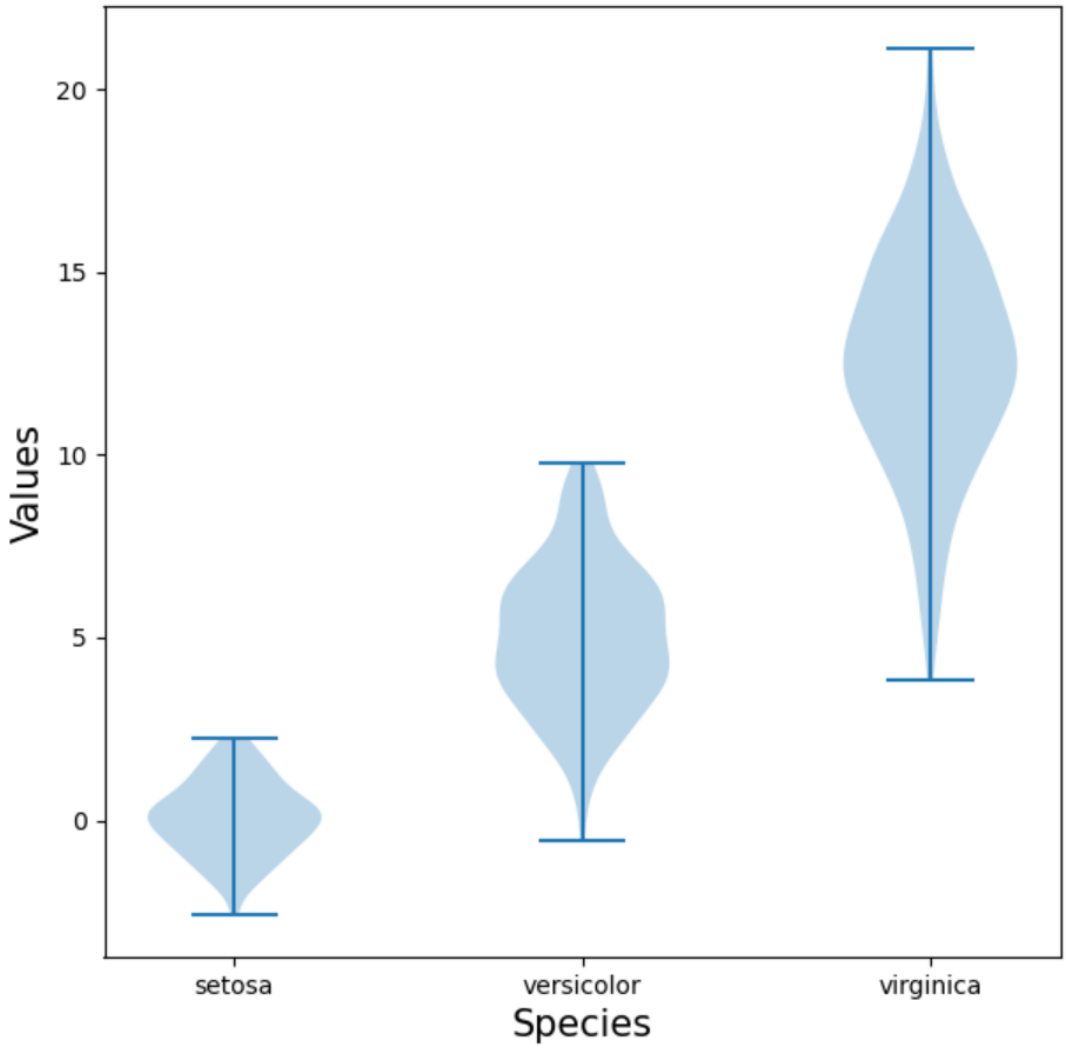
# ML Day19 (Matplotlib) (Violinplot)

### IRIS Data Plot

▼ **ax.violinplot (1)** : **violinplot = 데이터 별로 feature들이 어떻게 분포 되어있는지 확인할 수 있는 시각화 방법**

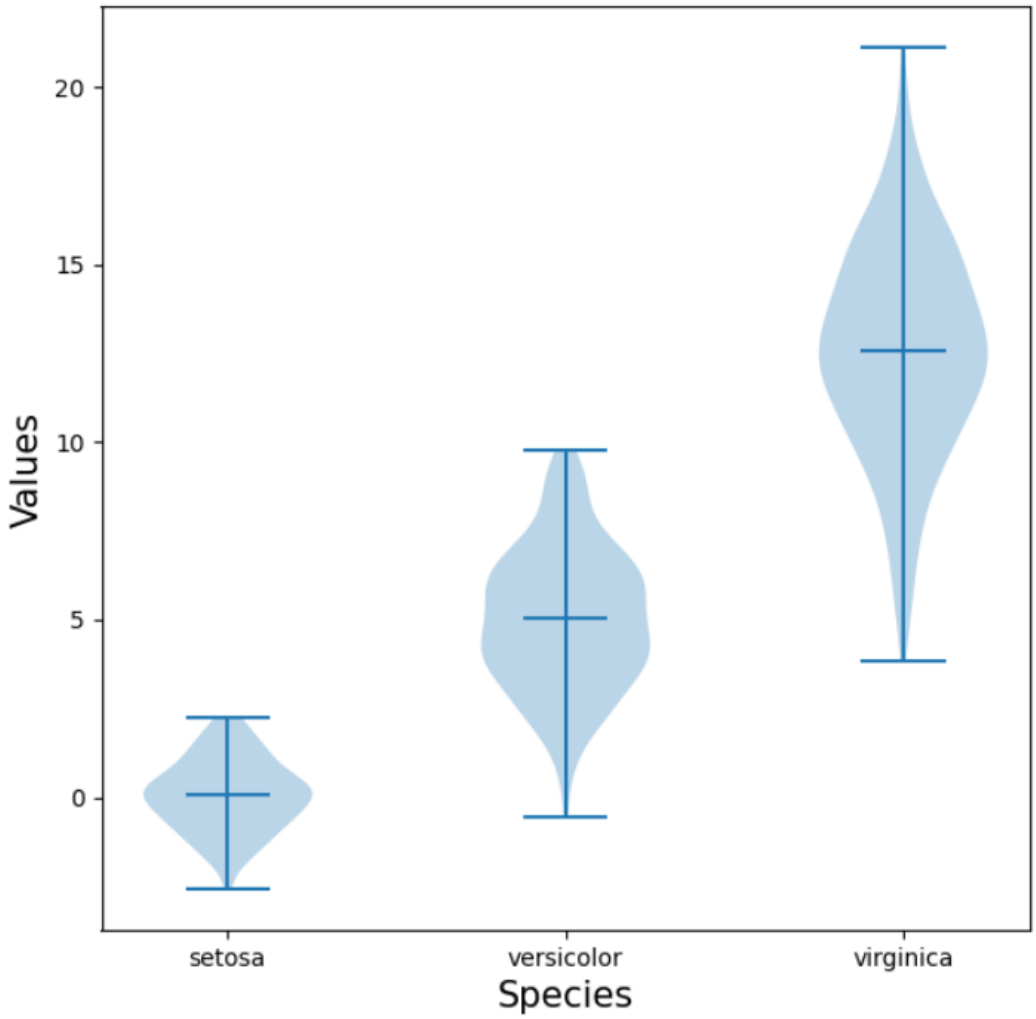
```
import matplotlib.pyplot as plt
import numpy as np
import matplotlib.cm as cm

np.random.seed(0)

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

data1 = np.random.normal(0, 1, 100)
data2 = np.random.normal(5, 2, 200)
data3 = np.random.normal(13, 3, 300)

xticks = np.arange(3)

# violinplot에 들어가는 데이터는 [] 형태로 들어가야 한다. positions : plot이 xticks에 위치하도록 설정
ax.violinplot([data1, data2, data3], positions=xticks)

ax.set_xticks(xticks)
ax.set_xticklabels(['setosa', 'versicolor', 'virginica'])
ax.set_xlabel('Species', fontsize=15)
ax.set_ylabel('Values', fontsize=15)

plt.show()
```




**▼ ax.violinplot (2) : plot의 평균들 나타내기**

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

np.random.seed(0)

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

data1 = np.random.normal(0, 1, 100)
data2 = np.random.normal(5, 2, 200)
data3 = np.random.normal(13, 3, 300)

xticks = np.arange(3)

ax.violinplot([data1, data2, data3],
              showmeans=True,          # 각 plot들의 means 표현
              showextrema=False,       # plot의 최대, 최소 표현(T/F) - default=True
              showmedians=True,        # plot의 중앙 값(데이터값들의 평균이 means, 데이터들(n)의 중앙값 표)
              positions=xticks)

ax.set_xticks(xticks)
```

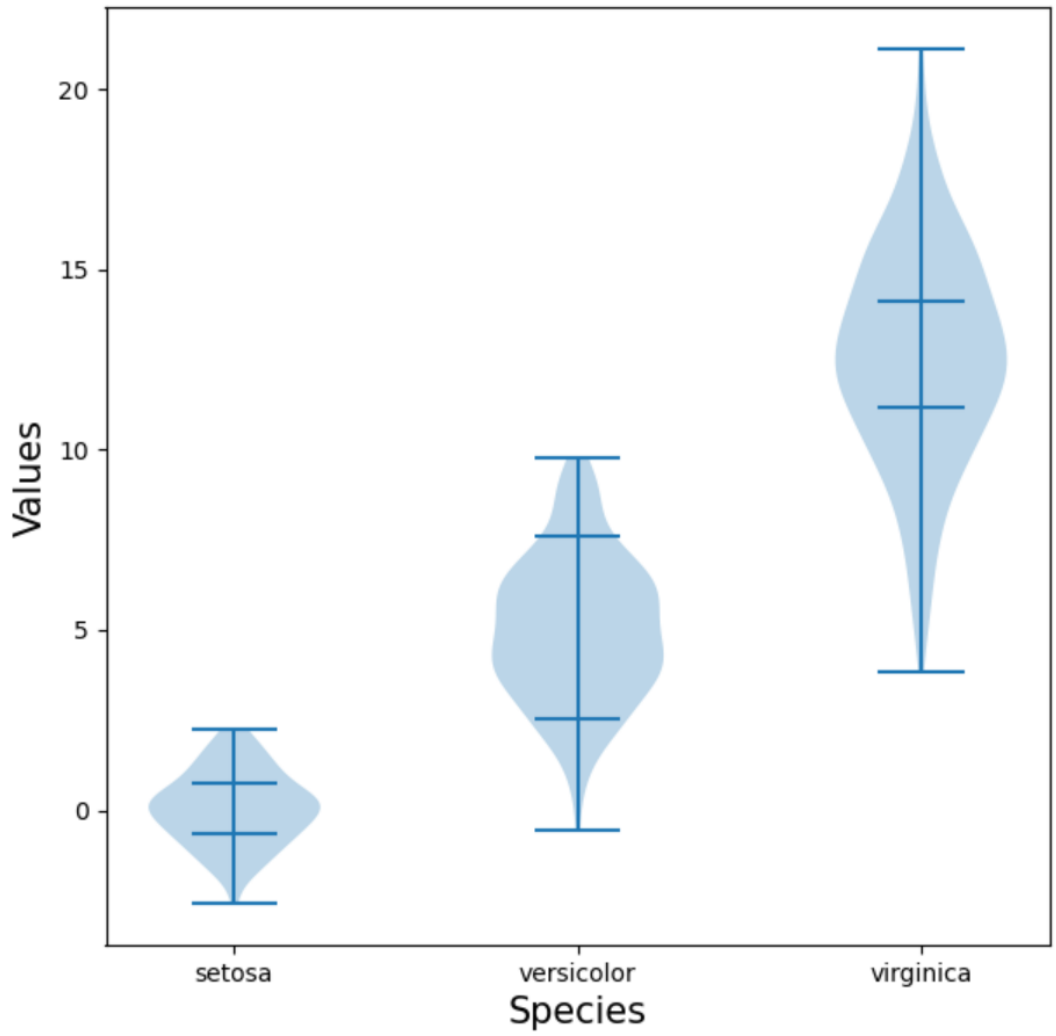
```
ax.set_xticklabels(['setosa', 'versicolor', 'virginica'])
ax.set_xlabel('Species', fontsize=15)
ax.set_ylabel('Values', fontsize=15)

plt.show()
```




**▼ ax.violinplot (3) : plot에 분위 수 표현하기**

```
np.random.seed(0)

data1 = np.random.normal(0, 1, 100)
data2 = np.random.normal(5, 2, 200)
data3 = np.random.normal(13, 3, 300)

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

xticks = np.arange(3)

ax.violinplot([data1, data2, data3],
              quantiles=[[0.25, 0.75], [0.1, 0.9], [0.3, 0.7]],   # 분위수 표현 ([25%, 75%], [10%, 90%], [30%, 70%])
              positions=xticks)

ax.set_xticks(xticks)
ax.set_xticklabels(['setosa', 'versicolor', 'virginica'])
ax.set_xlabel('Species', fontsize=15)
```

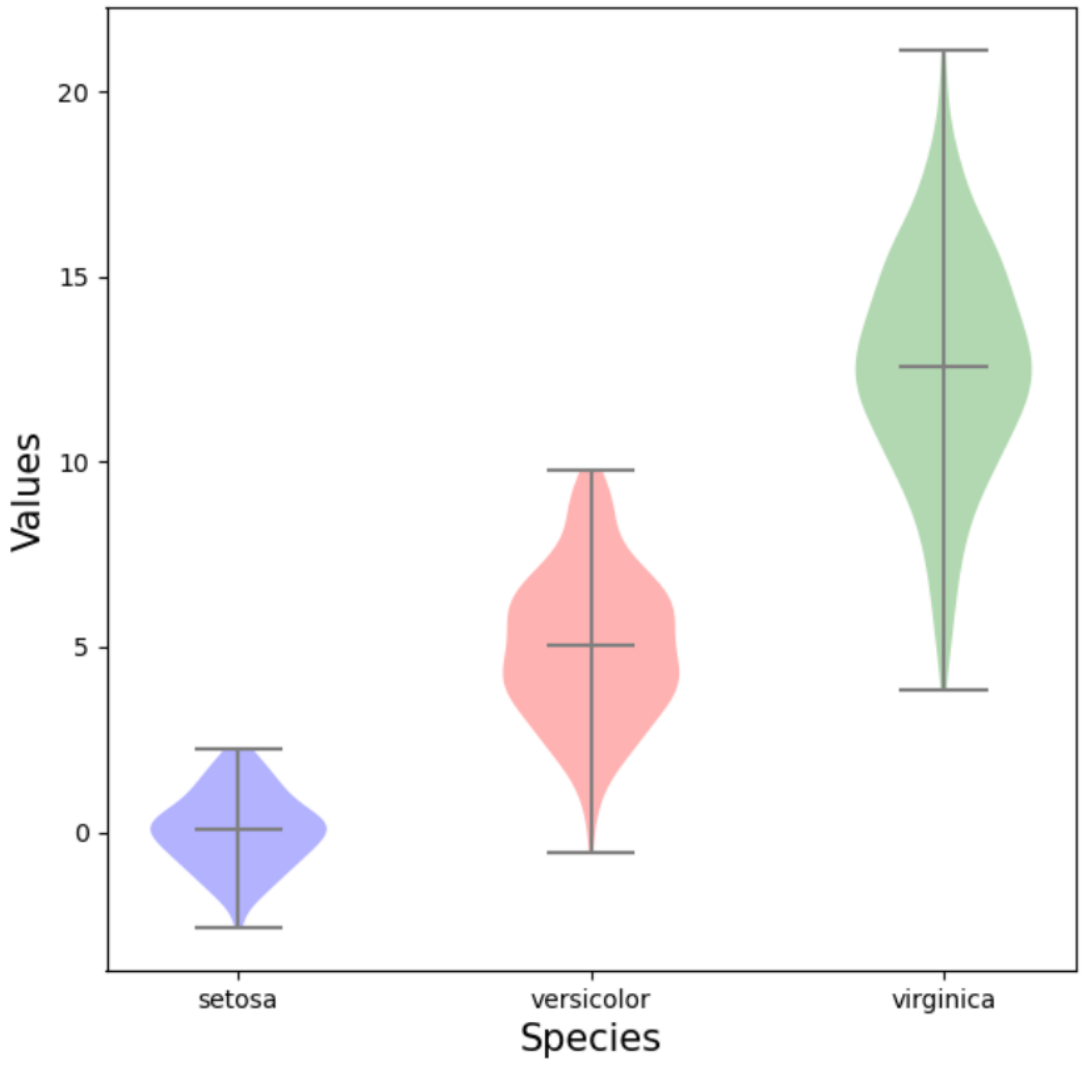
```
ax.set_ylabel('Values', fontsize=15)

plt.show()
```



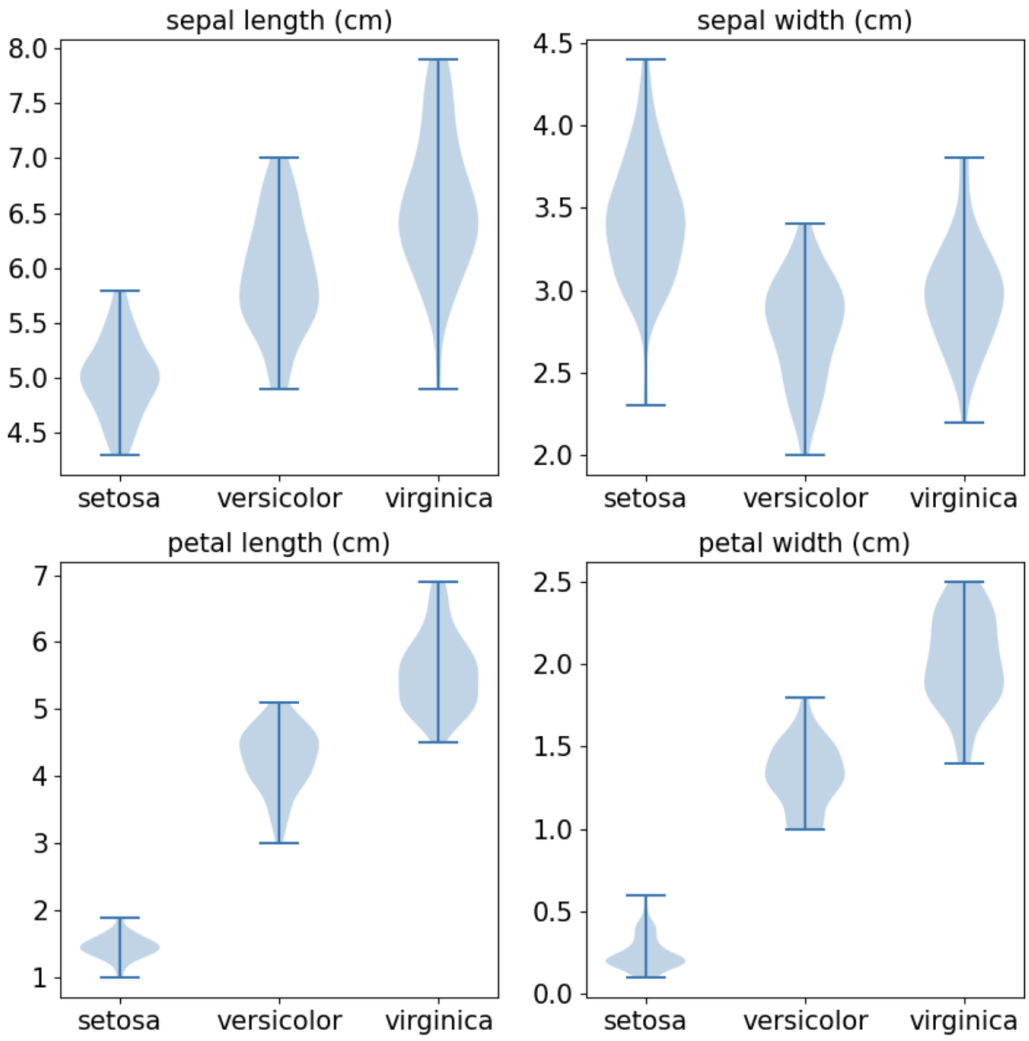

**▼ ax.violinplot (4) : 'bodies', 'cbars', 'cmaxes', 'cmins', 'cmeans' 색상 설정**

```
np.random.seed(0)
data1 = np.random.normal(0, 1, 100)
data2 = np.random.normal(5, 2, 200)
data3 = np.random.normal(13, 3, 300)

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

xticks = np.arange(3)

violin = ax.violinplot([data1, data2, data3],    # plot을 violin 변수에 대입(dictionary 형태로 저장)
                       showmeans=True,
                       positions=xticks)

ax.set_xticks(xticks)
ax.set_xticklabels(['setosa', 'versicolor', 'virginica'])
ax.set_xlabel('Species', fontsize=15)
ax.set_ylabel('Values', fontsize=15)
```

```
violin['bodies'][0].set_facecolor('blue')        # violin에 저장된 plot의 ['bodies']의 [0]번째 index 컬러 설정
violin['bodies'][1].set_facecolor('red')
violin['bodies'][2].set_facecolor('green')

violin['cbars'].set_edgecolor('gray')            # 각각의 plot들의 'cbars' 색상 설정
violin['cmaxes'].set_edgecolor('gray')           # 'cmaxes' 최댓값 표현 bar 색상 설정
violin['cmins'].set_edgecolor('gray')
violin['cmeans'].set_edgecolor('gray')

plt.show()
```




**시각화 실습 (4)**

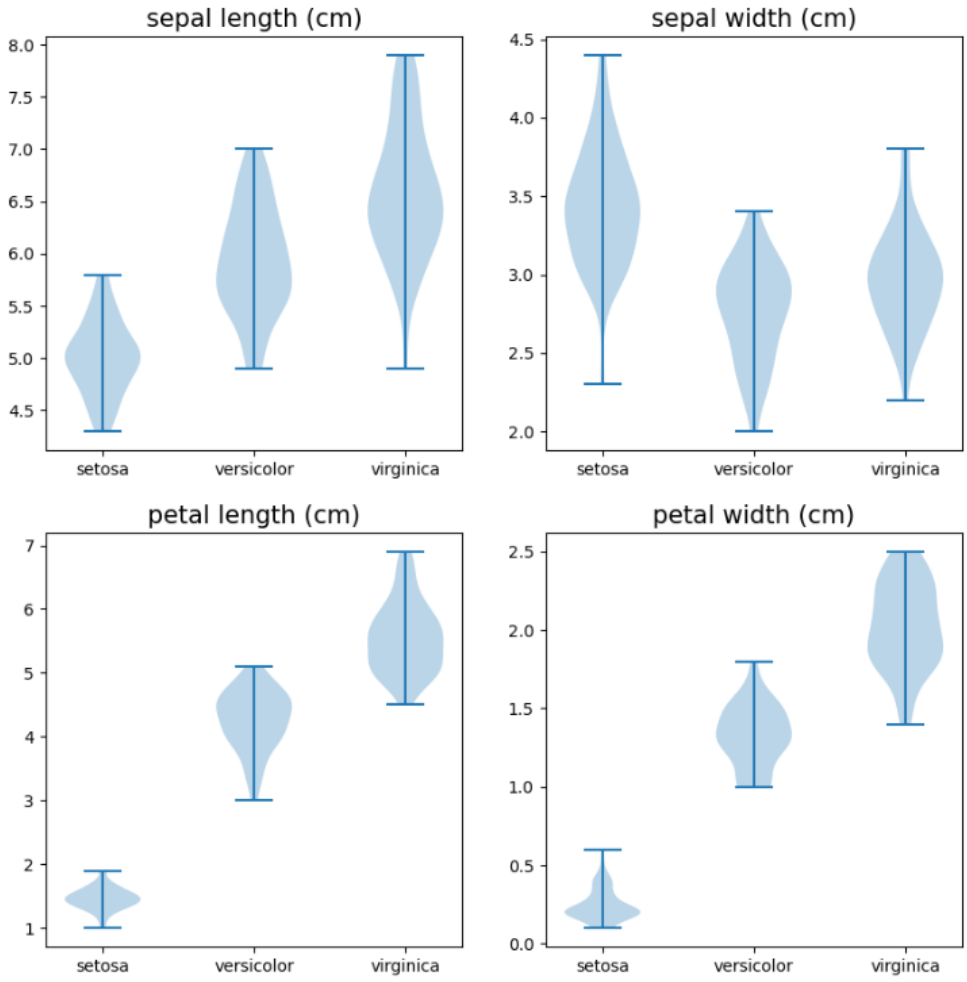



**▼ Iris data (Kaggle의 Iris data)**

```
from sklearn.datasets import load_iris

iris = load_iris()

feature_names = iris.feature_names
n_features = len(feature_names)
species = iris.target_names
n_species = len(species)

iris_X, iris_y = iris.data, iris.target
```

```
import matplotlib.pyplot as plt
import numpy as np
import pandas as pd
import matplotlib.cm as cm
from sklearn.datasets import load_iris

iris = load_iris()
iris_df = pd.DataFrame(iris['data'],
                       columns=['SepalLength', 'SepalWidth', 'PetalLength', 'Petalwidth'])
iris_target = pd.DataFrame(iris['target'],
                           columns=['target'])
```

```
se_pe_lw = iris_df[['SepalLength', 'SepalWidth', 'PetalLength', 'Petalwidth']].values
species = iris_target['target'].values

# target feature별로 데이터 분화
se_pe_lw0 = se_pe_lw[species == 0]     # species : 'setosa'
se_pe_lw1 = se_pe_lw[species == 1]     # species : 'versicolor'
se_pe_lw2 = se_pe_lw[species == 2]     # species : 'virginica'

feature_names = iris.feature_names
species_names = iris.target_names

xticks = np.arange(3)

fig, axes = plt.subplots(2, 2, figsize=(10, 10))

# enumerate를 이용하여 (2, 2) 그래프를 flat으로 왼위->오위->왼아래->오아래 순으로 차례대로 plot을 그려줌
for ax_idx, ax in enumerate(axes.flat):
    ax.violinplot([se_pe_lw0[:, ax_idx], se_pe_lw1[:, ax_idx], se_pe_lw2[:, ax_idx]], positions=xticks)
    ax.set_xticks(xticks)
    ax.set_xticklabels(species_names)
    ax.set_title(feature_names[ax_idx], fontsize=15)

plt.show()
```

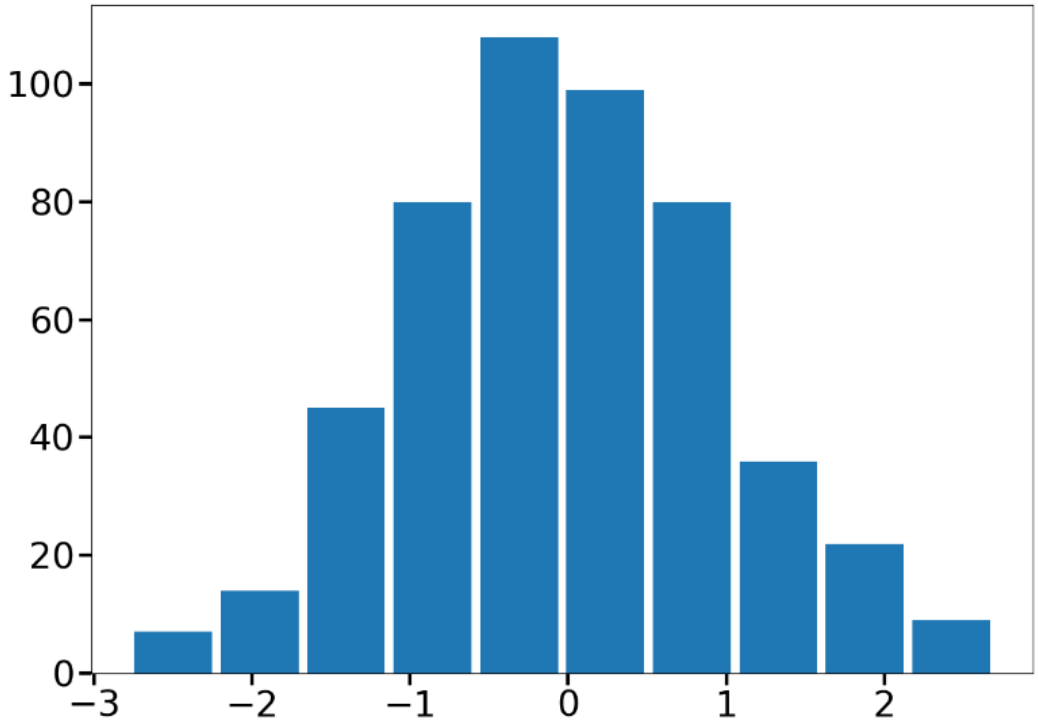



```
# 선생님 code

import matplotlib.pyplot as plt
from sklearn.datasets import load_iris

iris = load_iris()

feature_names = iris.feature_names
n_features = len(feature_names)
species = iris.target_names
n_species = len(species)
```

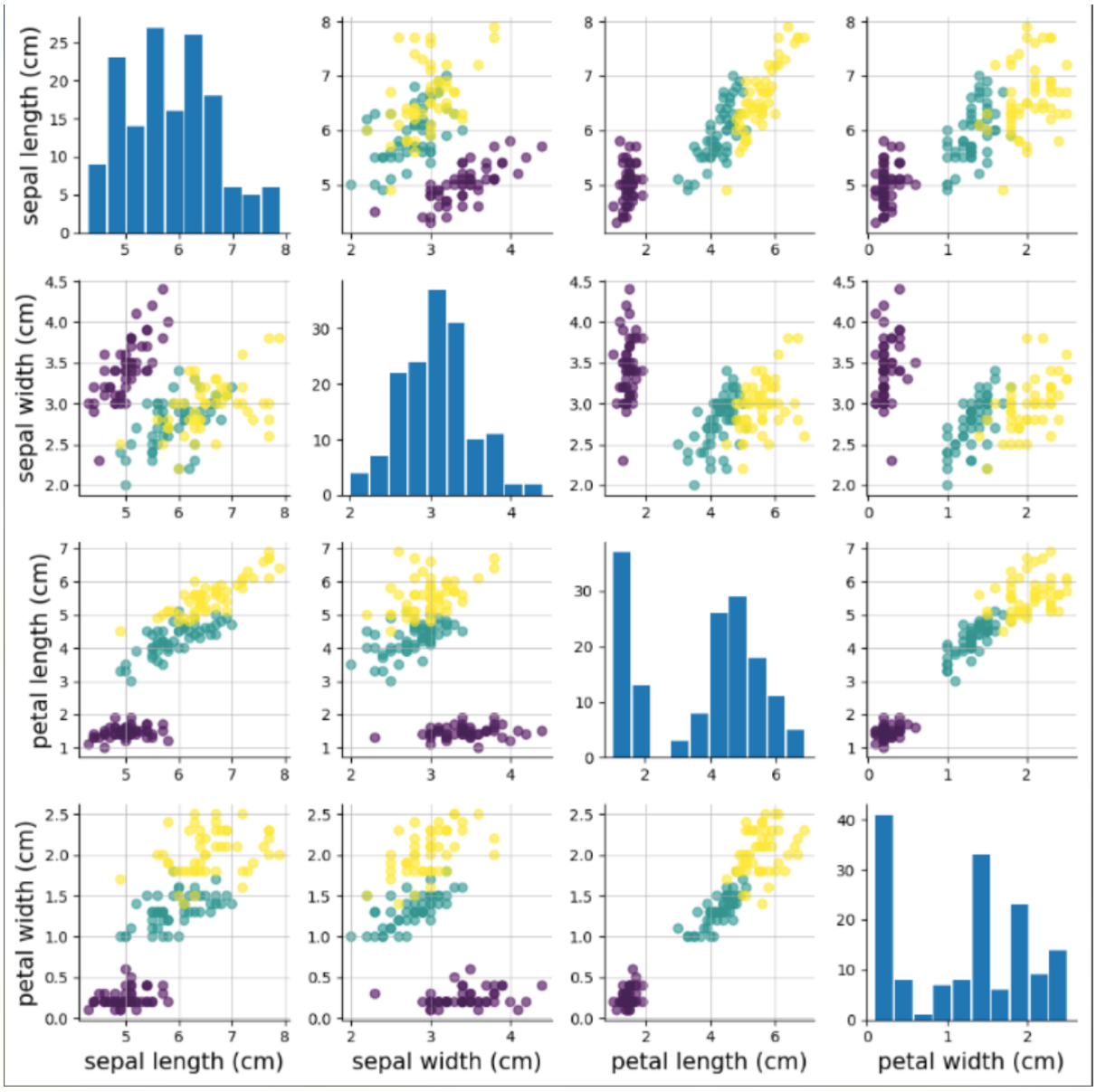
```
# iris_X에 데이터배열, iris_y에 target배열을 대입
iris_X, iris_y = iris.data, iris.target

fig, axes = plt.subplots(2, 2, figsize=(10, 10))
for ax_idx, ax in enumerate(axes.flatten()):   # axes.flatten()을 이용하여 (2, 2)형태의 fig에 위 왼->위 오->아래 왼->아래 오 순으로
    feature_data = []
    for species_idx in range(n_species):       # range(n_species) : 3, species_idx : 0, 1, 2  => data를 species_idx 종류에 따라 분류
        data = iris_X[iris_y == species_idx]   # iris_y값이 == species_idx(0~2)이 True면 해당하는 index위치의 값들을 반환
        data = data[:, ax_idx]                 # data[:, ax_idx] -> data의 컬럼별로 다시 data 변수에 대입

        feature_data.append(data)              # column에 해당하는 setosa, versicolor, virginica 로 구별되어 append

    ax.violinplot(feature_data)                # for문 안의 for문에서 column 별로 정리된 데이터를 가지고 ax.violinplot을 그림
    ax.set_title(feature_names[ax_idx], fontsize=15)
    ax.set_xticks([1, 2, 3])
    ax.set_xticklabels(species)
    ax.tick_params(labelsize=15)

plt.show()
```

### ▼ ax.hist : 막대그래프

```
np.random.seed(0)

n_data = 500
data = np.random.normal(0, 1, (n_data, ))

fig, ax = plt.subplots(figsize=(14, 10))
ax.tick_params(labelsize=30,
               length=10,
               width=3)

ax.hist(data, rwidth=0.9)  # rwidth : 각 막대그래프의 넓이 조절 param

plt.show()
```



### 시각화 실습 (5)




```
# 연습

import matplotlib.pyplot as plt
from sklearn.datasets import load_iris
import pandas as pd
import numpy as np

from sklearn.datasets import load_iris

iris = load_iris()
iris_df = pd.DataFrame(iris['data'],
                       columns=['SepalLength', 'SepalWidth', 'PetalLength', 'Petalwidth'])
iris_target = pd.DataFrame(iris['target'],
                           columns=['target'])
se_pe_lw = iris_df[['SepalLength', 'SepalWidth', 'PetalLength', 'Petalwidth']].values
```

```
species = iris_target['target'].values

# target feature별로 데이터 분화
se_pe_lw0 = se_pe_lw[species == 0]     # species : 'setosa'
se_pe_lw1 = se_pe_lw[species == 1]     # species : 'versicolor'
se_pe_lw2 = se_pe_lw[species == 2]     # species : 'virginica'

feature_names = iris.feature_names
species_names = iris.target_names

xticks = np.arange(3)

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

#ax.hist(iris_df['SepalLength'], rwidth=0.9)
ax.scatter(se_pe_lw0[:, 1], se_pe_lw0[:, 0], s=200, c='purple', alpha=0.5)
ax.scatter(se_pe_lw1[:, 1], se_pe_lw1[:, 0], s=200, c='green', alpha=0.5)
ax.scatter(se_pe_lw2[:, 1], se_pe_lw2[:, 0], s=200, c='yellow', alpha=0.5)

#fig, axes = plt.subplots(4, 4, figsize=(10, 10), sharex=True)
#for ax_idx, ax in enumerate(axes.flat):
#    ax.hist([se_pe_lw[:, ax_idx]], rwidth=0.9)
#    ax.scatter(se_pe_lw[:, ax_idx], se_pe_lw[:, ax_idx])
#    ax.set_xticks(xticks)
#    ax.set_xticklabels(species_names)
#    ax.set_title(feature_names[ax_idx], fontsize=15)

#ax[-1].set_xticks(xticks)
#ax[-1].set_xticklabels(species_names)
```

```
# 연습
import matplotlib.pyplot as plt
import numpy as np
import matplotlib.cm as cm
import pandas as pd

from sklearn.datasets import load_iris

iris = load_iris()

feature_names = iris.feature_names
n_feature = len(feature_names)
species = iris.target_names
n_species = len(species)

iris_X, iris_y = iris.data, iris.target

fig, axes = plt.subplots(2, 2, figsize=(10, 10))
for ax_idx, ax in enumerate(axes.flatten()):
    feature_data = []
    for species_idx in range(n_species):
        data = iris_X[iris_y == species_idx]
        data = data[:, ax_idx]
        feature_data.append(data)

    ax.violinplot(feature_data)
    ax.set_title(feature_names[ax_idx], fontsize=15)
    ax.set_xticks([1, 2, 3])
    ax.set_xticklabels(species)
    ax.tick_params(labelsize=15)

plt.show()
```

```
import matplotlib.pyplot as plt
import numpy as np
import pandas as pd
import matplotlib.cm as cm
from sklearn.datasets import load_iris

iris = load_iris()

feature_names = iris.feature_names
n_features = len(feature_names)
species = iris.target_names
n_species = len(species)
```

```
iris_df = pd.DataFrame(iris['data'],
                       columns=['SepalLength', 'SepalWidth', 'PetalLength', 'Petalwidth'])
iris_target = pd.DataFrame(iris['target'],
                           columns=['target'])
se_pe_lw = iris_df[['SepalLength', 'SepalWidth', 'PetalLength', 'Petalwidth']].values
species = iris_target['target'].values
colors = ['purple', 'green', 'yellow']
# target feature별로 데이터 분화
se_pe_lw0 = se_pe_lw[species == 0]     # species : 'setosa'
se_pe_lw1 = se_pe_lw[species == 1]     # species : 'versicolor'
se_pe_lw2 = se_pe_lw[species == 2]     # species : 'virginica'

feature_names = iris.feature_names
species_names = iris.target_names

xticks = np.arange(3)

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

# 1번 histogram
#ax.hist(iris_df['SepalLength'], rwidth=0.9)
# 1번 scatter
# ax.scatter(se_pe_lw0[:, 1], se_pe_lw0[:, 0], s=200, c='purple', alpha=0.5)
# ax.scatter(se_pe_lw1[:, 1], se_pe_lw1[:, 0], s=200, c='green', alpha=0.5)
# ax.scatter(se_pe_lw2[:, 1], se_pe_lw2[:, 0], s=200, c='yellow', alpha=0.5)

fig, axes = plt.subplots(4, 4, figsize=(10, 10), sharex=True)
for ax_idx, ax in enumerate(axes.flat):
    ax.hist([se_pe_lw[:, ax_idx]], rwidth=0.9)
    for idx in range(len(species)):
        data = np.hstack((se_pe_lw, ))
        ax.scatter(se_pe_lw0[:, 1], se_pe_lw0[:, idx], s=200, c=colors[idx], alpha=0.5)
        ax.scatter(se_pe_lw0[:, 2], se_pe_lw0[:, idx], s=200, c=colors[idx], alpha=0.5)
        ax.scatter(se_pe_lw0[:, 3], se_pe_lw0[:, idx], s=200, c=colors[idx], alpha=0.5)
        ax.set_xticks(xticks)
        ax.set_xticklabels(species_names)
        ax.set_title(feature_names[ax_idx], fontsize=15)

ax[-1].set_xticks(xticks)
ax[-1].set_xticklabels(species_names)
plt.show()
```