데이터 분석을 위한

Matplotlib

류영표 강사
youngpyoryu@dongguk.edu

강사 소개

류영표
Youngpyo Ryu

동국대학교 수학과/응용수학 석사수료
現 Upstage AI 부스트캠프 멘토
現 메가 IT아카데미(파이썬,빅데이터) 강사
한국파스퇴르연구소 Image Mining 인턴(Deep learing)
前 ㈜셈웨어(수학컨텐츠, 데이터 분석 개발 및 연구인턴)

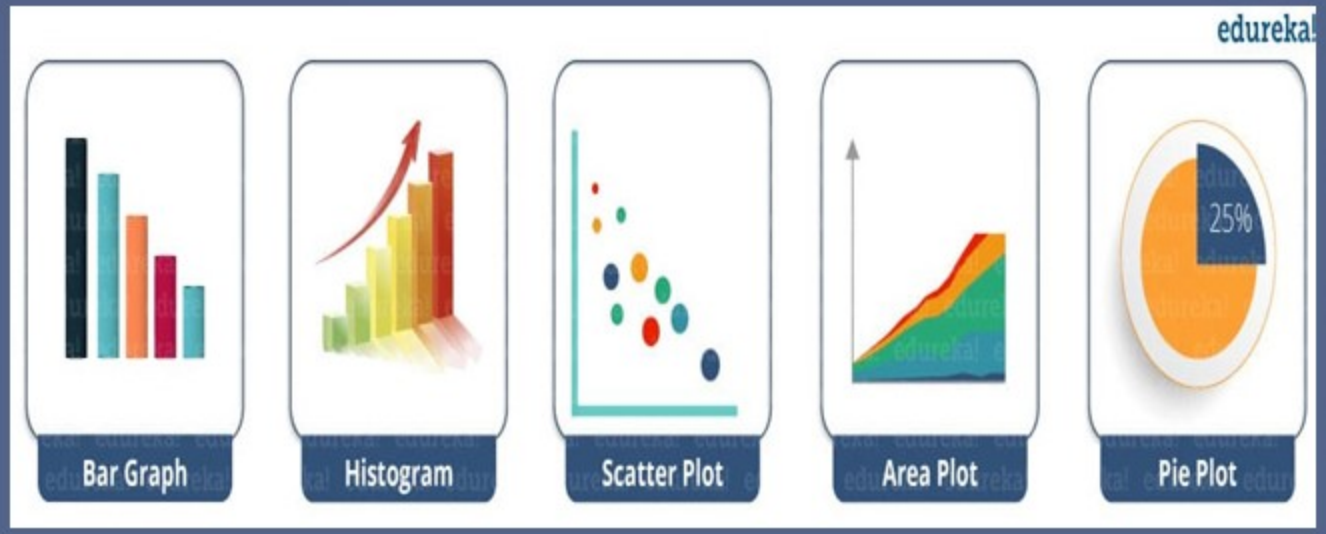
강의 경력

- 현대자동차 연구원 강의 (인공지능/머신러닝/딥러닝/강화학습)
- 딥러닝 집중 교육과정 강사
- (재)월튼블록체인 6일 과정 (파이썬기초, 크롤링,머신러닝)
- 서울특별시 X AI 양재허브 X 모두의연구소 (중급 NLP과정) 보조강사
- SK아카데미_HLP(임원) 1차/2차 보조강사
- ㈜ 모두의연구소 Aiffel 1기 퍼실리테이터(인공지능 교육)
- LG전자 / LG 인화원 보조강사
- 인공지능 자연어처리(NLP) 기업데이터 분석 전문가 양성과정 멘토
- 고려대학교 선도대학 소속 30명 딥러닝 집중 강의

주요 프로젝트 및 기타사항

- 제1회 인공지능(AI)기반 데이터사이언티스트 전문가 양성과정 최우수상 수상(Q&A 챗봇)
- 인공지능(AI)기반 데이터사이언티스트 전문가 양성과정 1기 수료
- 제 1회 산업 수학 스터디 그룹 (질병에 영향을 미치는 유전자 정보 분석)
- 제 4,5회 산업 수학 스터디 그룹 (피부암, 유방암 분류)
- 빅데이터 여름학교 참석 (혼잡도를 최소화하는 새로운 노선 건설 위치의 최적화 문제)

INDEX



출처 : https://www.edureka.co/blog/python-matplotlib-tutorial/

Matplotlib

- Matplotlib이란?
  - 가장 대표적인 시각화 패키지. 그러나 다소 복잡
  - seaborn, plotnine, Plotly 등 더 다양한 기능을 지원하는 패키지가 많음

- 시작하기 앞서

```
import matplotlib.pyplot as plt
```

→ 네이밍 컨벤션

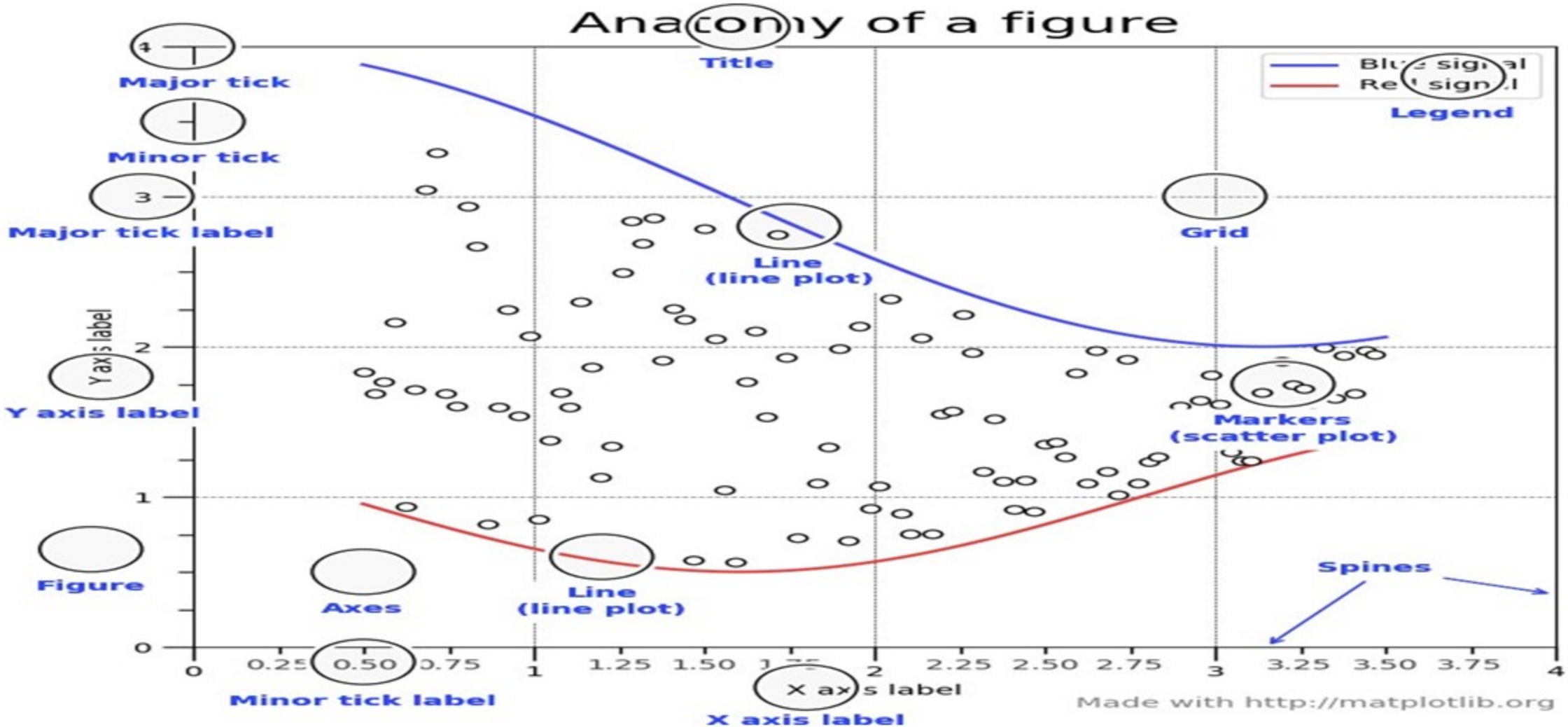
```
[ ]  %matplotlib notebook
```

→ 쥬피터 노트북 사용시 대화형 시각화 기능을 사용하기 위해선 코드를 반드시 실행

```
%matplotlib qt
```

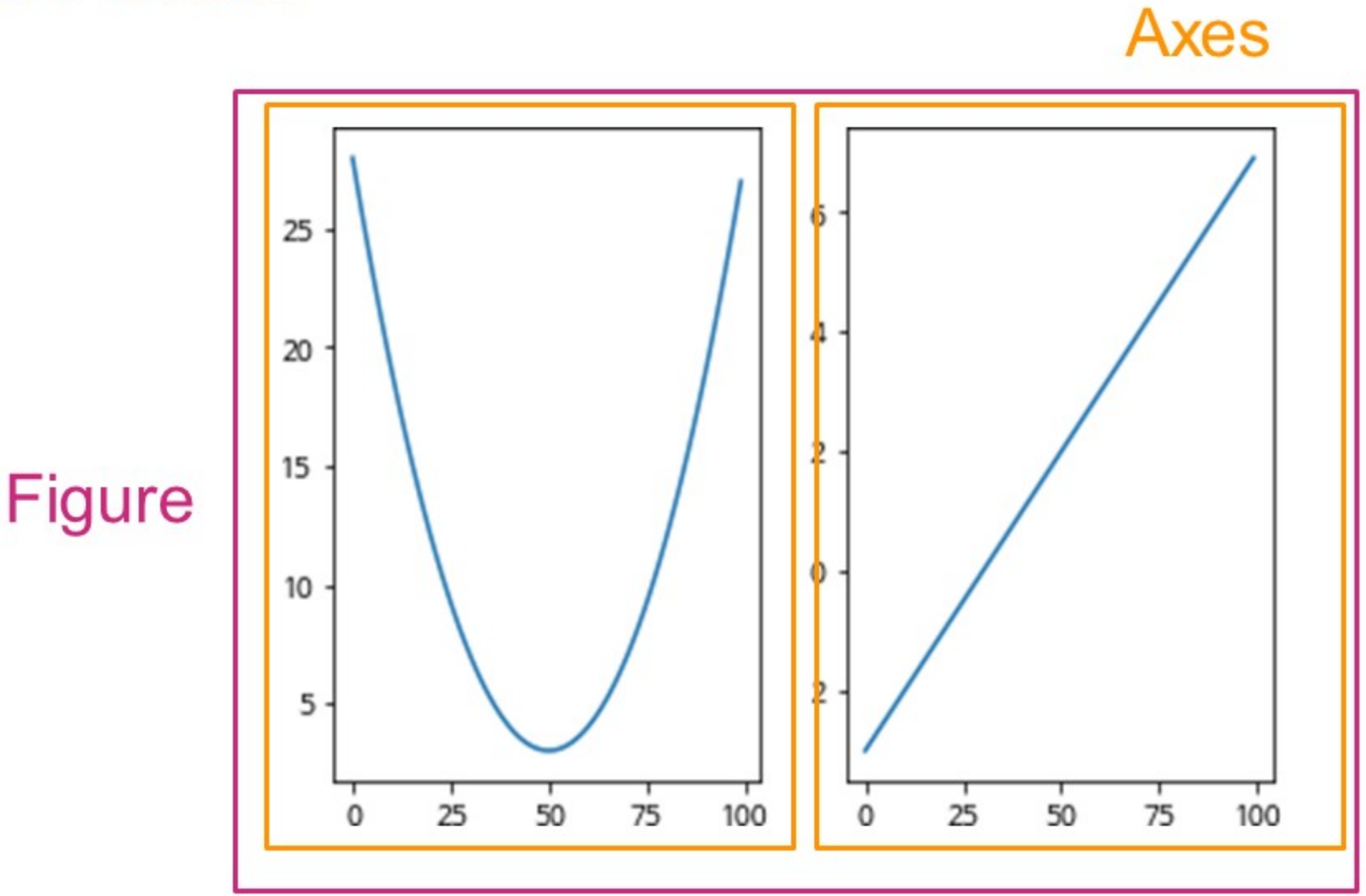
-> 별도의 팝업창에 그래프 출력(colab X)

Matplotlib



https://matplotlib.org/3.1.3/gallery/showcase/anatomy.html

Figure & Axes

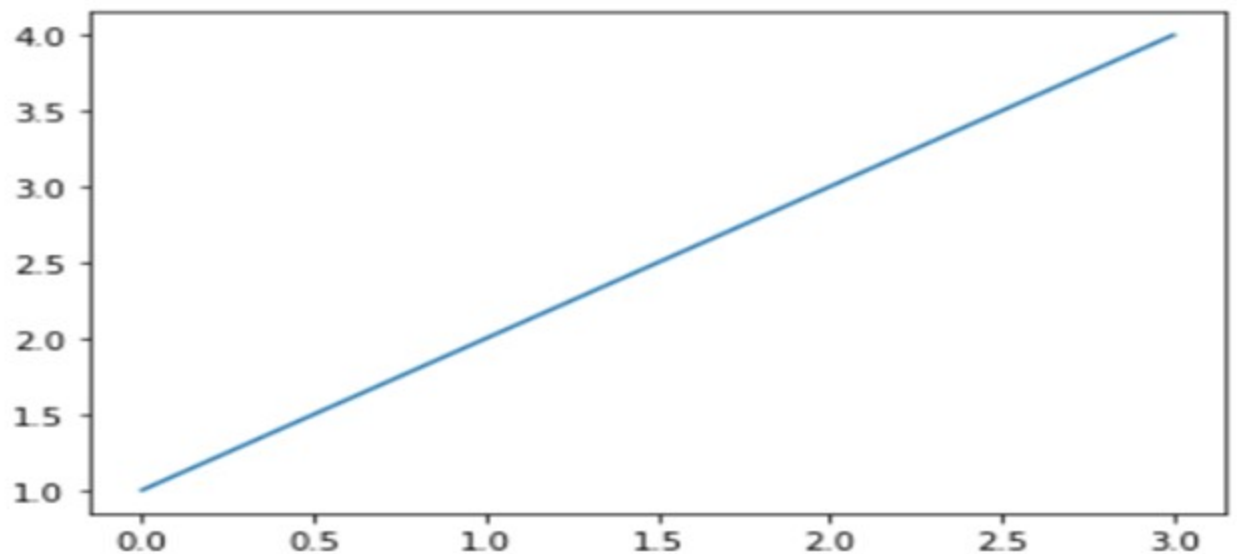


그래프 그리기

```
plot([x], y, [fmt], *, data=None, **kwargs)
plot([x], y, [fmt], [x2], y2, [fmt2], ..., **kwargs)
```

[x],[fmt]은 생략가능

```
import matplotlib.pyplot as plt

plt.plot([1,2,3,4])
plt.show()
```

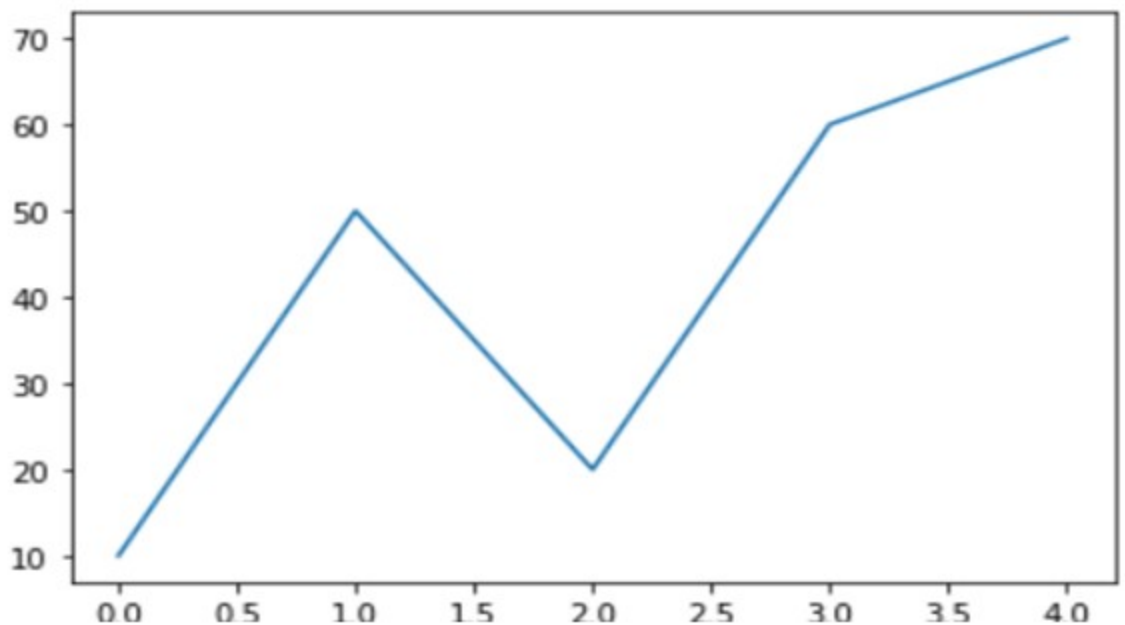


선 그래프 그리기

```
import matplotlib.pyplot as plt

data = [10,50,20,60,70]

%matplotlib inline

plt.plot(data)
```

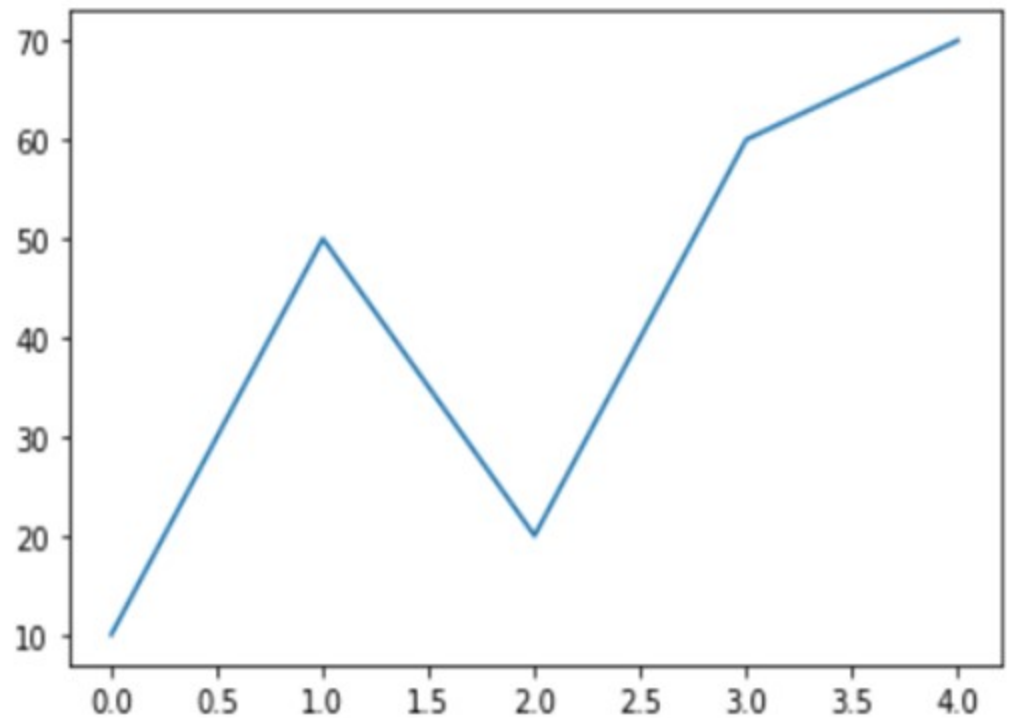
```
[<matplotlib.lines.Line2D at 0x2052afef308>]
```

<- 그래프 객체 정보



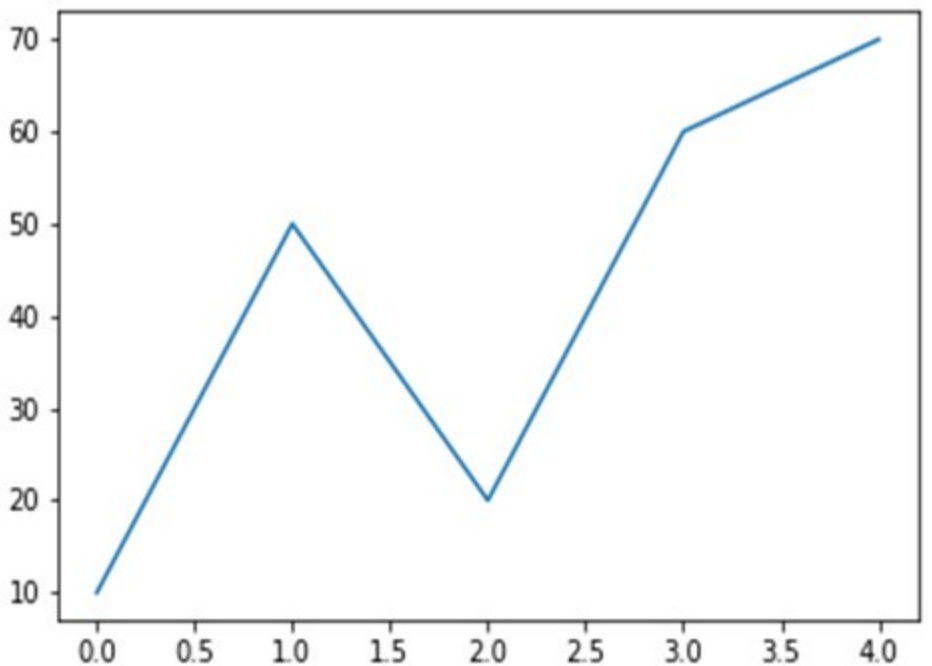
선 그래프 그리기

```
plt.plot(data)
# 객체 정보없이 그래프만 출력

plt.show()
```




```
%matplotlib qt

plt.plot(data)
```

```
[<matplotlib.lines.Line2D at 0x2052ffa3248>]
```



Matplotlib의 Formating의 종류

Colors

character	color
'b'	blue
'g'	green
'r'	red
'c'	cyan
'm'	magenta
'y'	yellow
'k'	black
'w'	white

Line Styles

character	description
'-'	solid line style
'--'	dashed line style
'-.'	dash-dot line style
':'	dotted line style

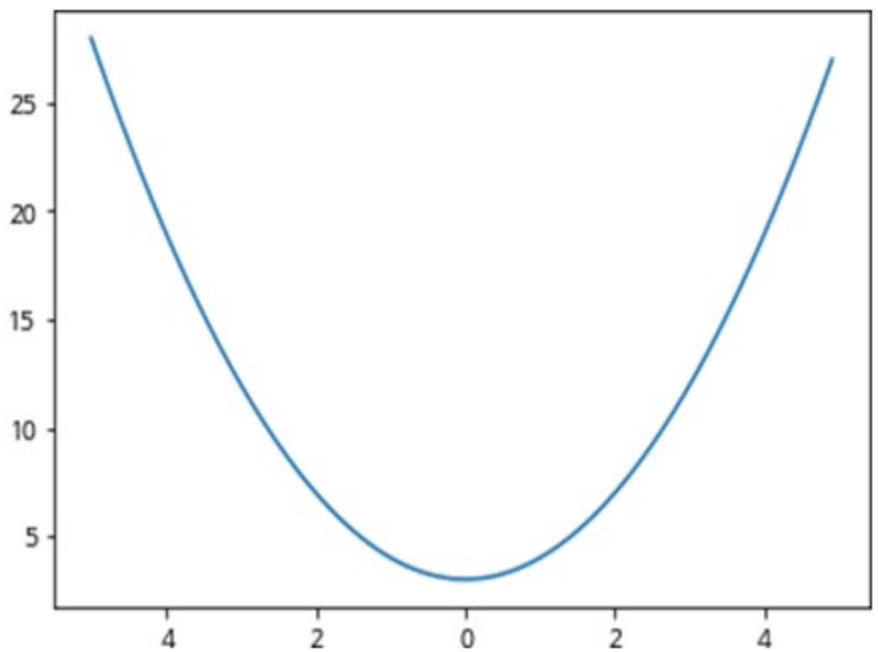
Markers

character	description
'.'	point marker
','	pixel marker
'o'	circle marker
'v'	triangle_down marker
'^'	triangle_up marker
'<'	triangle_left marker
'>'	triangle_right marker
'1'	tri_down marker
'2'	tri_up marker
'3'	tri_left marker
'4'	tri_right marker
's'	square marker
'p'	pentagon marker
'*'	star marker
'h'	hexagon1 marker
'H'	hexagon2 marker
'+'	plus marker
'x'	x marker
'D'	diamond marker
'd'	thin_diamond marker
'|'	vline marker
'_'	hline marker

Figure & Axes

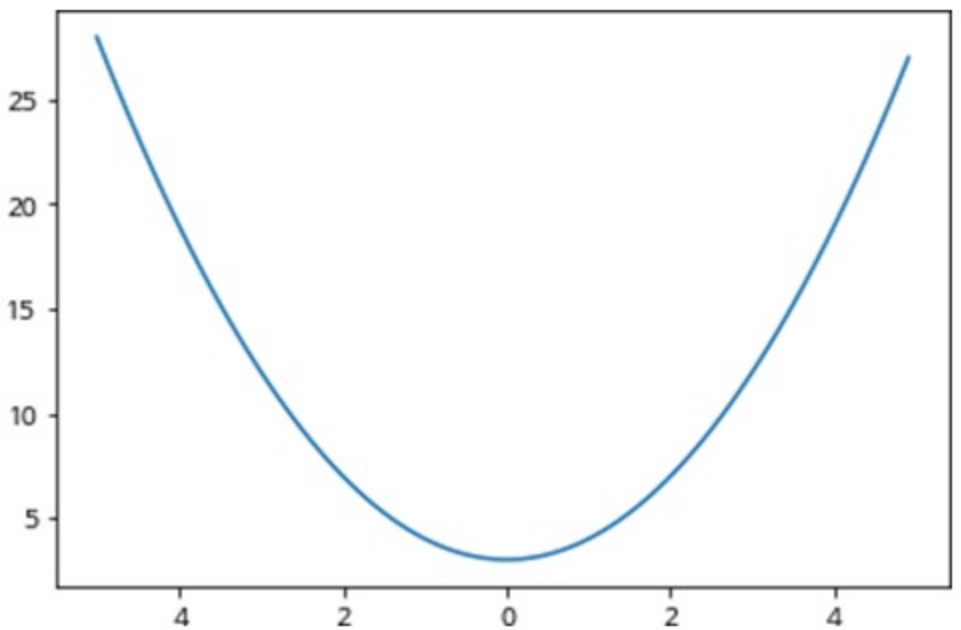
- The pyplot API

```
plt.plot(x, y1)
```

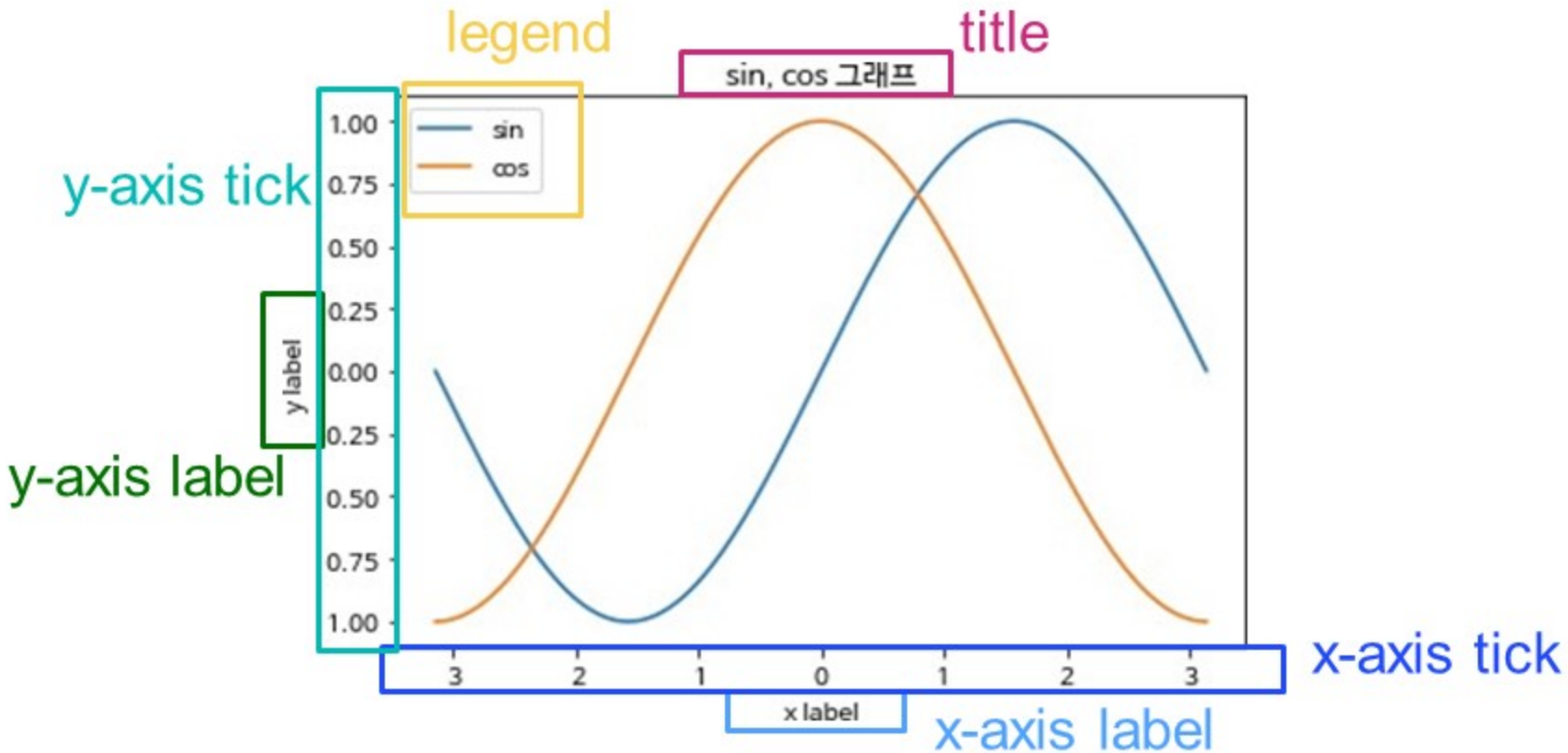


- The object-oriented API

```
fig, ax = plt.subplots()
ax.plot(x, y1)
```



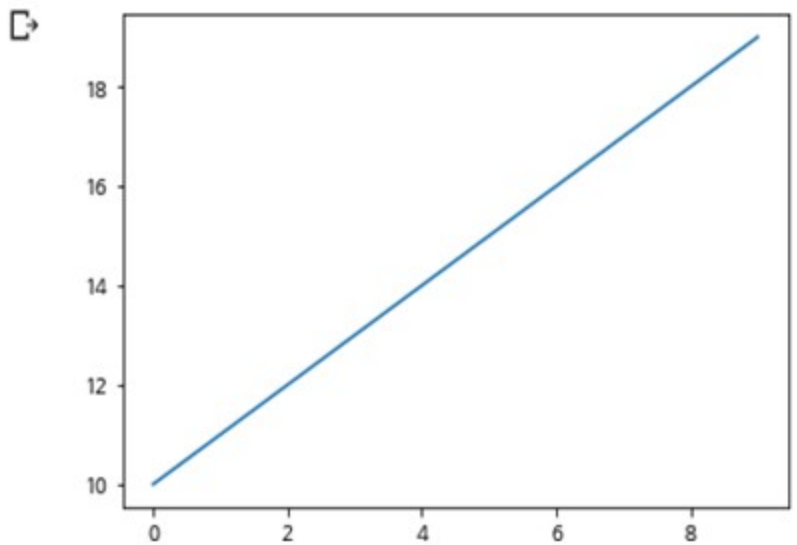
Axis


legend
title
sin, cos 그래프
y-axis tick
y label
y-axis label
sin
cos
1.00
0.75
0.50
0.25
0.00
0.25
0.50
0.75
1.00
3
2
1
0
1
2
3
x label
x-axis tick
x-axis label

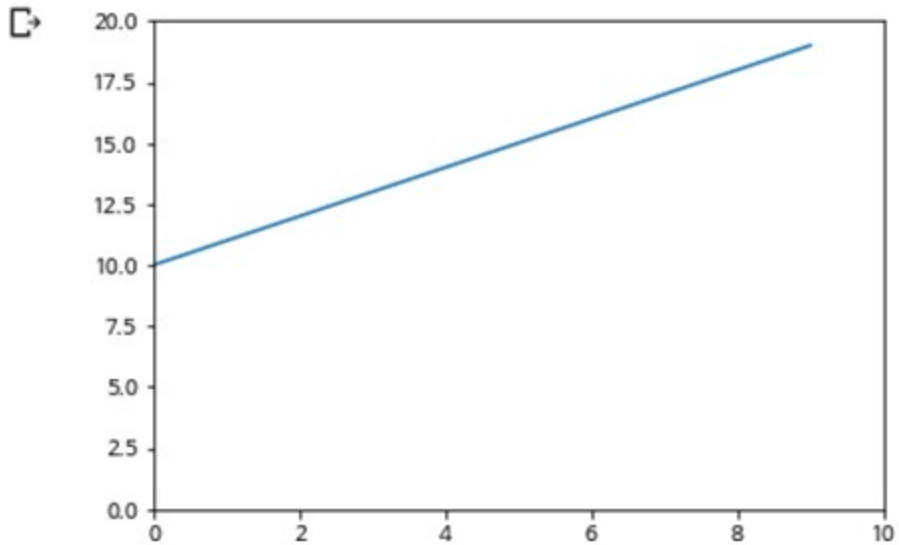
Axis

- xlim, yli

```
[15] x = np.arange(10)
     y = x+10

     plt.plot(x, y)

     plt.show()
```



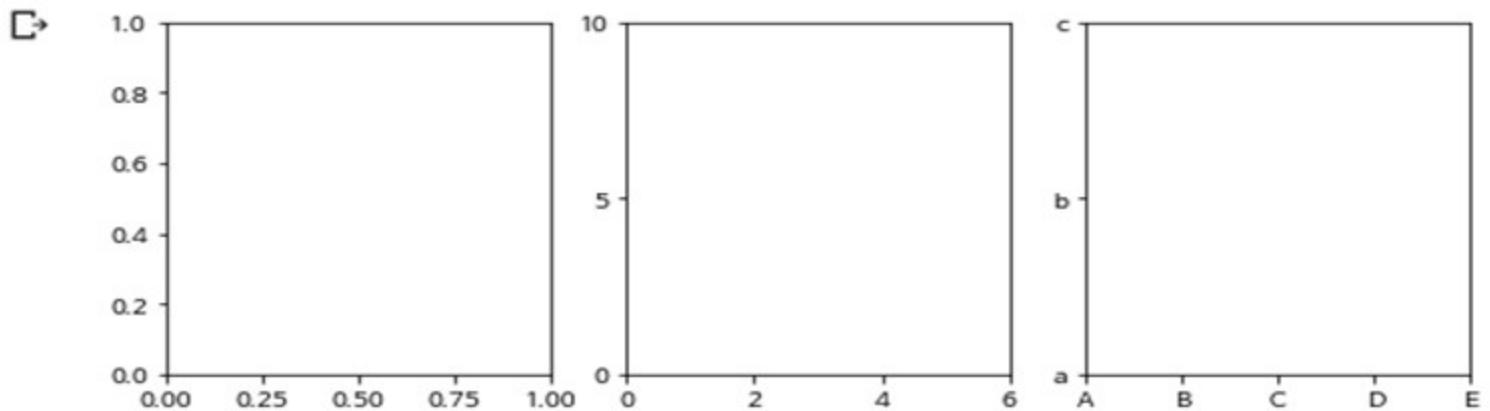

```
plt.xlim([0, 10])
plt.ylim([0, 20])

plt.plot(x, y)

plt.show()
```



Axis

- ticks, tick labels

```
fig, axs = plt.subplots(1, 3, figsize=(9, 3))

axs[1].set_xticks([0,2,4,6])
axs[1].set_yticks([0,5,10])

axs[2].set_xticklabels(['A', 'B', 'C', 'D', 'E'])
axs[2].set_yticks([0,1,2])
axs[2].set_yticklabels(['a', 'b', 'c'])

plt.show()
```



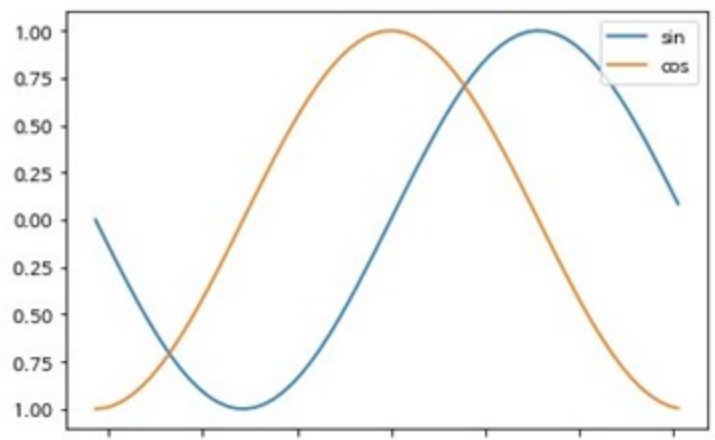
Axis

- 한글 표시

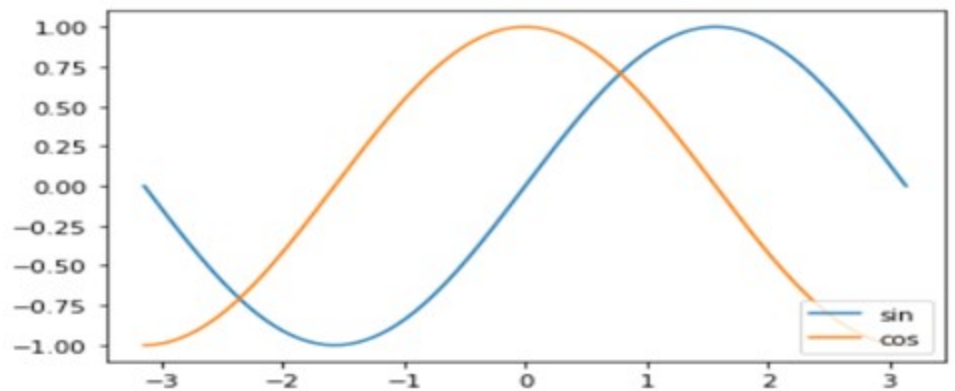
```
import matplotlib

matplotlib.rcParams['font.family'] = 'Malgun Gothic' #맑은 고딕으로 폰트 설정
matplotlib.rcParams['axes.unicode_minus'] = False   #마이너스(-) 폰트 깨짐 방지
```

Legend

```
fig, ax = plt.subplots()

ax.plot(x, y1, label = 'sin')
ax.plot(x, y2, label = 'cos')

ax.legend(loc=1)

plt.show()
```



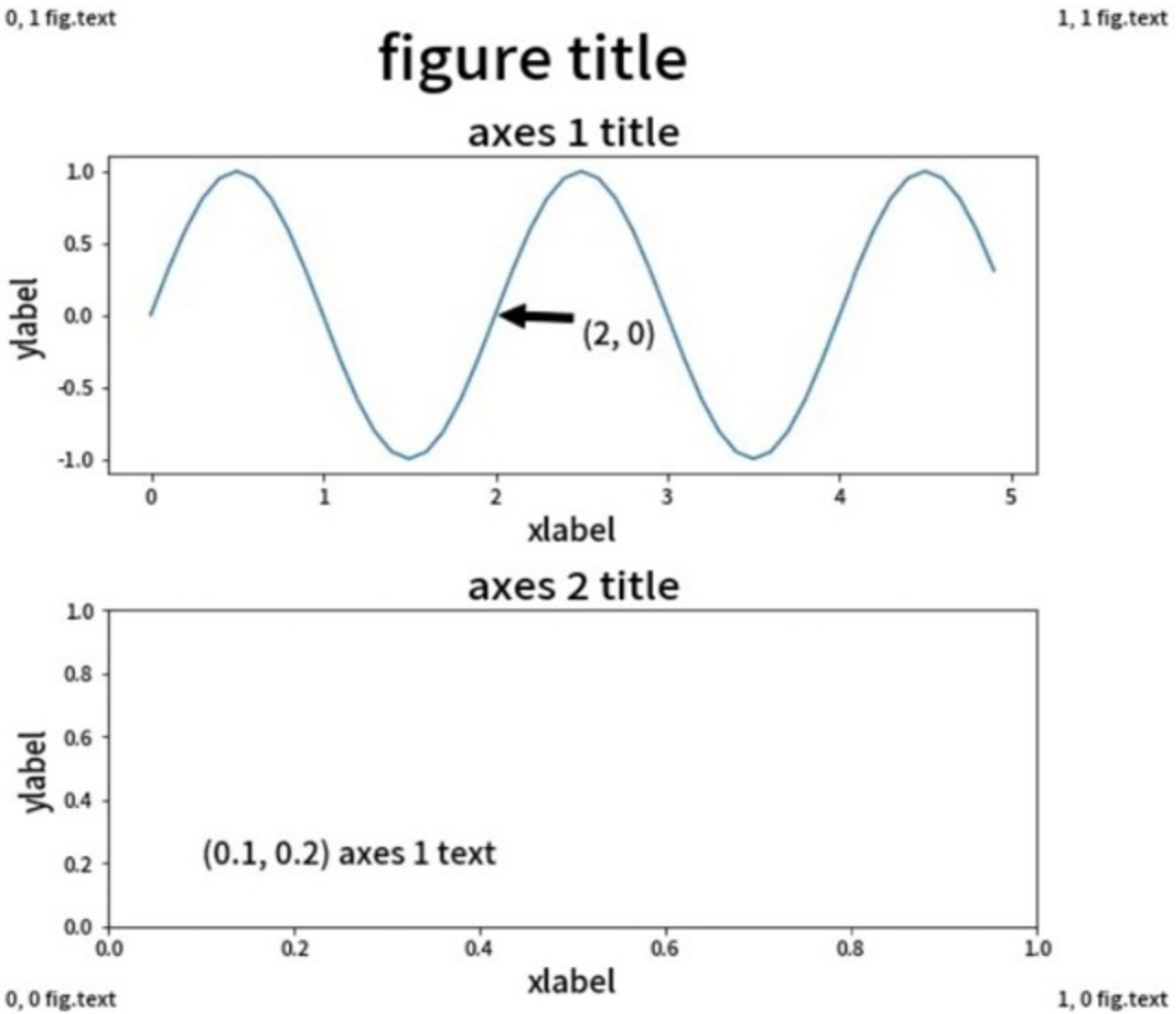

```
import matplotlib.pyplot as plt
import numpy as np

x = np.linspace(-np.pi, np.pi, 201)
y1 = np.sin(x)
y2 = np.cos(x)

fig,ax = plt.subplots()

ax.plot(x, y1, label = 'sin')
ax.plot(x, y2, label = 'cos')

ax.legend(loc='lower right')


plt.show()
```



참고 : https://matplotlib.org/3.1.1/api/_as_gen/matplotlib.pyplot.legend.html

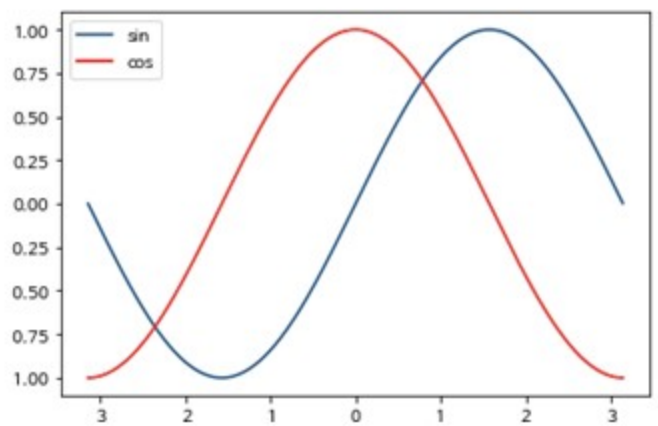
Text

- figure title
- axes title
- x-axis label
- y-axis label
- figure text
- axes text
- annotate



Color

```
x = np.arange(-np.pi, np.pi, 0.02)
y1 = np.sin(x)
y2 = np.cos(x)

plt.plot(x, y1, label = 'sin', color= (0.1, 0.3, 0.5))
plt.plot(x, y2, label = 'cos', color='r')
plt.legend()
plt.show()
```



- color 인자를 명시하여 색을 설정 할 수 있음

1) matplotlib에서 미리 정해둔 값으로 설정

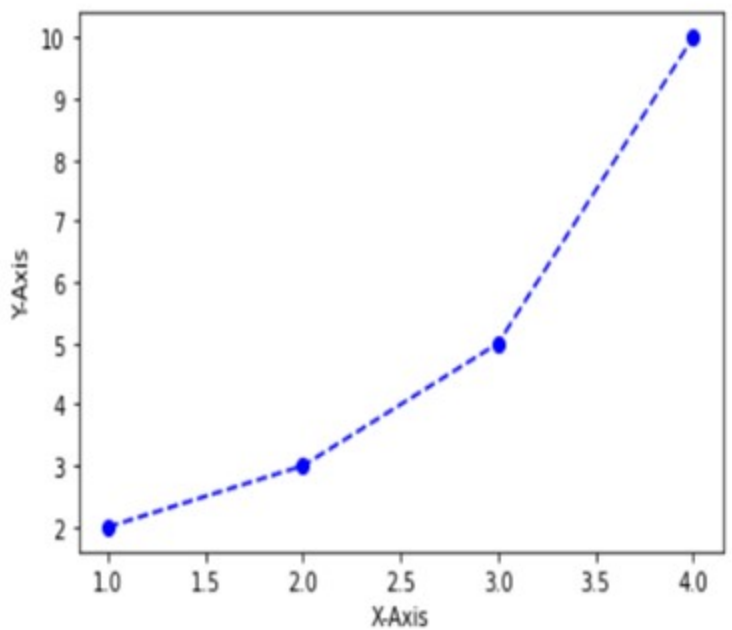
{'b', 'g', 'r', 'c', 'm', 'y', 'k', 'w'}

2)RGB 값을 튜플로 묶어서 주기

3)색상팔레트의 값을 string으로 주기

Markers

```
import matplotlib.pyplot as plt

# plt.plot([1, 2, 3, 4], [2, 3, 5, 10], 'bo-')    # 파란색 + 실선 + 마커
plt.plot([1, 2, 3, 4], [2, 3, 5, 10], 'bo--')     # 파란색 + 점선 + 마커
plt.xlabel('X-Axis')
plt.ylabel('Y-Axis')
plt.show()
```



- Marker 인자를 명시하여 색을 설정 할 수 있음

Colors

character	color
'b'	blue
'g'	green
'r'	red
'c'	cyan
'm'	magenta
'y'	yellow
'k'	black
'w'	white

Line Styles

character	description
'-'	solid line style
'--'	dashed line style
'-.'	dash-dot line style
':'	dotted line style

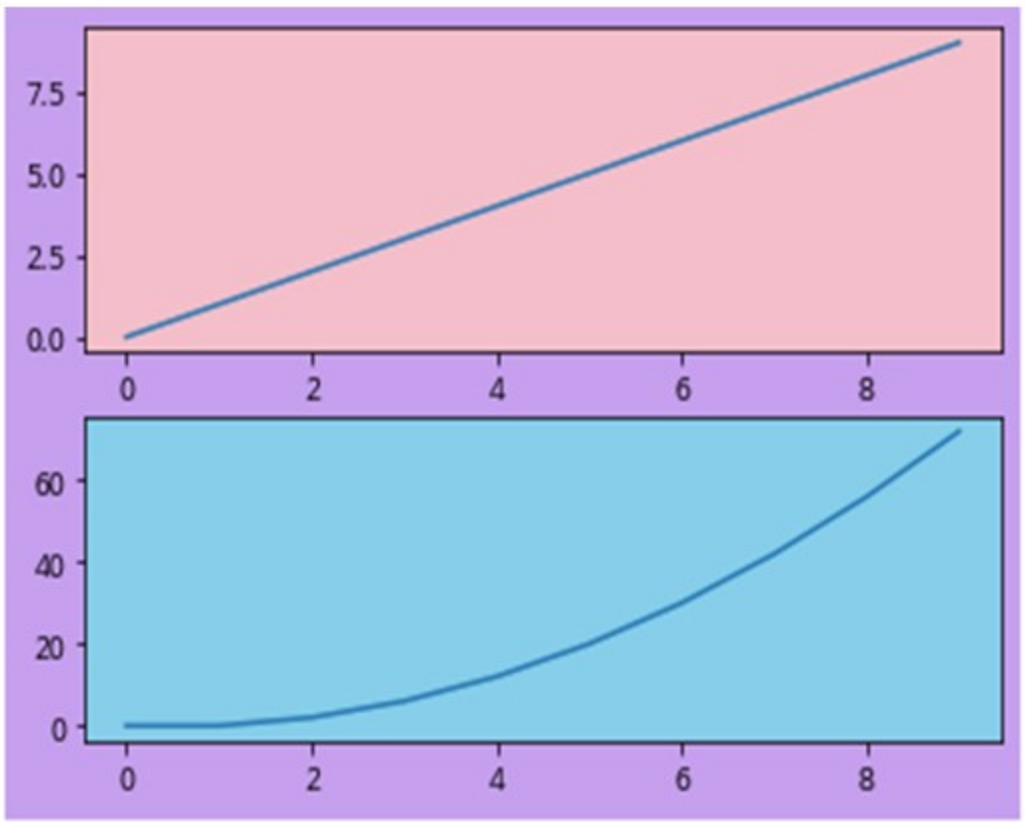
Markers

character	description
'.'	point marker
','	pixel marker
'o'	circle marker
'v'	triangle_down marker
'^'	triangle_up marker
'<'	triangle_left marker
'>'	triangle_right marker
'1'	tri_down marker
'2'	tri_up marker
'3'	tri_left marker
'4'	tri_right marker
's'	square marker
'p'	pentagon marker
'*'	star marker
'h'	hexagon1 marker
'H'	hexagon2 marker
'+'	plus marker
'x'	x marker
'D'	diamond marker
'd'	thin_diamond marker
'|'	vline marker
'_'	hline marker

Face Color

- 바탕색

```
x = np.arange(10)
y1 = x
y2 = x**2 - x

fig, axs = plt.subplots(2, 1)
fig.set_facecolor('#c79fef')

axs[0].plot(x, y1)
axs[1].plot(x, y2)

axs[0].set_facecolor('pink')
axs[1].set_facecolor('skyblue')

plt.show()
```



savefig

- plot을 image로 저장

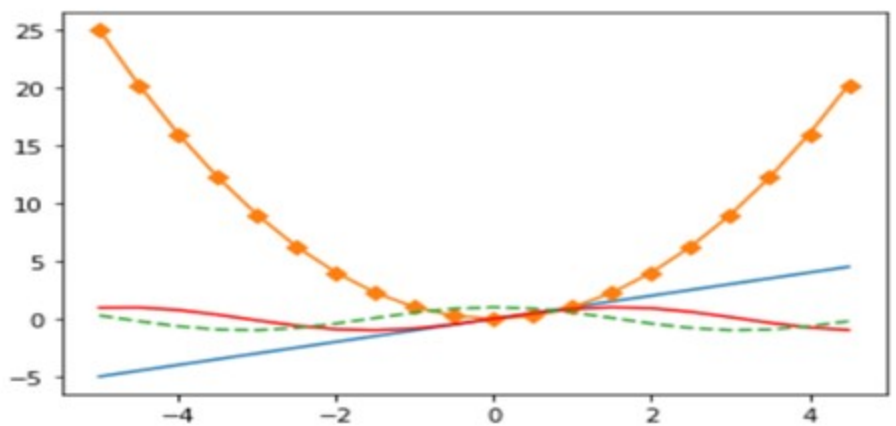
```
fig, ax = plt.subplots()
x = np.arange(10)
y1 = x**2
ax.plot(x, y1, label = 'sin')
fig.savefig('image_matplot_tmp.jpg')
```

Line Plot

- plot( )

```
import numpy as np
import matplotlib.pyplot as plt

%matplotlib inline

x = np.arange(-5, 5, 0.5)
y1 = x
y2 = x **2
y3 = np.sin(x)
y4 = np.cos(x)

plt.plot(x,y1)
plt.plot(x,y2, marker = 'D')
plt.plot(x,y3, color ='r')
plt.plot(x,y4, linestyle = 'dashed')
plt.show
```

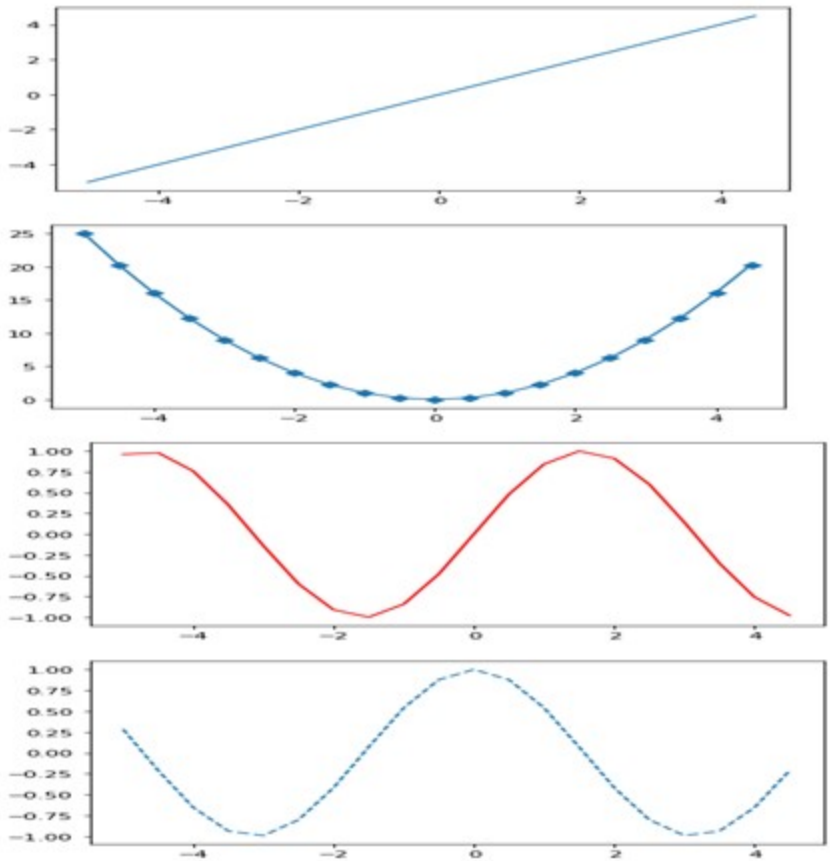
```
<function matplotlib.pyplot.show(*args, **kw)>
```



Line Plot

- plot( )

```
plt.plot(x,y1)

plt.figure()
plt.plot(x,y2, marker = 'D')

plt.figure()
plt.plot(x,y3, color ='r')

plt.figure()
plt.plot(x,y4, linestyle = 'dashed')
plt.show()
```



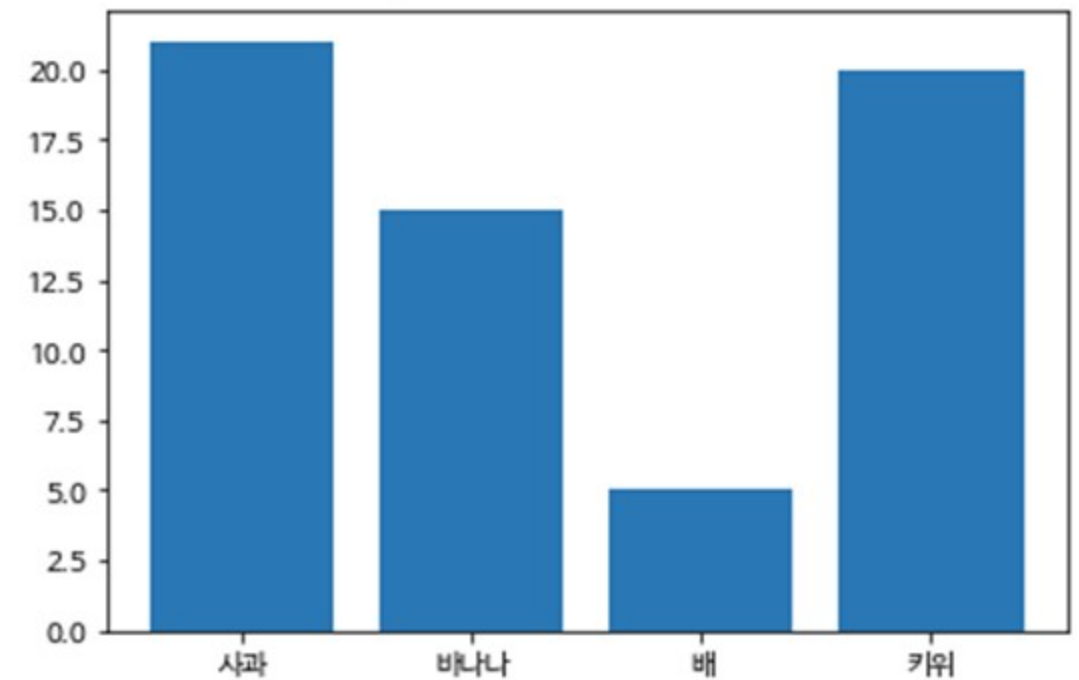
참고 :
https://matplotlib.org/3.1.1/api/_as_gen/matplotlib.lines.Line2D.html#matplotlib.lines.Line2D.set_marker

Bar Plot

- bar( )

```
data = {'사과': 21, '바나나': 15, '배': 5, '키위': 20}
names = list(data.keys())
values = list(data.values())

fig, ax = plt.subplots()
ax.bar(names, values)
```

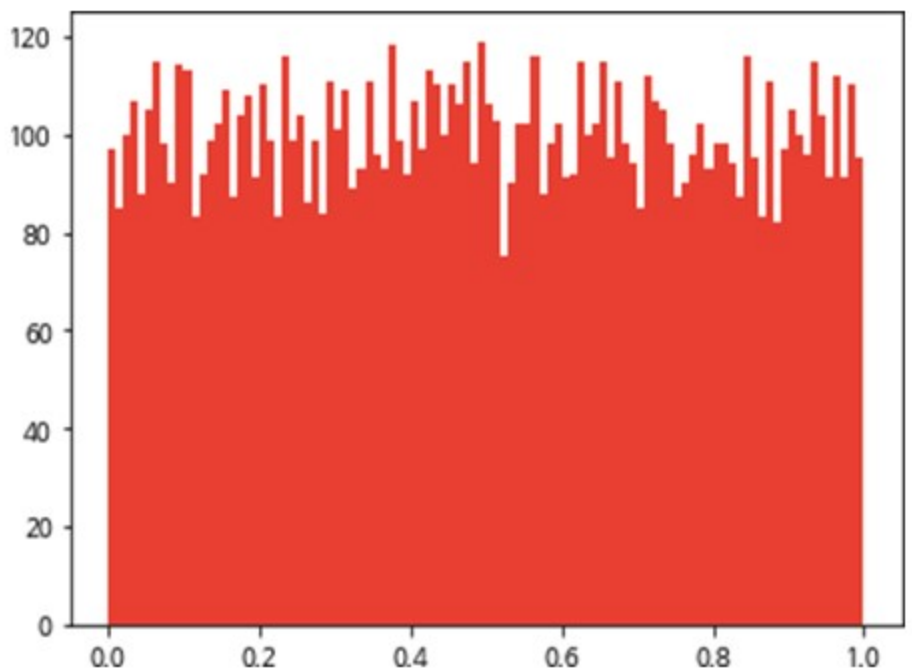
```
<BarContainer object of 4 artists>
```



Histogram

- hist( )

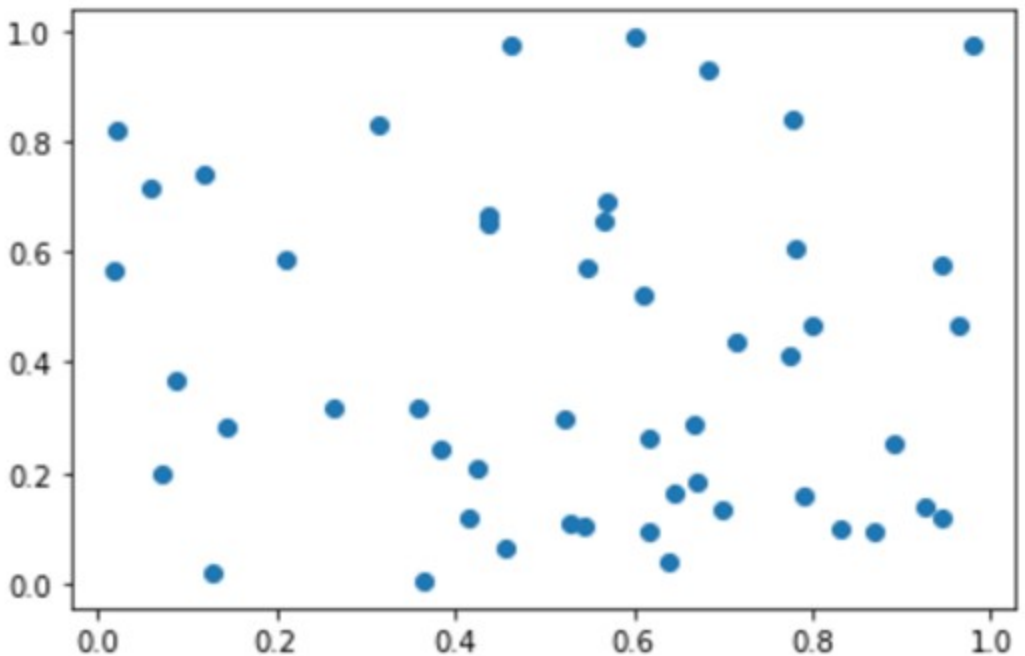
```
data = np.random.rand(10000)
fig, ax = plt.subplots()
ax.hist(data, bins = 100, facecolor='r')
plt.show()
```



2D Scatter Plot

- Scatter( )

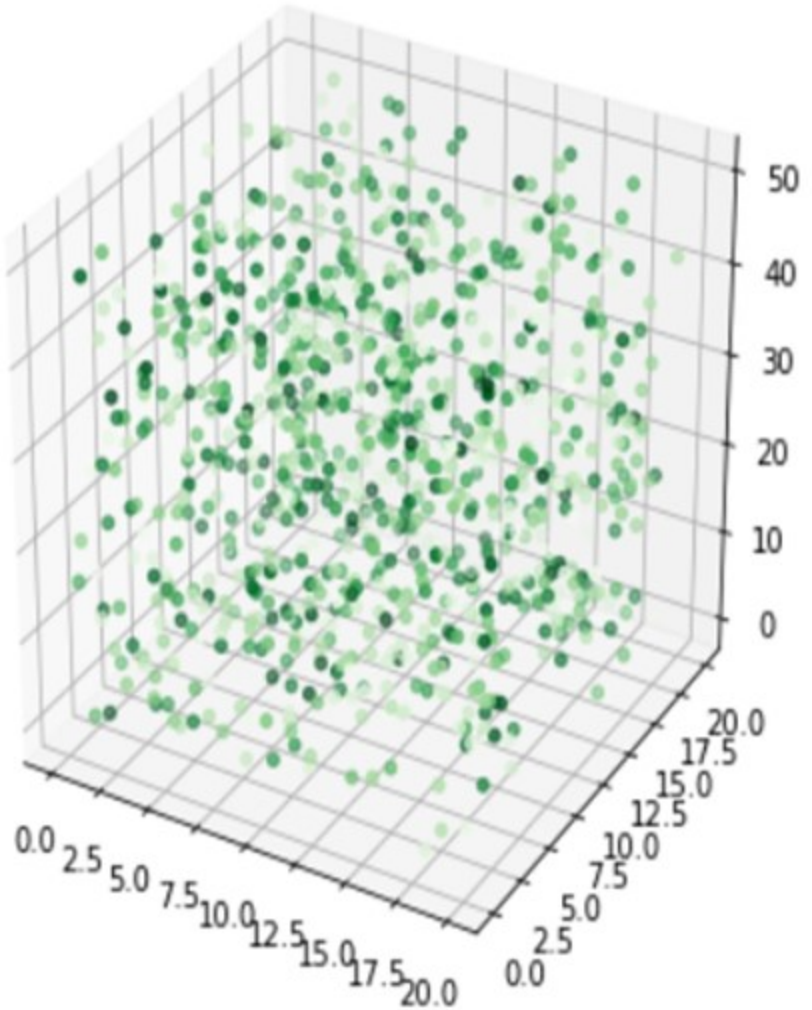
```
import matplotlib.pyplot as plt
import numpy as np

np.random.seed(0)

n = 50
x = np.random.rand(n)
y = np.random.rand(n)

plt.scatter(x, y)
plt.show()
```



3D Scatter Plot

- Scatter( )

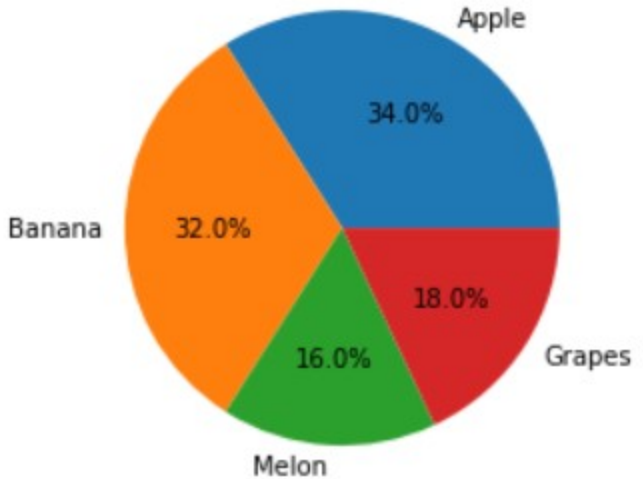
```
from mpl_toolkits.mplot3d import Axes3D
import matplotlib.pyplot as plt
import numpy as np

n = 1000
xmin, xmax, ymin, ymax, zmin, zmax = 0, 20, 0, 20, 0, 50
cmin, cmax = 0, 2

xs = np.array([(xmax - xmin) * np.random.random_sample() + xmin for i in range(n)])
ys = np.array([(ymax - ymin) * np.random.random_sample() + ymin for i in range(n)])
zs = np.array([(zmax - zmin) * np.random.random_sample() + zmin for i in range(n)])
color = np.array([(cmax - cmin) * np.random.random_sample() + cmin for i in range(n)])

fig = plt.figure(figsize=(6, 6))
ax = fig.add_subplot(111, projection='3d')
ax.scatter(xs, ys, zs, c=color, marker='o', s=15, cmap='Greens')

plt.show()
```



Pie chart

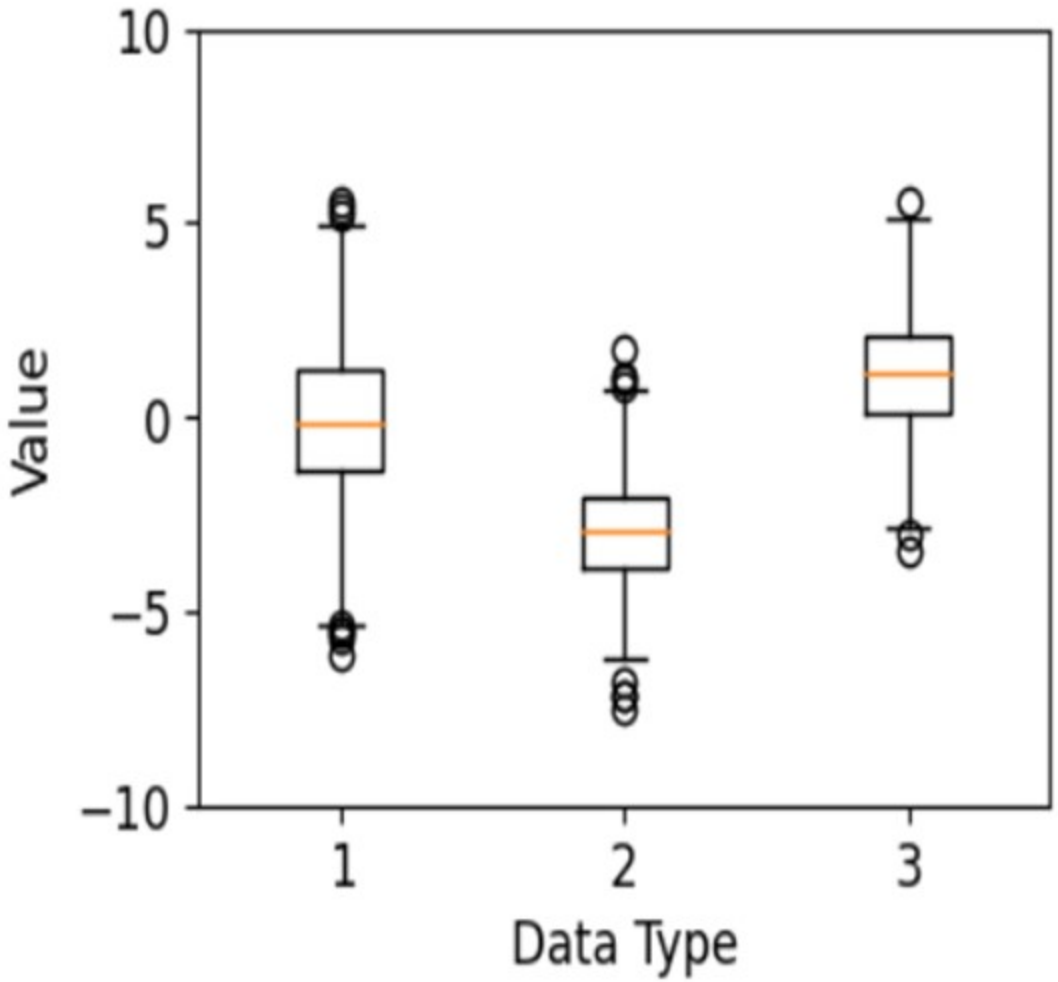
- Pie( )

```
import matplotlib.pyplot as plt

ratio = [34, 32, 16, 18]
labels = ['Apple', 'Banana', 'Melon', 'Grapes']

plt.pie(ratio, labels=labels, autopct='%.1f%%')
plt.show()
```



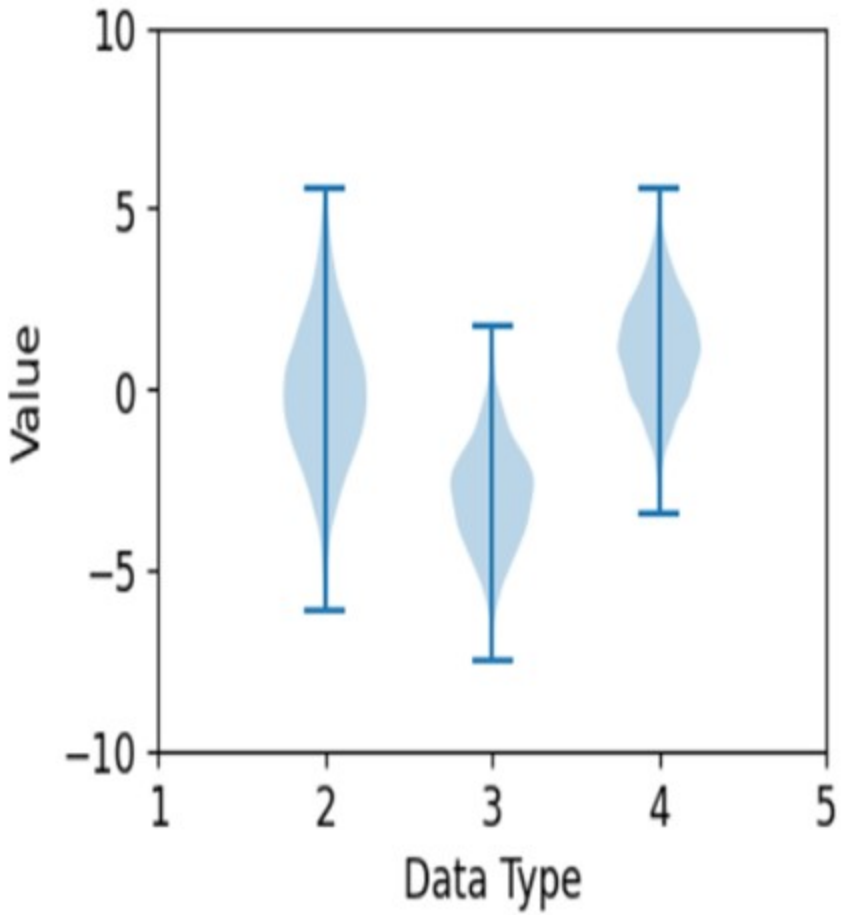
Box Plot

```python
import matplotlib.pyplot as plt
import numpy as np

# 1. 기본 스타일 설정
plt.style.use('default')
plt.rcParams['figure.figsize'] = (4, 3)
plt.rcParams['font.size'] = 12

# 2. 데이터 준비
np.random.seed(0)
data_a = np.random.normal(0, 2.0, 1000)
data_b = np.random.normal(-3.0, 1.5, 500)
data_c = np.random.normal(1.2, 1.5, 1500)

# 3. 그래프 그리기
fig, ax = plt.subplots()

ax.boxplot([data_a, data_b, data_c])
ax.set_ylim(-10.0, 10.0)
ax.set_xlabel('Data Type')
ax.set_ylabel('Value')

plt.show()
```



Violin Plot

```python
# 기본 사용
import matplotlib.pyplot as plt
import numpy as np

# 1. 기본 스타일 설정
plt.style.use('default')
plt.rcParams['figure.figsize'] = (4, 3)
plt.rcParams['font.size'] = 12

# 2. 데이터 준비
np.random.seed(0)
data_a = np.random.normal(0, 2.0, 1000)
data_b = np.random.normal(-3.0, 1.5, 500)
data_c = np.random.normal(1.2, 1.5, 1500)

# 3. 그래프 그리기
fig, ax = plt.subplots()

violin = ax.violinplot([data_a, data_b, data_c], positions=[2, 3, 4])
ax.set_ylim(-10.0, 10.0)
ax.set_xticks([1, 2, 3, 4, 5])
ax.set_xlabel('Data Type')
ax.set_ylabel('Value')

plt.show()
```



Image

```
[48] import matplotlib.image as mpimg
```

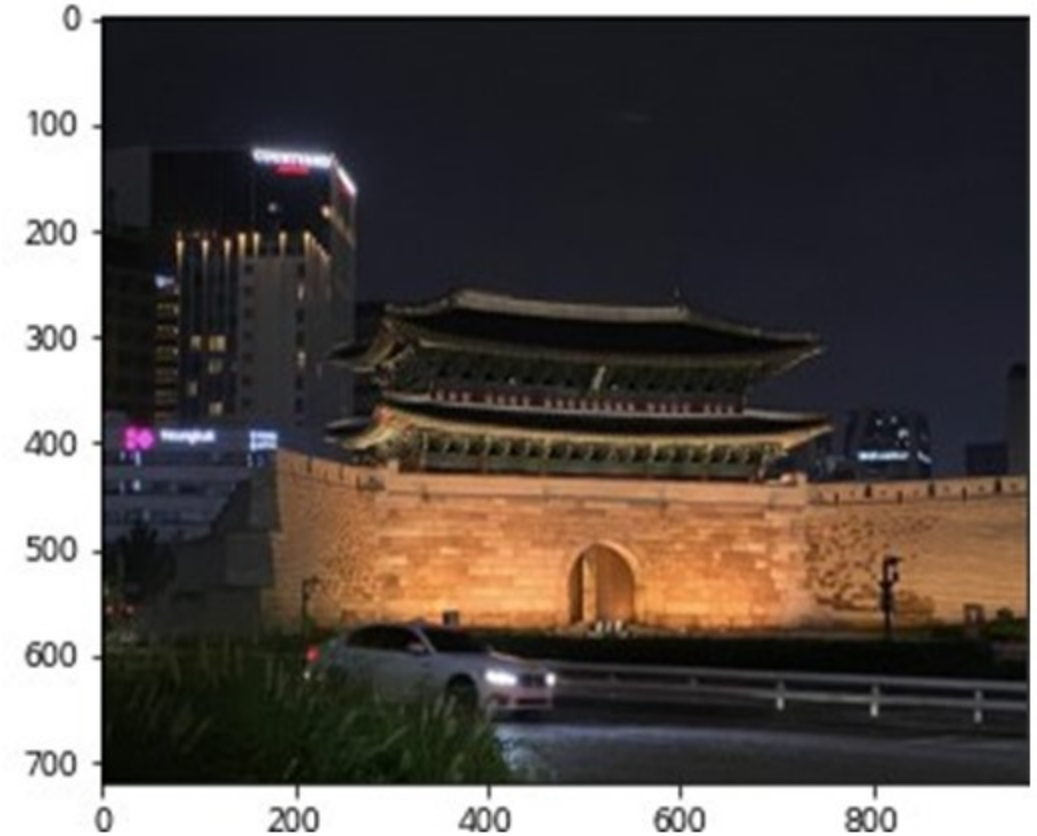
```
img = mpimg.imread('img_.jpeg')
print(type(img))
```

```
<class 'numpy.ndarray'>
```

```
imgplot = plt.imshow(img)
print(type(imgplot))
```

```
<class 'matplotlib.image.AxesImage'>
```

```
<class 'matplotlib.image.AxesImage'>
```



Seaborn

- seaborn은 matplotlib을 기반으로 다양한 색 테마, 차트 기능을 추가한 라이브러리입니다
  - matplotlib에 의존성을 가지고 있습니다
  - matplotlib에 없는 그래프(히트맵, 카운트플랏 등)을 가지고 있습니다

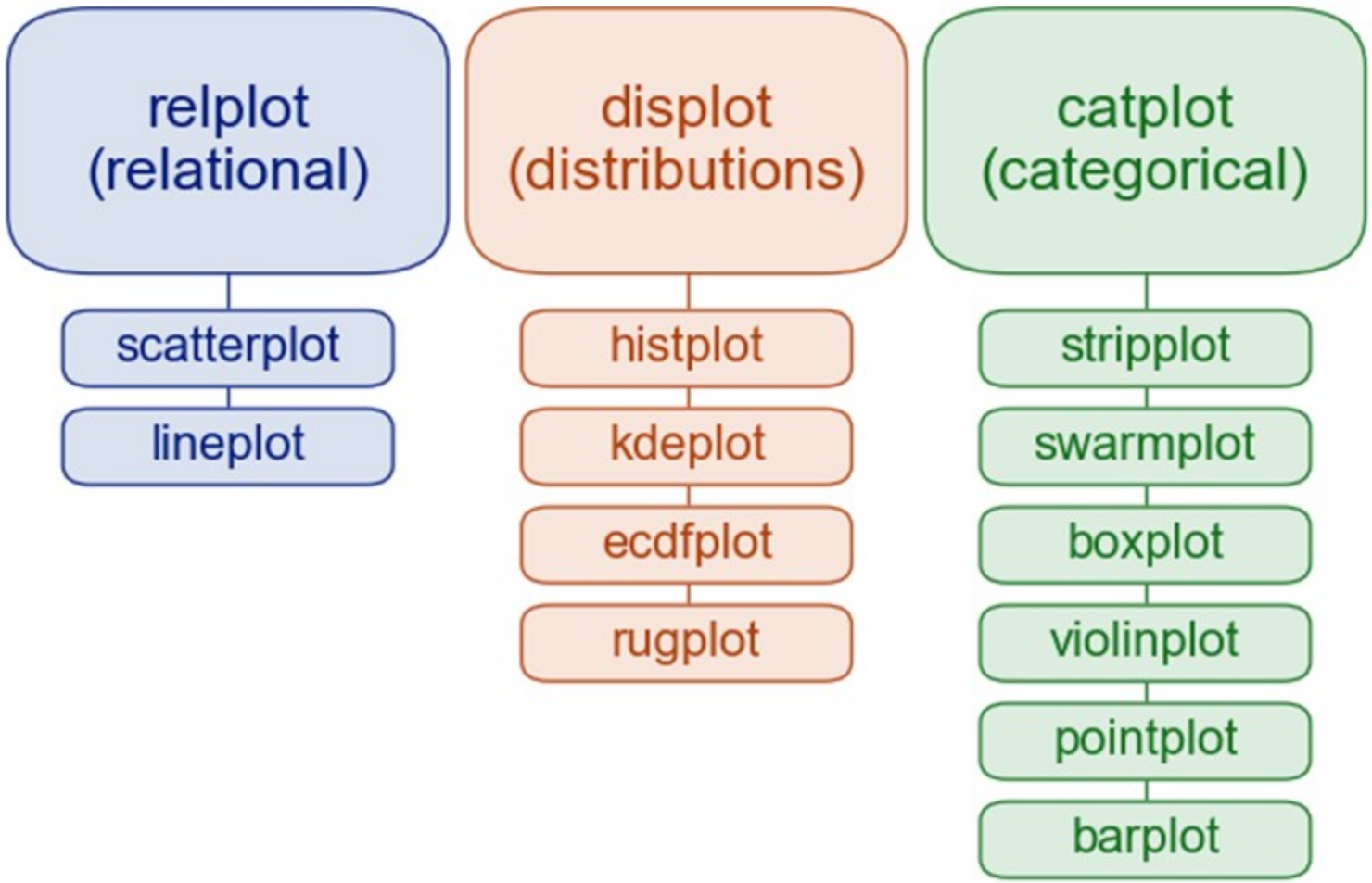
설치

pip install seaborn

```
import seaborn as sns

print("Seaborn version : ", sns.__version__)
```

```
Seaborn version :  0.9.0
```

Seaborn의 종류

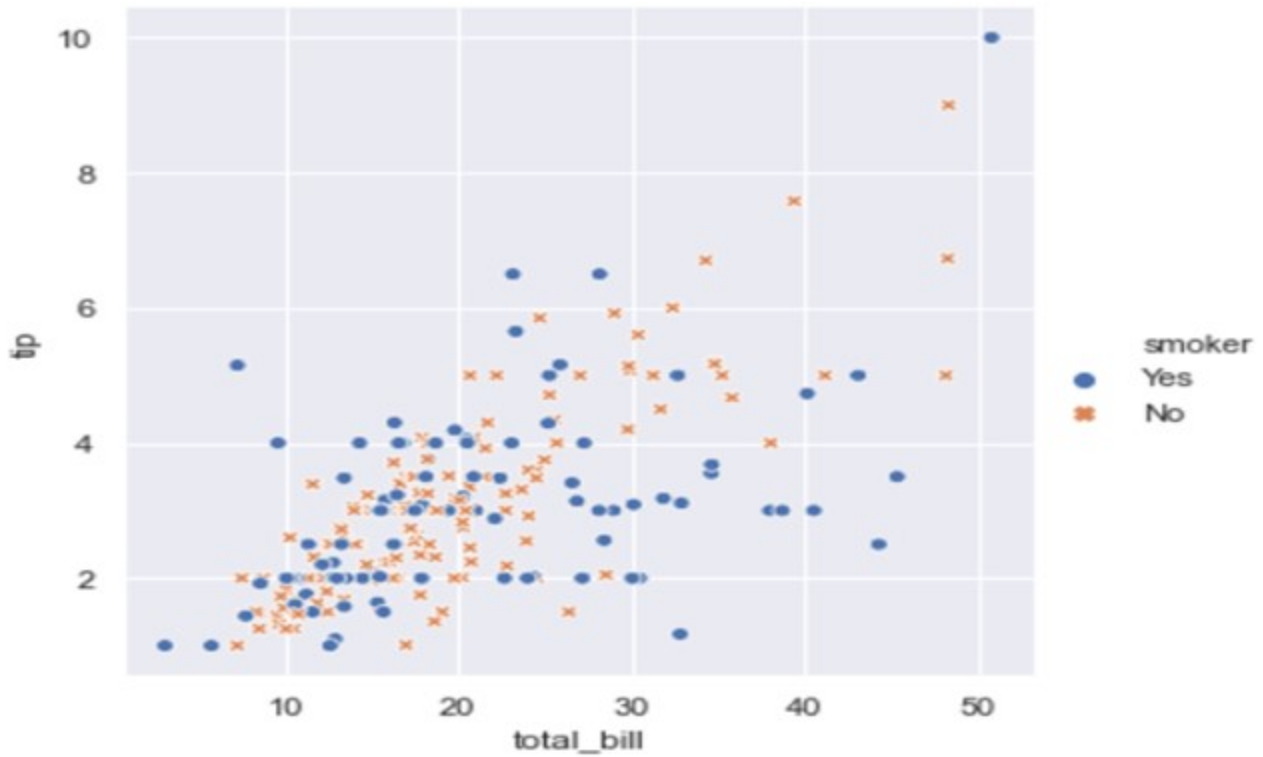


참고 : https://seaborn.pydata.org/tutorial.html

Seaborn_relplot

```
tips = sns.load_dataset("tips")

sns.relplot(x="total_bill", y="tip", hue="smoker", style="smoker",
            data=tips)
```

```
<seaborn.axisgrid.FacetGrid at 0x1bd18b8b390>
```



Seaborn_relplot

```
import seaborn as sns
import pandas as pd
import numpy as np
df = pd.DataFrame(dict(time=np.arange(500),
                       value=np.random.randn(500).cumsum()))
g = sns.relplot(x="time", y="value", kind="line", data=df)
g.fig.autofmt_xdate()
```

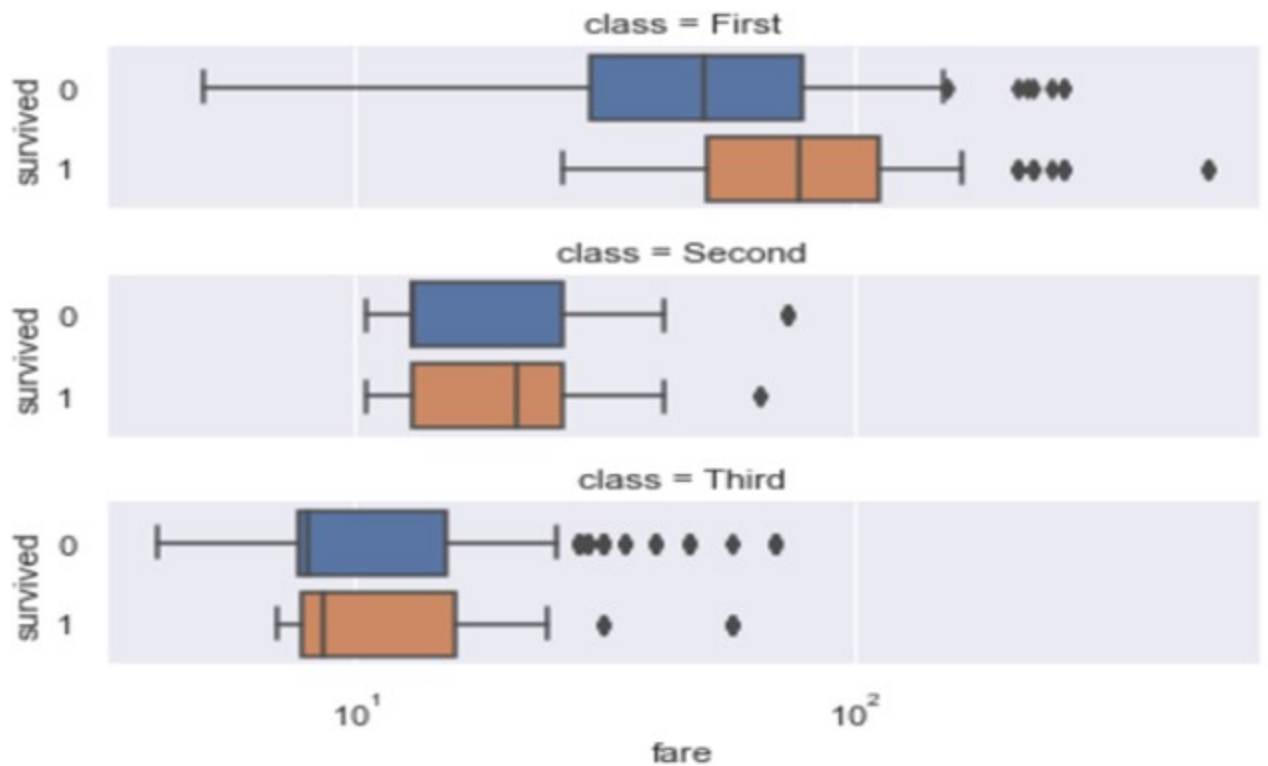
Seaborn_catplot

```
sns.catplot(x="day", y="total_bill", hue="smoker",
            col="time", aspect=.6,
            kind="swarm", data=tips)
```

```
<seaborn.axisgrid.FacetGrid at 0x1bd1acb92e8>
```

Seaborn_catplot

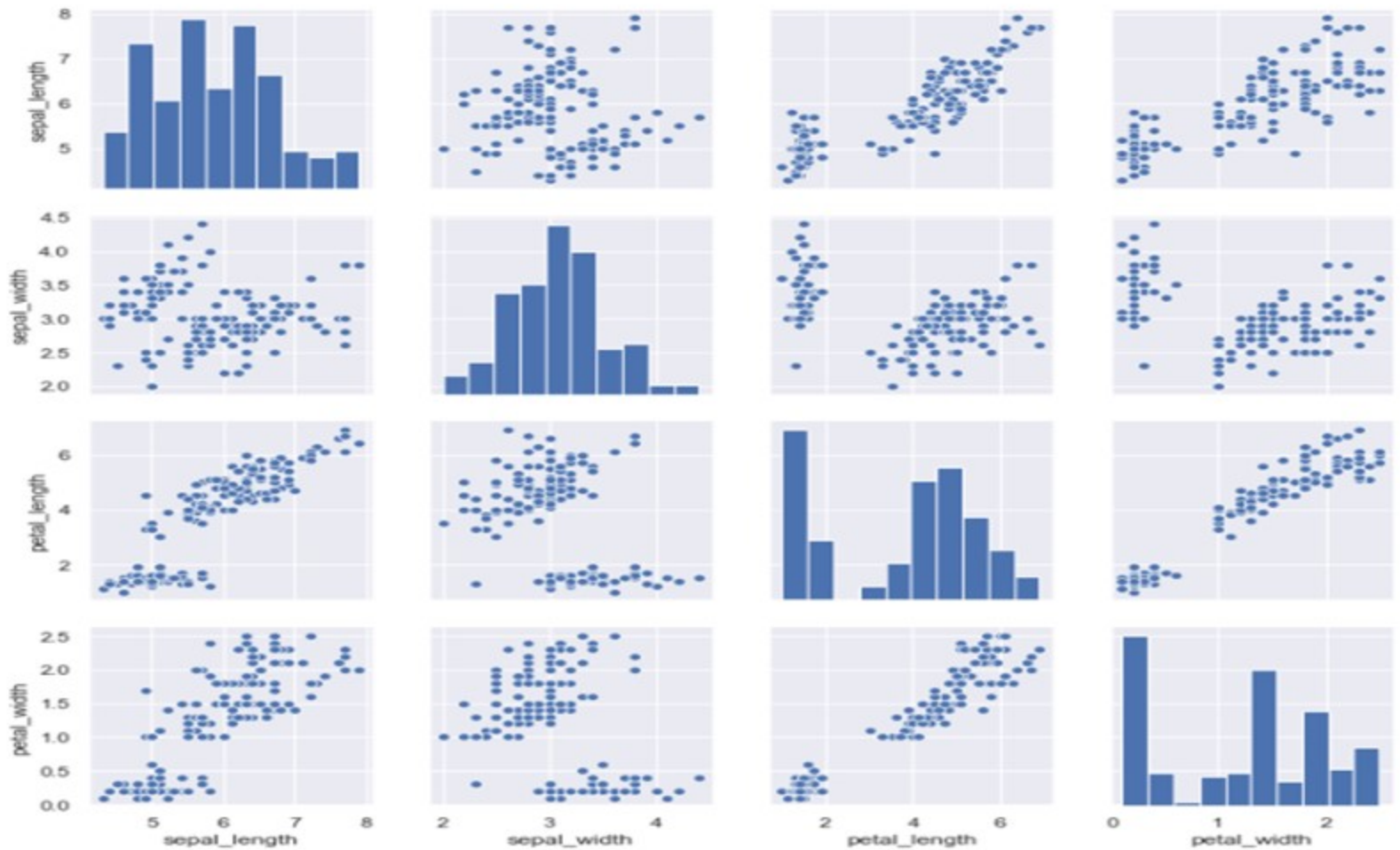
```
titanic = sns.load_dataset("titanic")
g = sns.catplot(x="fare", y="survived", row="class",
                kind="box", orient="h", height=1.5, aspect=4,
                data=titanic.query("fare > 0"))
g.set(xscale="log");
```



Seaborn_pairplot

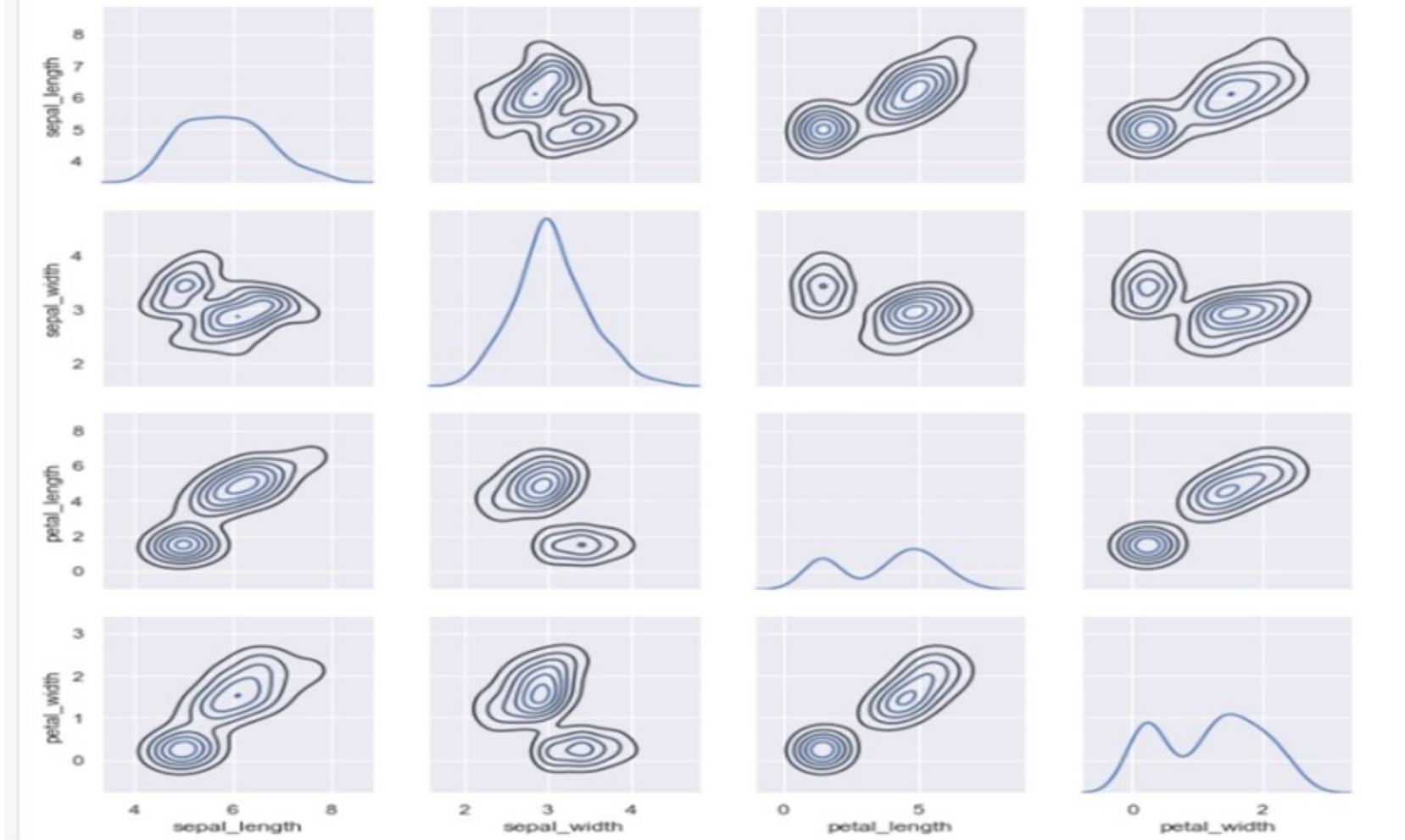
```
iris = sns.load_dataset("iris")
sns.pairplot(iris)
```

```
<seaborn.axisgrid.PairGrid at 0x1bd1b07f4e0>
```



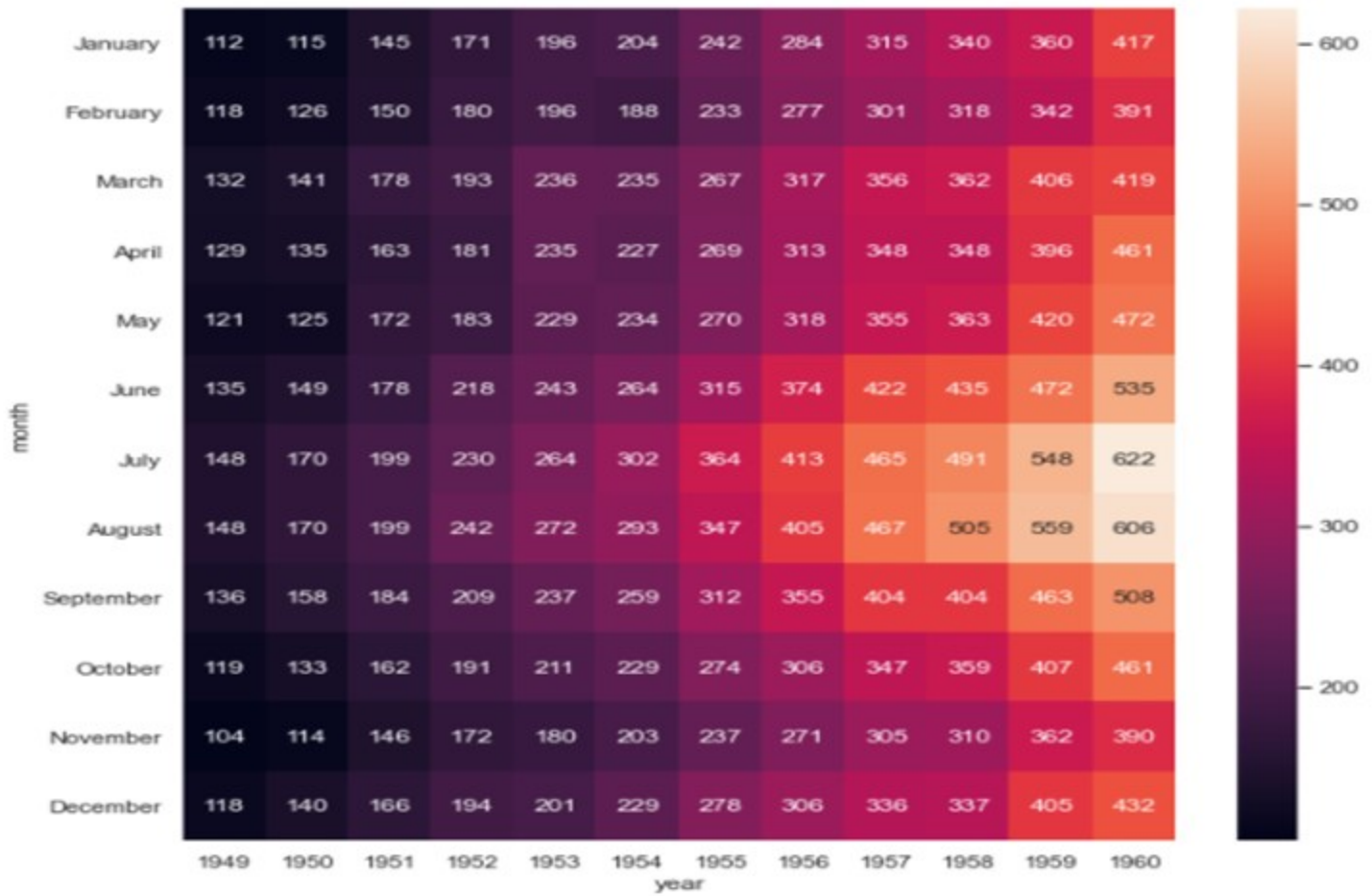
Seaborn_pairplot

```
g = sns.PairGrid(iris)
g.map_diag(sns.kdeplot)
g.map_offdiag(sns.kdeplot, n_levels=6);
```



Seaborn_Heatmap

```
import matplotlib.pyplot as plt
flights = sns.load_dataset("flights")
flights = flights.pivot("month", "year", "passengers")
plt.figure(figsize=(10, 10))
ax = sns.heatmap(flights, annot=True, fmt="d")
```



plotnine

plotnine은 R의 ggplot2에 기반해 그래프를 그려주는 라이브러리입니다

```
!pip install plotnine
```

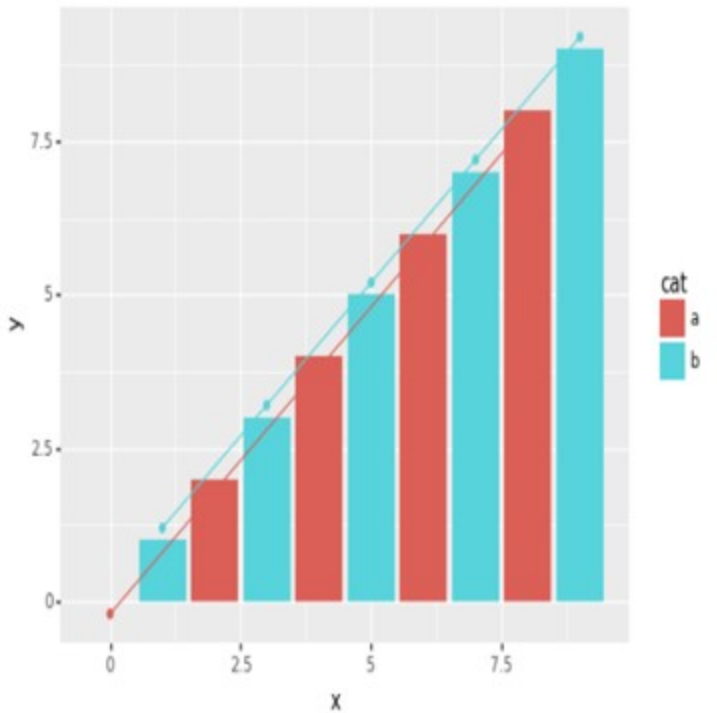
```
!pip3 install plotnine
```

```
import plotnine
import pandas as pd
import numpy as np
from plotnine import *
print("plontnine version :",plotnine.__version__)
```

```
plontnine version : 0.7.1
```

```
n = 10
df = pd.DataFrame({'x': np.arange(n),
                   'y': np.arange(n),
                   'yfit': np.arange(n) + np.tile([-.2, .2], n//2),
                   'cat': ['a', 'b']*(n//2)})
```

```
(ggplot(df)
 + geom_col(aes('x', 'y', fill='cat'))
 + geom_point(aes('x', y='yfit', color='cat'))
 + geom_path(aes('x', y='yfit', color='cat'))
)
```



Folium

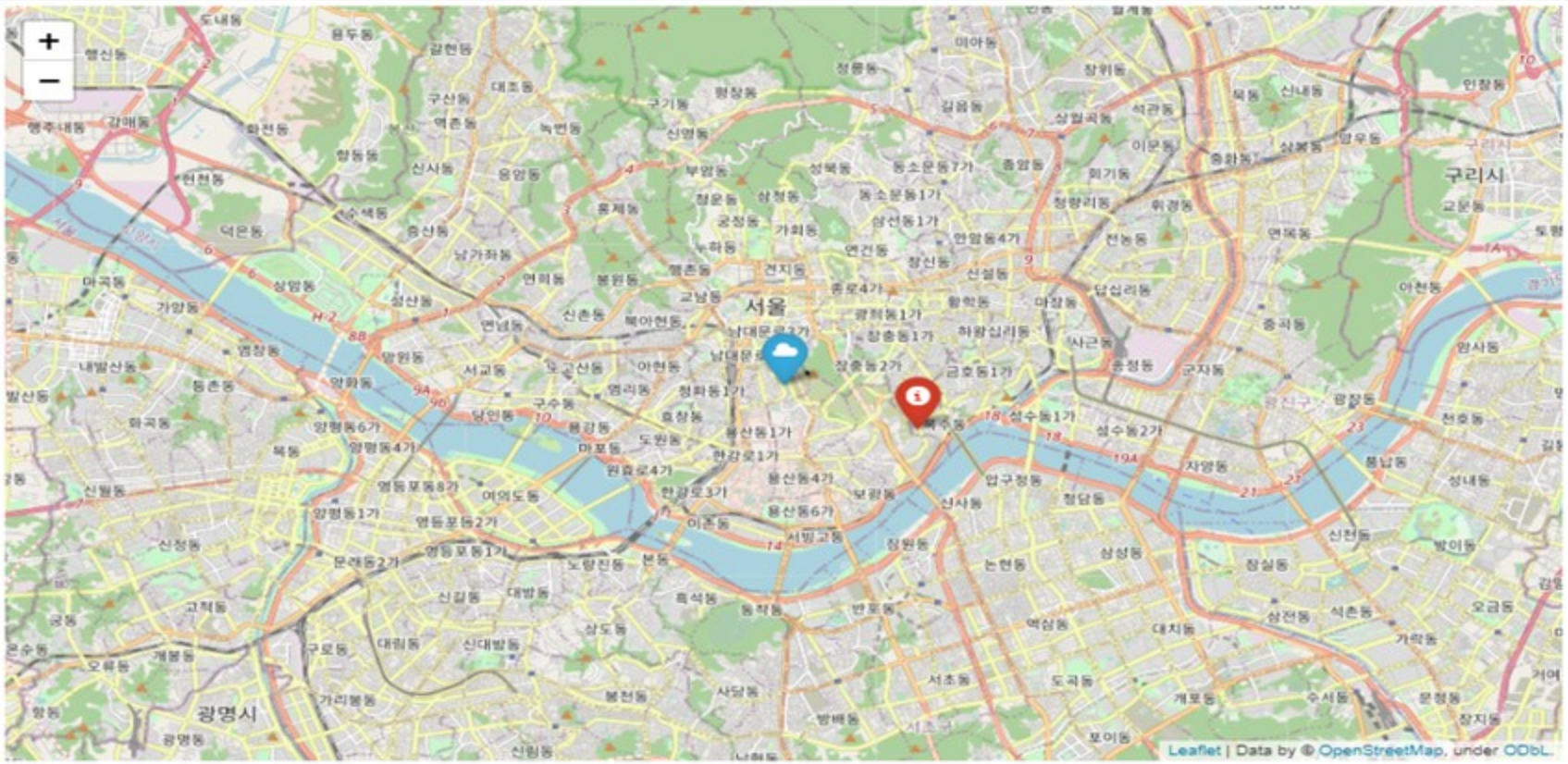
- Folium은 지도 데이터(Open Street Map)에 leaflet.js를 이용해 위치 정보를 시각화하는 라이브러리
- 자바스크립트 기반이라 interactive하게 그래프를 그릴 수 있음.
- 그 외에도 pydeck, ipyleaflet 등으로 지도 시각화를 할 수 있습니다

```
!pip install folium
```

참고 : 공식문서(http://python-visualization.github.io/folium/)
: 서울지역 5대 강력범죄 데이터 분석(https://github.com/HyunSu-Jin/seoul_crime/blob/master/seoul_crime.ipynb)

Folium

```
import folium

m = folium.Map(location=[37.5502, 126.982], zoom_start=12)
folium.Marker(location=[37.5502, 126.982], popup="Marker A",
              icon=folium.Icon(icon='cloud')).add_to(m)
folium.Marker(location=[37.5411, 127.0107], popup="한남동",
              icon=folium.Icon(color='red')).add_to(m)
m
```



Plot.ly

- Plotly는 Interactive 그래프를 그려주는 라이브러리
- Scala , R , Python, JavaScript, MATLAB 등에서 사용할 수 있음.
- 시각화를 위해 D3.js를 사용하고 있음
- 사용해보면 사용이 쉽고, 세련된 느낌을 받음
- Plotly cloud라는 유료 모델이 있음.

```
!pip install plotly
```

참고 : https://plotly.com/python/ipython-notebook-tutorial/

pyecharts

- Baidu 에서 데이터 시각화를 위해 만든 Echarts.js의 파이썬 버전
- 정말 다양한 그래프들이 내장되어 있어 레포트를 작성할 때 좋다.
- 자바스크립트 기반이기 때문에 Interactive한 그래프를 그려줌.

```
pip3 install pyecharts
```

참고 : https://pyecharts.org/#/en-us/

Thank you.