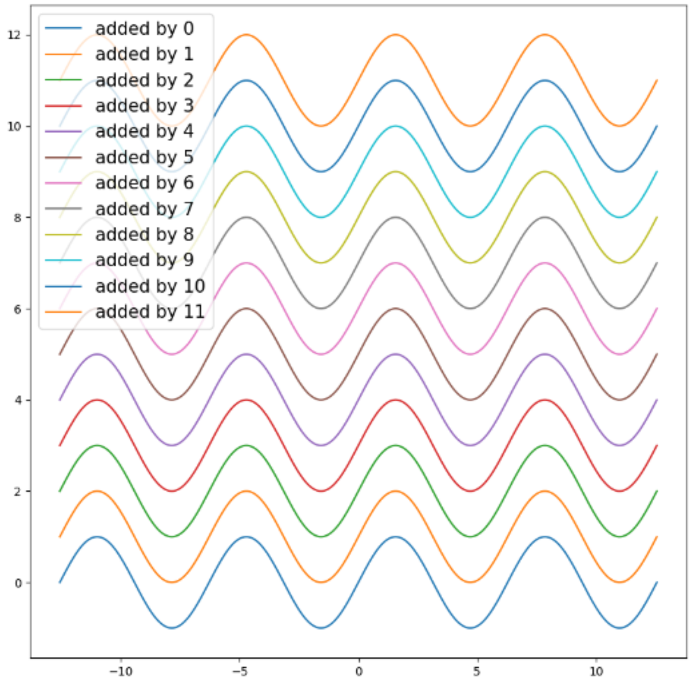
ML Day15 (Matplotlib)

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

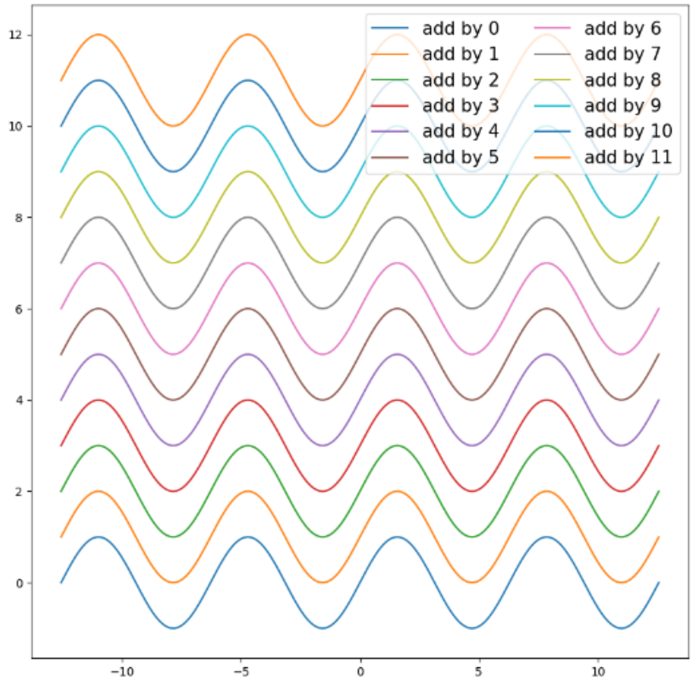
▼ ncol Argument (1)

```
PI = np.pi
t = np.linspace(-4*PI, 4*PI, 300)
sin = np.sin(t)

fig, ax = plt.subplots(figsize=(10, 10))

for ax_idx in range(12):               # 생성하고자 하는 임의의 수
    label_template = 'added by {}'
    ax.plot(t, sin+ax_idx,
            label=label_template.format(ax_idx))    # for문을 돌며 label_template에 .format(ax_idx) 들어간다.

ax.legend(fontsize=15)
plt.show()
```

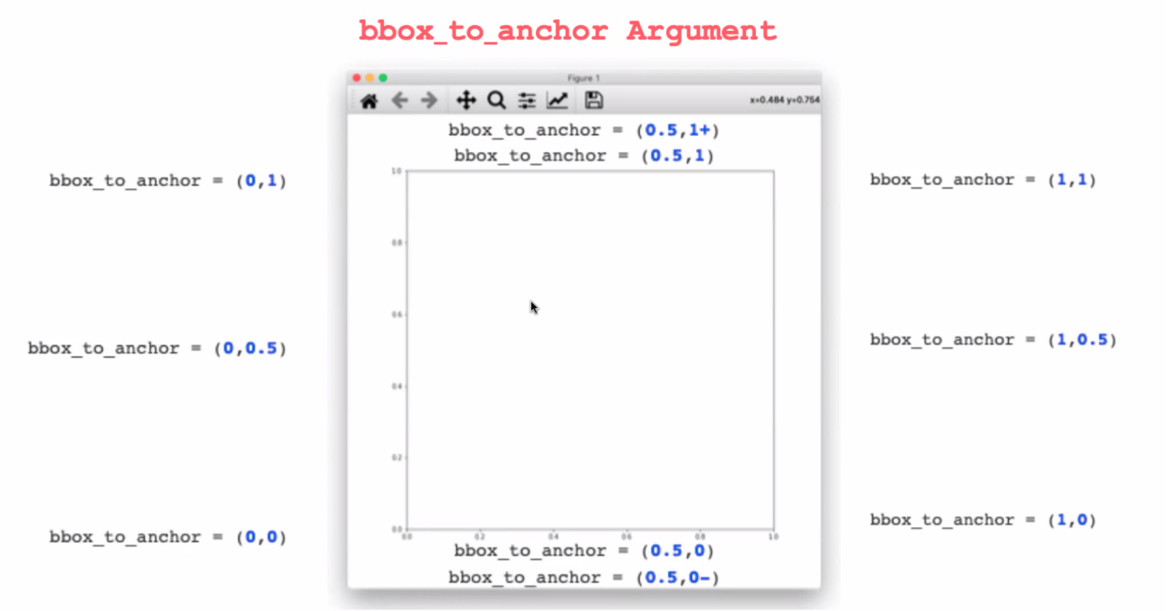


▼ ncol Argument (2)

```
PI = np.pi
t = np.linspace(-4*PI, 4*PI, 300)
sin = np.sin(t)

fig, ax = plt.subplots(figsize=(10, 10))

for ax_idx in range(12):
    label_template = 'add by {}'
    ax.plot(t, sin+ax_idx,
            label=label_template.format(ax_idx))

ax.legend(fontsize=15,
          ncol=2)             # legend의 column 갯수 지정

plt.show()
```



▼ **bbox_to_anchor Argument (1)**



bbox_to_anchor Argument

bbox_to_anchor = (0.5,1+)
bbox_to_anchor = (0.5,1)

bbox_to_anchor = (0,1)

bbox_to_anchor = (1,1)

bbox_to_anchor = (0,0.5)

bbox_to_anchor = (1,0.5)

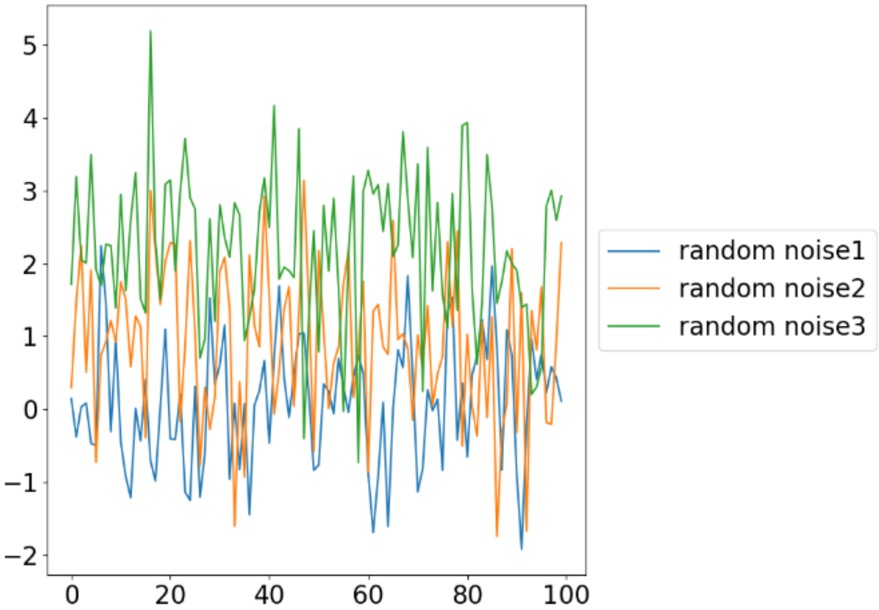
bbox_to_anchor = (0,0)

bbox_to_anchor = (1,0)

bbox_to_anchor = (0.5,0)
bbox_to_anchor = (0.5,0-)

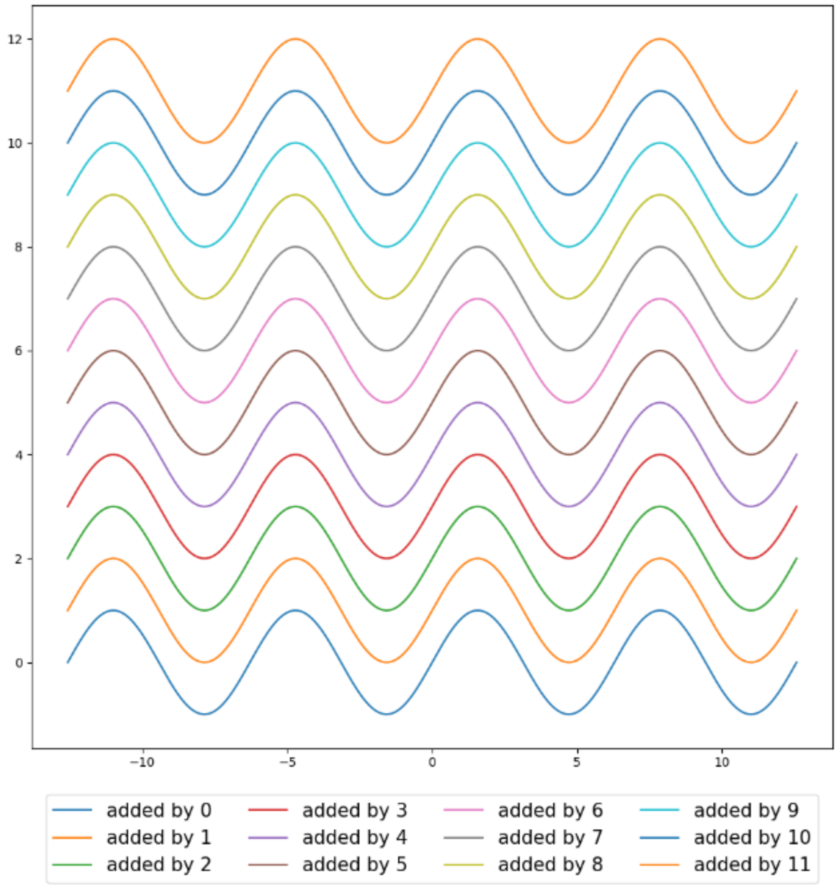
```
n_data = 100
random_noise1 = np.random.normal(0, 1, (n_data,))
random_noise2 = np.random.normal(1, 1, (n_data,))
random_noise3 = np.random.normal(2, 1, (n_data,))

fig, ax = plt.subplots(figsize=(10, 7))
ax.tick_params(labelsize=20)

ax.plot(random_noise1,
        label='random noise1')
ax.plot(random_noise2,
        label='random noise2')
```

```
ax.plot(random_noise3,
        label='random noise3')

ax.legend(fontsize=20,
          bbox_to_anchor=(1, 0.5),     # plot내에서 legend 기준인 anchor의 위치 지정
          loc='center left')           # bbox_to_anchor를 입력하면 legend안에서 anchor의 위치를 조정하게 된다.

fig.tight_layout()
plt.show()
```



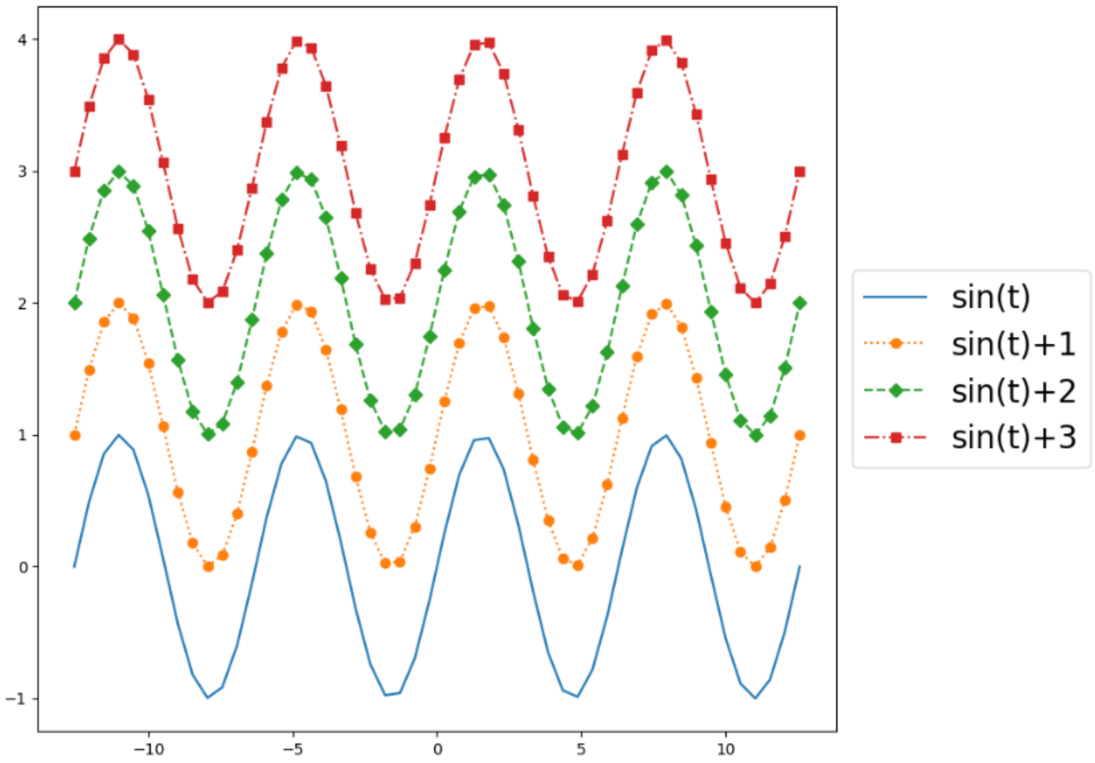
▼ bbox_to_anchor Argument (2)

```
PI = np.pi
t = np.linspace(-4*PI, 4*PI, 300)
sin = np.sin(t)

fig, ax = plt.subplots(figsize=(10, 10))

for ax_idx in range(12):
    label_template = 'added by {}'
    ax.plot(t, sin+ax_idx,
            label=label_template.format(ax_idx))

ax.legend(fontsize=15,
          ncol=4,
          bbox_to_anchor=(0.5, -0.05),
          loc='upper center')
fig.tight_layout()

plt.show()
```



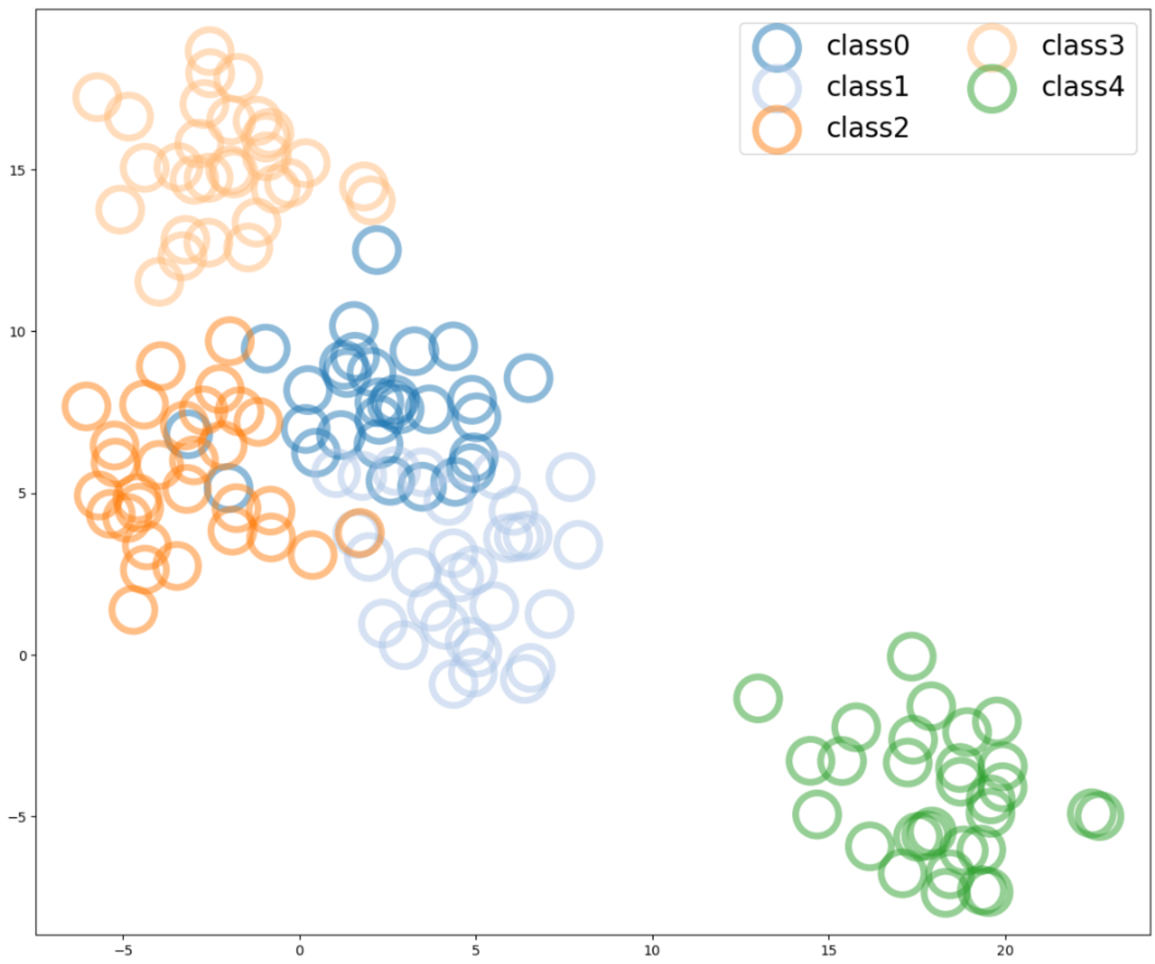
▼ Line Styles/Markers with Legend

```
PI = np.pi
t = np.linspace(-4*PI, 4*PI, 50)
sin = np.sin(t)

fig, ax = plt.subplots(figsize=(10, 7))

ax.plot(t, sin,
        label='sin(t)')
ax.plot(t, sin+1,
        marker='o',
        label='sin(t)+1',
        linestyle=':')
ax.plot(t, sin+2,
        marker='D',
        label='sin(t)+2',
        linestyle='--')
ax.plot(t, sin+3,
        marker='s',
        label='sin(t)+3',
        linestyle='-.')
ax.legend(loc='center left',              # 위 code에서 marker와 linestyle을 지정해준 것이 legend에도 적용된다.
          bbox_to_anchor=(1, 0.5),
          fontsize=20)
fig.tight_layout()
plt.show()
```



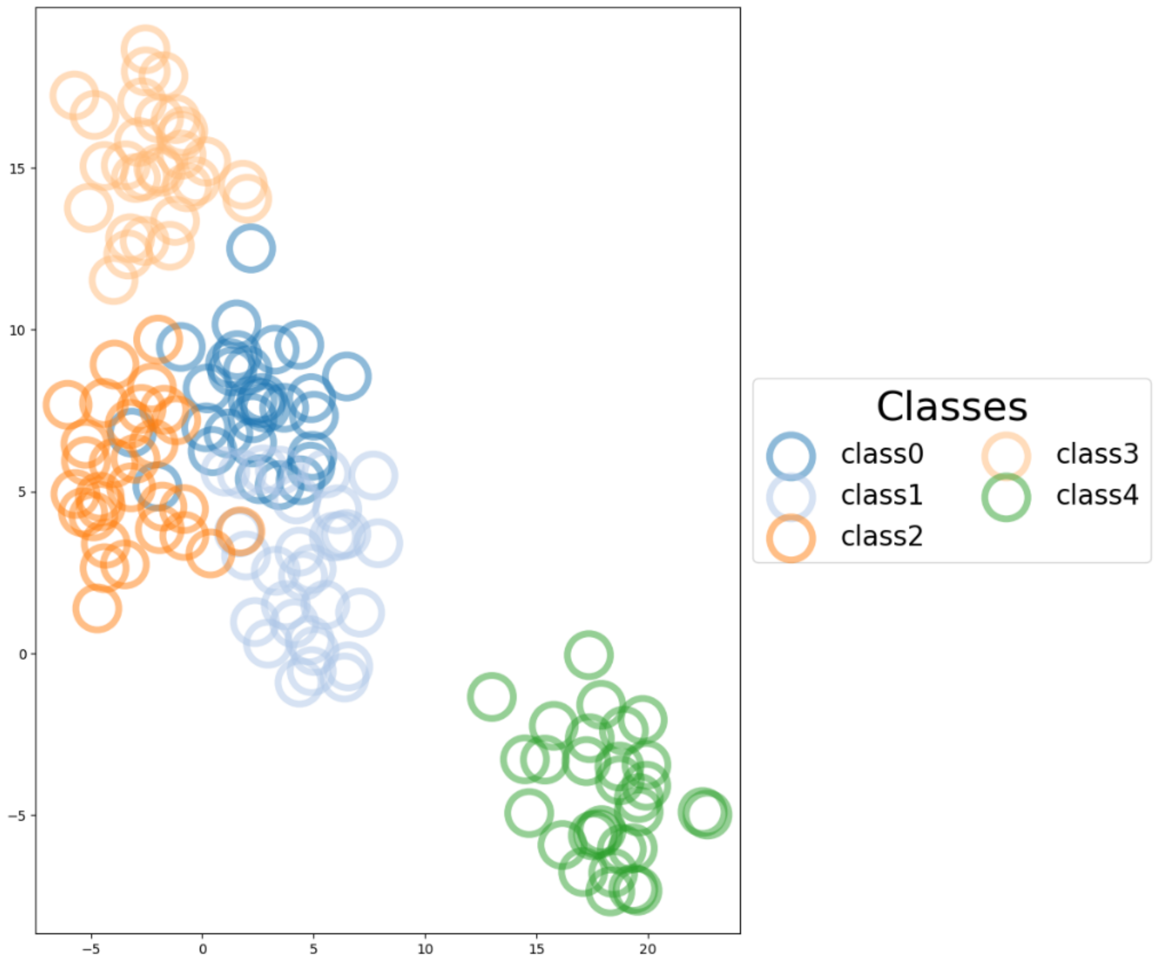
▼ Advanced Legend (1)

```
np.random.seed(0)

n_class = 5
n_data = 30
center_pt = np.random.uniform(-20, 20, (n_class, 2))    # uniform(최소값, 최대값, 범위) -> 범위 안에서 발생할 확률이 동일하게 랜덤추출, -20~20까
cmap = cm.get_cmap('tab20')
colors = [cmap(i) for i in range(n_class)]

data_dict = {'class' + str(i) : None for i in range(n_class)}   # 딕셔너리 컴프리헨션, class명(key)을 지정해주기 위해 for문을 돌림
for class_idx in range(n_class):
    center = center_pt[class_idx]                              # center_pt[class_idx]에 해당하는 값을 center변수에 대입

    x_data = center[0] + 2 * np.random.normal(0, 1, (1, n_data))
    y_data = center[1] + 2 * np.random.normal(0, 1, (1, n_data))
    data = np.vstack((x_data, y_data))

    data_dict['class' + str(class_idx)] = data                     # 딕셔너리 생성 : key = 'class' + str(class_idx) : value = data

fig, ax = plt.subplots(figsize=(12, 10))
for class_idx in range(n_class):
    data = data_dict['class' + str(class_idx)]
    ax.scatter(data[0], data[1],
               s=1000,
               facecolor='None',
               edgecolor=colors[class_idx],
               linewidth=5,
               alpha=0.5,
               label='class' + str(class_idx))

ax.legend(loc='upper right',
          bbox_to_anchor=(1, 1),
          fontsize=20,
          ncol=2)
fig.tight_layout()

plt.show()
```



▼ Advanced Legend (2)

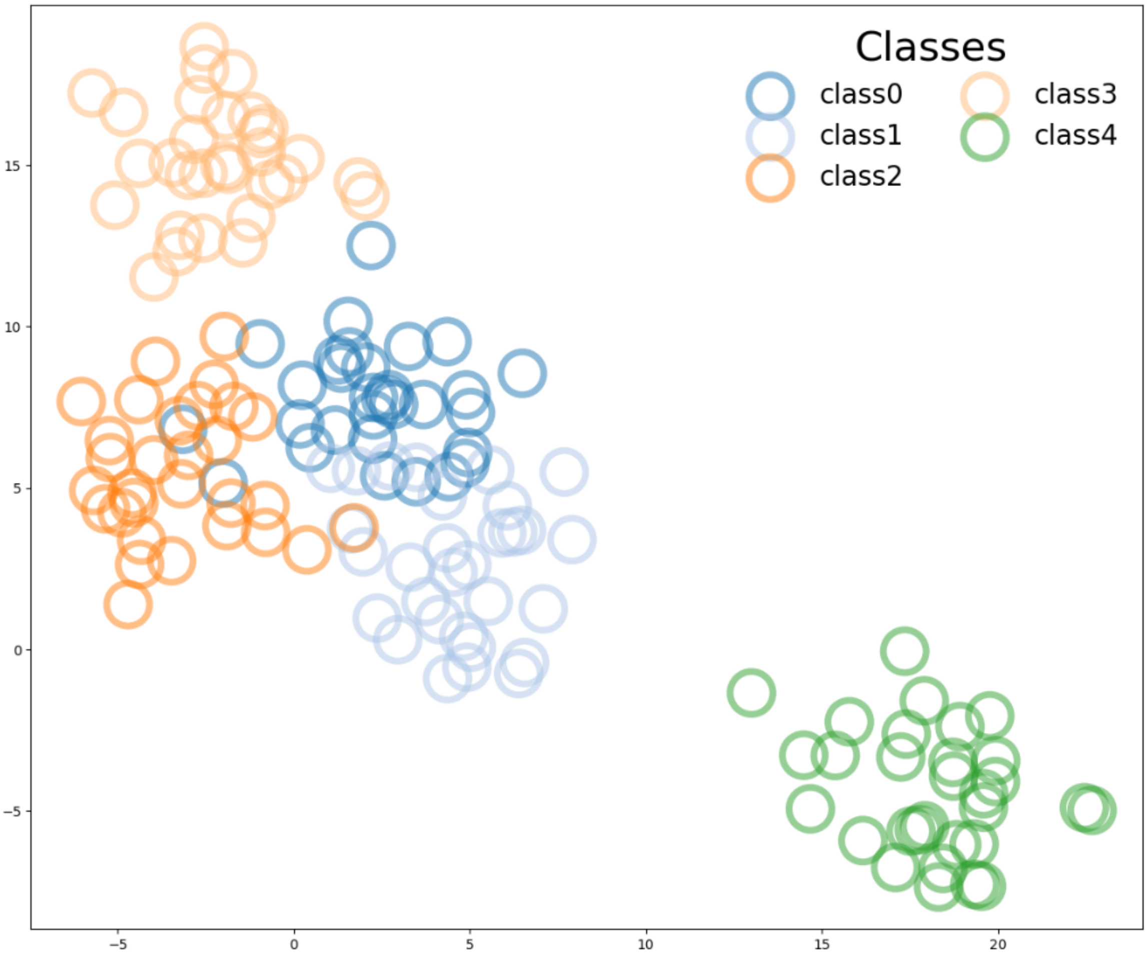
```
np.random.seed(0)

n_class = 5
n_data = 30
center_pt = np.random.uniform(-20, 20, (n_class, 2))
cmap = cm.get_cmap('tab20')
colors = [cmap(i) for i in range(n_class)]

data_dict = {'class' + str(i) : None for i in range(n_class)}
for class_idx in range(n_class):
    center = center_pt[class_idx]

    x_data = center[0] + 2 * np.random.normal(0, 1, (1, n_data))
    y_data = center[1] + 2 * np.random.normal(0, 1, (1, n_data))
    data = np.vstack((x_data, y_data))

    data_dict['class' + str(class_idx)] = data

fig, ax = plt.subplots(figsize=(12, 10))
for class_idx in range(n_class):
    data = data_dict['class' + str(class_idx)]
    ax.scatter(data[0], data[1],
               s=1000,
               facecolor='None',
               edgecolor=colors[class_idx],
               linewidth=5,
               alpha=0.5,
               label='class' + str(class_idx))

ax.legend(loc='center left',
          bbox_to_anchor=(1, 0.5),
          fontsize=20,
          ncol=2,
          title='Classes',    # legend의 title명 지정
          title_fontsize=30)
fig.tight_layout()
plt.show()
```



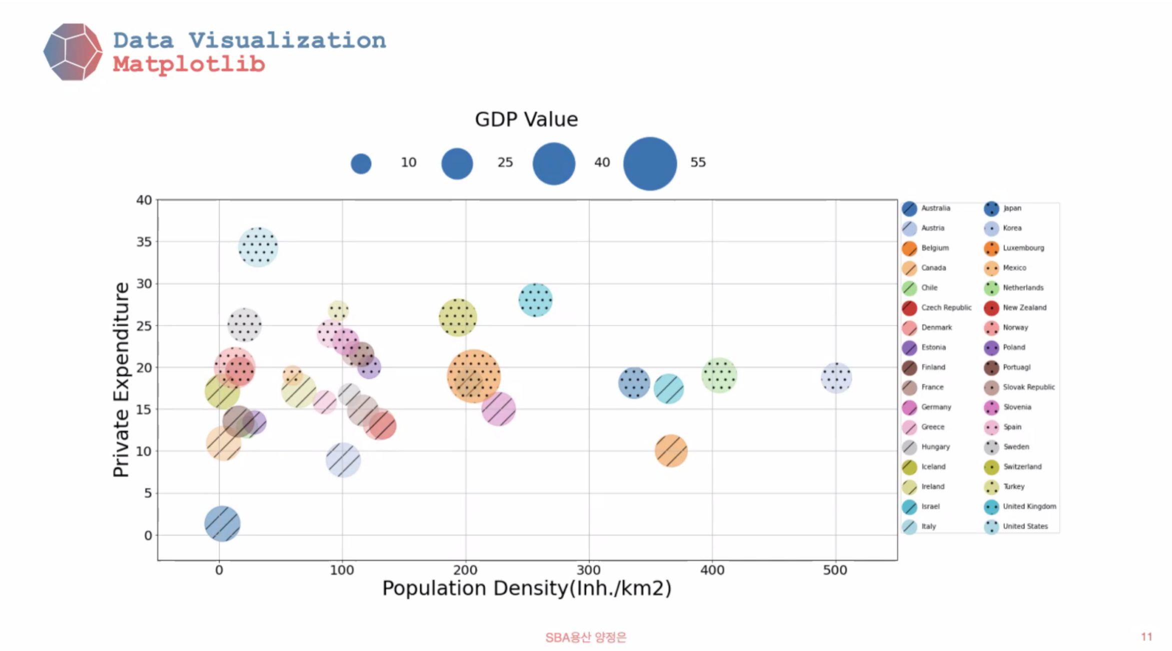
Advanced Legend (3)

```
np.random.seed(0)

n_class = 5
n_data = 30
center_pt = np.random.uniform(-20, 20, (n_class, 2))
cmap = cm.get_cmap('tab20')
colors = [cmap(i) for i in range(n_class)]

data_dict = {'class' + str(i) : None for i in range(n_class)}
for class_idx in range(n_class):
    center = center_pt[class_idx]

    x_data = center[0] + 2 * np.random.normal(0, 1, (1, n_data))
    y_data = center[1] + 2 * np.random.normal(0, 1, (1, n_data))
    data = np.vstack((x_data, y_data))

    data_dict['class' + str(class_idx)] = data

fig, ax = plt.subplots(figsize=(12, 10))
for class_idx in range(n_class):
    data = data_dict['class' + str(class_idx)]
    ax.scatter(data[0], data[1],
               s=1000,
               facecolor='None',
               edgecolor=colors[class_idx],
               linewidth=5,
               alpha=0.5,
               label='class' + str(class_idx))
```

```
ax.legend(loc='upper right',
          bbox_to_anchor=(1, 1),
          fontsize=20,
          ncol=2,
          title='Classes',
          title_fontsize=30,
          edgecolor='None',            # legend의 테두리색과 배경색 'None'
          facecolor='None',
          labelspacing=3,              # 각 label명 간의 간격 조정
          columnspacing=5)             # column 간격 조정
fig.tight_layout()
plt.show()
```



▼ 시각화 실습(1)



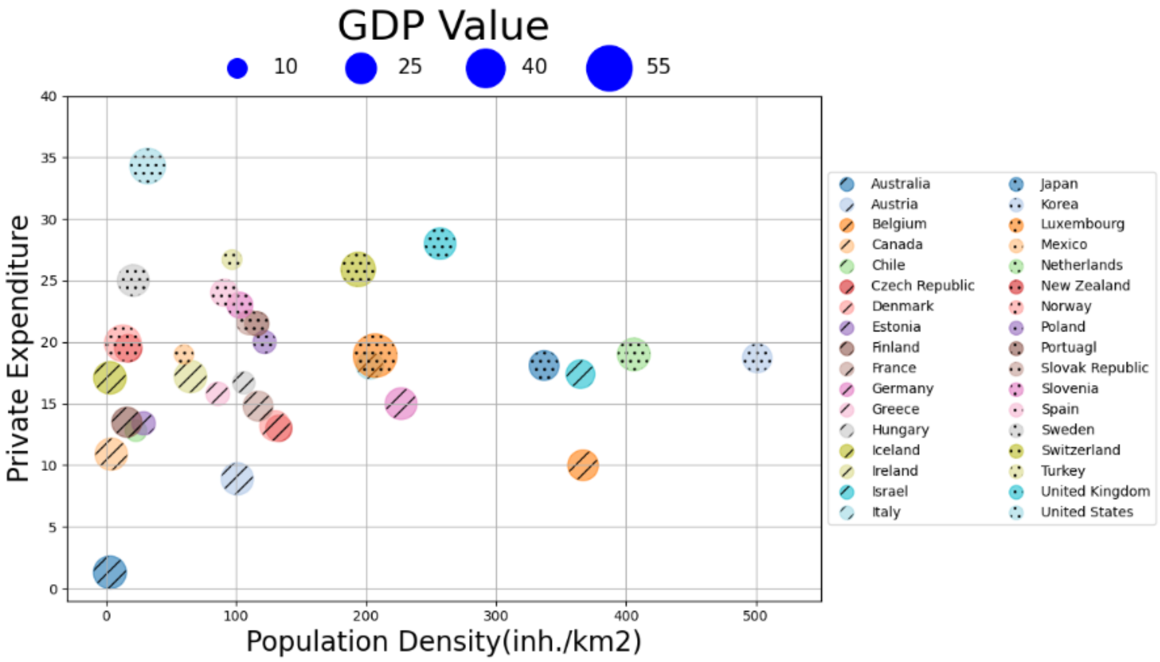
Data Visualization
Matplotlib

SBA용산 양정은

11

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


countries = ['Australia', 'Austria', 'Belgium', 'Canada', 'Chile',
             'Czech Republic', 'Denmark', 'Estonia', 'Finland', 'France',
             'Germany', 'Greece', 'Hungary', 'Iceland', 'Ireland',
             'Israel', 'Italy', 'Japan', 'Korea', 'Luxembourg',
             'Mexico', 'Netherlands', 'New Zealand', 'Norway', 'Poland',
             'Portuagl', 'Slovak Republic', 'Slovenia', 'Spain', 'Sweden',
             'Switzerland', 'Turkey', 'United Kingdom', 'United States']

population_density = [3, 101, 367, 4, 23,
                      133, 130, 29, 16, 117,
                      227, 86, 106, 3, 65,
                      365, 203, 337, 501, 207,
                      60, 406, 17, 13, 122,
                      116, 110, 103, 91, 21,
                      194, 97, 257, 32]
private_expenditure = [1.3, 8.9, 10.0, 10.9, 12.8,
                       13.0, 13.2, 13.4, 13.5, 14.8,
                       15.0, 15.8, 16.7, 17.1, 17.2,
                       17.4, 18.1, 18.1, 18.7, 18.9,
                       19.0, 19.0, 19.5, 19.9, 20.0,
                       21.5, 21.6, 23.0, 24.0, 25.0,
                       25.9, 26.7, 28.0, 34.3]
gdp = [38.7, 37.4, 33.6, 37.5, 16.4,
       24.5, 33.2, 19.3, 32.1, 32.0,
       36.2, 19.8, 17.8, 37.7, 37.7,
       29.4, 26.6, 32.0, 31.0, 67.9,
       13.4, 38.4, 27.0, 48.2, 18.9,
       20.9, 21.8, 24.2, 26.8, 36.2,
       42.5, 13.9, 35.6, 45.7]

gdp_arr = np.array(gdp)


cmap = cm.get_cmap('tab20', lut=17)
colors = [cmap(i) for i in range(17)]



# Main graph

fig, ax = plt.subplots(figsize=(12, 7))

for c_idx in range(len(countries))[:17]:
    ax.scatter(population_density[c_idx], private_expenditure[c_idx],
               s=gdp[c_idx]*15,
               c=colors[c_idx],
               alpha=0.6,
               hatch='/'*2)
```

```
for c_idx_2 in range(len(countries))[17:]:
    ax.scatter(population_density[c_idx_2], private_expenditure[c_idx_2],
               s=gdp[c_idx_2]*15,
               c=colors[c_idx_2 - 17],
               alpha=0.6,
               hatch='.' * 2)

# Legend를 만들기 위해 2개의 빈 그래프 생성
for m in range(len(countries))[:17]:
    ax.scatter([], [],
               s=100,
               c=colors[m],
               alpha=0.6,
               hatch='/'*2,
               label=countries[m])            # legend의 국가명을 표기하기 위해 label지정

for n in range(len(countries))[17:]:
    ax.scatter([], [],
               s=100,
               c=colors[n - 17],
               alpha=0.6,
               hatch='.'*2,
               label=countries[n])             # legend의 국가명을 표기하기 위해 label지정

# 2개의 빈 그래프의 색과 국가명을 가져오는 legend 생성
ax.legend(loc='center left',
          bbox_to_anchor=(1, 0.5),
          fontsize=30,
          ncol=2)

# 상단의 GDP Value(legend)를 만들기 위한 과정
gdp_size = [10, 25, 40, 55]

ax2 = ax.twinx()                                  # ax.twinx()로 ax와 같은 축을 가지도록 설정

# 빈 scatter인 ax.2 생성(color와 label을 가짐)
for k in range(len(gdp_size)):                    # for문으로 gdp_size의 길이만큼 빈 scatter생성
    ax2.scatter([], [],
                s=gdp_size[k]*20,                 # 크기를 gdp_size[k]값의 20배
                color='blue',
                label=gdp_size[k])                # 표시해야 할 label이 gdp_size의 수치이므로 gdp_size[k]로 indexing

ax2.tick_params(axis='y',
                right=False, labelright=False)    # ax2의 tick_params로 우측 tick과 labeltick 삭제
ax2.set_xlim([100, 400])                          # ax2의 x축 최소, 최대범위 설정
ax2.legend(loc='lower center',                    # 비어있는 ax2 scatter의 legend 설정
           bbox_to_anchor=(0.5, +1),
           fontsize=15,
           ncol=4,
           title='GDP Value',
           title_fontsize=30,
           edgecolor='None',
           facecolor='None')

ax.legend(loc='center left',
          bbox_to_anchor=(1, 0.5),
          fontsize=10,
          ncol=2)


ax.set_xlim([-30, 550])
ax.set_ylim([-1, 40])

ax.set_xlabel('Population Density(inh./km2)', fontsize=20)
ax.set_ylabel('Private Expenditure', fontsize=20)

ax.grid()

fig.tight_layout()
plt.show()
```


GDP Value
10
25
40
55
40
35
30
25
20
15
10
5
0
Private Expenditure
0
100
200
300
400
500
Population Density(inh./km2)
Australia
Austria
Belgium
Canada
Chile
Czech Republic
Denmark
Estonia
Finland
France
Germany
Greece
Hungary
Iceland
Ireland
Israel
Italy
Japan
Korea
Luxembourg
Mexico
Netherlands
New Zealand
Norway
Poland
Portuagl
Slovak Republic
Slovenia
Spain
Sweden
Switzerland
Turkey
United Kingdom
United States