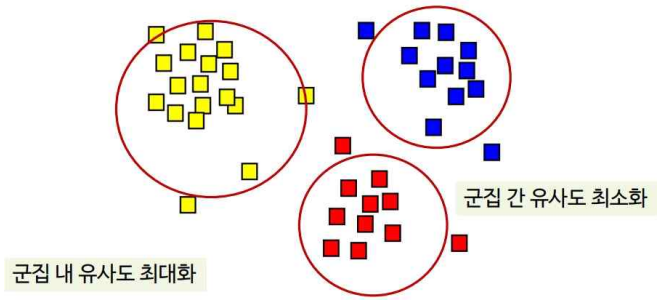
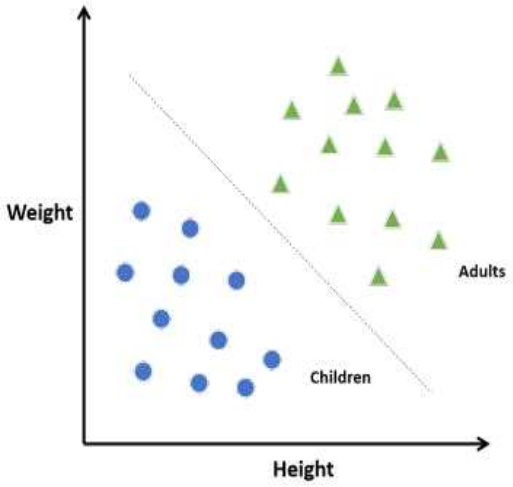
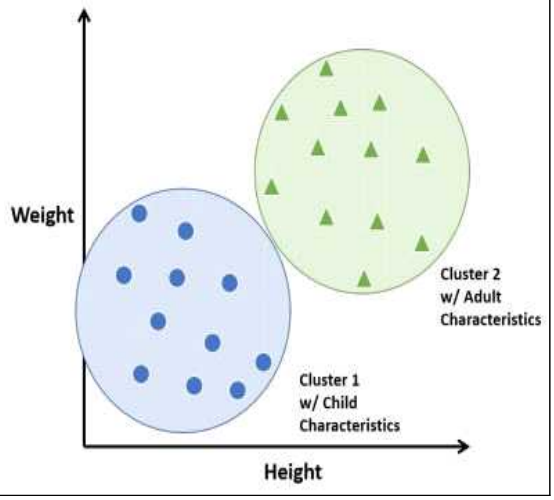
군집화(Clustering)

: 유사한 속성을 갖는 데이터를 묶어 전체 데이터를 몇 개의 군집으로 나누기



좋은 군집화 기준

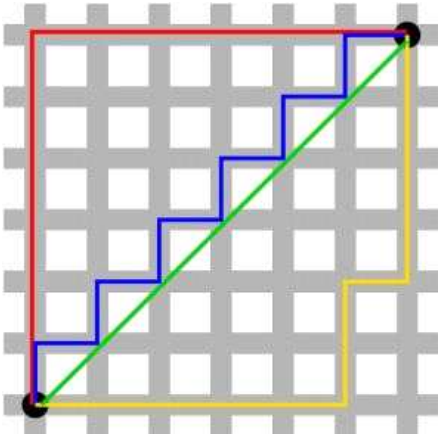
1. 같은 군집의 데이터가 서로 유사할수록 좋다(inter-class similarity)
2. 다른 군집의 데이터는 서로 다를수록 좋다(intra-class dissimilarity)

분류	군집화
지도 학습(Supervised Learning)	비지도 학습(Unsupervised Learning)
사전에 정의된 범주가 있는 데이터로 예측 모델 학습	사전에 정의된 범주가 없는 데이터로 최적의 그룹 찾기
	

고려사항

1. 거리 측도(유사도 측도)
2. 군집화 알고리즘
3. 최적 군집 수(k)
4. 군집화 결과 측정/평가

거리 측도(유사도 척도)



1. 유클리디언 거리(Euclidiean distance)

L_2 거리. 두 관측치 사이의 직선 거리

$$d(x,y)=\sqrt{\sum_{i=1}^{p}(x_i-y_i)^2}=\parallel x-y \parallel_2$$

2. 맨하탄 거리(Manhattan distance)

L_1 거리. 택시거리

$$d_{Manhattan}(x,y)=\sum_{i=1}^{p}\left|x_i-y_i\right|=\parallel x-y \parallel_1$$

3. 마할라노비스 거리(Mahalanobis distance)

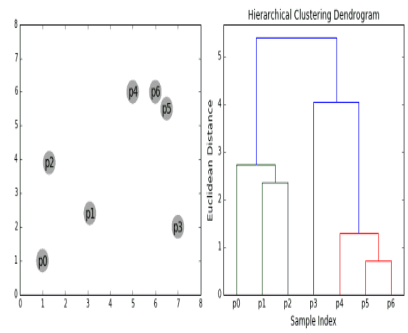
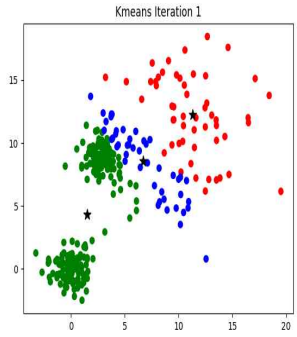
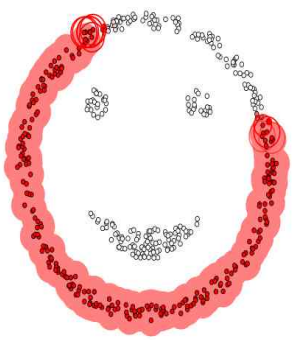
: 변수 내 분산과 공분산을 모두 반영하여 X, Y 간 거리를 계산하는 방법

$$d_{Mahalanobis}(X,Y)=\sqrt{(X-Y)\Sigma^{-1}(X-Y)}\ where\ \Sigma^{-1}=inverse\ of\ convariance\ matrix$$

데이터의 $convariance\,matrix$ = $identity\,matrix$이면

마할라노비스 거리 = 유클리디언 거리

군집화 알고리즘

계층적 군집화	분리형 군집화	분포 기반 군집화
개체들을 가까운 집단부터 묶어나가는 방식	전체 데이터 영역을 특정 기준으로 동시에 구분하는 방식	데이터 분포 기반으로 높은 밀도를 갖는 세부 영역들로 전체 영역 구분하는 방식
군집화 결과뿐만 아니라 유사한 개체들이 결합하는 덴드로그램 생성	각 객체들은 사전에 정의된 개수의 군집 중 하나에 속함	
		

1. 계층적 군집화(Hierarchical Clustering)

: 계층적 트리 모델로 개별 대체를 순차적/계층적 유사한 개체/군집과 통합

덴드로그램(Dendrogram)으로 시각화할 수 있다

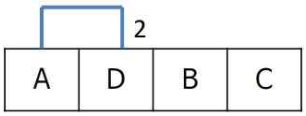
덴드로그램: 결합하는 순서를 나타내는 트리 구조

사전에 군집 개수를 정하지 않아도 수행이 가능하다

덴드로그램 생성하고 적절한 수준에서 자르면 해당하는 군집화 결과

1. 모든 데이터 사이의 거리에 대한 유사도 행렬 계산

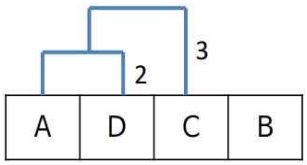
	A	B	C	D
A		20	7	2
B			10	25
C				3
D				



2. 거리가 인접한 데이터 끼리 군집 형성

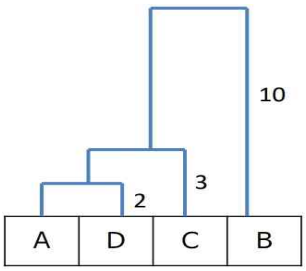
3. 유사도 행렬 갱신

	AD	B	C	
AD		20	3	
B			10	
C				

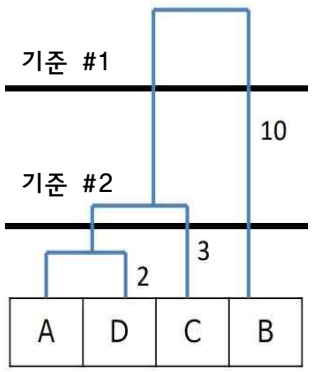


4. 1~3번 반복

	ADC	B		
ADC		10		
B				

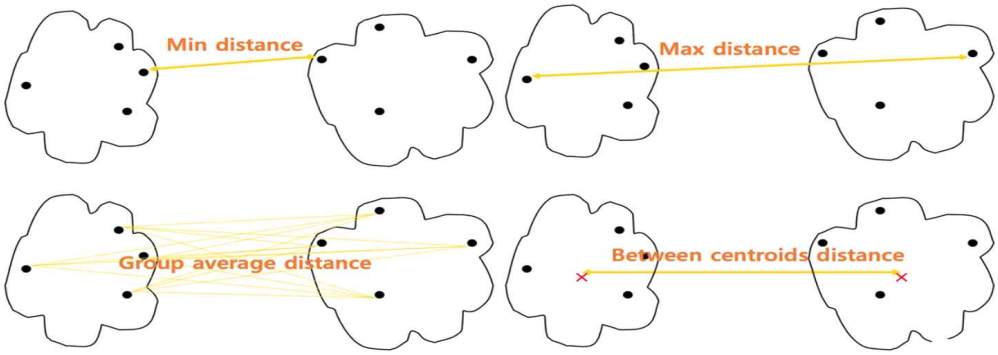


최종 결과



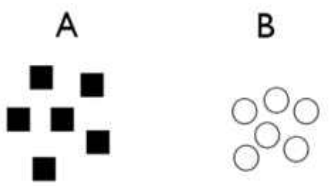
군집과 군집의 거리 계산법

1. Min distance: 군집 내 데이터 간 거리를 모두 계산하여 가장 작은 거리 값
2. Max distance: 군집 내 데이터 간 거리를 모두 계산하여 가장 큰 거리 값
3. Group average distance
   : 군집 내 데이터 간 거리를 모두 계산하여 평균 거리 값
4. Between centroid distance
   : 각 군집 내 데이터의 평균을 계산하여 평균과 평균 사이의 거리 값

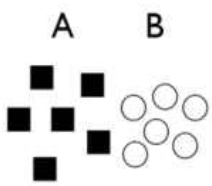


5. Ward 거리 계산법(Ward linkage method)
   1. 서로 다른 군집 내 모든 데이터를 포함한 중심(centroid)을 구하고
   2. 구한 중심과 서로 다른 군집 내 모든 데이터 사이의 거리를 구한다
   3. 각 군집 내 데이터를 통해 개별 중심을 구하고
   4. 개별 중심과 해당 군집 내 데이터 사이의 거리를 구한다
   5. Ward 거리 = AuB의 중심거리 − (A의 중심거리 + B의 중심거리)

$$Ward\ distance = \sum_{i \in A \cup B} \| x_i - m_{A \cup B} \|^2 - \left\{ \sum_{i \in A} \| x_i - m_A \|^2 - \sum_{i \in A} \| x_i - m_B \|^2 \right\}$$



Ward's distance = 10−(3+2) = 5



Ward's distance = 7−(3+2) = 2

최종 결과가 클수록 서로 다른 군집의 유사도가 낮아 멀리 있고,
최종 결과가 작을수록 서로 다른 군집의 유사도가 높아 가까이에 있다

2. 분리형 군집화

K 평균 군집화(K-means Clustering)

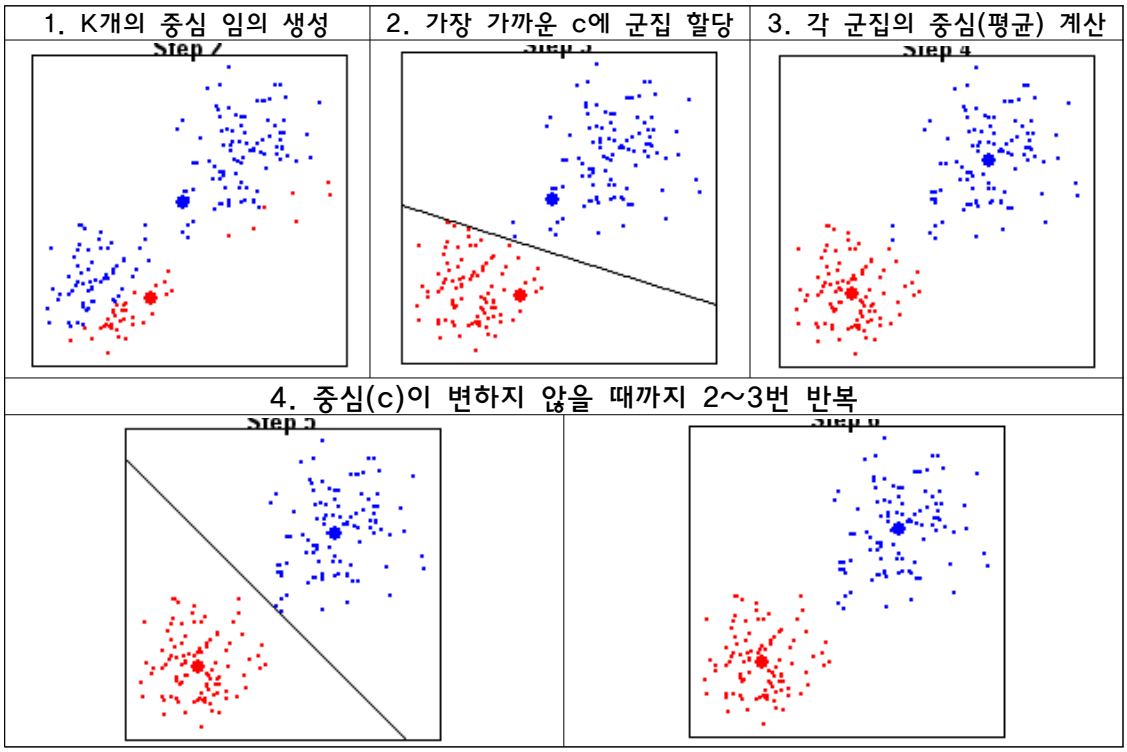
* 사전에 군집의 수(K)가 정해져야 한다

$$\underset{c}{argmax}\sum_{i=1}^{K}\sum_{x_j \in C_i} \| x_j - c_i \|^2$$

$$X = C_i \cup C_2 \cup \cdots \cup C_k,\ C_i \cap C_j = \varnothing,\ i \neq j$$

X: 데이터, C: 군집, c_i : 각 군집의 중심

1. K개의 중심(c)을 임의로 생성한다(random)
2. 모든 데이터에 가까운 중심(c)으로 군집을 할당한다
3. 각 군집의 중심(평균)을 다시 계산한다(update)
4. 중심이 변하지 않을 때까지 2~3번을 반복한다



랜덤 초기화의 단점 해결 방법

1. 앙상블(Ensemble) 결과 통합
   :여러 번 K-means 군집화해서 가장 여러 번 나타나는 군집을 사용한다
2. 데이터 분포 정보를 활용한 초기화 방법
   데이터가 Gaussian 분포라면, 분포의 중심을 초깃값으로 선정

3. (데이터가 많을 때)샘플링 데이터를 활용해서 계층적 군집화를 수행하고
계층적 군집화의 중심을 초기 군집 중심으로 사용
BUT 많은 경우 초기 중심이 최종 결과에 영향을 미치지 않는다

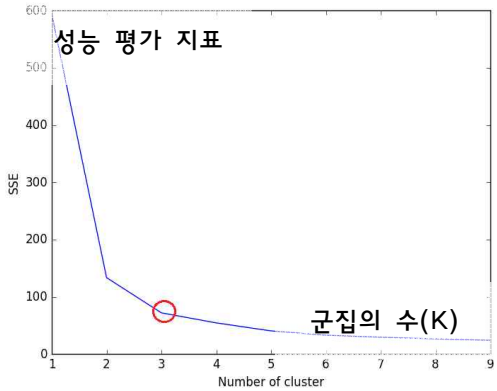
단점

1. 서로 다른 크기의 군집을 잘 찾아내지 못한다
2. 서로 다른 밀도의 군집을 잘 찾아내지 못한다
3. 지역적 패턴이 있는(구조가 특이한) 군집을 판별하지 못한다

K 값 선정 방법

다양한 군집 수에 대한 성능 평가 지표를 통해 최적 군집 수 선택한다

→ 일반적으로 elbow point에서
최적 군집 수 결정한다



성능 평가 지표

but 지도학습 문제처럼 모든 상황에 적용할 수 있는 평가 지표가 없다

1. 내부 평가 지표: Sum of Squared Error(SSE), Dunn Index
2. 외부 평가 지표: Rand Index, Jaccard Coefficient, 등

Sum of Squard Error(SSE)

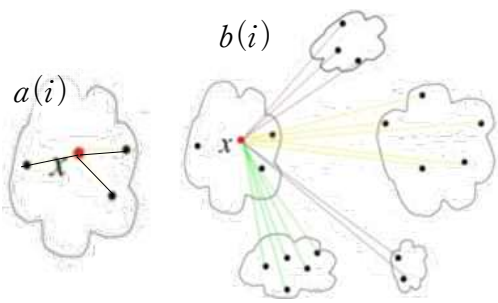
군집 내 거리 최소화(만족), 군집 간 거리 최대화(불만족)

$$SSE = \sum_{i=1}^{K} \sum_{x \in C_i} dist(x, c_i)^2$$

Sihouette 통계량

$$s(i) = \frac{b(i) - a(i)}{max\ \{a(i), b(i)\}},\ -1 \le s(i) \le 1$$

$$\bar{s} = \frac{1}{n} \sum_{i=1}^{n} s(i)$$



$a(i)$: i번째 데이터와 같은 군집에 있는 모든 데이터 사이의 평균 거리

$b(i)$: i번째 데이터와 다른 군집에 있는 모든 데이터 사이의 최소 거리

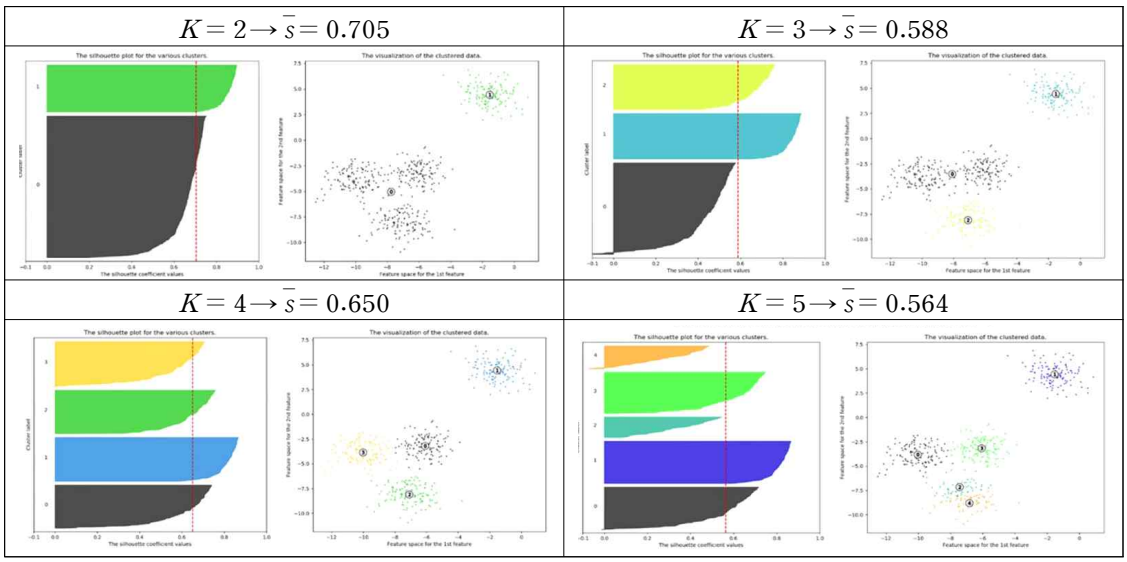
→ a가 작을수록 유사한 데이터가 잘 모여 있고

b가 클수록 서로 다른 데이터가 잘 흩어져 있다

s값이 1에 가까울수록 good, −1에 가까울수록 bad

일반적으로 s 〉 0.5이면 군집 결과가 타당하다고 판단한다

※K=2인 경우 통계량이 best인 경우가 많아 2등 K를 선정하는 것 좋다



3. 분포 기반 군집화

DBSCAN(Density Based Clustering of Application with Noise)

높은 밀도로 모여 있는 데이터들 → 그룹으로 분류

낮은 밀도를 가지고 있는 데이터 → 이상치 또른 잡음으로 분류

데이터의 $\epsilon-neighborhood$가 M개 이상의 데이터가 포함하는지 고려한다

ϵ와 M: 분포 기반 군집화의 파라미터

1. 핵심 자료(core point)

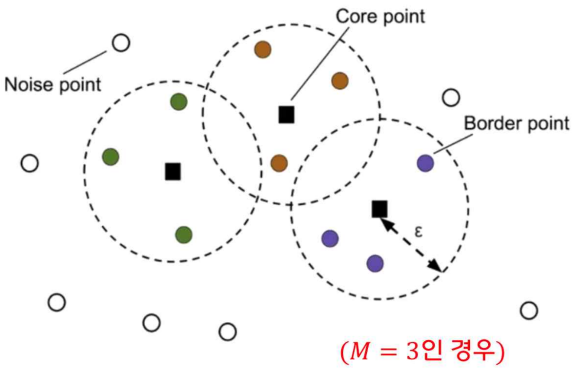
: $\epsilon-neighborhood$가 M개 이상의 데이터 포함하는 자료

2. 주변 자료(border point)

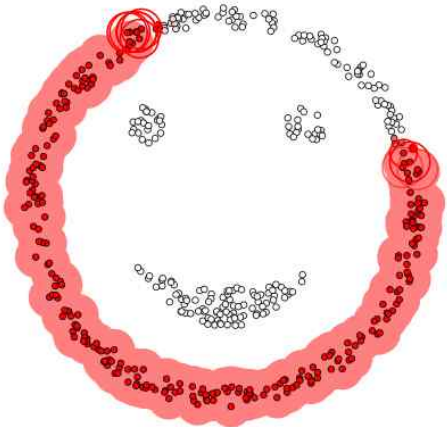
: 핵심자료는 아니지만 $\epsilon-neighborhood$에 핵심 자료를 포함하는 자료

3. 잡음 자료(noisr point)

: 핵심 자료도 주변 자료도 아닌 자료



($M = 3$인 경우)



1. 임의 데이터를 선택하고 군집 1을 부여한다
2 임의 데이터의 $\epsilon - NN$을 구한다

    데이터 수가 M보다 작으면 잡음 자료로 부여한다
3. M보다 크면 $\epsilon - NN$ 모두 군집 1로 부여한다

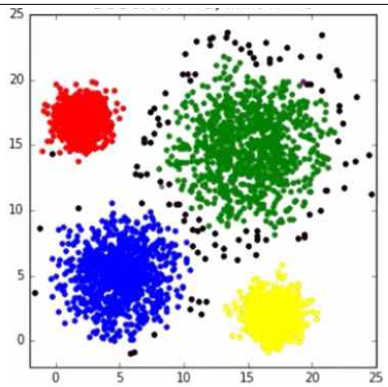
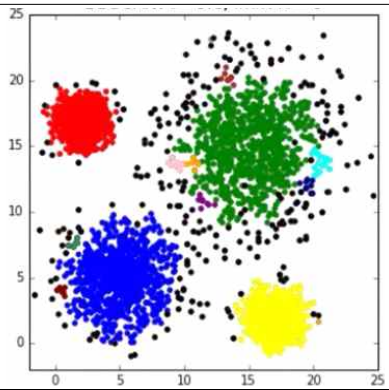
    군집 1 모든 데이터의 $\epsilon - NN$에 크기가 M보다 큰 것이 없을 때까지 반복
4. 군집 2에 대해서 동일하게 1~3번을 반복한다
5. 모든 데이터에 군집이 할당되거나 잡음으로 분류될 때까지 반복한다

파라미터 설정

ϵ: 너무 작으면 많은 데이터가 잡음으로 분류되고, 너무 크면 군집 개수가 적다

    HDBSCAN(Hierarchical DBCAN)의 경우 ϵ을 자동으로 설정한다

M: 일반적으로 "특성 변수 개수 +1“

$\epsilon = 1 \rightarrow cluster : 4$	$\epsilon = 0.6 \rightarrow cluster : 12$
	

군집화 api

1. 계층적 군집화

sklearn.cluster.AgglomerativeClustering(n_clusters=2, affinity='euclidean', linkage='ward')

n_clusters: 찾을 군집의 개수(int or None, default = 2)

affinity: 거리 측도(유사도 측도)

str or callable, default = 'euclidean

{'euclidean', 'l1', 'l2', 'manhattan', 'cosine', or 'precomputed'}

linkage: 군집과 군집 사이의 거리 계산법

{'ward', 'complete', 'average', 'single'}, default = 'ward'

2. 분리형 군집화: K 평균 군집화

sklearn.cluster.KMeans(n_cluster=8, init='k-means++', n_init=10, max_iter=300, tol=0.0001, random_state=None)

n_clusters: 생성할 군집의 개수(K. 중심의 개수) (int, default = 8)

inint: method for initialization

{'k-means++', 'random'}, default = 'k-means++"

n_init: number of time the K-means algorithm will be run with different centroid seeds(int, default = 10)

max_iter: maximum number of iterations of the K-means algorithm for a single run(int, default = 300)

tol: relative tolerance(float, default = 1e-4)

3. 분포 기반 군집화: DBSCAN

sklearn.cluster.DBSCAN(eps=0.5, min_samples=5, metric='euclidean')

eps: $neighborhood$의 최대 거리(엡실론 ϵ) (float, default = 0.5)

min_samples: $\epsilon - neighborhood$에 최소 개수(M) (int, default = 5)

metric: 거리 측도(string, or callable, default = 'euclidean')

4. HDBCAN: ϵ은 자동 설정된다

hdbscan.HDBSCAN(min_samples=None)

min_samples: $\epsilon - neighborhood$에 최소 개수(M)

int, optional, default = 5