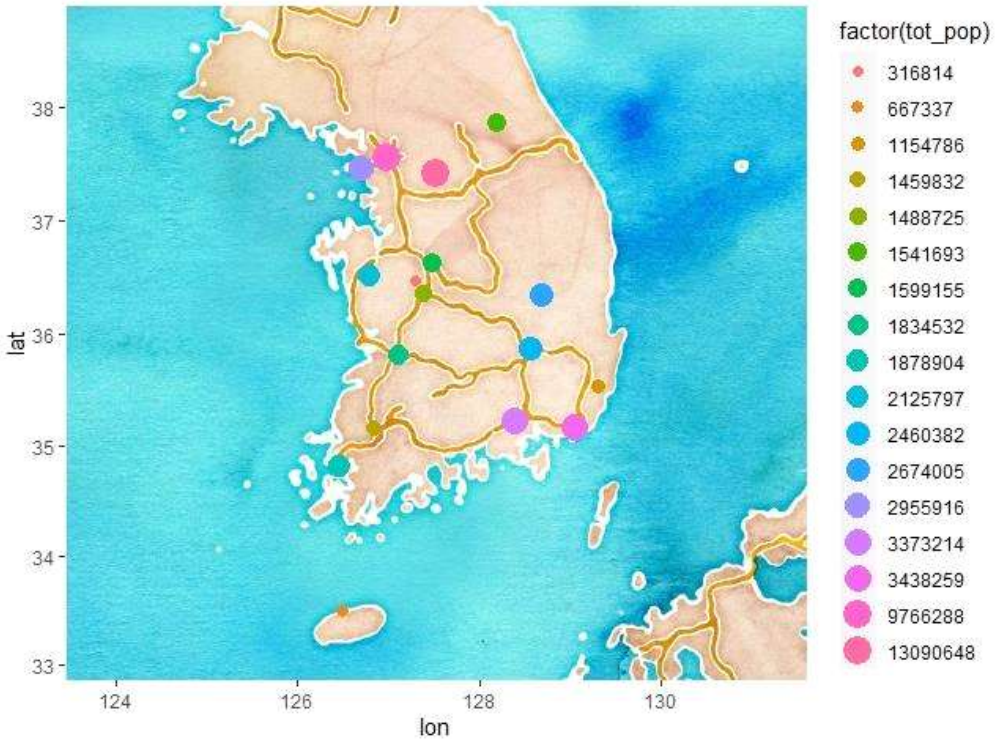


- 사용된 methods

data.scatter(x,y,z)를 이용하여 3차원 그래프를 그릴 수 있다.

12) 지도 시각화

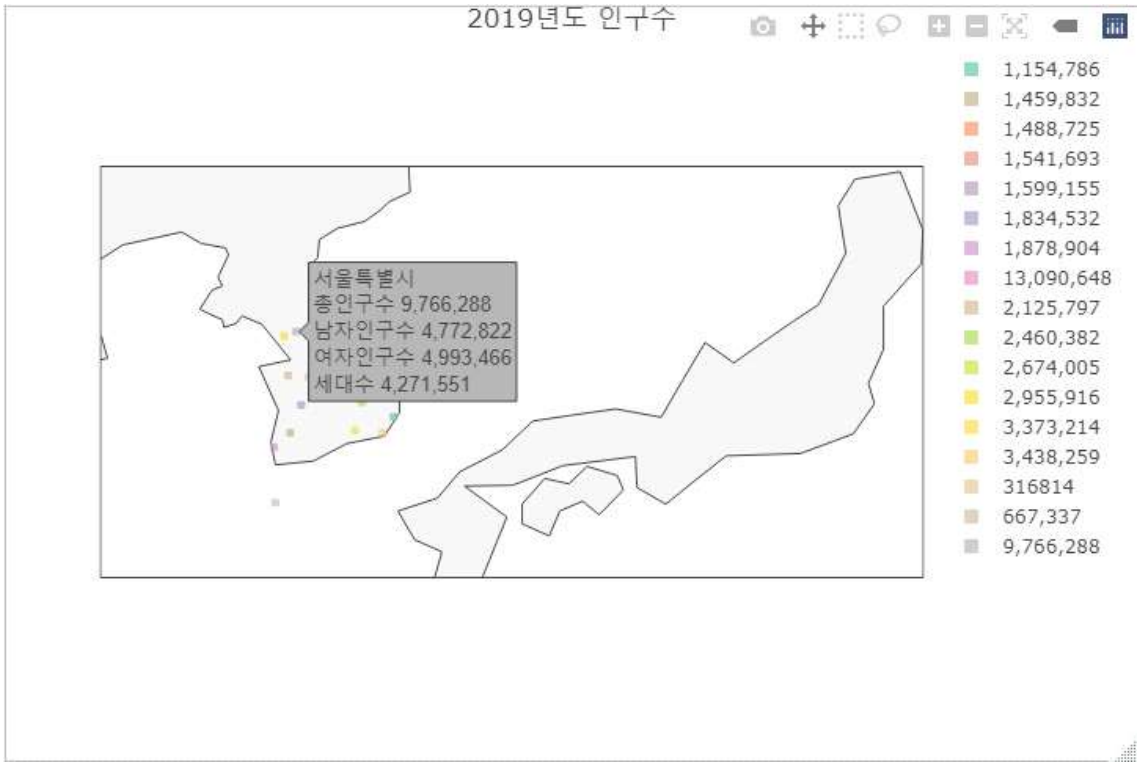
- ggplot2 패키지(R)



- 사용된 methods

ggmap을 이용하여 지도를 가져온뒤 **geom_point()**를 이용하여 위도 경도에 점을 찍고 **geom_text()**를 이용하여 위도경도에 텍스트를 삽입 할 수 있다.

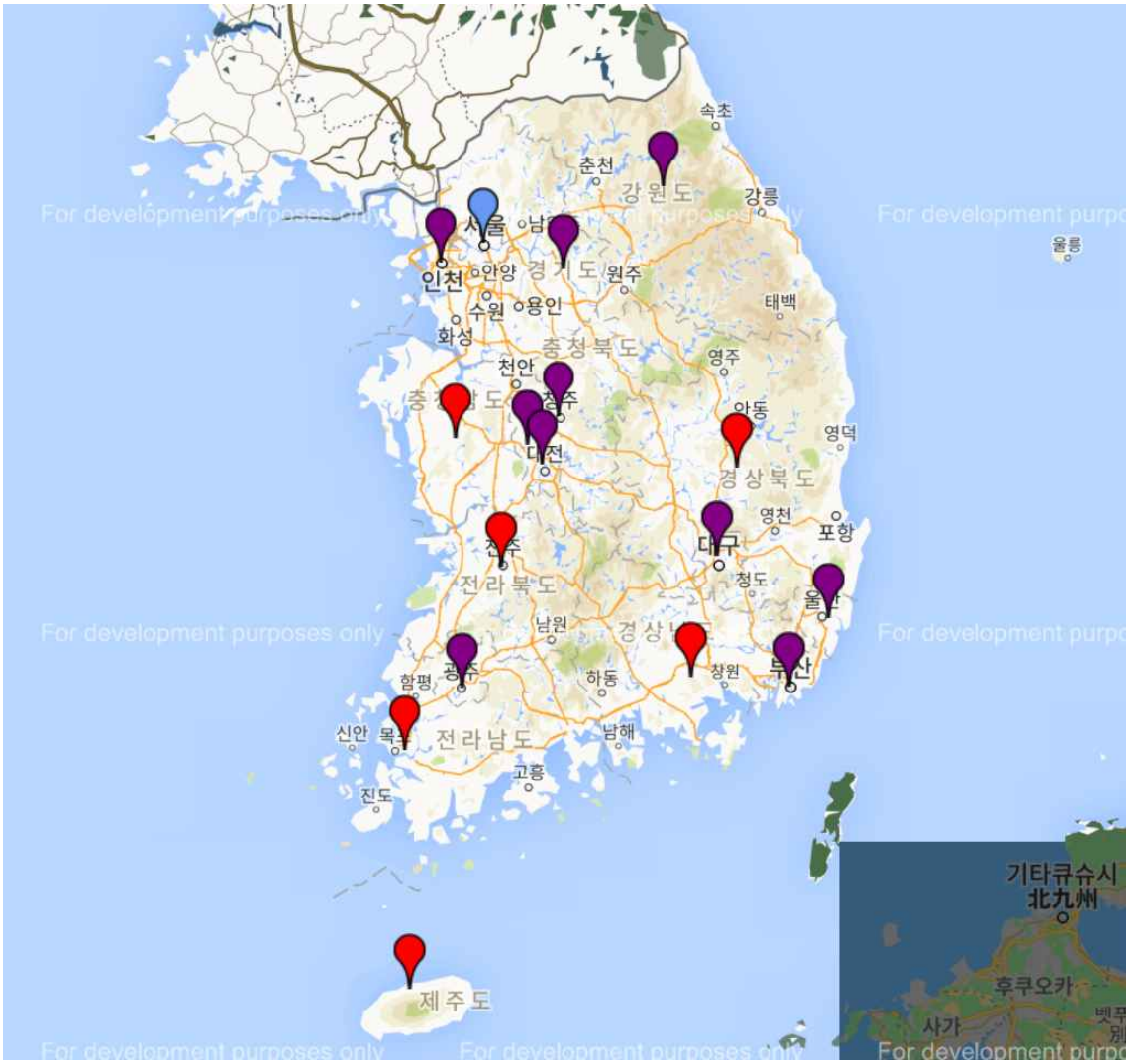
- plotly패키지(R)



- 사용된 methods

Plot_geo(data,lat,lon)을 이용하여 지도 데이터를 가져오고 text를 추가할수 있다.

- Matplotlib 패키지(Python)



- 사용된 methods

gmap을 이용하여 지도를 가져온뒤 **gmap.Plolygon()**를 이용하여 위도 경도를 추가한다.

Ⅲ. 결론

R과 Python의 패키지인 ggplot2,plotly,matplotlib에 같은 데이터를 입력하여 시각화 결과를 비교 했다. ggplot2의 경우 +를 이용하여 점과 선, 텍스트를 편하게 추가 할 수 있는 장점이 있었으며, plotly는 반응형 그래프를 그릴 수 있는 장점이 있었다. matplotlib의 경우 R에서 이용하는 패키지에 비해 다소 복잡하고 불편하다고 생각이 되었다. 하지만 파이썬에서 구동할 수 있다는 장점이 있다.

첨부자료

사례연구6 사용 R, Python코드.zip