K-means Clustering - 군집분석

```
import warnings
warnings.filterwarnings('ignore')
```

I. Import Packages and Lead Dataset

1) Import Packages

```
import matplotlib.pyplot as plt
import numpy as np
import pandas as pd
```

2) Load Dataset

- Load iris Dataset

```
from sklearn.datasets import load_iris

iris = load_iris()
```

- iris : Dictionary
  - X : iris.data
  - y : iris.target

```
iris
```

```
[7.0, 3. , 6.0, 2.1],
[4.9, 2.5, 4.5, 1.7],
[7.3, 2.9, 6.3, 1.8],
[6.7, 2.5, 5.8, 1.8],
[7.2, 3.6, 6.1, 2.5],
[6.5, 3.2, 5.1, 2. ],
[6.4, 2.7, 5.3, 1.9],
[6.8, 3. , 5.5, 2.1],
[5.7, 2.5, 5. , 2. ],
[5.8, 2.8, 5.1, 2.4],
[6.4, 3.2, 5.3, 2.3],
[6.5, 3. , 5.5, 1.8],
[7.7, 3.8, 6.7, 2.2],
[7.7, 2.6, 6.9, 2.3],
[6. , 2.2, 5. , 1.5],
```

```
       [6. , 2.2, 5. , 1.5],
       [6.9, 3.2, 5.7, 2.3],
       [5.6, 2.8, 4.9, 2. ],
       [7.7, 2.8, 6.7, 2. ],
       [6.3, 2.7, 4.9, 1.8],
       [6.7, 3.3, 5.7, 2.1],
       [7.2, 3.2, 6. , 1.8],
       [6.2, 2.8, 4.8, 1.8],
       [6.1, 3. , 4.9, 1.8],
       [6.4, 2.8, 5.6, 2.1],
       [7.2, 3. , 5.8, 1.6],
       [7.4, 2.8, 6.1, 1.9],
       [7.9, 3.8, 6.4, 2. ],
       [6.4, 2.8, 5.6, 2.2],
       [6.3, 2.8, 5.1, 1.5],
       [6.1, 2.6, 5.6, 1.4],
       [7.7, 3. , 6.1, 2.3],
       [6.3, 3.4, 5.6, 2.4],
       [6.4, 3.1, 5.5, 1.8],
       [6. , 3. , 4.8, 1.8],
       [6.9, 3.1, 5.4, 2.1],
       [6.7, 3.1, 5.6, 2.4],
       [6.9, 3.1, 5.1, 2.3],
       [5.8, 2.7, 5.1, 1.9],
       [6.8, 3.2, 5.9, 2.3],
       [6.7, 3.3, 5.7, 2.5],
       [6.7, 3. , 5.2, 2.3],
       [6.3, 2.5, 5. , 1.9],
       [6.5, 3. , 5.2, 2. ],
       [6.2, 3.4, 5.4, 2.3],
       [5.9, 3. , 5.1, 1.8]]),
 'feature_names': ['sepal length (cm)',
  'sepal width (cm)',
  'petal length (cm)',
  'petal width (cm)'],
 'filename': '/usr/local/lib/python3.7/dist-packages/sklearn/datasets/data/iris.csv',
 'target': array([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, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,
       0, 0, 0, 0, 0, 0, 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, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2,
       2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2,
       2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2]),
 'target_names': array(['setosa', 'versicolor', 'virginica'], dtype='<U10')}
```

- pandas DataFrame

```
DF = pd.DataFrame(data = iris.data,
                  columns = ['sepal_length',
                             'sepal_width',
                             'petal_length',
                             'petal_width'])
```

```
DF.head(3)
```

	sepal_length	sepal_width	petal_length	petal_width
0	5.1	3.5	1.4	0.2
1	4.9	3.0	1.4	0.2
2	4.7	3.2	1.3	0.2

II. K-means Clustering

1) Modeling

- n_clusters : 군집 개수 지정
- init : 초기 중심 설정 방식(기본값)
- max_iter : 최대 반복 횟수

```
from sklearn.cluster import KMeans

kmeans_3 = KMeans(n_clusters = 3,
                  init ='k-means++',
                  max_iter = 15,
                  random_state = 2045)

kmeans_3.fit(DF)
```

```
KMeans(algorithm='auto', copy_x=True, init='k-means++', max_iter=15,
       n_clusters=3, n_init=10, n_jobs=None, precompute_distances='auto',
       random_state=2045, tol=0.0001, verbose=0)
```

2) Clustering Results

- 반복 횟수

```
kmeans_3.n_iter_
```

```
3
```

- 군집별 중심점

```
kmeans_3.cluster_centers_
```

```
array([[5.9016129 , 2.7483871 , 4.39354839, 1.43387097],
```

```
       [5.006     , 3.428     , 1.462     , 0.246     ],
       [6.85      , 3.07368421, 5.74210526, 2.07105263]])
```

- 군집 결과 레이블

```
kmeans_3.labels_
```

```
array([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, 1,
       1, 1, 1, 1, 1, 1, 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, 2, 0, 0, 0, 0, 0, 0, 0, 0, 0,
       0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 2, 0, 2, 2, 2, 2, 0, 2, 2, 2,
       2, 2, 2, 0, 0, 2, 2, 2, 2, 0, 2, 0, 2, 0, 2, 2, 0, 0, 2, 2, 2, 2,
       2, 0, 2, 2, 2, 2, 0, 2, 2, 2, 0, 2, 2, 2, 0, 2, 2, 0], dtype=int32)
```

- 군집 중심까지의 제곱 거리의 합

```
kmeans_3.inertia_
```

```
78.85144142614601
```

III. Scree Plot

1) DataFrame

```
Z = pd.DataFrame(data = iris.data,
                 columns = ['sepal_length',
                            'sepal_width',
                            'petal_length',
                            'petal_width'])

Z.head(3)
```

	sepal_length	sepal_width	petal_length	petal_width
0	5.1	3.5	1.4	0.2
1	4.9	3.0	1.4	0.2
2	4.7	3.2	1.3	0.2

2) K(1~10) 군집분석

```
inertia = []
```

```
inertia = []

K = range(1,10)

for k in K:
    kmeanModel = KMeans(n_clusters = k).fit(Z)
    kmeanModel.fit(Z)
    inertia.append(kmeanModel.inertia_)
```

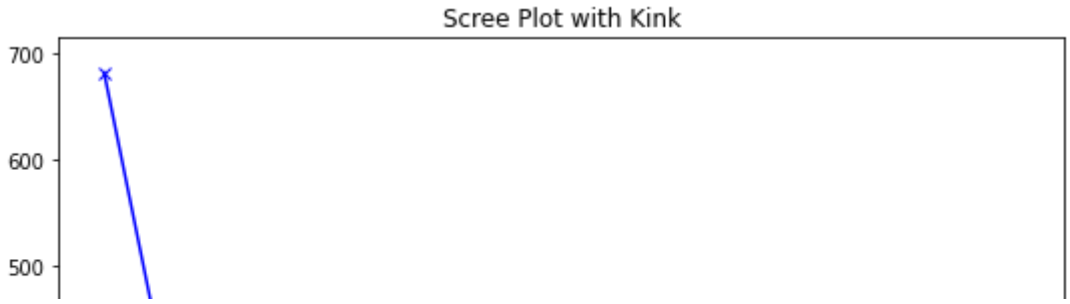
3) 군집 중심까지의 제곱 거리의 합

```
inertia
```

```
[681.3706,
 152.34795176035792,
 78.85144142614601,
 57.25600931571815,
 46.44618205128205,
 39.03998724608725,
 34.29822966507177,
 30.06459307359308,
 28.347370851370854]
```

4) Plot the elbow

```
plt.figure(figsize = (9, 7))
plt.plot(K, inertia, 'bx-')
plt.xlabel('k')
plt.ylabel('inertia')
plt.title('Scree Plot with Kink')
plt.show()
```



IV. Visualization with PCA(Principal Component Analysis)

1) target 및 cluster 추가

```
DF['cluster'] = kmeans_3.labels_
DF['target'] = iris.target
```

```
DF.head(3)
```

	sepal_length	sepal_width	petal_length	petal_width	cluster	target
0	5.1	3.5	1.4	0.2	1	0
1	4.9	3.0	1.4	0.2	1	0
2	4.7	3.2	1.3	0.2	1	0

2) 군집 결과 확인

```
DF.groupby('target')['cluster'].value_counts()
```

```
target  cluster
0       1          50
1       0          48
        2           2
2       2          36
        0          14
Name: cluster, dtype: int64
```

3) PCA 차원 축소(4차원 -> 2차원)

```
from sklearn.decomposition import PCA

pca = PCA(n_components = 2)
pca_transformed = pca.fit_transform(iris.data)
```

```
pca_transformed[:5]
```

```
array([[-2.68412563,  0.31939725],
       [-2.71414169, -0.17700123],
       [-2.88899057, -0.14494943],
       [-2.74534286, -0.31829898],
       [-2.72871654,  0.32675451]])
```

4) pca_x와 pca_y 추가

```
DF['pca_x'] = pca_transformed[:, 0]
DF['pca_y'] = pca_transformed[:, 1]
```

```
DF.head(5)
```

	sepal_length	sepal_width	petal_length	petal_width	cluster	target	pca_x
0	5.1	3.5	1.4	0.2	1	0	-2.684126
1	4.9	3.0	1.4	0.2	1	0	-2.714142
2	4.7	3.2	1.3	0.2	1	0	-2.888991
3	4.6	3.1	1.5	0.2	1	0	-2.745343
4	5.0	3.6	1.4	0.2	1	0	-2.728717

5) 2차원 시각화

- 군집 값 0, 1, 2 인덱스 추출

```
idx_0 = DF[DF['cluster'] == 0].index
idx_1 = DF[DF['cluster'] == 1].index
idx_2 = DF[DF['cluster'] == 2].index
```

```
idx_0, idx_1, idx_2
```

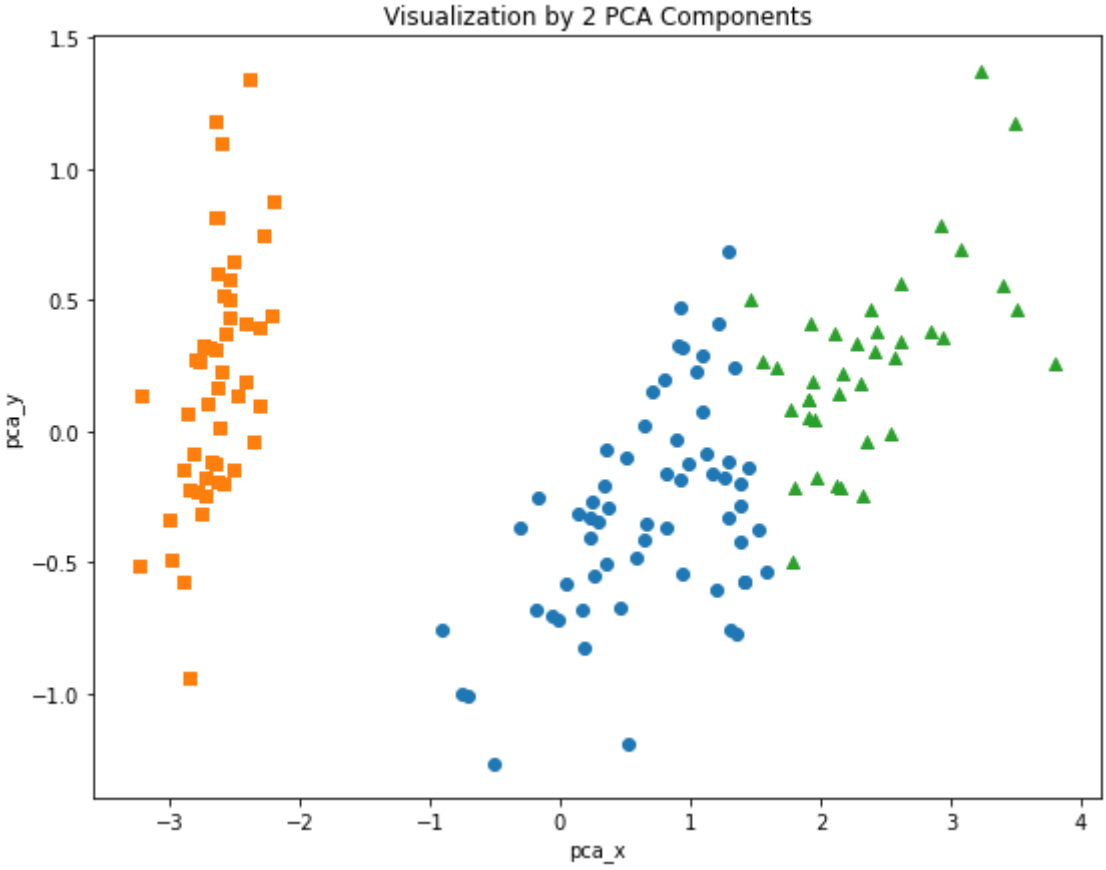
```
(Int64Index([ 50,  51,  53,  54,  55,  56,  57,  58,  59,  60,  61,  62,  63,
              64,  65,  66,  67,  68,  69,  70,  71,  72,  73,  74,  75,  76,
              78,  79,  80,  81,  82,  83,  84,  85,  86,  87,  88,  89,  90,
              91,  92,  93,  94,  95,  96,  97,  98,  99, 101, 106, 113, 114,
             119, 121, 123, 126, 127, 133, 138, 142, 146, 149],
            dtype='int64'),
 Int64Index([ 0,  1,  2,  3,  4,  5,  6,  7,  8,  9, 10, 11, 12, 13, 14, 15, 16,
             17, 18, 19, 20, 21, 22, 23, 24, 25, 26, 27, 28, 29, 30, 31, 32, 33,
             34, 35, 36, 37, 38, 39, 40, 41, 42, 43, 44, 45, 46, 47, 48, 49],
            dtype='int64'),
 Int64Index([ 52,  77, 100, 102, 103, 104, 105, 107, 108, 109, 110, 111, 112,
```

```
 115, 116, 117, 118, 120, 122, 124, 125, 128, 129, 130, 131, 132,
 134, 135, 136, 137, 139, 140, 141, 143, 144, 145, 147, 148],
dtype='int64'))
```

- 0, 1, 2 인덱스 시각화

```
plt.figure(figsize = (9, 7))
plt.scatter(x = DF.loc[idx_0, 'pca_x'],
            y = DF.loc[idx_0, 'pca_y'],
            marker = 'o')
plt.scatter(x = DF.loc[idx_1, 'pca_x'],
            y = DF.loc[idx_1, 'pca_y'],
            marker = 's')
plt.scatter(x = DF.loc[idx_2, 'pca_x'],
            y = DF.loc[idx_2, 'pca_y'],
            marker = '^')

plt.xlabel('pca_x')
plt.ylabel('pca_y')
plt.title('Visualization by 2 PCA Components')
plt.show()
```



#

#

#

The End

#

#

#