데이터 시각화 (2024)

데이터과학부 정진명
(jmjung@suwon.ac.kr, 글로벌경상관 918호)

11 주차

Contents

- tick param
- text
- legend

tick_params

Tick_params

이미 **주어진 tick**의 각종 style을 수정하는 명령어:
tick_params

axis: x, y, both
direction: in, out, inout
length: tick length
width: tick width
pad: distance between tick and tick_label
labelsize: tick label size
color: tick color
labelcolor: tick label color

matplotlib.axes.Axes.tick_params

Axes.tick_params(*self*, *axis='both'*, ***kwargs*)

Change the appearance of ticks, tick labels, and gridlines.

Parameters:

axis : {'x', 'y', 'both'}, optional
Which axis to apply the parameters to.

Other Parameters:

axis : {'x', 'y', 'both'}
Axis on which to operate; default is 'both'.

reset : bool
If *True*, set all parameters to defaults before processing other keyword arguments. Default is *False*.

which : {'major', 'minor', 'both'}
Default is 'major'; apply arguments to *which* ticks.

direction : {'in', 'out', 'inout'}
Puts ticks inside the axes, outside the axes, or both.

length : float
Tick length in points.

width : float
Tick width in points.

color : color
Tick color; accepts any mpl color spec.

pad : float
Distance in points between tick and label.

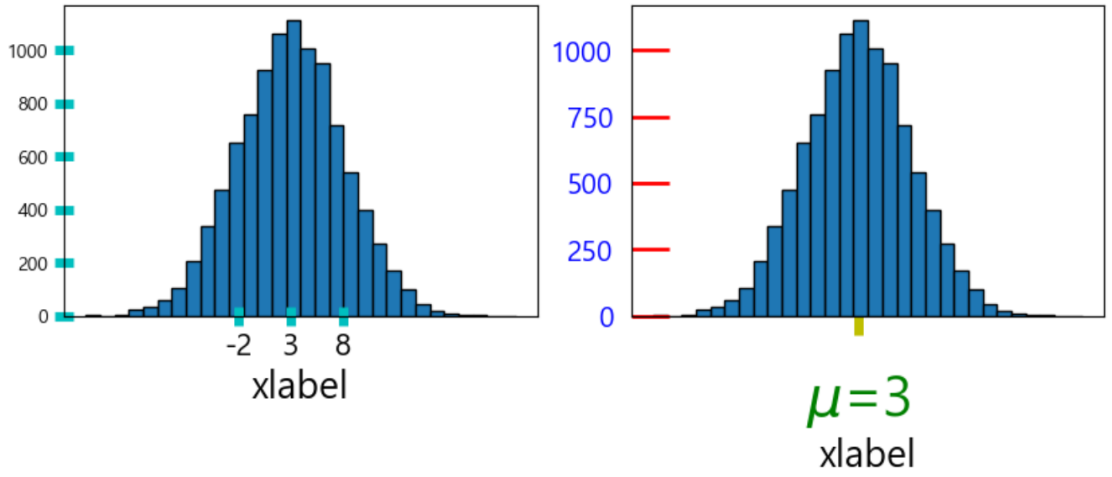
labelsize : float or str
Tick label font size in points or as a string (e.g., 'large').

labelcolor : color
Tick label color; mpl color spec.

colors : color
Changes the tick color and the label color to the same value: mpl color spec.

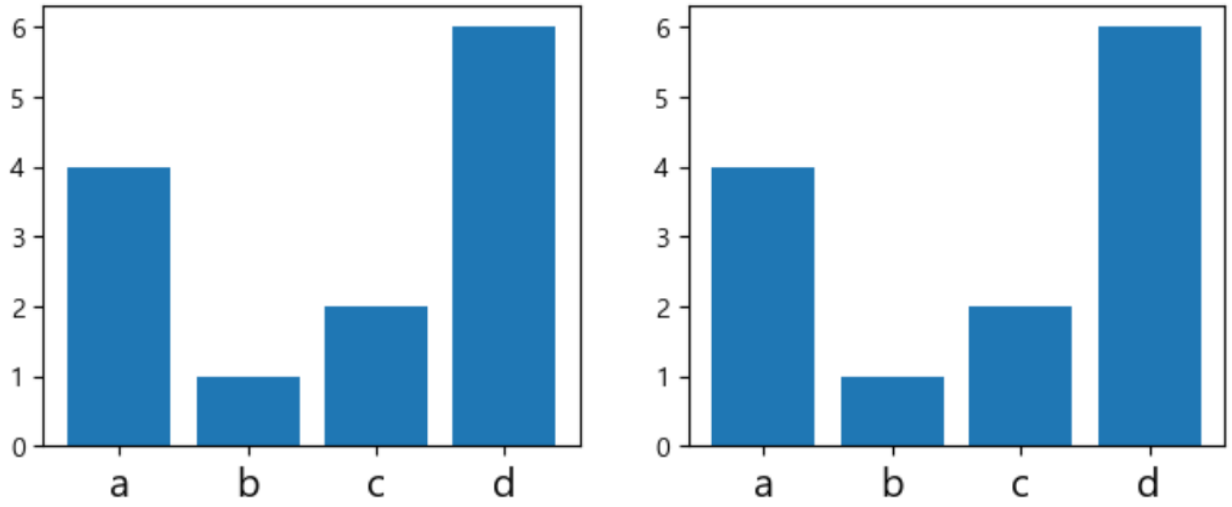
Tick_params 예제1

```
_=ax1.tick_params(axis='both',
                  direction='inout',
                  width=5,
                  length=10,
                  color='c')

_=ax2.tick_params(axis='x',
                  direction='out',
                  width=5,
                  length=10,
                  pad=20,
                  color='y',
                  labelcolor='g',labelsize=30)

_=ax2.tick_params(axis='y',
                  direction='in',
                  width=2, length=20,
                  pad=10,
                  color='r',
                  labelcolor='b',
                  labelsize=15)
```



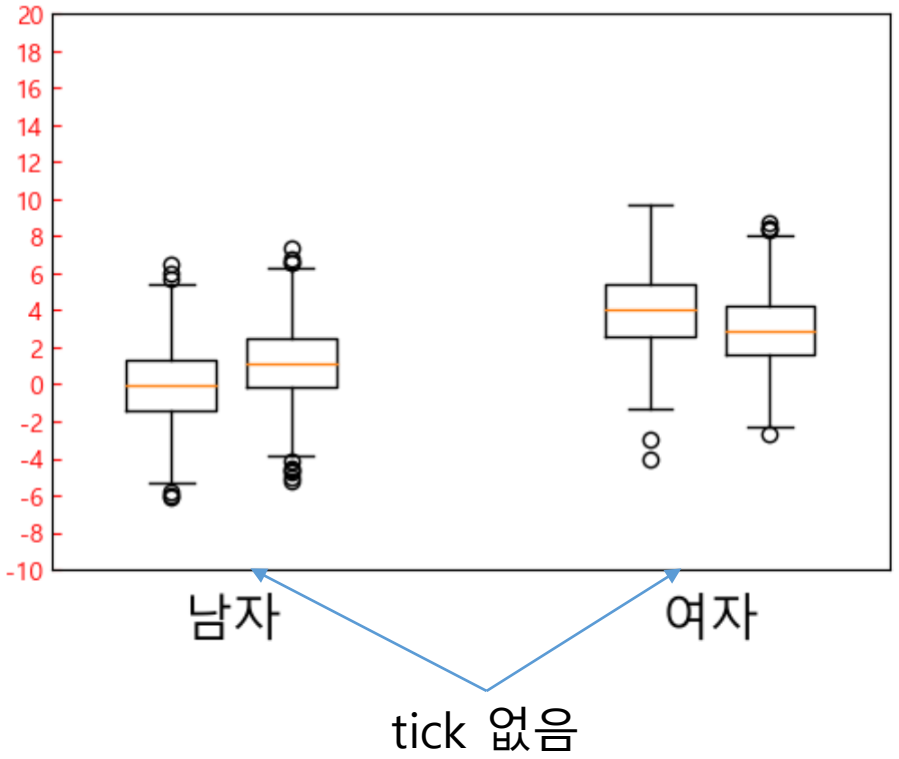
Tick_params 예제2

```
fig=plt.figure(figsize=(8,3), dpi=100)
ax1, ax2=fig.subplots(1,2)

sr1=pd.Series([4,1,2,6], index=list('abcd'))

## 방법 1
ax1.bar(sr1.index, sr1)
ax1.set_xticks(range(len(sr1)), labels=sr1.index, fontsize=15)

## 방법 2
ax2.bar(sr1.index, sr1)
ax2.tick_params(axis='x',labelsize=15)
```



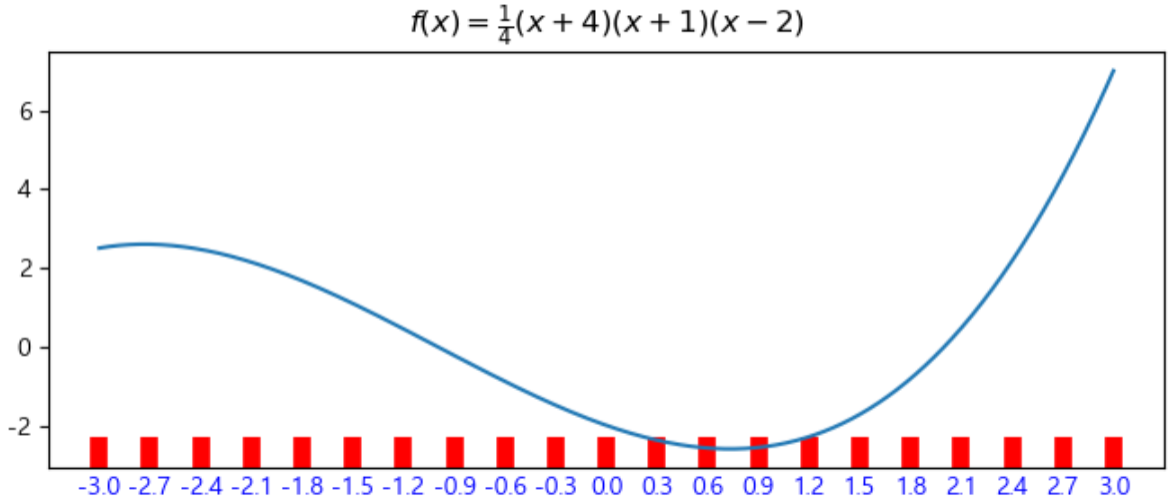
실습1

set_ticks와 tick_params를 활용하여 아래 그림을 그리시오



실습2

set_ticks와 tick_params를 활용하여 아래 그림을 그리시오
(x=0.6, 0.9에서의 tick이 함수 곡선을 넘어가도록 세팅하시오)



text

Text

ax.text(x, y, 'string', fontsize, ha, va)

ha: horizontal alignment
- left (default)
- center
- right

va: vertical alignment
- top
- baseline (default) (bottom 과 center 중간)
- center
- bottom

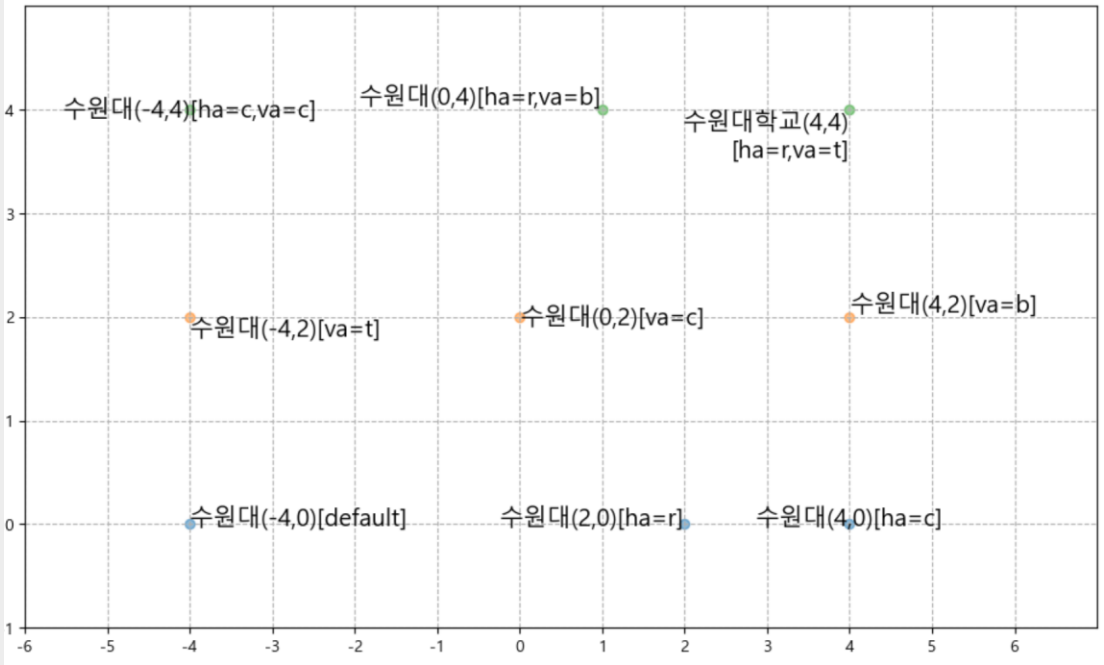
→ Text box의 해당 alignment를 x,y 좌표에 맞춤

Ex)

```
ax1.text(-2,6,'수원대(-2,6,va=c,ha=c)', fontsize=20, ha='center', va='center')
```

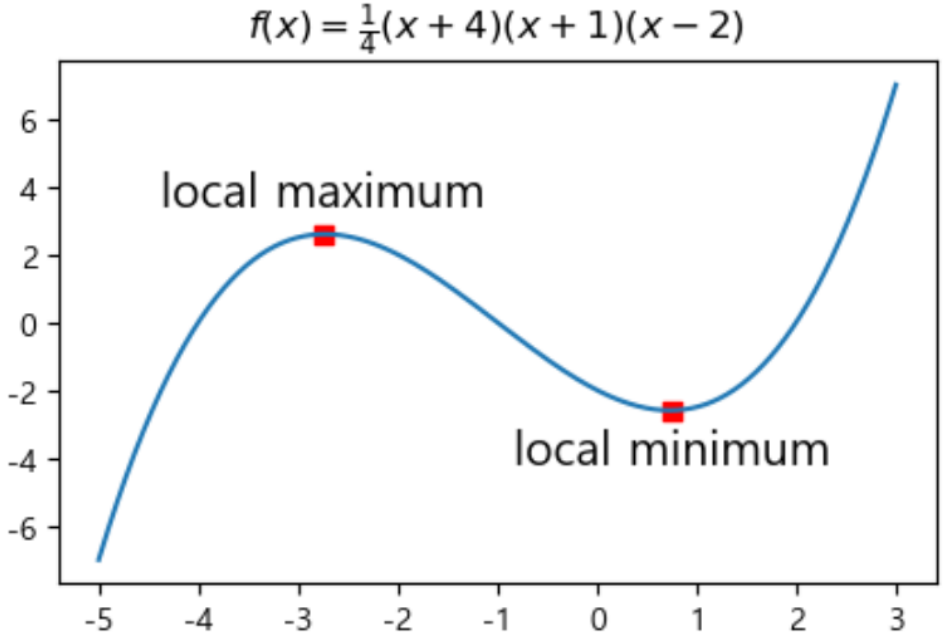
Text 예제1

```
fig=plt.figure(figsize=(12,7), dpi=100)
ax1=fig.subplots()

_=ax1.text(-4,0,'수원대(-4,0)[default]', fontsize=15) # va=baseline, ha=left
_=ax1.text( 2,0,'수원대(2,0)[ha=r]'    , ha='right', fontsize=15) # va=baseline
_=ax1.text( 4,0,'수원대(4,0)[ha=c]'    , ha='center', fontsize=15) # va=baseline
_=ax1.scatter([-4,2,4],[0,0,0], alpha=0.5)

_=ax1.text(-4,2,'수원대(-4,2)[va=t]', fontsize=15, va='top') # ha=left
_=ax1.text( 0,2,'수원대(0,2)[va=c]', fontsize=15, va='center') # ha=left
_=ax1.text( 4,2,'수원대(4,2)[va=b]', fontsize=15, va='bottom') # ha=left
_=ax1.scatter([-4,0,4],[2,2,2], alpha=0.5)

_=ax1.text(-4,4,'수원대(-4,4)[ha=c,va=c]', fontsize=15, ha='center', va='center')
_=ax1.text( 1,4,'수원대(0,4)[ha=r,va=b]', fontsize=15, ha='right', va='bottom')
_=ax1.text( 4,4,'수원대학교(4,4)\n[ha=r,va=t]', fontsize=15, ha='right', va='top')
_=ax1.scatter([-4,1,4],[4,4,4], alpha=0.5)

_=ax1.set_xticks(np.arange(-6,7))
_=ax1.set_yticks(np.arange(-1,5))
_=ax1.grid(ls='--')    ## grid 사용
_=ax1.set_xlim(-6,7)
_=ax1.set_ylim(-1,5)
```



실습3

```
fig=plt.figure(figsize=(5,5), dpi=100)
ax1=fig.subplots()

X=np.linspace(-5,3,1000)
Y=0.25*(X+4)*(X+1)*(X-2)

_=ax1.plot(X,Y)
_=ax1.set_title('$f(x)=\\frac{1}{4}(x+4)x(x+1)(x-2)$')

max_x=-2.75
min_x=0.75
```



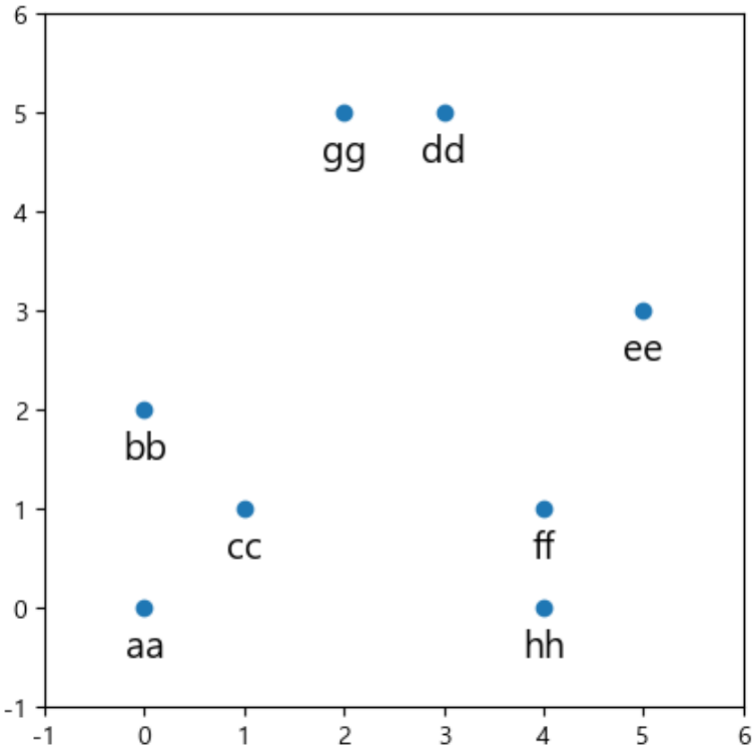
- 오른쪽 그래프의 local maximum과 minimum의 x좌표는 각각 -2.75, 0.75 이다.
- 오른쪽 그래프와 같이, Local maximum과 minimum에 빨간 사각점을 찍고, 함수 곡선을 가리지 않도록 text를 넣으시오

Text 예제 2

'data/11w_d3.txt ' 을 읽어서 x,y를 오른쪽과 같이 scatter 하고 index를 해당 x,y 아래에 넣으시오

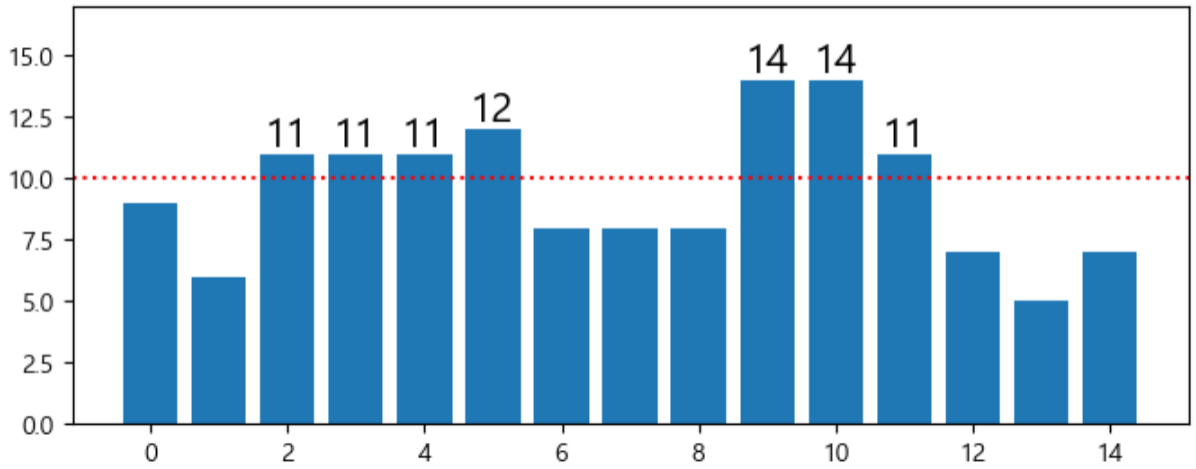
```
fig=plt.figure(figsize=(10,5), dpi=100)
axs=fig.subplots(1,2)

d1=pd.read_table('data/11w_d3.txt', index_col=0, sep='\t')
d1

## axs[0], scatter 함수 한번 호출
_=axs[0].scatter(d1['x'],d1['y'])
for ind, x, y in zip(d1.index, d1['x'], d1['y']):
    _=axs[0].text(x,y-0.2,ind, ha='center',va='top', fontsize=15)

_=axs[0].set_xlim(-1,6)
_=axs[0].set_ylim(-1,6)

## axs[1], scatter 함수 각 점의 수만큼 호출
for ind, x, y in zip(d1.index, d1['x'], d1['y']):
    _=axs[1].scatter(x,y, color='b')
    _=axs[1].text(x,y-0.2,ind, ha='center',va='top', fontsize=15)

_=axs[1].set_xlim(-1,6)
_=axs[1].set_ylim(-1,6)
```

	x	y
aa	0	0
bb	0	2
cc	1	1
dd	3	5
ee	5	3
ff	4	1
gg	2	5
hh	4	0



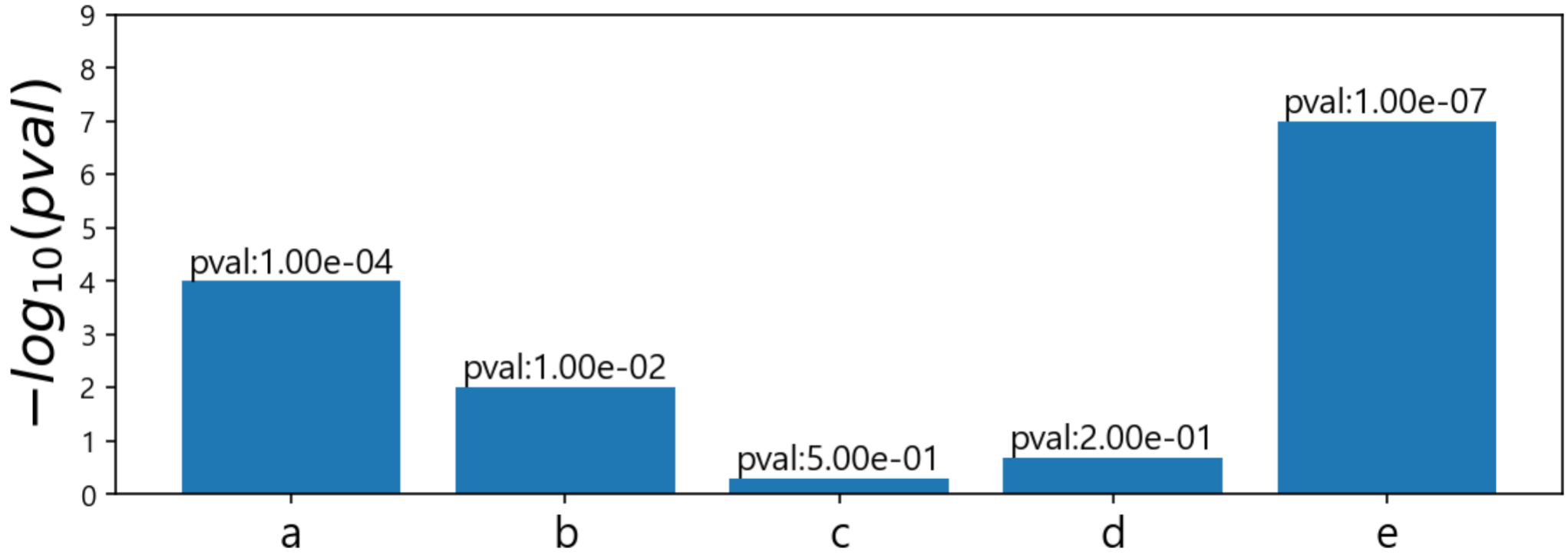
실습4

랜덤하게 생성되는 15개의 정수 Y를 bar그래프로 그리고,
10이상인 값에 대해서만 bar위에 해당 y값을 text로 넣으시오



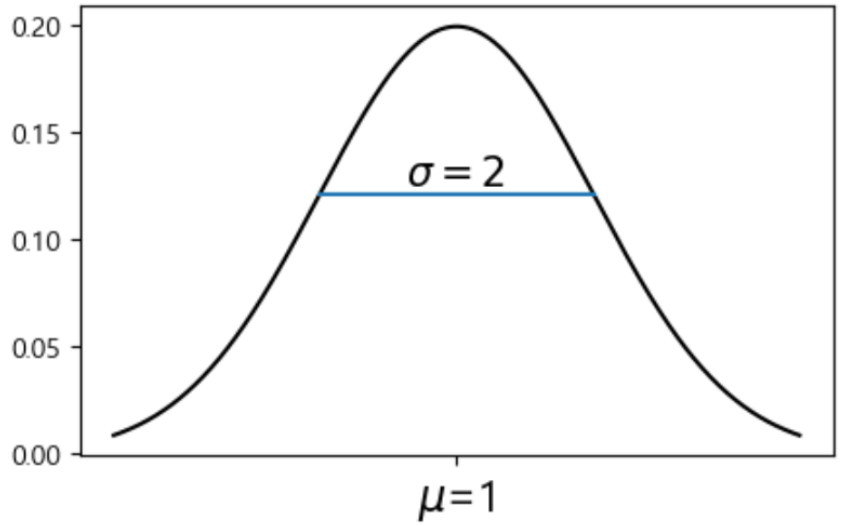
실습5

- p_val_sr에 있는 다섯개의 p_value를 –log10 scale로 bar plot하고, bar 위에 원래 p_value를 text로 표시하시오
  - step1) bar 그리기
  - step2) y label 수정
  - step3) text 넣기
  - step4) y lim 조정



실습6

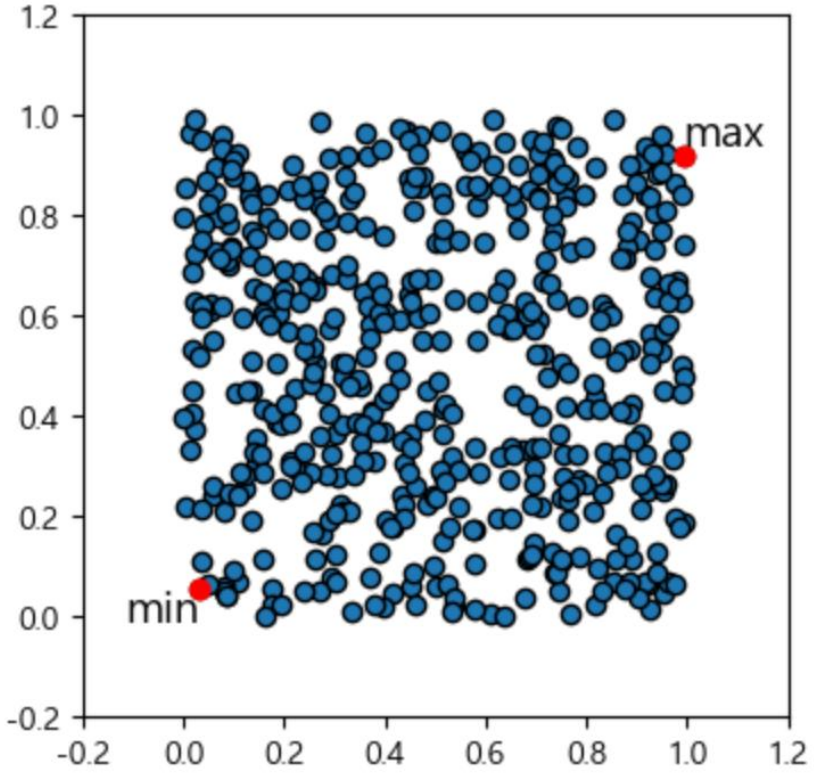
```
fig=plt.figure(figsize=(5,3), dpi=100)
ax=fig.subplots()

def pdf(X, mu, sigma):
    a = 1/(sigma * np.sqrt(2*np.pi))
    b = -1/(2*(sigma**2))
    return a * np.exp(b * ((X - mu)** 2))

m1=1
s1=2

X = np.linspace(m1-5, m1+5, 1000)
_=ax.plot(X, pdf(X, m1, s1), color = 'k')
```



위 코드에 추가하여 오른쪽과 같이 그리시오
추가된 직선의 x 범위: m1-2 < X < m1+2

실습7

- Uniform 분포 (0~1)에서 뽑은 500개의 값을 각각 X,Y로 하여 scatter plot하시오
- x+y의 값이 가장 큰 점과 가장 작은 점을 빨간색으로 scatter 한 뒤, 각각 max, min의 text를 추가하시오
- text 가 점을 가리지 않도록 하시오



legend

legend (범례)

- **plot, scatter, hist, bar** 함수에서는 다음 두가지 step에 따라서 **legend**를 사용할 수 있다 (첫번째 방법).
  1. 그림을 그리는 함수에 **label** parameter를 추가한다.
  2. **ax.legend()**를 호출한다.

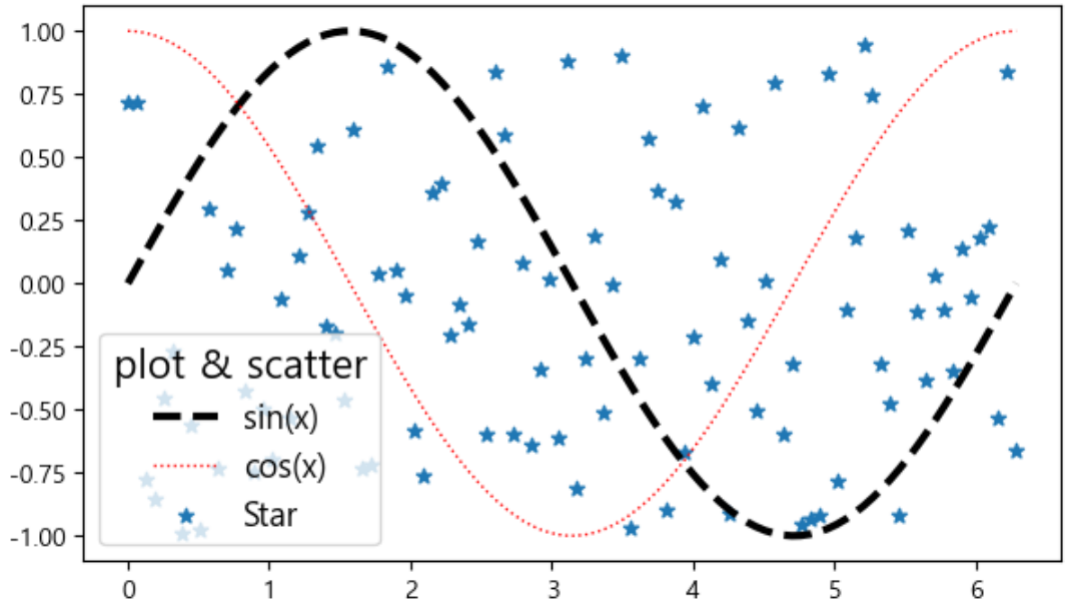
- **boxplot** 함수에서는 legend를 직접 명시하는 복잡한 방법을 사용해야 한다. (두번째 방법).

legend 사용 – 첫번째 방법 (plot, scatter)

```
fig=plt.figure(figsize=(7,4), dpi=100)
ax1=fig.subplots()

X=np.linspace(0,2*np.pi, 100)
Y1=np.sin(X)
Y2=np.cos(X)
Y3=np.random.uniform(-1,1,size=100)

_=ax1.plot(X,Y1, c='k', lw=3, ls='--',label='sin(x)')
_=ax1.plot(X,Y2, c='r', lw=1, ls=':',label='cos(x)')
_=ax1.scatter(X,Y3, marker='*' , label='Star')
_=ax1.legend(title='plot & scatter',fontsize=13, title_fontsize=16)
```

1. 그림을 그리는 함수에 **label**을 추가한다.

2. legend함수를 호출한다.

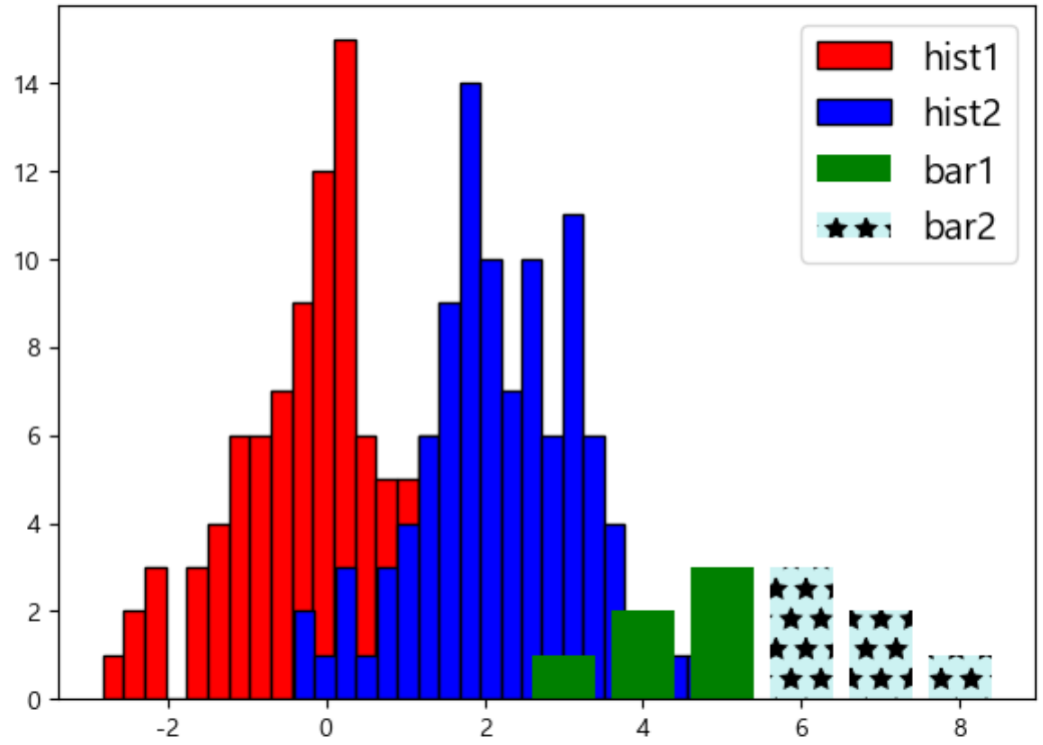


legend 사용 – 첫번째 방법 (hist, bar)

```
fig=plt.figure(figsize=(7,5), dpi=100)
ax1=fig.subplots()

_=ax1.hist(np.random.normal(0,1,100), bins=20, edgecolor='k', color='r', label='hist1')
_=ax1.hist(np.random.normal(2,1,100), bins=20, edgecolor='k', color='b', label='hist2')
_=ax1.bar([3,4,5],[1,2,3], color='g',label='bar1')
_=ax1.bar([6,7,8],[3,2,1], color='c',hatch='*', alpha=0.2,label='bar2')
_=ax1.legend(fontsize=15)
```

1. 그림을 그리는 함수에 **label**을 추가한다.

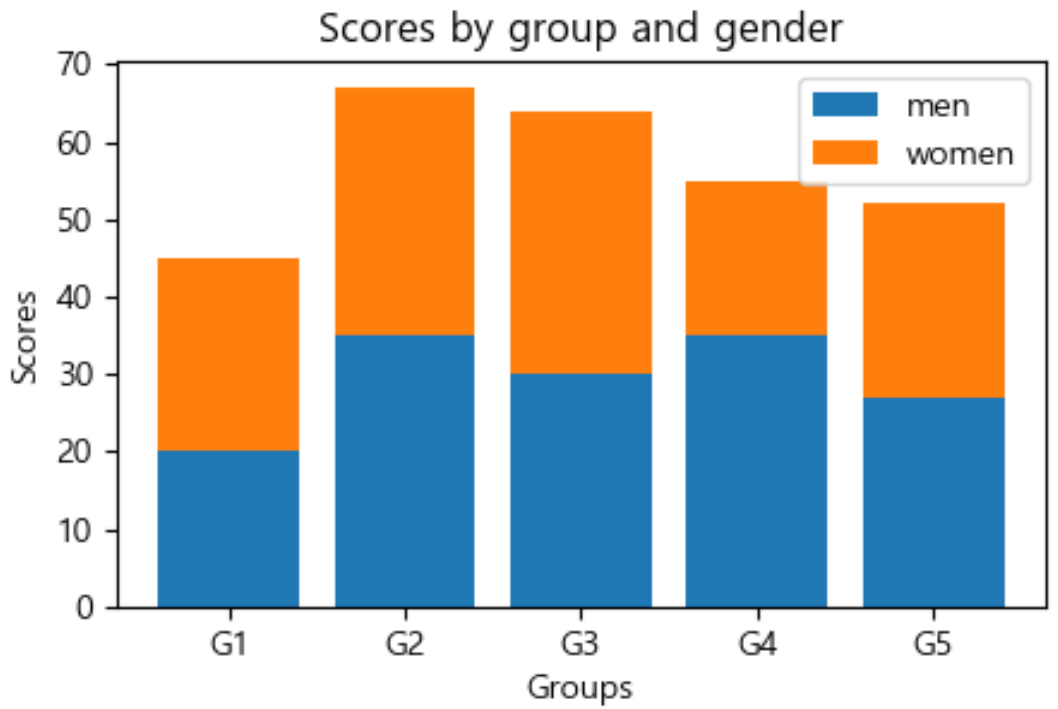
2. legend함수를 호출한다.



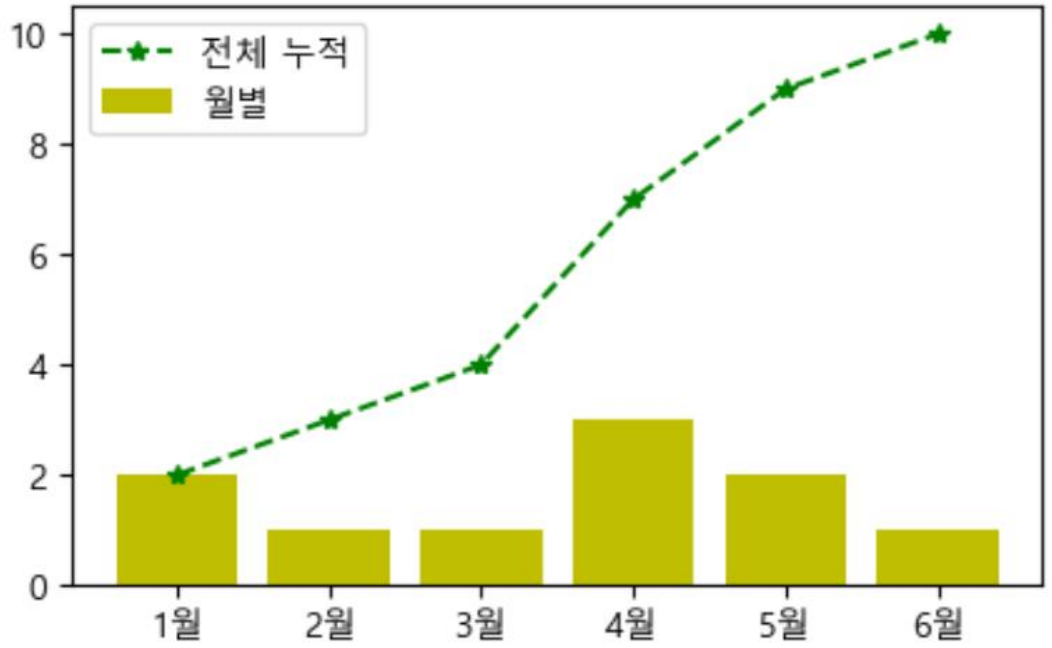
실습8

```
fig=plt.figure(figsize=(5,3), dpi=100)
ax1=fig.subplots()

mens = (20, 35, 30, 35, 27)
womens = (25, 32, 34, 20, 25)
```

- 위 코드를 이어서 작성하여 오른쪽 그림처럼 그리시오

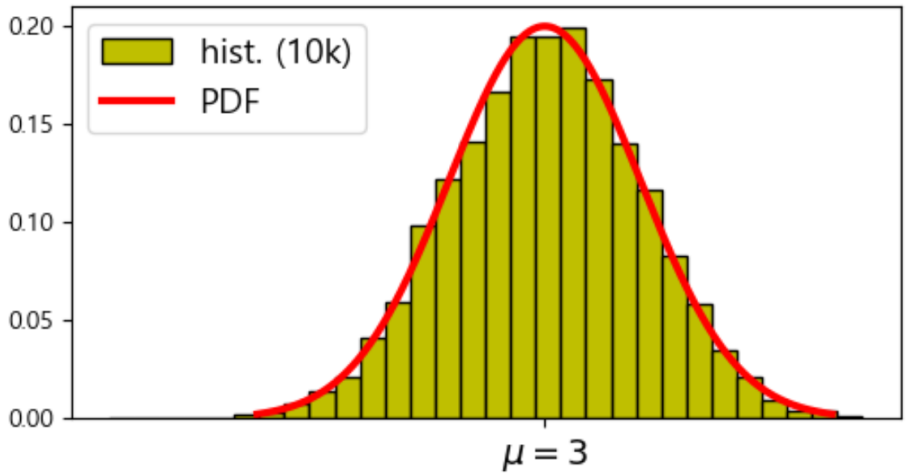


실습9



실습10

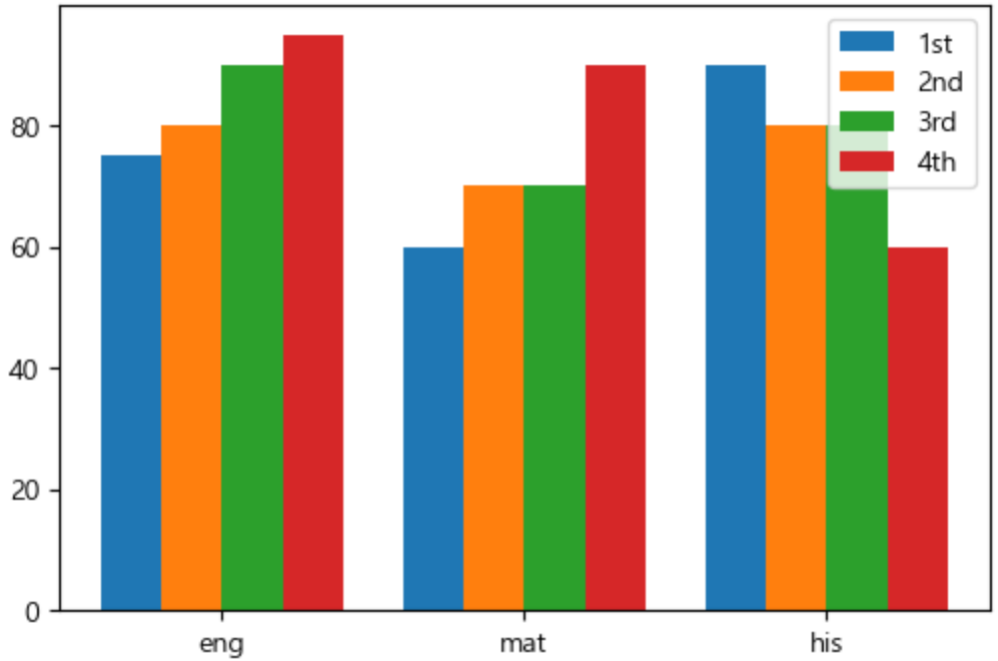
- 평균이 3, 표준편차가 2인 정규분포에서 뽑은 10000개 값으로 histogram을 그리시오 (density=True)
- 해당 분포의 PDF를 그리시오 (-3< X < 9)
- Title, tick label, legend를 추가하시오



실습11

	eng	mat	his
1st	75	60	90
2nd	80	70	80
3rd	90	70	80
4th	95	90	60

위와 같이 11w_d4.txt를 읽어서
오른쪽 그림처럼 그리시오

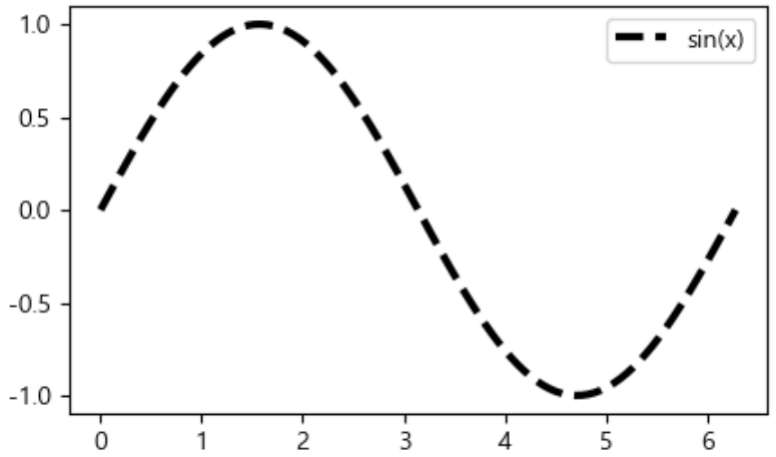


legend의 사용 - 두번째 방법 (plot)

- 시각화 함수의 리턴 객체를 변수(l1)로 받는다
- 이제까지는 dummy variable(_)로 받거나, 받지 않았음

```
fig=plt.figure(figsize=(10,6), dpi=100)
ax=fig.subplots()

X=np.linspace(0,2*np.pi, 100)
Y1=np.sin(X)

l1=ax.plot(X,Y1, c='k', lw=3, ls='--')
l1
l1[0]
_=ax.legend(handles=(l1[0],), labels=('sin(x)',))
```

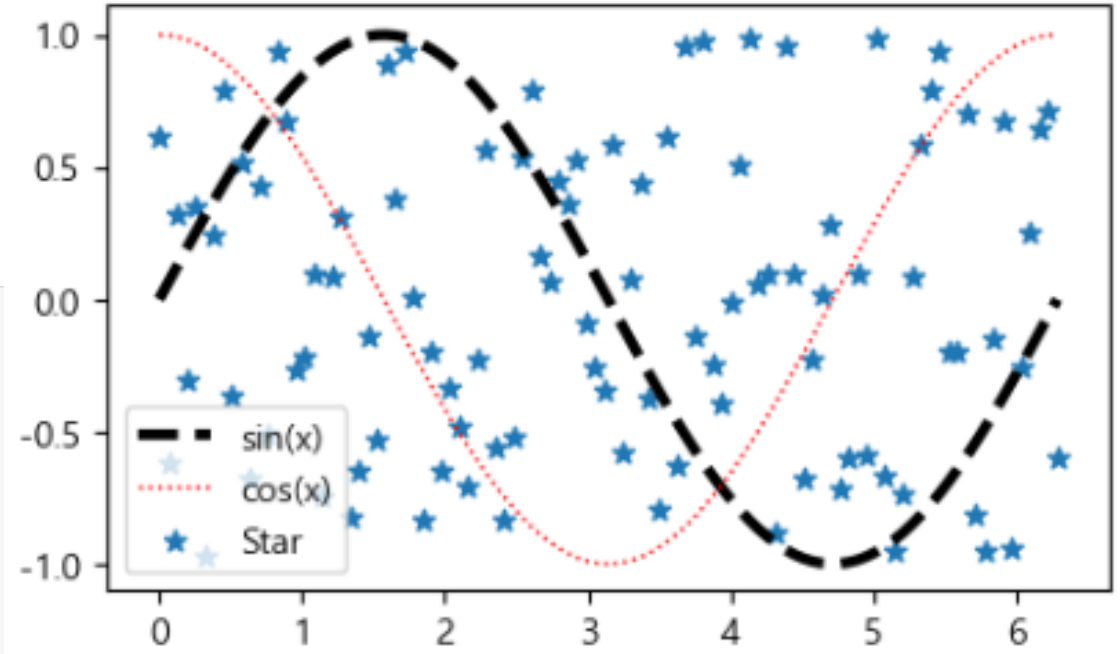
```
[<matplotlib.lines.Line2D at 0x19f38a604c0>]

<matplotlib.lines.Line2D at 0x19f38a604c0>
```



- legend의
  1) handles와 2) labels 파라미터 사용
- - handles: 수행한 시각화 함수의 리턴 객체
  - labels: legend text
- 시각화 함수마다 리턴 객체의 구성이 다르기 때문에, 리턴 객체의 주소를 잘 선정하여 handles에 넣어야 한다.
- 튜플 형태로 넣는다 (element가 하나일 경우 쉼표 (,) 꼭 추가)

legend의 사용 - 두번째 방법 (plot, scatter)

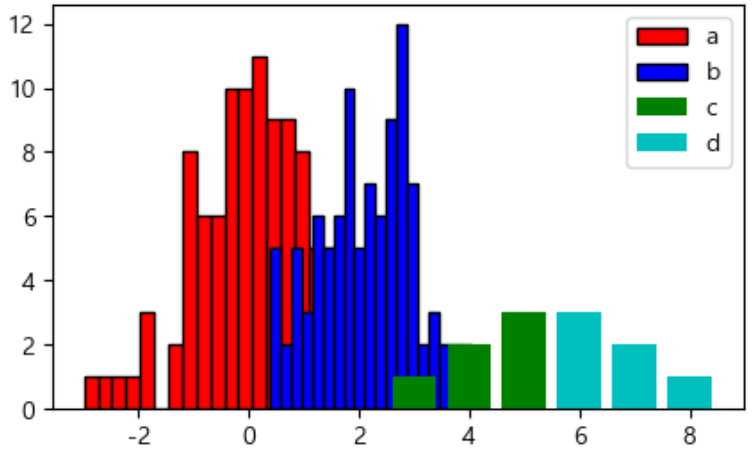
```
fig=plt.figure(figsize=(5,3), dpi=100)
ax1=fig.subplots()

X=np.linspace(0,2*np.pi, 100)
Y1=np.sin(X)
Y2=np.cos(X)

plot1=ax1.plot(X,Y1, c='k', lw=3, ls='--')
plot2=ax1.plot(X,Y2, c='r', lw=1, ls=':')
scat1=ax1.scatter(X,np.random.uniform(-1,1,100), marker='*')

_=ax1.legend(handles=(plot1[0], plot2[0], scat1), labels=('sin(x)', 'cos(x)', 'Star'))
```



- legend에 사용할 정보가 있는 주소에 접근해서 그 정보를 가져와야 한다. (plot: var[0], scatter: var)
- 튜플의 element가 여러 개일 경우에는 마지막에 쉼표(,) 추가하지 않아도 된다

legend의 사용 - 두번째 방법 (hist, bar)

```
fig=plt.figure(figsize=(5,2), dpi=100)
ax1=fig.subplots()

hist1=ax1.hist(np.random.normal(0,1,100), bins=20, edgecolor='k', color='r')
hist2=ax1.hist(np.random.normal(2,1,100), bins=20, edgecolor='k', color='b')
bar3=ax1.bar([3,4,5],[1,2,3], color='g')
bar4=ax1.bar([6,7,8],[3,2,1], color='c')
hist2
hist2[2]
bar3
_=ax1.legend(handles=(hist1[2],hist2[2],bar3,bar4), labels=(list('abcd')))
```



- legend에 사용할 정보가 있는 주소에 접근해서 그 정보를 가져와야 한다.
  (hist: var[2], bar: var)

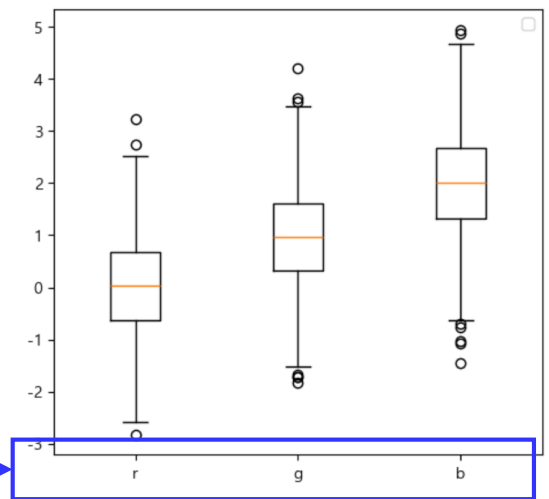
legend의 사용 - 두번째 방법 (boxplot)

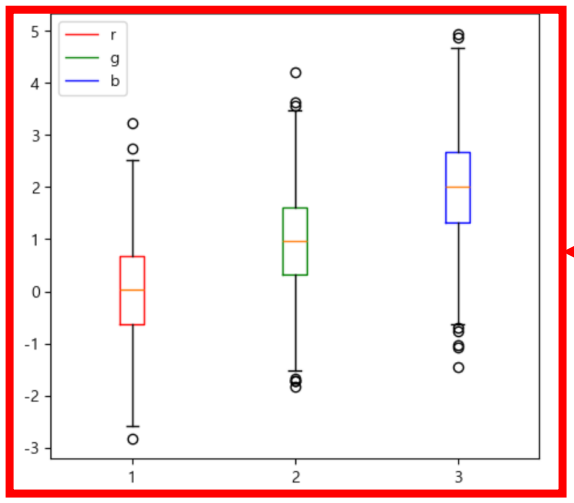
```
fig=plt.figure(figsize=(12,5), dpi=100)
ax1, ax2=fig.subplots(1,2)

d1=np.random.normal(0,1,1000)
d2=np.random.normal(1,1,1000)
d3=np.random.normal(2,1,1000)

## ax1
_=ax1.boxplot([d1,d2,d3], labels=['r','g','b'])
_=ax1.legend()

## ax2
bp1=ax2.boxplot(d1,positions=[1],boxprops = {'color':'r'})
bp2=ax2.boxplot(d2,positions=[2],boxprops = {'color':'g'})
bp3=ax2.boxplot(d3,positions=[3],boxprops = {'color':'b'})
bp1
bp1["boxes"][0]

_=ax2.legend(handles=(bp1["boxes"][0],bp2["boxes"][0], bp3["boxes"][0]), labels=list('rgb'))
```

박스 위치 다르게 설정

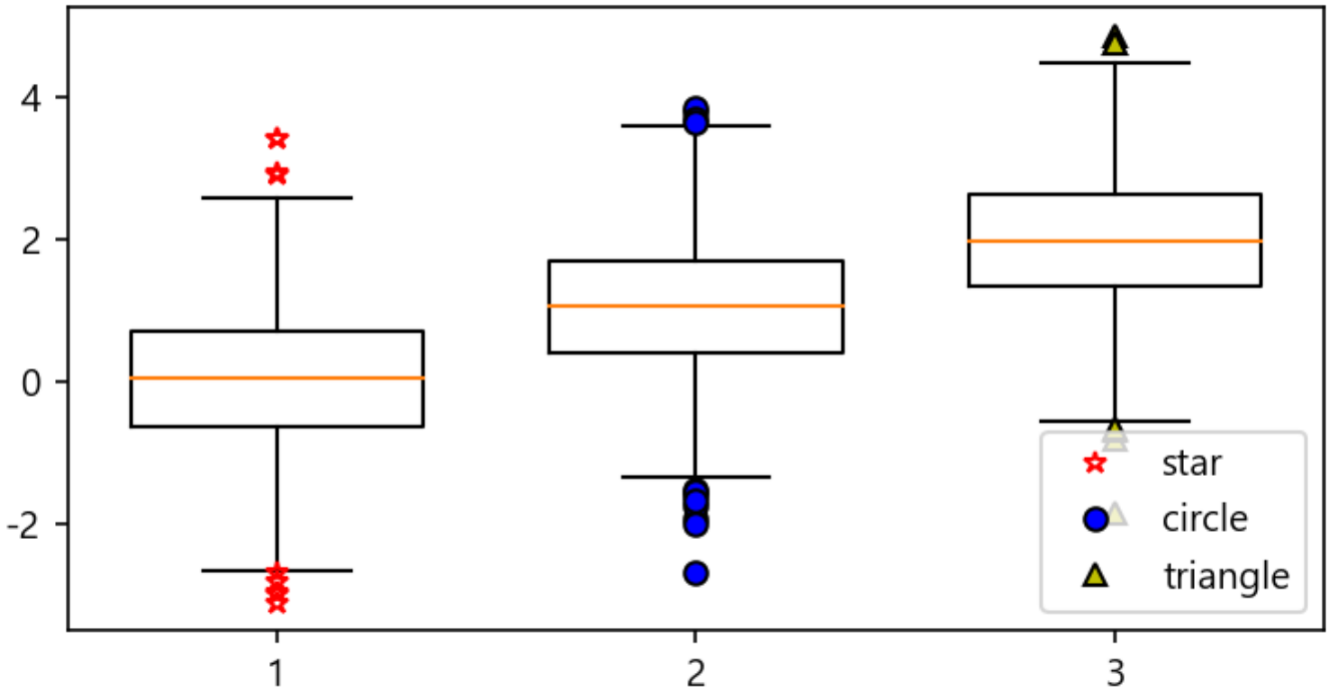
박스 색깔만 다르게 설정

박스의 색깔정보



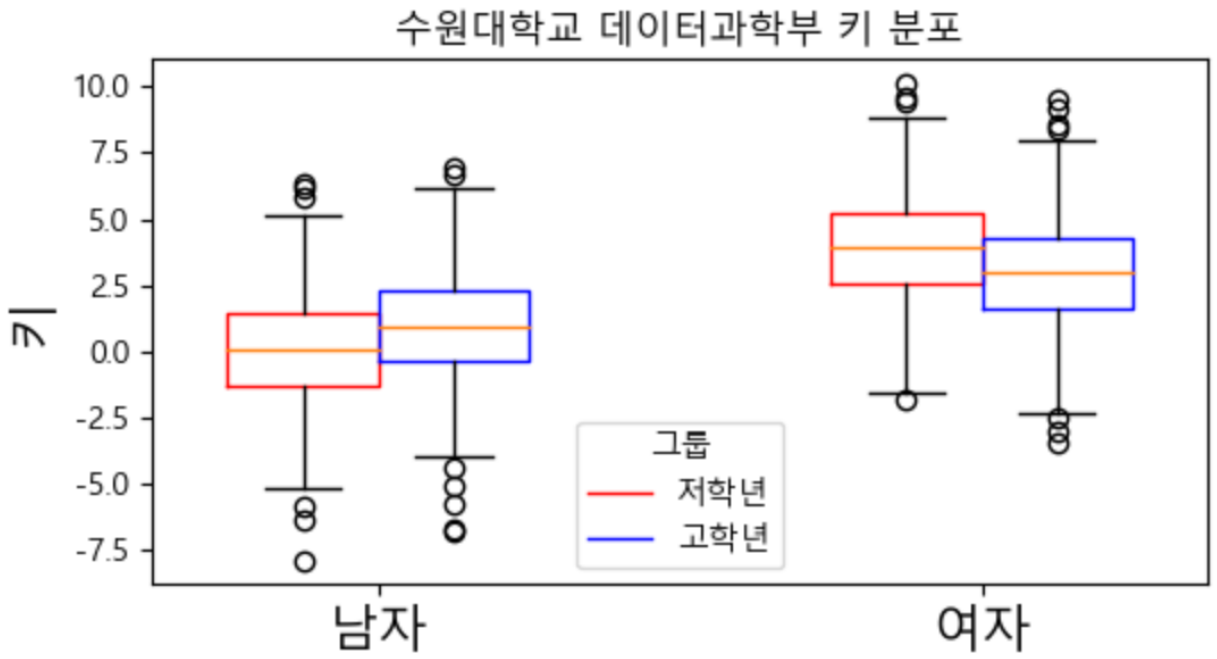


실습12




```
flierprops = {'marker':'*', 'markeredgecolor':'r'})
flierprops = {'marker':'o', 'markerfacecolor':'b'})
flierprops = {'marker':'^', 'markerfacecolor':'y'})
```

실습13



Q & A

Thank you