계층적 군집 (2)

kaggle customer 데이터 셋 적용

#01. 패키지 참조

```
import seaborn as sb
import numpy as np
from pandas import read_excel
from matplotlib import pyplot as plt
from scipy.cluster.hierarchy import dendrogram, linkage
from sklearn.cluster import AgglomerativeClustering
from sklearn.preprocessing import StandardScaler
```

#02. 데이터 가져오기

```
origin = read_excel("https://data.hossam.kr/G02/customer.xlsx")
print(origin.info())
origin.head()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 200 entries, 0 to 199
Data columns (total 5 columns):
 #   Column  Non-Null Count  Dtype
---  ------  --------------  -----
 0   고객ID    200 non-null    int64
 1   성별      200 non-null    object
 2   나이      200 non-null    int64
 3   연수입     200 non-null    int64
 4   지출점수    200 non-null    int64
dtypes: int64(4), object(1)
memory usage: 7.9+ KB
None
```

	고객ID	성별	나이	연수입	지출점수
0	1	Male	19	15	39
1	2	Male	21	15	81
2	3	Female	20	16	6
3	4	Female	23	16	77
4	5	Female	31	17	40

#03. 데이터 전처리

1. 필요한 변수만 추출

```
x = origin.filter(['연수입', '지출점수'])
x.head()
```

	연수입	지출점수
0	15	39
1	15	81
2	16	6
3	16	77
4	17	40

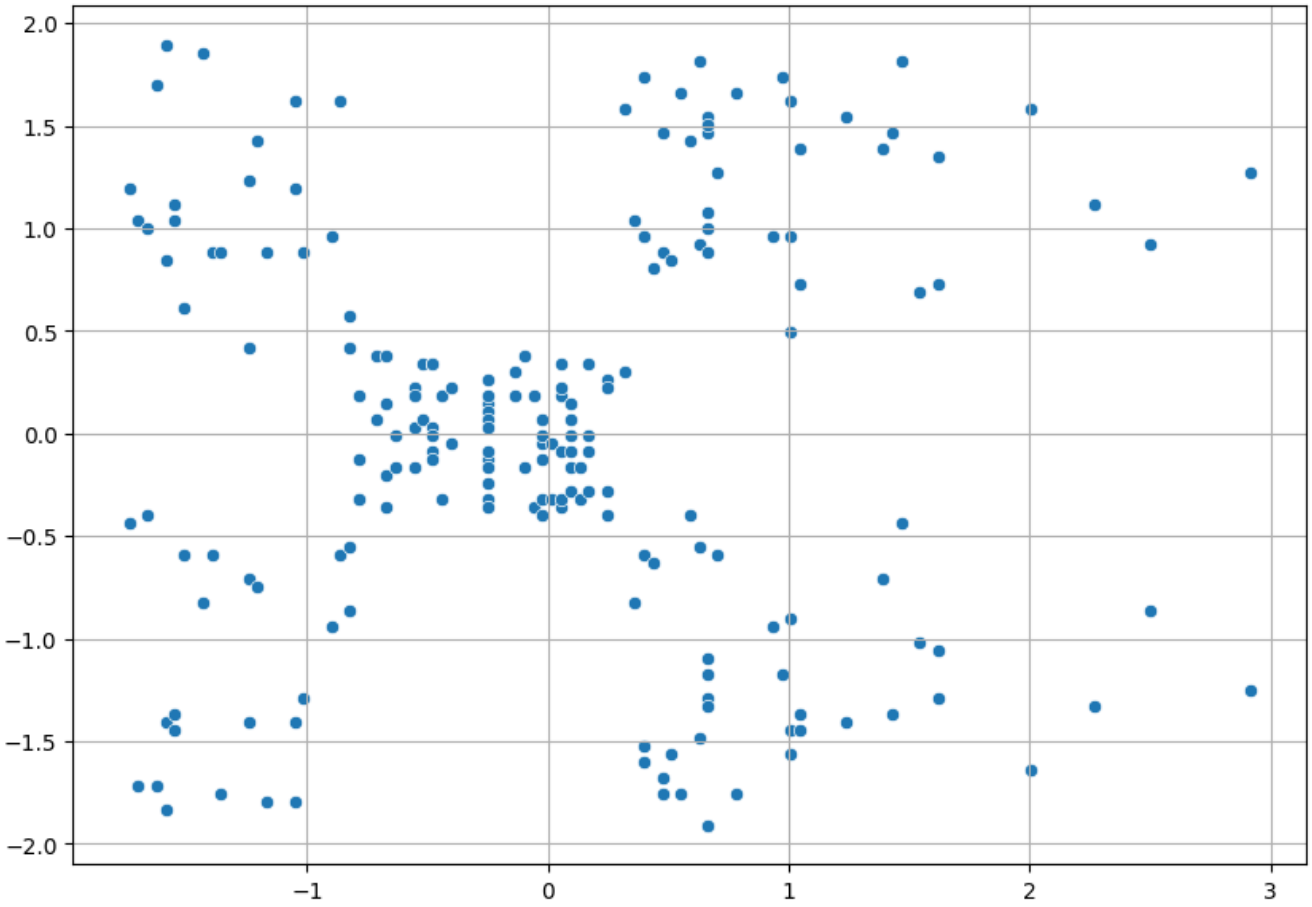
2. 데이터 표준화

```
scaler = StandardScaler()
n_data = scaler.fit_transform(x)
n_data[:5]
```

```
array([[-1.73899919, -0.43480148],
       [-1.73899919,  1.19570407],
       [-1.70082976, -1.71591298],
       [-1.70082976,  1.04041783],
       [-1.66266033, -0.39597992]])
```

3. 데이터 분포 확인

```
plt.figure(figsize=(10, 7))
sb.scatterplot(x=n_data[:, 0], y=n_data[:, 1])
plt.grid()
plt.show()
plt.close()
```

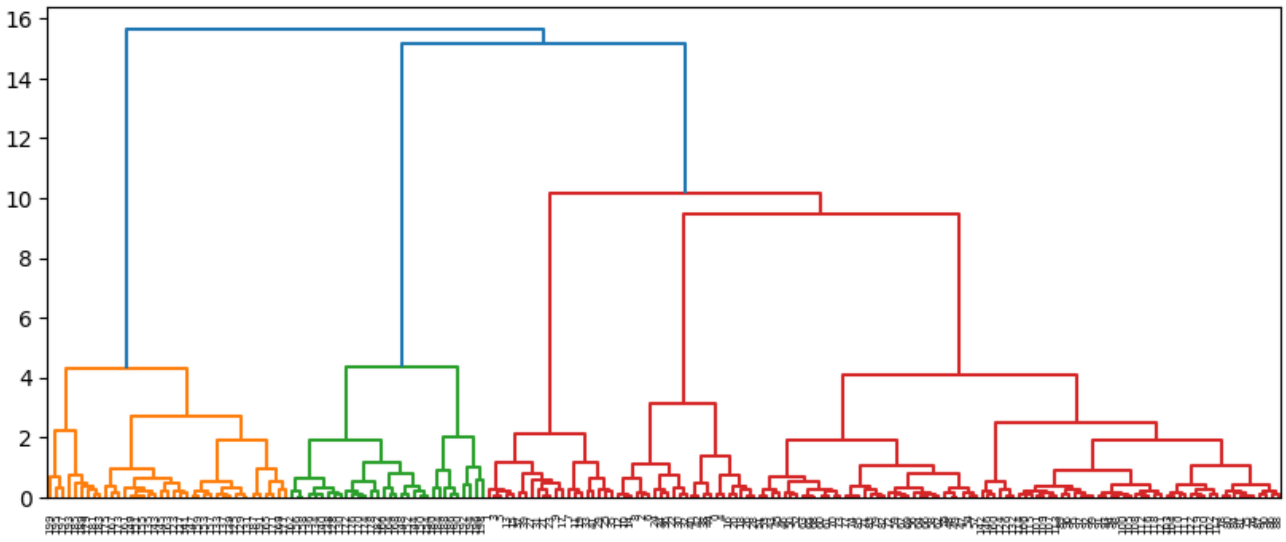


#04. 계층 군집 수행

1. scipy 패키지

```
lnk = linkage(n_data, method='ward')

plt.figure(figsize=(10, 4))
dendrogram(lnk, show_leaf_counts=True)
plt.show()
plt.close()
```



2. sklearn 패키지

```
ac = AgglomerativeClustering(n_clusters=2, metric='euclidean', linkage='ward',
                             compute_distances=True)
clustering = ac.fit(n_data)

counts = np.zeros(clustering.children_.shape[0])
n_samples = len(clustering.labels_)

for i, merge in enumerate(clustering.children_):
    current_count = 0
    for child_idx in merge:
        if child_idx < n_samples:
            current_count += 1  # leaf node
        else:
            current_count += counts[child_idx - n_samples]
    counts[i] = current_count

linkage_matrix = np.column_stack(
    [clustering.children_, clustering.distances_, counts]
).astype(float)

plt.figure(figsize=(10, 4))
dendrogram(linkage_matrix)
plt.show()
plt.close()
```