Data Mining

군집화 과제

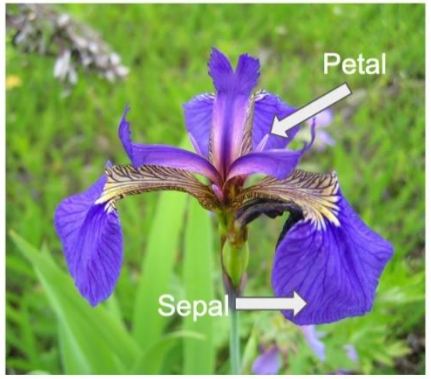
2021년 5월 6일

1. 데이터셋

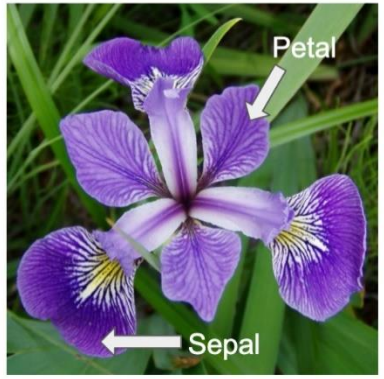
붓꽃 데이터셋 (Iris dataset)

세가지 붓꽃 종에 대한 데이터셋

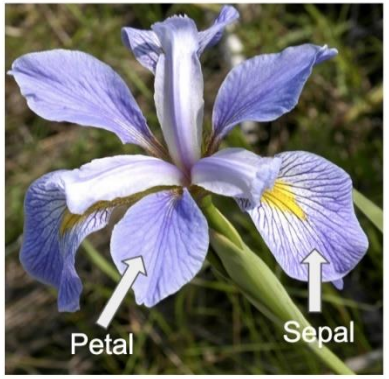
Iris setosa



Iris versicolor



Iris virginica



속성	설명	타입
sepal length (cm)	꽃받침 길이	continuous
sepal width (cm)	꽃받침 폭	continuous
petal length (cm)	꽃잎 길이	continuous
petal width (cm)	꽃잎 폭	continuous
target	붓꽃 종류 • Iris Setosa • Iris Versicolour • Iris Virginica	multi-valued discrete

- 150개 (각 종별로 50개씩)
- 1936년 영국 통계학자이자 생물학자인 도널드 피셔 (Ronald Fisher)의 논문에서 사용됨

https://archive.ics.uci.edu/ml/machine-learning-databases/iris/

붓꽃 데이터셋 (Iris dataset)

데이터 파일 :

5.1,3.5,1.4,0.2,Iris-setosa
4.9,3.0,1.4,0.2,Iris-setosa
4.7,3.2,1.3,0.2,Iris-setosa
4.6,3.1,1.5,0.2,Iris-setosa
5.0,3.6,1.4,0.2,Iris-setosa
5.4,3.9,1.7,0.4,Iris-setosa

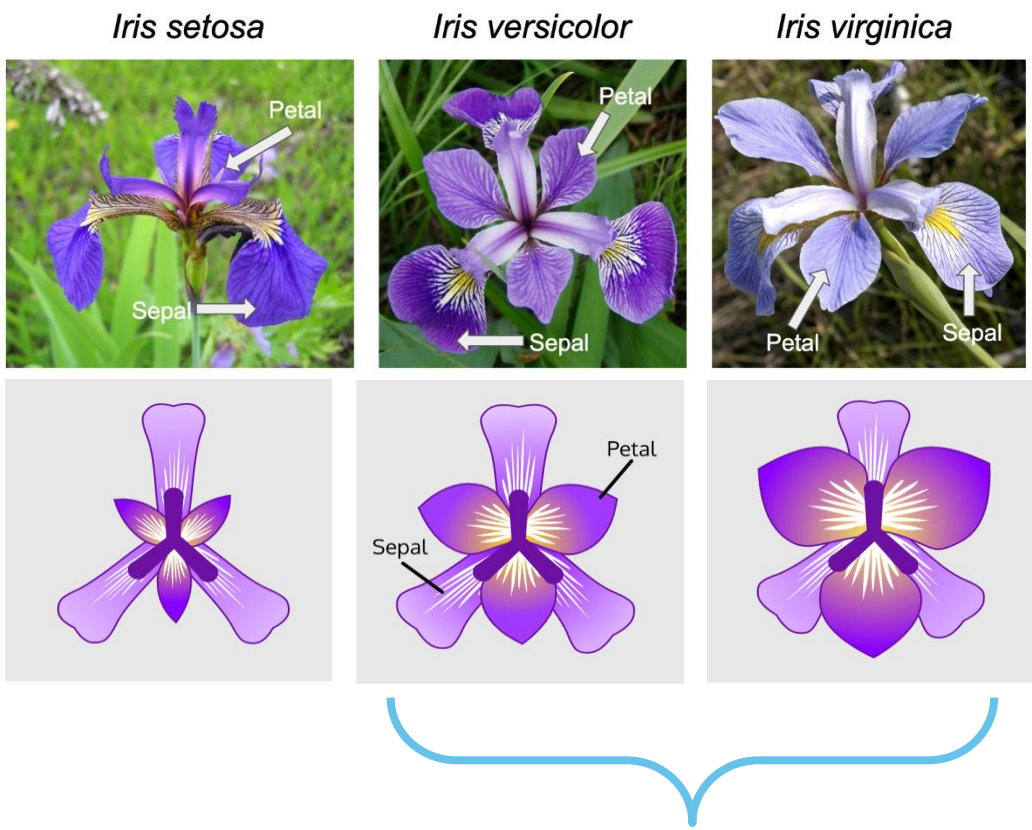
엑셀과 같은 형태로 포맷팅

sepal_length	sepal_width	petal_length	petal_width	species
5.1	3.5	1.4	0.2	setosa
4.9	3.0	1.4	0.2	setosa
4.7	3.2	1.3	0.2	setosa
4.6	3.1	1.5	0.2	setosa
5.0	3.6	1.4	0.2	setosa
5.4	3.9	1.7	0.4	setosa
4.6	3.4	1.4	0.3	setosa
5.0	3.4	1.5	0.2	setosa

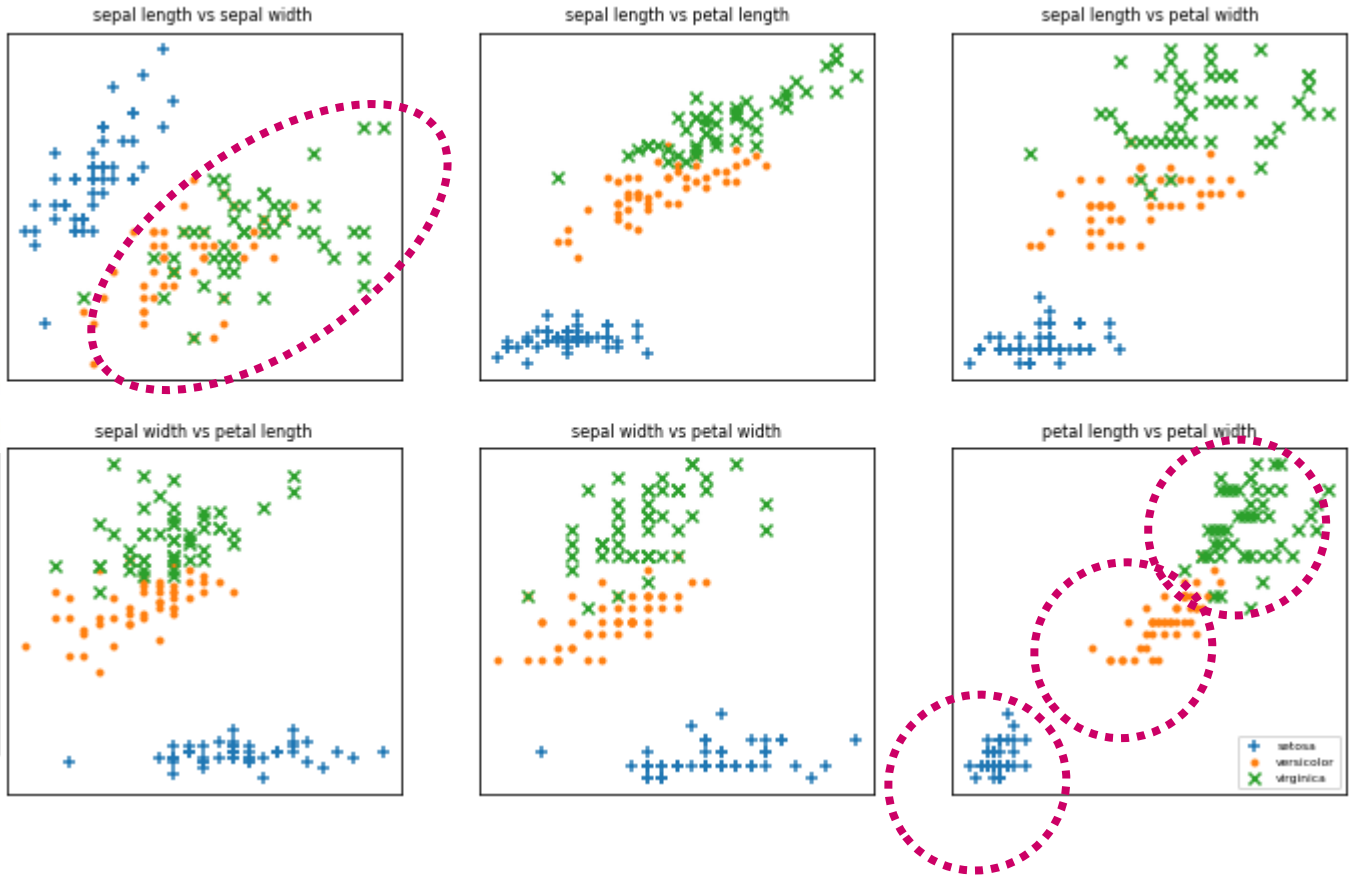
데이터 탐색

꽃받침의 길이와 너비로 'versicolor'와 'virginica' 종이 구분되지 않음

마커 : + • X



이 두 종은 꽃받침이 비슷하다.



꽃잎의 길이와 너비로 종별 구분이 잘됨

데이터셋 다운로드

데이터 다운로드

```
import requests
import os

data = requests.get("https://archive.ics.uci.edu/ml/machine-learning-
databases/iris/iris.data")
path = os.path.join('data', 'iris.data')
with open(path, "w") as f:
  f.write(data.text)
```

- URL에서 데이터를 다운로드해서 “iris.dat” 파일에 저장

데이터셋 읽기

데이터 읽기

```
import pandas as pd
column_names = ['sepal length', 'sepal width', 'petal length', 'petal width',
'species']
dataset = pd.read_csv(path, names=column_names)
dataset.sample(5)
```

⇒ 컬럼정보가 없어서 일부러 정해둠.

- Csv 파일에 column 이름이 없으므로 이름을 지정해서 읽어 옴

	sepal length	sepal width	petal length	petal width	species
0	5.1	3.5	1.4	0.2	Iris-setosa
1	4.9	3.0	1.4	0.2	Iris-setosa
2	4.7	3.2	1.3	0.2	Iris-setosa
3	4.6	3.1	1.5	0.2	Iris-setosa
4	5.0	3.6	1.4	0.2	Iris-setosa

데이터셋 읽기

데이터 정보

```
dataset.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 150 entries, 0 to 149
Data columns (total 5 columns):
 #   Column        Non-Null Count  Dtype  
---  ------        --------------  -----  
 0   sepal length  150 non-null    float64
 1   sepal width   150 non-null    float64
 2   petal length  150 non-null    float64
 3   petal width   150 non-null    float64
 4   species       150 non-null    object 
dtypes: float64(4), object(1)
memory usage: 6.0+ KB
```

2. 데이터 탐색

요약 통계량

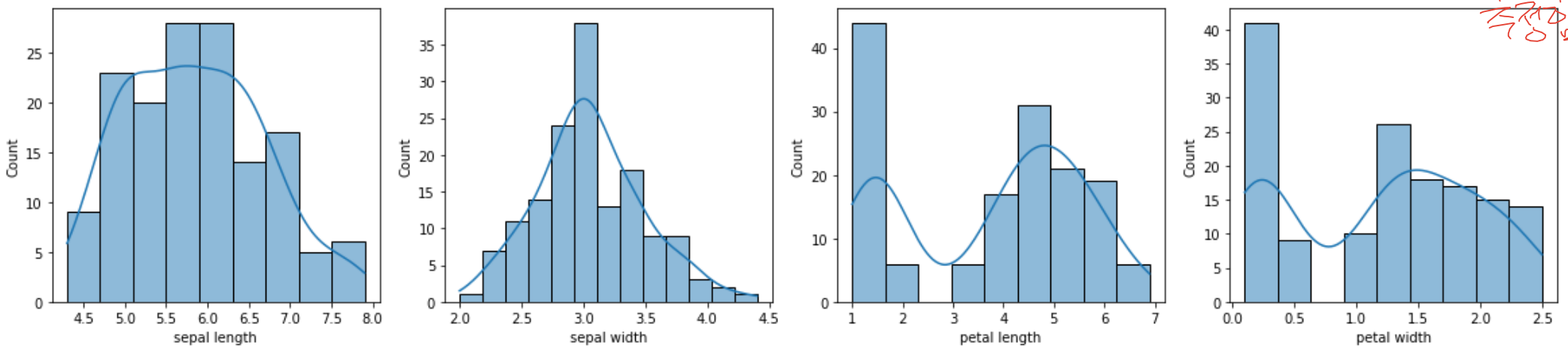
```
dataset.describe()
```

	sepal length	sepal width	petal length	petal width
count	150.000000	150.000000	150.000000	150.000000
mean	5.843333	3.054000	3.758667	1.198667
std	0.828066	0.433594	1.764420	0.763161
min	4.300000	2.000000	1.000000	0.100000
25%	5.100000	2.800000	1.600000	0.300000
50%	5.800000	3.000000	4.350000	1.300000
75%	6.400000	3.300000	5.100000	1.800000
max	7.900000	4.400000	6.900000	2.500000

히스토그램 (단일 변수 분석)

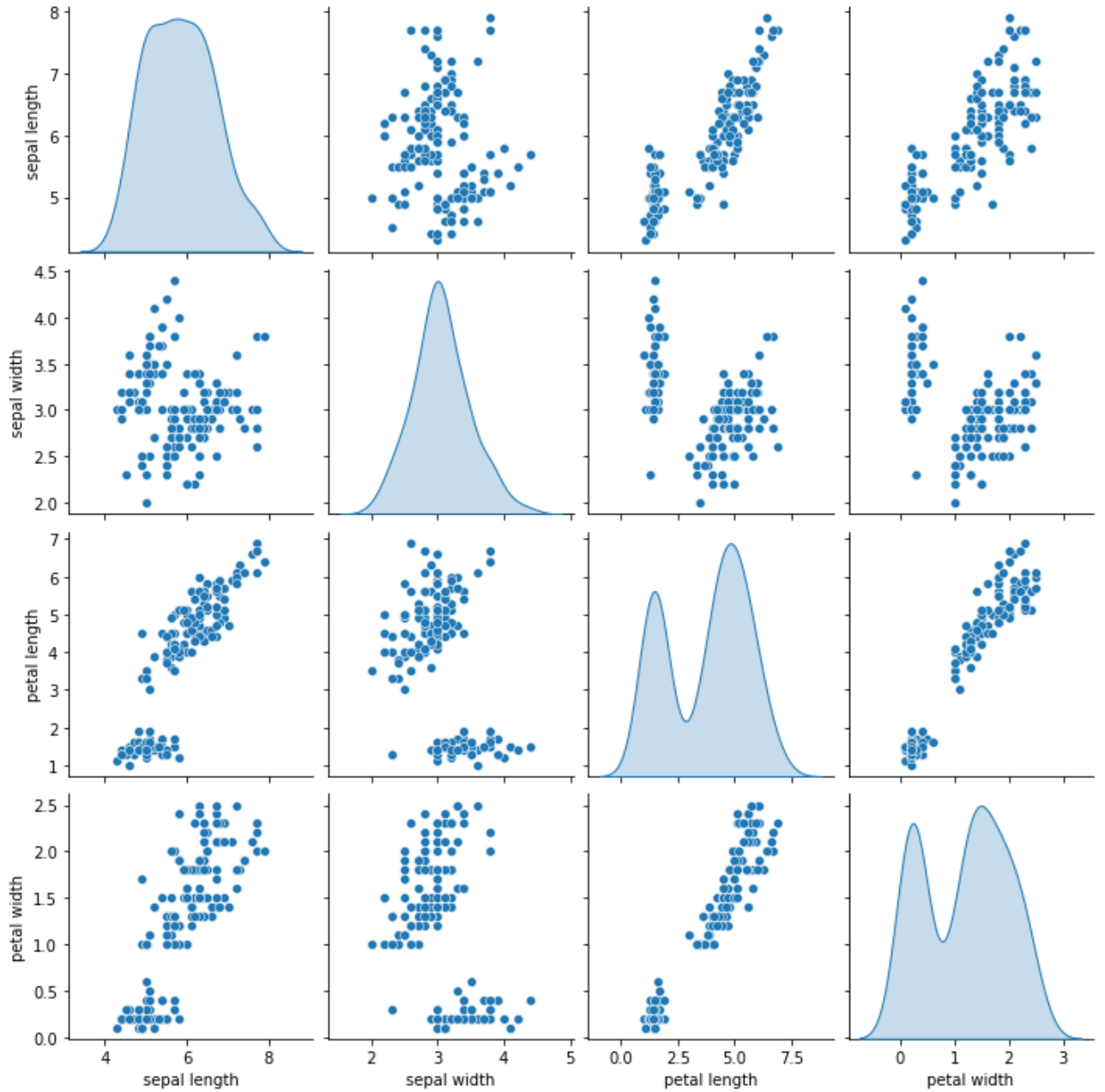
```
import matplotlib.pyplot as plt
import seaborn as sns

plt.figure(figsize=[20,4])
for i, column in enumerate(dataset.describe().columns):
    plt.subplot(1,4,i+1)
    sns.histplot(data=dataset, x=column, kde=True)
plt.show()
```



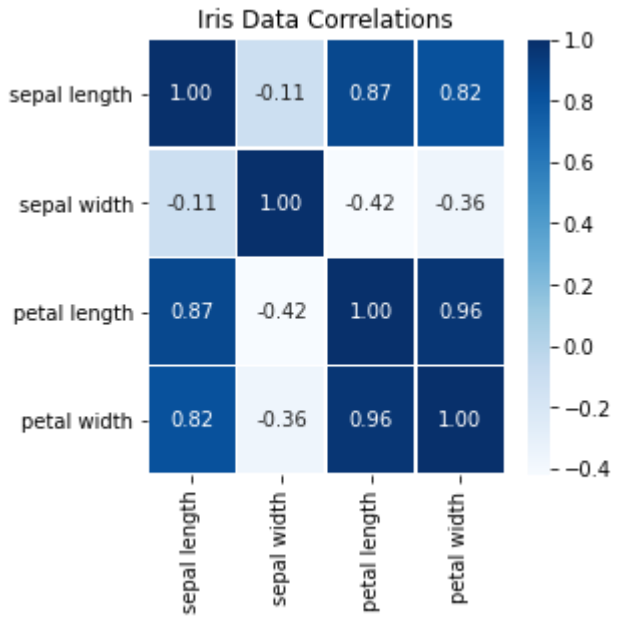
두 변수 관계 분석

```
sns.pairplot(dataset)
plt.show()
```



히트맵

```
fig, ax = plt.subplots(figsize=(4, 4))
sns.heatmap(dataset.corr(), linewidths=.5, annot=True, fmt=".2f", cmap='Blues')
plt.title('Iris Data Correlations')
plt.show()
```



3. 데이터 전처리

데이터 추출

```
clusterdata = dataset.iloc[:,:-1]
inputs = clusterdata.iloc[:,:].values.tolist()
columns = clusterdata.keys().tolist()
column2index ={ column : i for i, column in enumerate(columns)}
print('columns = ', columns)
print('column2index = ', column2index)
```

columns = ['sepal length', 'sepal width', 'petal length', 'petal width']
column2index = {'sepal length': 0, 'sepal width': 1, 'petal length': 2, 'petal width': 3}

데이터 표준화

```
from scratch.working_with_data import scale, rescale, Vector
from typing import List

inputs_normed = rescale(inputs)
```

4. K-평균 군집화

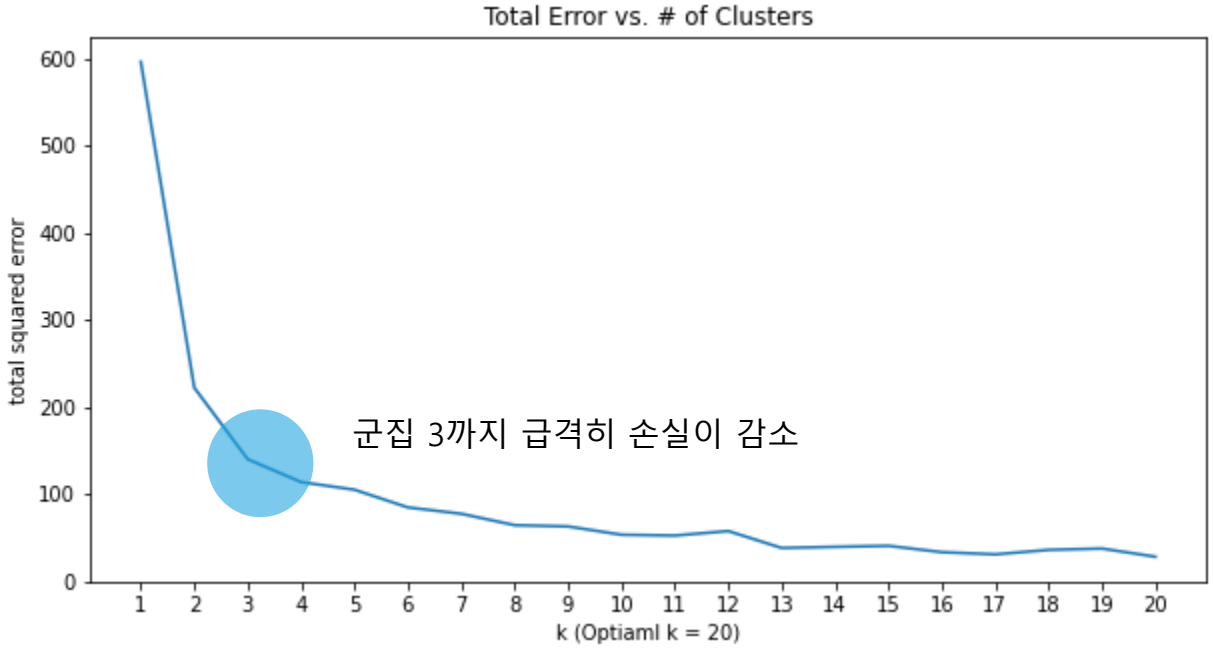
군집 개수 K 선택 (Q1)

손실 그래프를 그려서 군집 개수 K를 찾아보시오.
단, K는 20까지 확인해 볼 것

그래프 그리기

```
import random
optimal_k = errors.index(min(errors)) + 1

fig, ax = plt.subplots(figsize=(10, 5))
plt.plot(ks, errors)
plt.xticks(ks)
plt.xlabel(f"k (Optiaml k = {optimal_k})")
plt.ylabel("total squared error")
plt.title("Total Error vs. # of Clusters")
plt.show()
```



$K = 3$ 군집화 (Q2)

$K = 3$으로 군집화를 해서 다음과 같이 군집화 결과를 확인해 보라.

Dataset에 k_means 군집화 결과 추가

```
dataset["k_means"] = assignments
dataset.head()
```

	sepal length	sepal width	petal length	petal width	species	k_means
0	5.1	3.5	1.4	0.2	Iris-setosa	1
1	4.9	3.0	1.4	0.2	Iris-setosa	1
2	4.7	3.2	1.3	0.2	Iris-setosa	1
3	4.6	3.1	1.5	0.2	Iris-setosa	1
4	5.0	3.6	1.4	0.2	Iris-setosa	1

$K = 3$ 군집화

Cluster 0에 데이터 확인

```
dataset[dataset['k_means']==0].head()
```

	sepal length	sepal width	petal length	petal width	species	k_means
50	7.0	3.2	4.7	1.4	Iris-versicolor	0
51	6.4	3.2	4.5	1.5	Iris-versicolor	0
52	6.9	3.1	4.9	1.5	Iris-versicolor	0
56	6.3	3.3	4.7	1.6	Iris-versicolor	0
65	6.7	3.1	4.4	1.4	Iris-versicolor	0

$K = 3$ 군집화

K_means 결과와 species 비교

```
dataset.groupby(["k_means", "species"])['k_means'].count()
```

```
k_means  species
0        Iris-setosa         1
         Iris-versicolor    37
         Iris-virginica      8
1        Iris-setosa        49
2        Iris-versicolor    13
         Iris-virginica     42
Name: k_means, dtype: int64
```

classify. assignment 한뭉텅을 콕 바꿔서 머신 러닝 호출

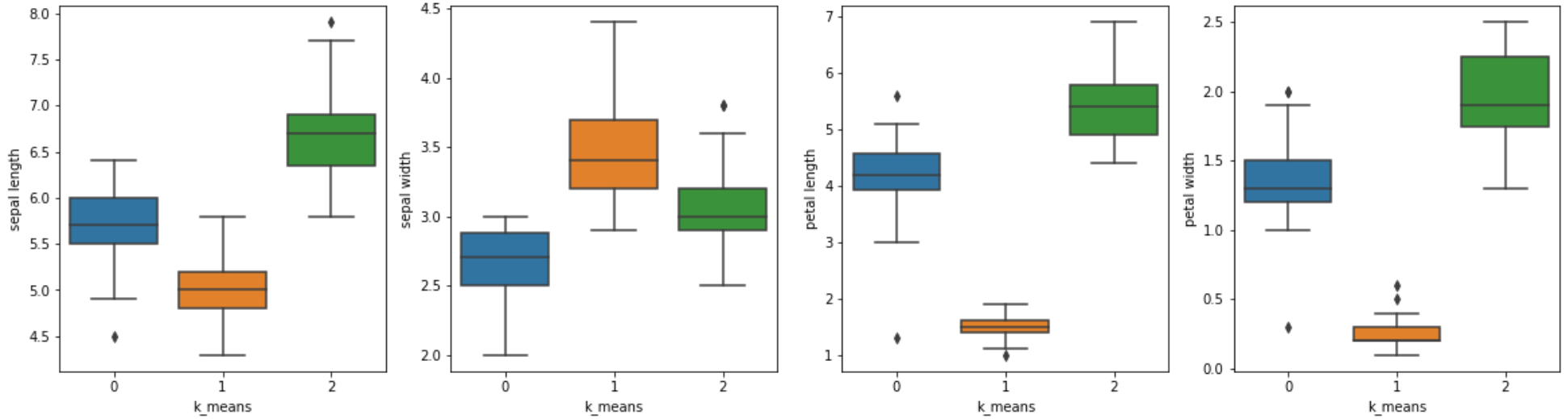
군집 0에는 versicolor가 군집 1에는 setosa가 군집 2에는 versicolor와 virginica가 군집화 됨

군집화 및 결과 확인

박스 플롯으로 군집 별 범위 확인

```
plt.subplots(figsize=(20, 5))
plt.subplot(1,4,1)
sns.boxplot(x = 'k_means', y = 'sepal length', data= dataset)
plt.subplot(1,4,2)
sns.boxplot(x = 'k_means', y = 'sepal width', data= dataset)
plt.subplot(1,4,3)
sns.boxplot(x = 'k_means', y = 'petal length', data= dataset)
plt.subplot(1,4,4)
sns.boxplot(x = 'k_means', y = 'petal width', data= dataset)
plt.show()
```

Cluster 0 : versicolor
Cluster 1 : setosa
Cluster 2 : versicolor , virginica



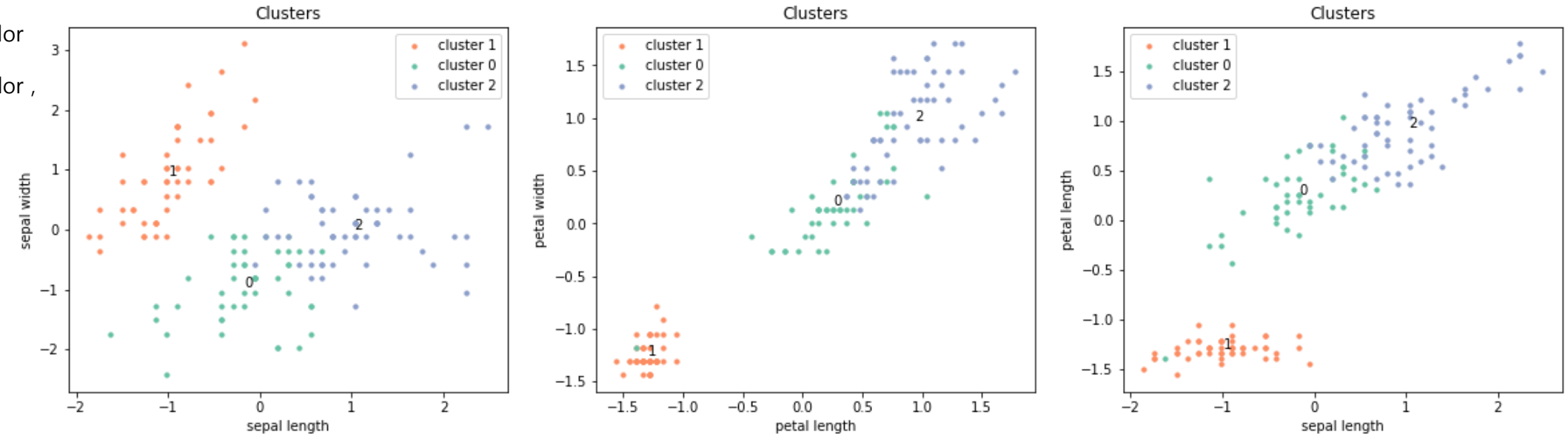
군집화 및 결과 확인 (Q3)

각 군집이 구분되도록 두 변수의 산포도를 그리는 함수 plot_cluster 구현하시오.

꼰약강만 스정

```
plt.subplots(figsize=(20, 5))
plt.subplot(1,3,1)
plot_cluster(clusters, column2index["sepal length"], column2index["sepal width"])
plt.subplot(1,3,2)
plot_cluster(clusters, column2index["petal length"], column2index["petal width"])
plt.subplot(1,3,3)
plot_cluster(clusters, column2index["sepal length"], column2index["petal length"])
plt.show()
```

Cluster 0 : versicolor
Cluster 1 : setosa
Cluster 2 : versicolor , virginica



5. 상향식 계층 군집화

$K = 3$ 군집화 (Q4)

$K = 3$으로 최장 거리(max) 기준으로 군집화를 해서 다음과 같이 군집화 결과를 확인해 보라.

Dataset에 h_clustering 군집화 결과 추가

```
dataset["h_clustering"] = h_assignments
dataset.head()
```

	sepal length	sepal width	petal length	petal width	species	k_means	h_clustering
0	5.1	3.5	1.4	0.2	Iris-setosa	1	1
1	4.9	3.0	1.4	0.2	Iris-setosa	1	1
2	4.7	3.2	1.3	0.2	Iris-setosa	1	1
3	4.6	3.1	1.5	0.2	Iris-setosa	1	1
4	5.0	3.6	1.4	0.2	Iris-setosa	1	1

$K = 3$ 군집화

군집화 결과는 (군집 번호(k), 데이터 포인트의 리스트) 딕셔너리

```
h_clusters = {0 : [[2.242171976289676, -0.12454037930146161, 1.32697020894639, 1.4431210532119516],
                   [2.1214086741555813, -0.12454037930146161, 1.6103493822692243, 1.181053065340532],...],
              1 : [...                                                                                ],
              2 : [...                                                                                ]}
```

```
class Leaf(NamedTuple):
    index : int
    value: Vector
```

1. **Leaf에 index를 저장**
2. **군집 별로 index 목록 생성**
3. **각 index 별로 배정된 k 리스트를 Assignment로 생성**

이런형태로 바꾸자.

h_assignment는 각 데이터 포인트가 속한 군집 번호 (k) 리스트

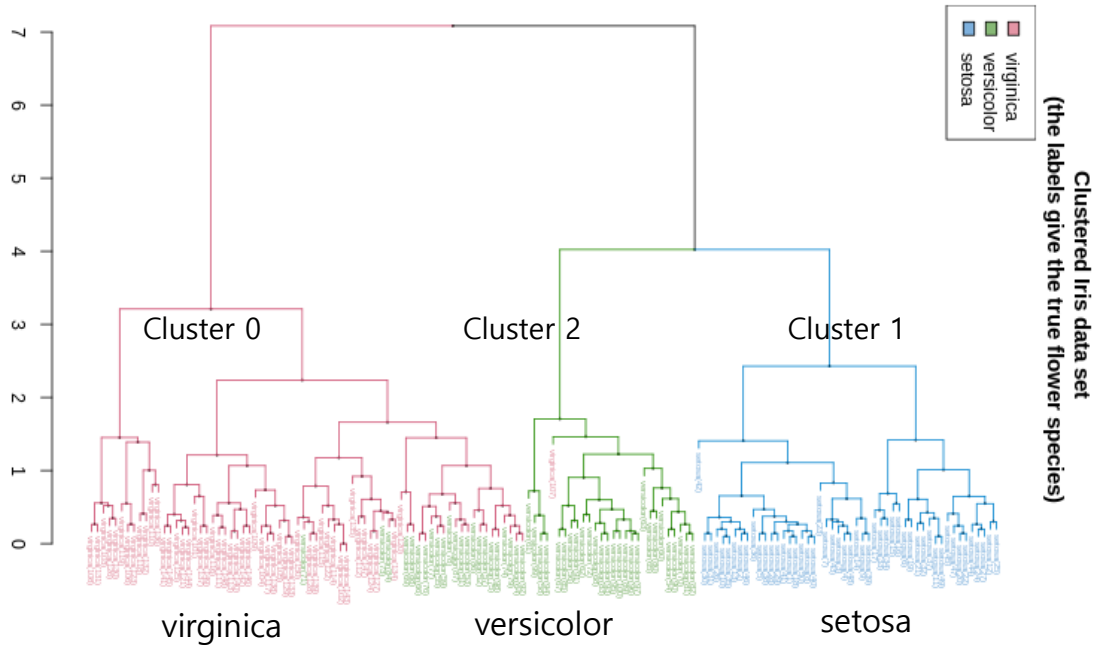
h_assignments= [1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1,
1, 1, 1, 2, 1, 1, 1, 1, 1, 1, 1, 1, 0, 0, 0, 2, 0, 2, 0, 2, 0, 2, 2, 0, 2, 0, 0, 0, 0, 2, 2, 2, 0, 0, 0, 0, 0, 0, 0, 0, 0, 2, 2, 2, 2,
0, 0, 0, 0, 2, 0, 2, 2, 0, 2, 2, 2, 0, 0, 0, 2, 2, 0, 0, 0, 0, 0, 0, 2, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 2, 0, 0, 0, 0, 0, 0, 0, 0,
0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0]

$K = 3$ 군집화

K_clustering 결과와 species 비교

```
dataset.groupby(["h_clustering", "species"])[h_clustering'].count()
```

```
h_clustering   species
0              Iris-versicolor    29
               Iris-virginica     48
1              Iris-setosa        49
2              Iris-setosa         1
               Iris-versicolor    21
               Iris-virginica      2
Name: h_clustering, dtype: int64
```

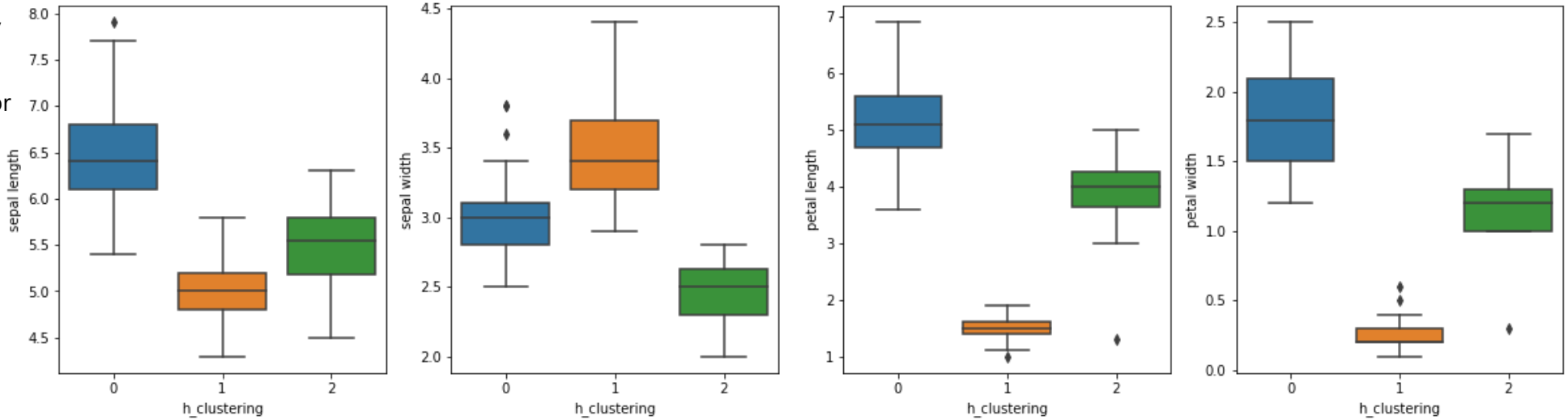


군집화 및 결과 확인

박스 플롯으로 군집 별 범위 확인

```
plt.subplots(figsize=(20, 5))
plt.subplot(1,4,1)
sns.boxplot(x = 'h_clustering', y = 'sepal length', data= dataset)
plt.subplot(1,4,2)
sns.boxplot(x = 'h_clustering', y = 'sepal width', data= dataset)
plt.subplot(1,4,3)
sns.boxplot(x = 'h_clustering', y = 'petal length', data= dataset)
plt.subplot(1,4,4)
sns.boxplot(x = 'h_clustering', y = 'petal width', data= dataset)
plt.show()
```

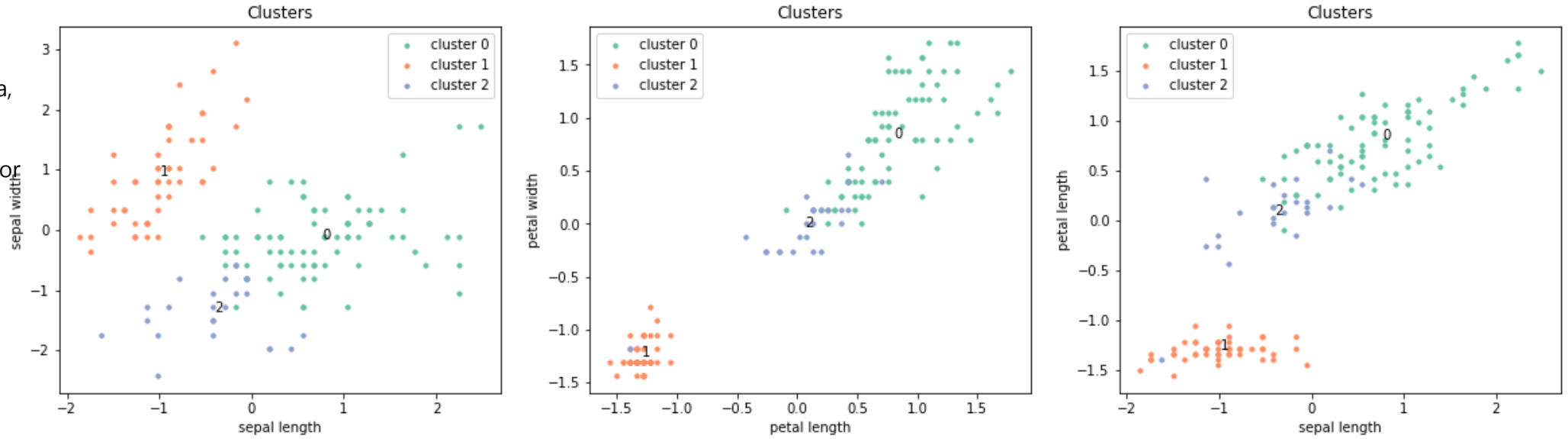
Cluster 0 : virginica, versicolor
Cluster 1 : setosa
Cluster 2 : versicolor



군집화 및 결과 확인

```
plt.subplots(figsize=(20, 5))
plt.subplot(1,3,1)
plot_cluster(h_clusters, column2index["sepal length"], column2index["sepal width"])
plt.subplot(1,3,2)
plot_cluster(h_clusters, column2index["petal length"], column2index["petal width"])
plt.subplot(1,3,3)
plot_cluster(h_clusters, column2index["sepal length"], column2index["petal length"])
plt.show()
```

Cluster 0 : virginica, versicolor
Cluster 1 : setosa
Cluster 2 : versicolor



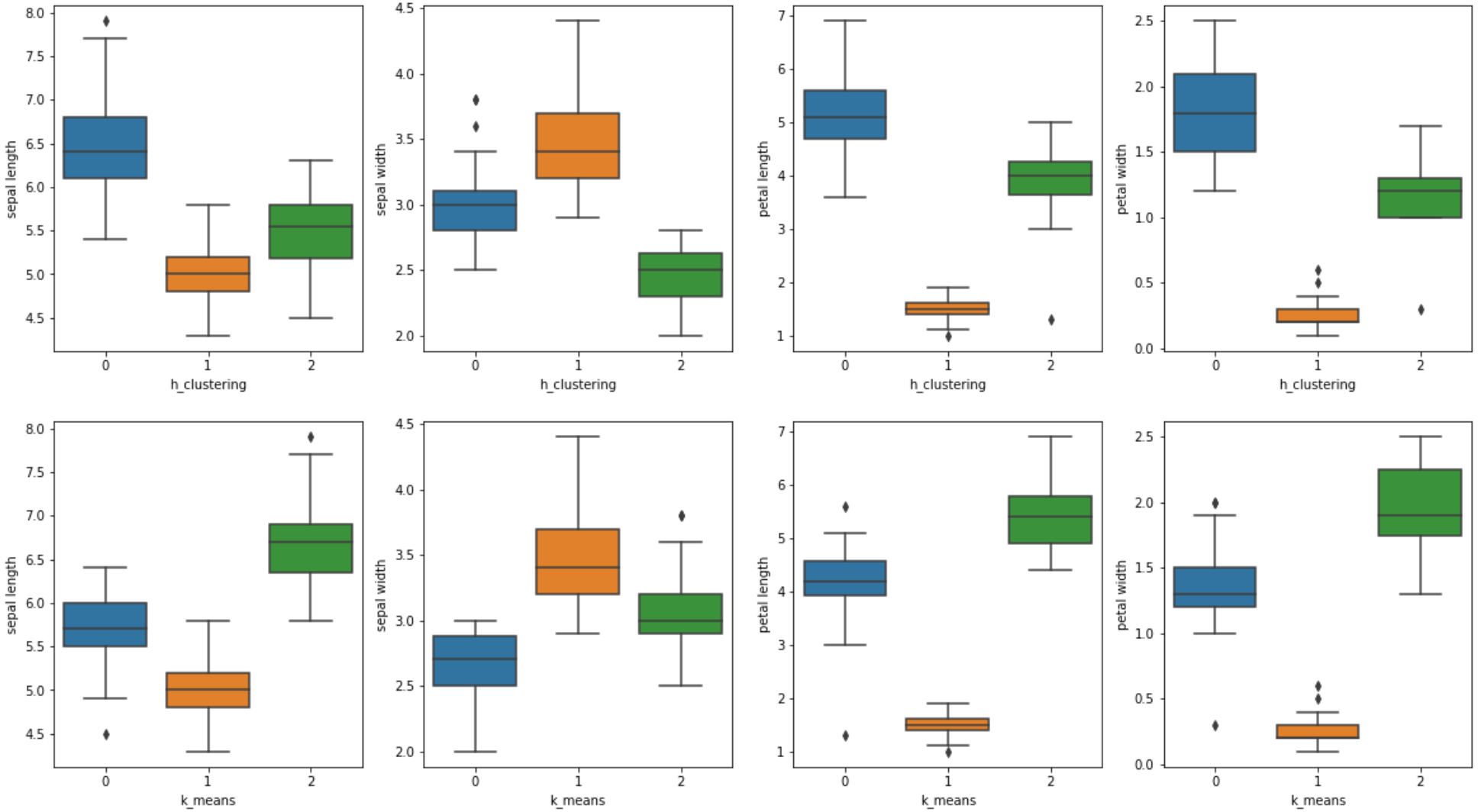
k-평균 vs. 상향식 계층 군집화 결과 비교

상향식 계층 군집화

Cluster 0 : virginica, versicolor
Cluster 1 : setosa
Cluster 2 : versicolor

K-평균

Cluster 0 : versicolor
Cluster 1 : setosa
Cluster 2 : versicolor , virginica



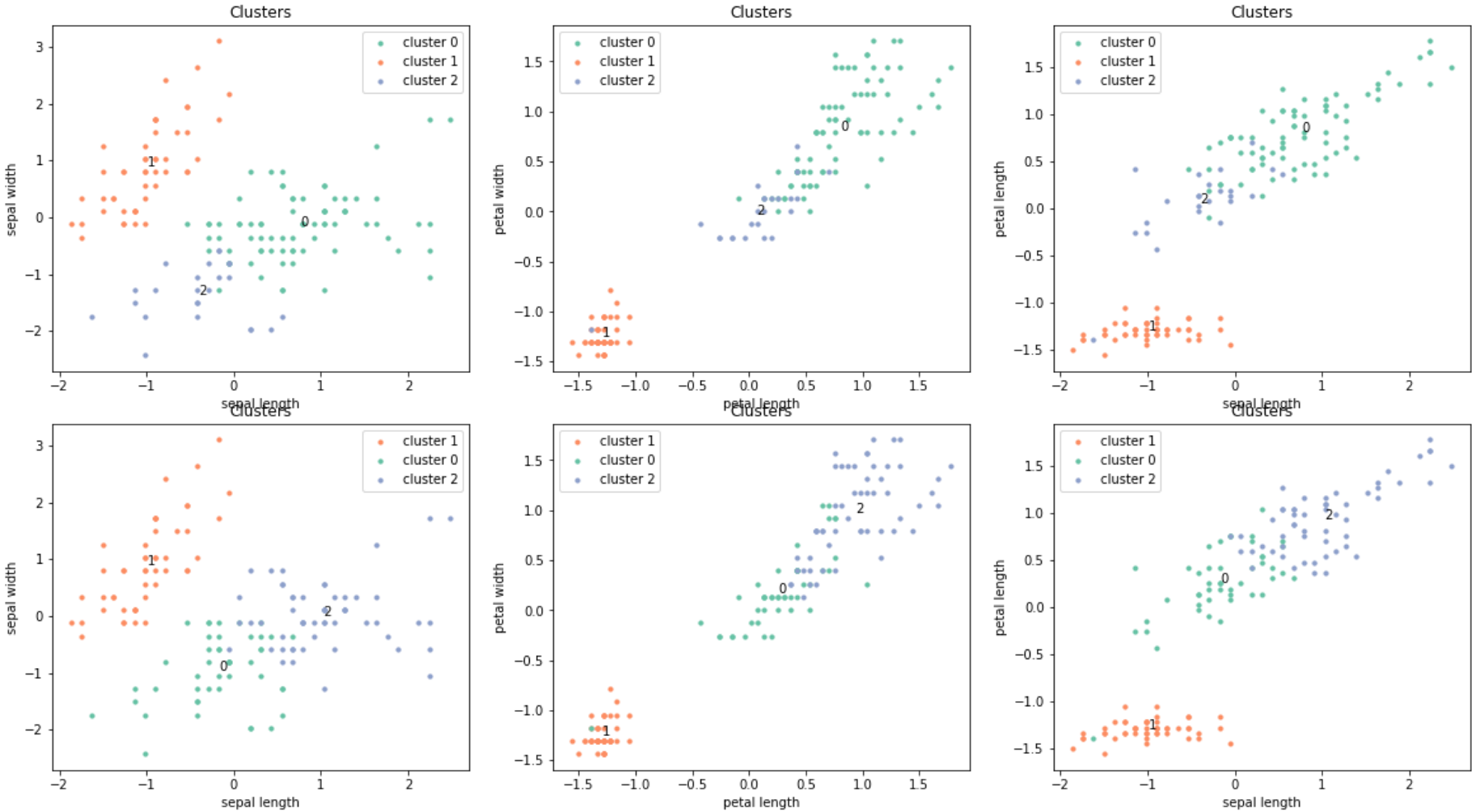
k-평균 vs. 상향식 계층 군집화 결과 비교

상향식 계층 군집화

Cluster 0 : virginica, versicolor
Cluster 1 : setosa
Cluster 2 : versicolor

K-평균

Cluster 0 : versicolor
Cluster 1 : setosa
Cluster 2 : versicolor , virginica



Thank you!