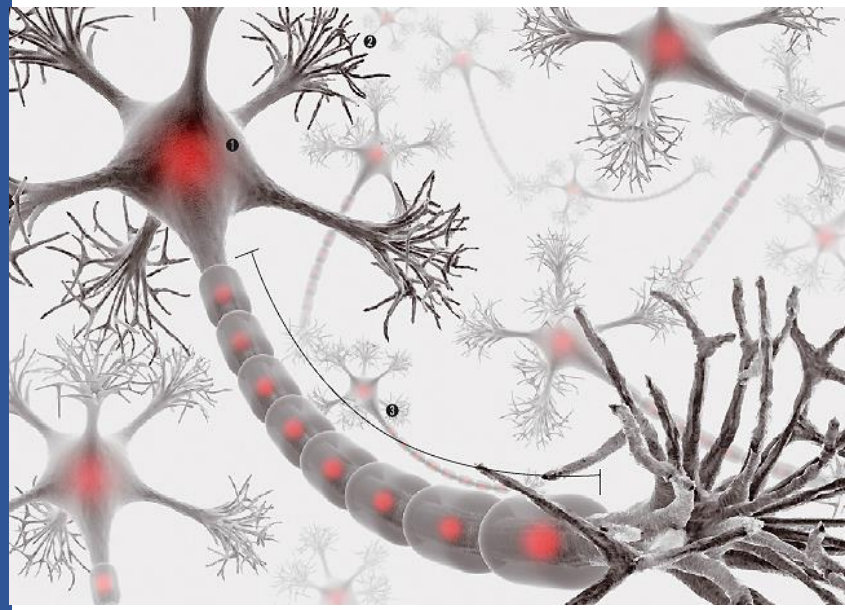
Matplotlib

학습 목표

- 그래프 라이브러리인 Matplotlib의 사용법을 이해한다.

주요 내용

- Matplotlib
- 기본 그래프
- 스타일 추가
- 카테고리 변수 그리기
- 여러 subplot 그리기
- Plot 컴포넌트
- 텍스트 처리
- 이미지 그리기



https://matplotlib.org/tutorials/introductory/pyplot.html

Matplotlib

Matplotlib는 MATLAB과 비슷하게 데이터를
그리기 위한 파이썬 그래프 함수 라이브러리

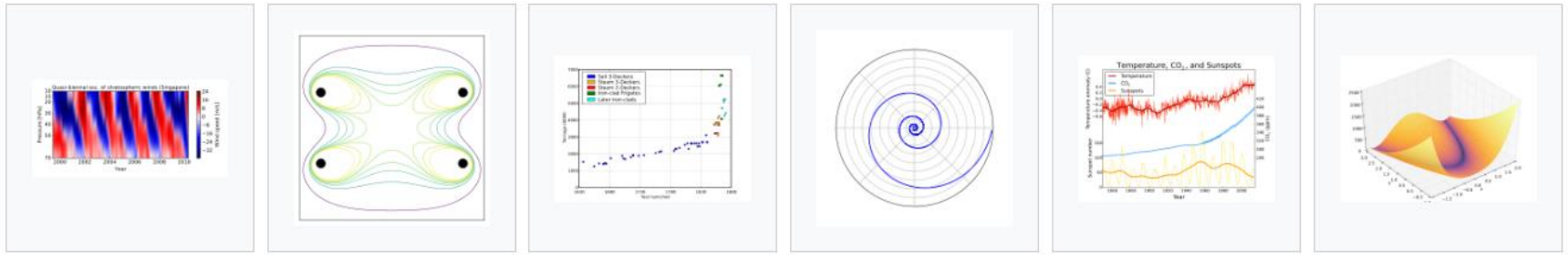


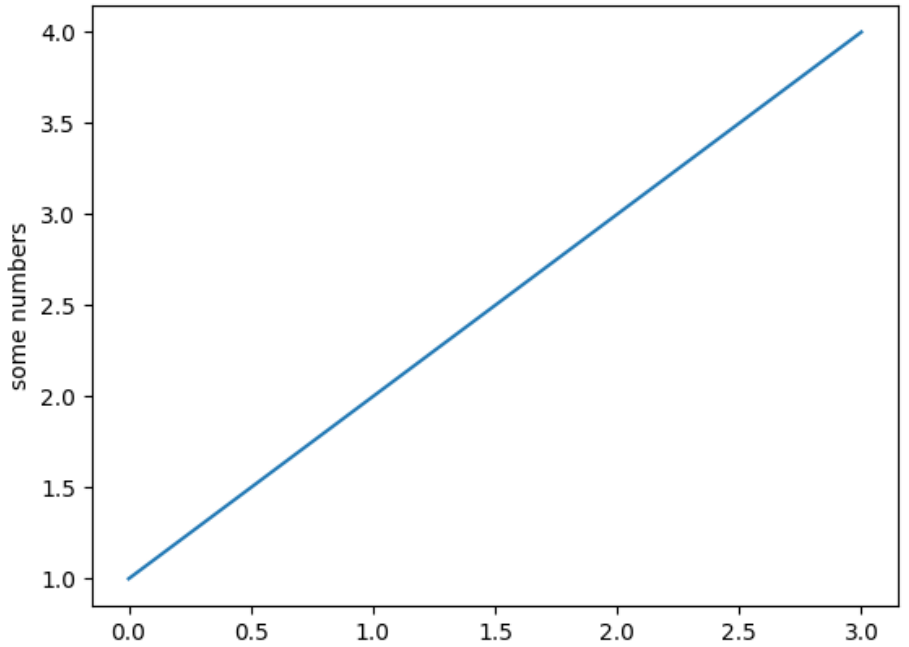
Image plot | Contour plot | Scatter plot | Polar plot | Line plot | 3-D plot

https://matplotlib.org/gallery.html

기본 그래프

```
import matplotlib.pyplot as plt
plt.plot([1, 2, 3, 4])
plt.ylabel('some numbers')
plt.show()
```

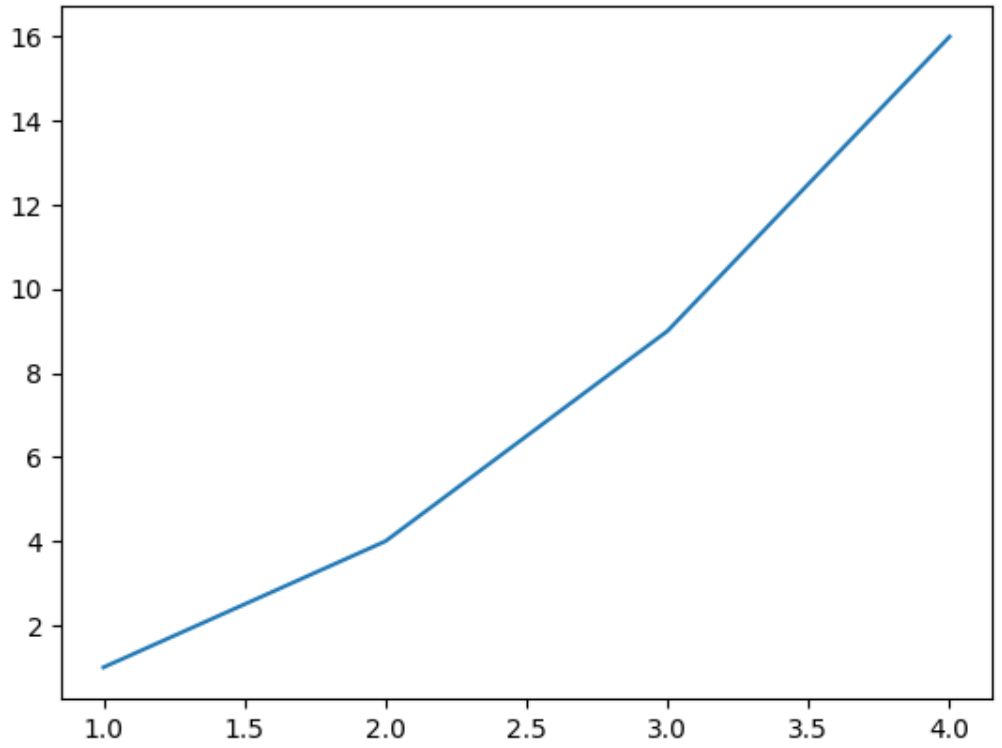
- Plot 함수의 리스트가 1개면 y 값으로 간주



기본 그래프

```
plt.plot([1, 2, 3, 4], [1, 4, 9, 16])
```

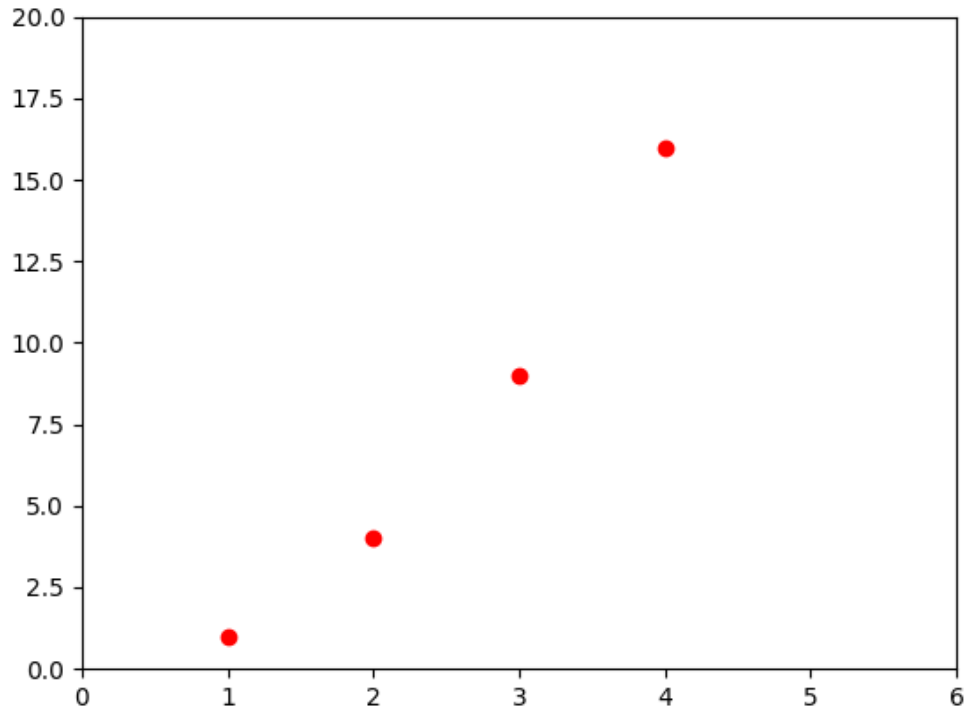
- Plot 함수의 리스트가 2개면 x값과 y값으로 간주



가로 세로 축 범위

```
plt.plot([1, 2, 3, 4], [1, 4, 9, 16], 'ro')
plt.axis([0, 6, 0, 20])
plt.show()
```

- `axis : [xmin, xmax, ymin, ymax]`



스타일 추가

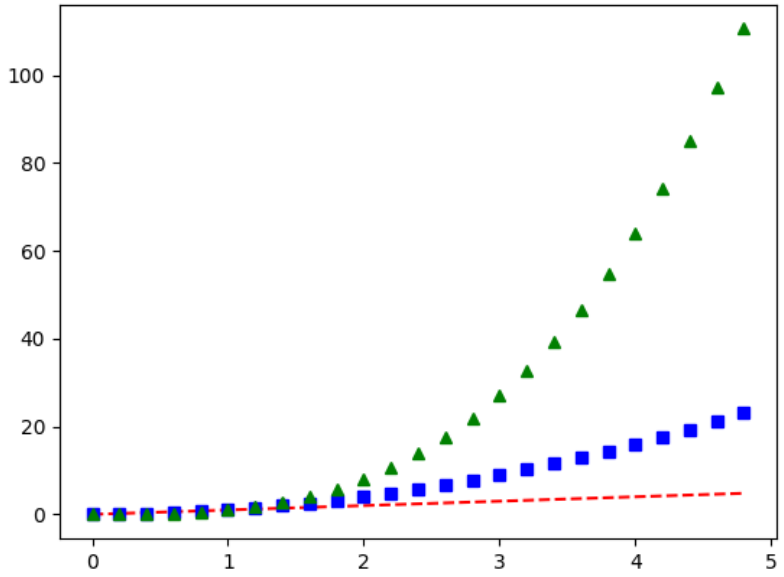
색깔 :
- 'r' (red), 'g' (green), 'b' (blue)
- 'c' (cyan), 'm' (magenta), 'y' (yellow)
- 'k' (black), 'w' (white)

마커:
- '.' (point marker), ',' (pixel marker), '*' (star marker), '+' (plus marker), 'x' (cross marker)
- 'o' (circle marker), 's' (square marker), 'h' (hexagon1 marker), 'H' (hexagon2 marker), 'd' (thin-diamond marker), 'D' (diamond marker)
- 'v' (triangle-down marker), '^' (triangle-up marker), '<' (triangle-left marker), '>' (triangle-right marker)
- '1' (triangle-down marker), '2' (triangle-up marker), '3' (triangle-left marker), '4' (triangle-right marker)

라인 스타일:
- '-' or 'solid'
- '--' or 'dashed'
- '|' (vline marker), '_' (hline marker)
- '-.' or 'dashdot'
- ':' or 'dotted'

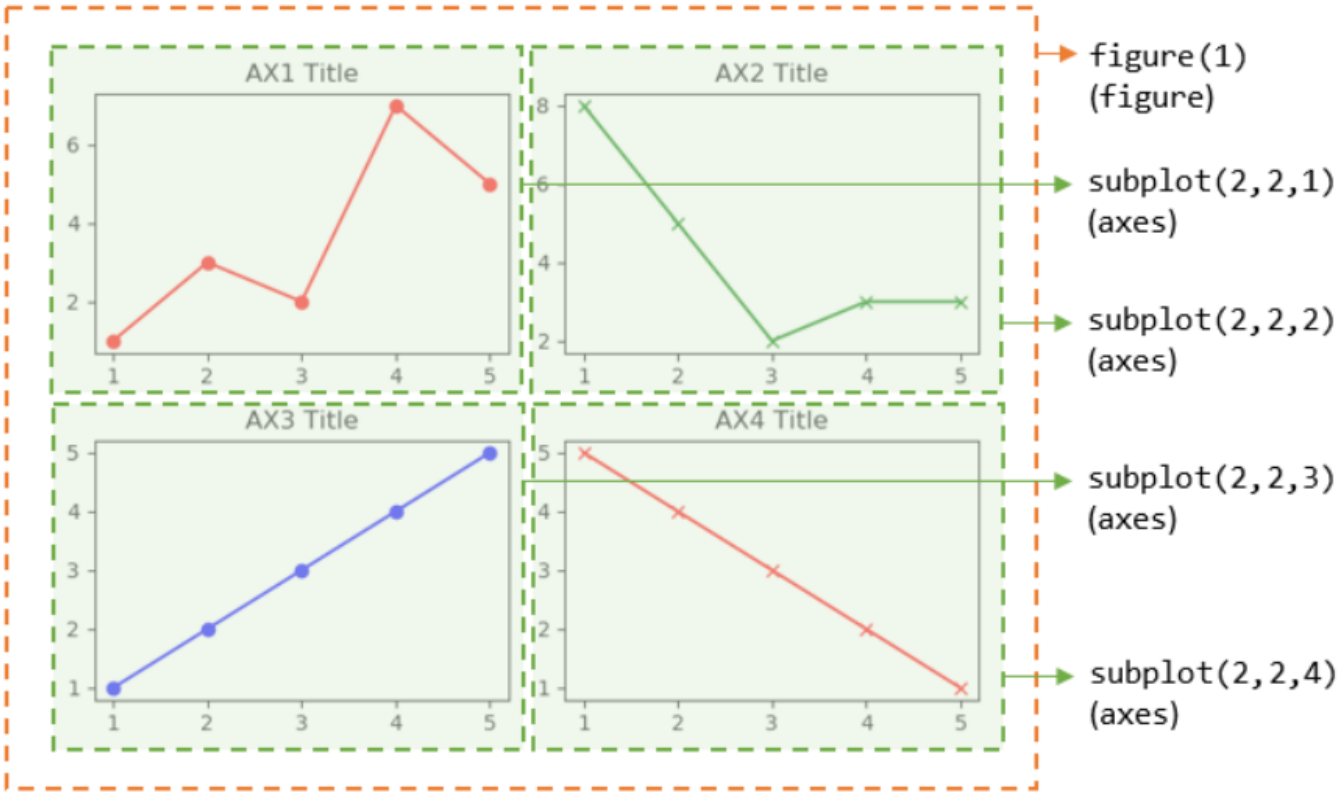
스타일 추가

```
import numpy as np

# evenly sampled time at 200ms intervals
t = np.arange(0., 5., 0.2)

# red dashes, blue squares and green triangles
plt.plot(t, t, 'r--', t, t**2, 'bs', t, t**3, 'g^')
plt.show()
```



Plot 컴포넌트

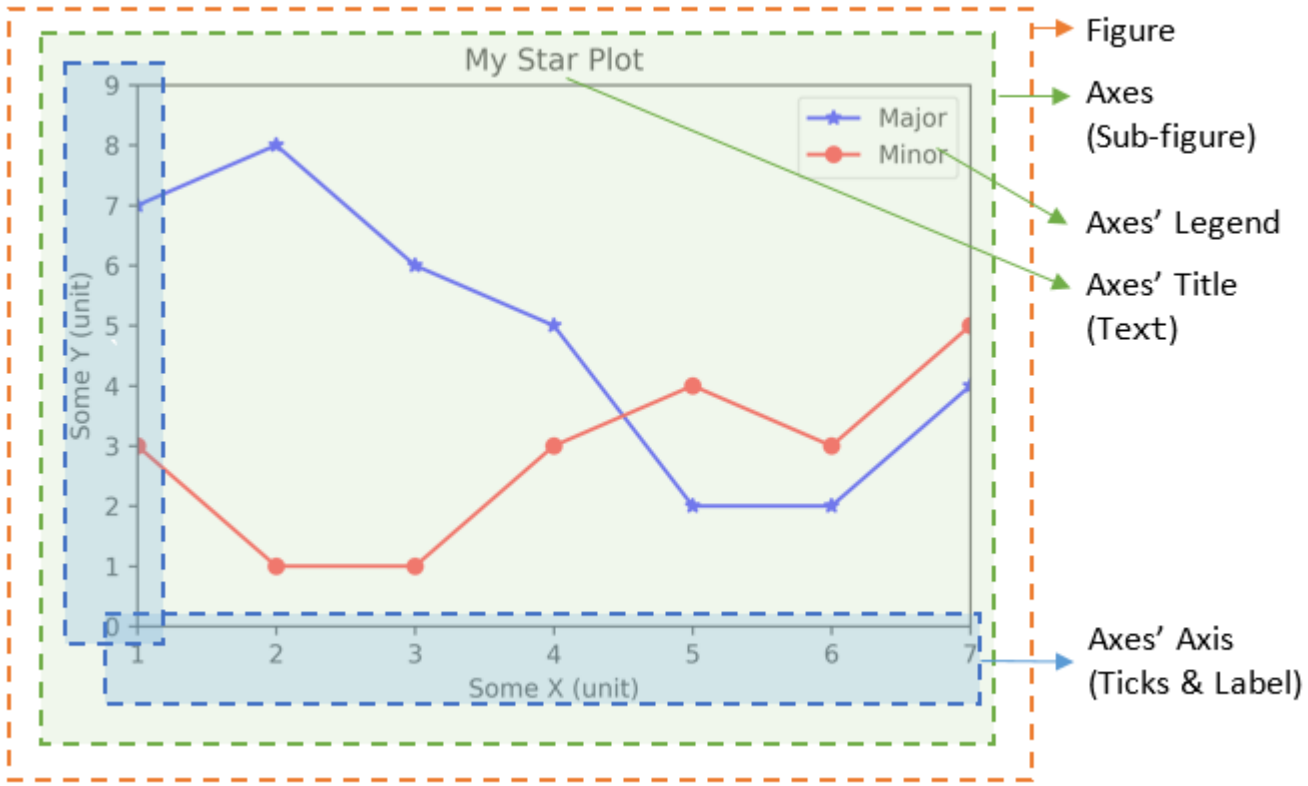
여러 Subplot을 그리드로 배열하기




```
fig1, ((ax1, ax2), (ax3, ax4)) = plt.subplots(2, 2)
```

Plot 컴포넌트

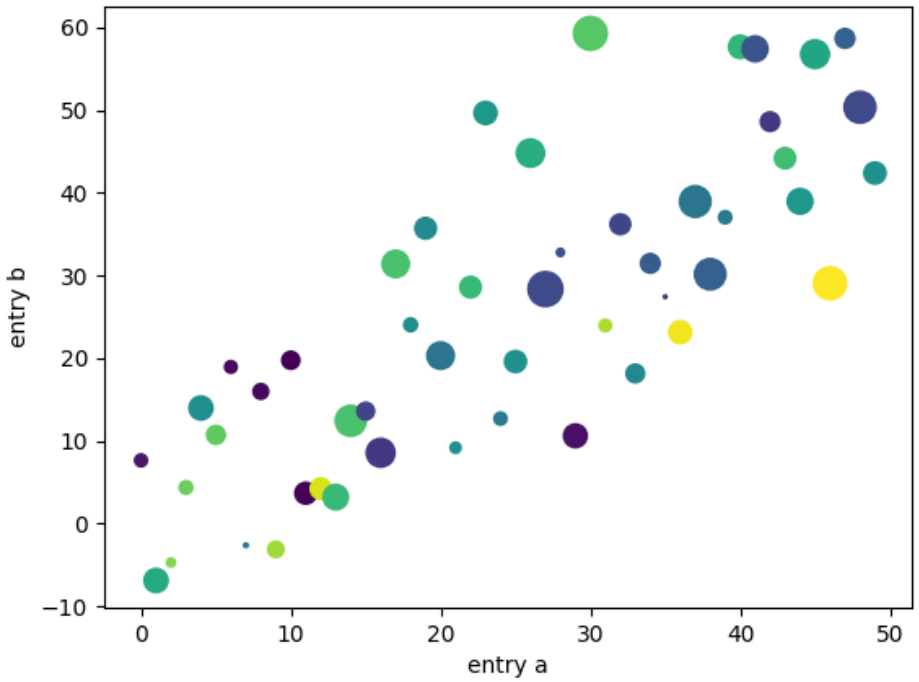
Subplot의 구성요소



Plotting with keyword strings

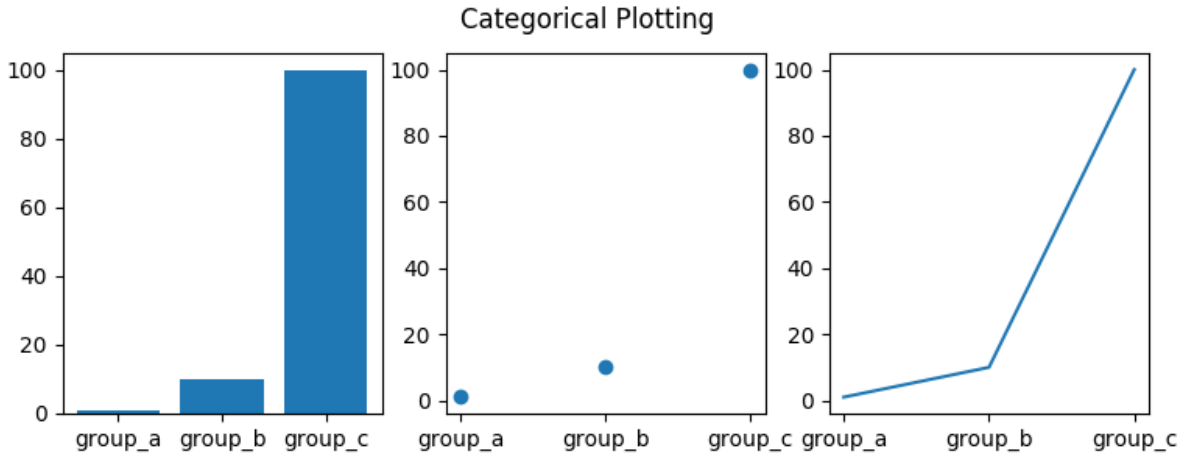
```
data = {'a': np.arange(50),
        'c': np.random.randint(0, 50, 50),
        'd': np.random.randn(50)}
data['b'] = data['a'] + 10 * np.random.randn(50)
data['d'] = np.abs(data['d']) * 100
plt.scatter('a', 'b', c='c', s='d', data=data)
plt.xlabel('entry a')
plt.ylabel('entry b')
plt.show()
```

c: marker color, s : marker size



Plotting with categorical variables

```
names = ['group_a', 'group_b', 'group_c']
values = [1, 10, 100]

plt.figure(figsize=(9, 3))

plt.subplot(131)
plt.bar(names, values)
plt.subplot(132)
plt.scatter(names, values)
plt.subplot(133)
plt.plot(names, values)
plt.suptitle('Categorical Plotting')
plt.show()
```

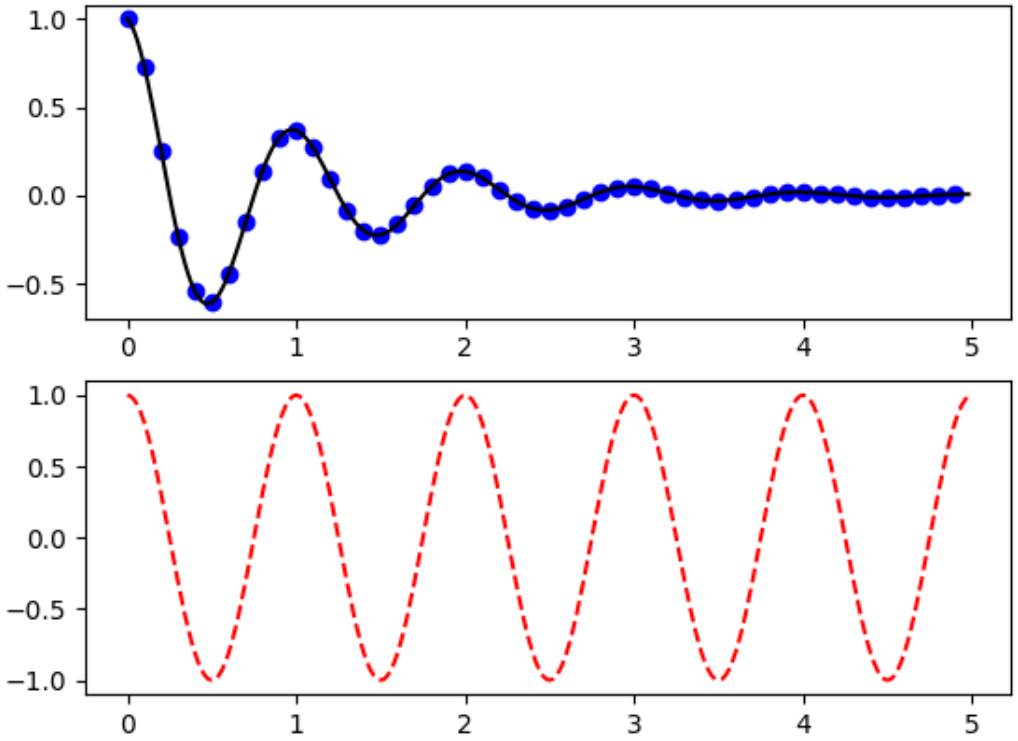


https://matplotlib.org/tutorials/introductory/pyplot.html

Working with multiple figures and axes

```
def f(t):
    return np.exp(-t) * np.cos(2*np.pi*t)

t1 = np.arange(0.0, 5.0, 0.1)
t2 = np.arange(0.0, 5.0, 0.02)

plt.figure()
plt.subplot(211)
plt.plot(t1, f(t1), 'bo', t2, f(t2), 'k’)

plt.subplot(212)
plt.plot(t2, np.cos(2*np.pi*t2), 'r--’)
plt.show()
```

$$f(t) = e^{-x}\cos(t * 2\pi)$$



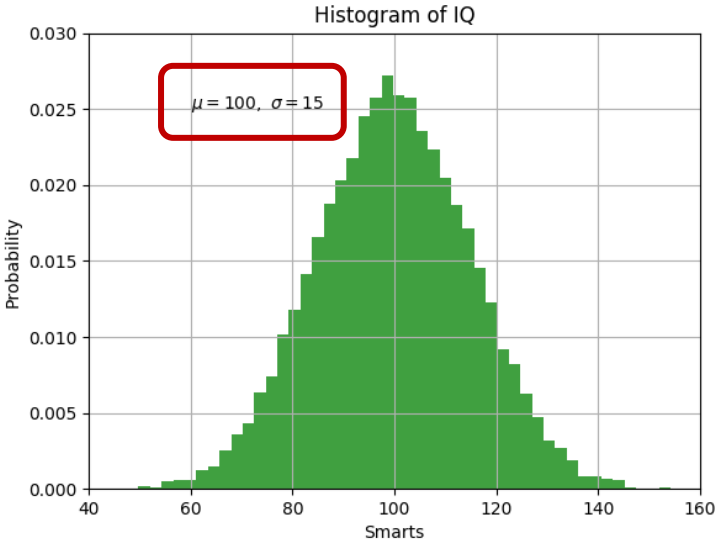
텍스트 처리

```
mu, sigma = 100, 15
x = mu + sigma * np.random.randn(10000)

# the histogram of the data
n, bins, patches = plt.hist(x, 50, density=1, facecolor='g', alpha=0.75)
```

Bin의 개수

확률 분포로 normalize할 지 여부

```
plt.xlabel('Smarts')
plt.ylabel('Probability')
plt.title('Histogram of IQ')
plt.text(60, .025, r'$\mu=100,\ \sigma=15$')
plt.axis([40, 160, 0, 0.03])
plt.grid(True)
plt.show()
```

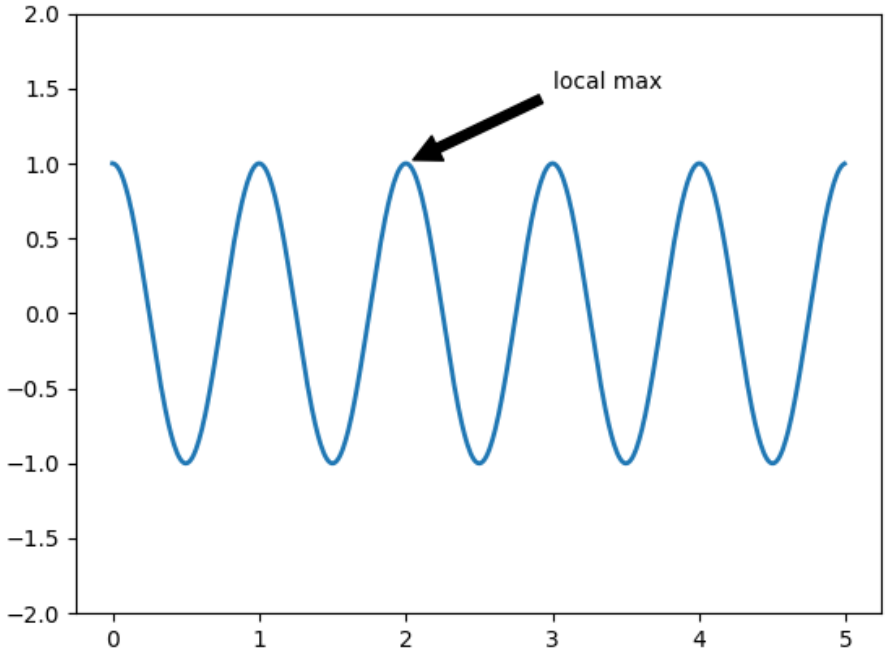


- 수식 표현에는 TeX equation expression을 사용

텍스트 처리

Annotating text

```
ax = plt.subplot(111)

t = np.arange(0.0, 5.0, 0.01)
s = np.cos(2*np.pi*t)
line, = plt.plot(t, s, lw=2)
                        lw : linewidth

plt.annotate('local max', xy=(2, 1), xytext=(3, 1.5),
                    arrowprops=(facecolor='black', shrink=0.05), )


plt.ylim(-2, 2)
plt.show()
```



annoate 함수

- xy : annotation 될 위치
- xytext : 텍스트 위치

이미지 그리기

```
import numpy as np
from scipy.misc import imread, imresize
import matplotlib.pyplot as plt

img = imread('assets/cat.jpg')
img_tinted = img * [1, 0.95, 0.9]

# 원본 이미지 나타내기
plt.subplot(1, 2, 1)
plt.imshow(img)

# 색변화된 이미지 나타내기
plt.subplot(1, 2, 2)

# imshow를 이용하며 주의할 점은 데이터의 자료형이
# uint8이 아니라면 이상한 결과를 보여줄 수도 있다는 것입니다.
# 그러므로 이미지를 나타내기 전에 명시적으로 자료형을 uint8로 형변환 해줍니다.

plt.imshow(np.uint8(img_tinted))
plt.show()
```

이미지 그리기

```
import matplotlib.pyplot as plt
import matplotlib.patches as patches
import matplotlib.cbook as cbook

image_file = cbook.get_sample_data('grace_hopper.png')
image = plt.imread(image_file)

fig, ax = plt.subplots()
im = ax.imshow(image)
patch = patches.Circle((260, 200), radius=200, transform=ax.transData)
im.set_clip_path(patch)

ax.axis('off')
plt.show()
```

Thank you!