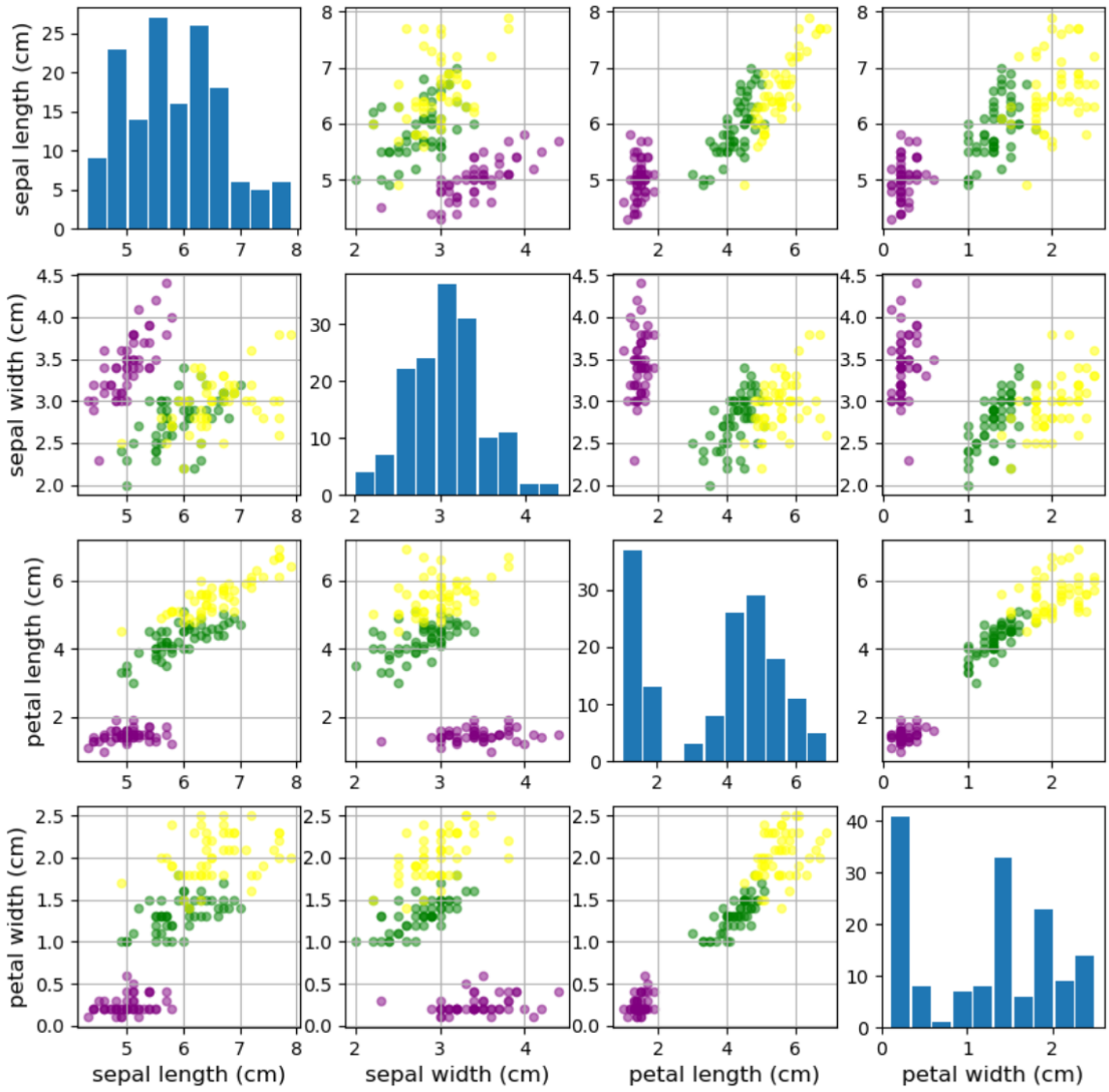
ML Day20 (Matplotlib) (Iris data)

```
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'])

iris_col_df = iris_df[['SepalLength', 'SepalWidth', 'PetalLength', 'Petalwidth']].values
iris_target_df = iris_target['target'].values

feature_names = iris.feature_names
n_feature = len(feature_names)
species = iris.target_names
n_species = len(species)
colors = ['purple', 'green', 'yellow']

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

iris_X0 = iris_col_df[iris_target_df == 0]     # species : 'setosa'
iris_X1 = iris_col_df[iris_target_df == 1]     # species : 'versicolor'
iris_X2 = iris_col_df[iris_target_df == 2]     # species : 'virginica'

xticks = np.arange(3)

fig, ax = plt.subplots(4, 4, figsize=(10, 10))

hist_idx_li = [0, 5, 10, 15]                   # histogram을 그리기 위해 histogram이 들어가는 plot index번호를 list에 저장

for i in range(4):                             # for문을 range(4)만큼에서 다시 for문을 range(4) = 4 x 4 plot을 그린다.
    for j in range(4):
        if i != j:                             # i != j (scatter를 그리는 plot)
            ax[i][j].scatter([iris_X0[:, j]], [iris_X0[:, i]], s=20, c='purple', alpha=0.5)     # for문의 변수에 해당하는 ax의 scatter를 구현
            ax[i][j].scatter([iris_X1[:, j]], [iris_X1[:, i]], s=20, c='green', alpha=0.5)
            ax[i][j].scatter([iris_X2[:, j]], [iris_X2[:, i]], s=20, c='yellow', alpha=0.5)
            ax[i][j].grid()
        if j == 0:                                                                           # j == 0 : ylabel name이 표시되어야할 부분
            ax[i][0].set_ylabel(feature_names[i], fontsize=12)

for i in range(4):
    for j in range(4):                                                                       # scatter를 그렸던 방식인 4x4 방식으로 hist
        if i == j:                                                                           # i == j: histogram이 그려져야할 plot
            ax[i][j].hist(iris_col_df[:, j], rwidth=0.9)
        if i == 3:                                                                           # i == 3: xlabel name들이 표시되어야 할 부
            ax[3][j].set_xlabel(feature_names[j], fontsize=12)

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

iris_X, iris_y = iris.data, iris.target

fig, axes = plt.subplots(4, 4, figsize=(10, 10))
for feature_idx, feature_name in enumerate(feature_names):        # feature_names를 enumerate를 사용하여 feature들의 idx와 name으로 for문을 
    feature_data = iris_X[:, feature_idx]                        # feature_data : iris_X의 feature_idx에 해당하는(각 column 별) 값들을 저
    axes[feature_idx, feature_idx].hist(feature_data, rwidth=0.9) # axes[feature_idx, feature_idx]에 해당하는 plot위치에 histogram을 그림

for row_idx in range(n_features):                                 # row_idx -> 0, 1, 2, 3
    for col_idx in range(n_features):                             # col_idx -> 0, 1, 2, 3
        if row_idx != col_idx:                                    # scatter를 그리려는 조건
            x_data = iris_X[:, col_idx]                           # col_idx(1~)로 iris_X를 slicing
            y_data = iris_X[:, row_idx]                           # row_idx(0~)로 iris_X를 slicing
            axes[row_idx, col_idx].scatter(x_data, y_data,        # axes[row_idx, col_idx] : 그려져야할 plot index
                                           c=iris_y, alpha=0.6)

for row_idx in range(n_features):
```

```
    for col_idx in range(n_features):
        if col_idx == 0:
            ylabel = feature_names[row_idx]
            axes[row_idx, col_idx].set_ylabel(ylabel, fontsize=15)
        if row_idx == 3:
            xlabel = feature_names[col_idx]
            axes[row_idx, col_idx].set_xlabel(xlabel, fontsize=15)

for row_idx in range(n_features):
    for col_idx in range(n_features):
        ax = axes[row_idx, col_idx]

        for spine_loc, spine in ax.spines.items():
            if spine_loc in ['right', 'top']:
                spine.set_visible(False)

        if row_idx != col_idx:
            ax.grid(lw=0.5)

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