15기 정규세션

ToBig's 14기 김민경

KNN & Clustering

Unit 00 | 들어가기 전에

머신러닝의 학습 방법

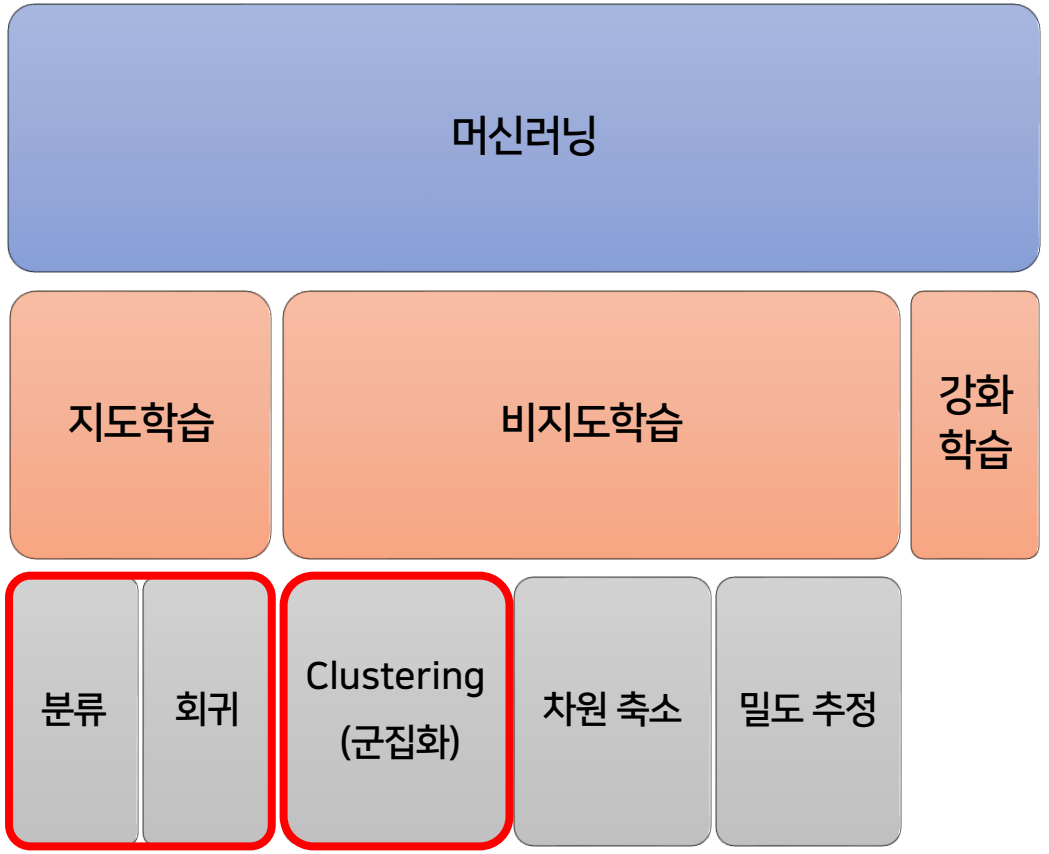
1) 지도학습 (Supervised Learning)

: 정답 라벨이 있는 데이터를 이용해서 상태나 값을 예측

2) 비지도학습 (Unsupervised Learning)

: 정답 라벨이 없는 데이터 자체에서 숨겨진 구조나 특징을 발견

3) 강화학습



15기 정규세션

ToBig's 14기 김민경

KNN

K – Nearest Neighbors

Contents

Unit 01 | KNN

사례 기반 학습 (Instance-Based Learning = Memory-Based Learning)

: 별도의 모델 생성 없이 인접 데이터를 예측에 사용

- KNN
- K-Means, 계층적 클러스터링
- etc.

모델 기반 학습 (Model-Based Learning)

: 데이터로부터 모델을 생성하여 예측 진행

- Linear Regression, Logistic Regression
- Neural Network
- etc.

Unit 01 | KNN

K N N

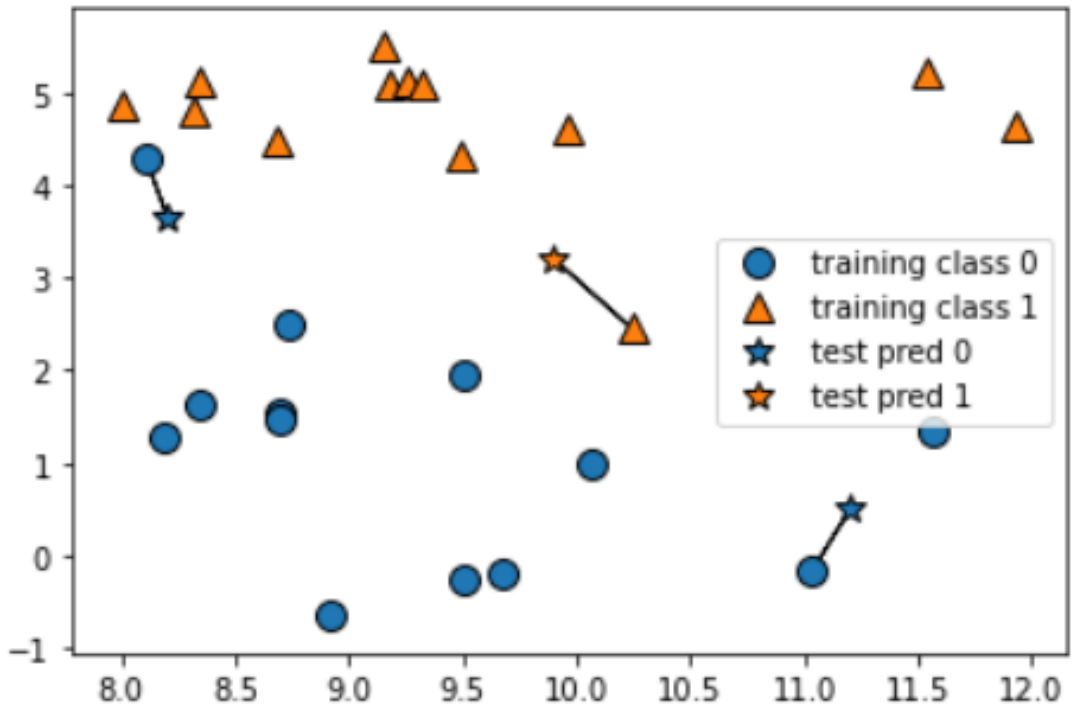
K개의 Nearest 가까운 Neighbors 이웃

K개의 가까운 이웃을 찾자!

Unit 01 | KNN

<Classification>

K = 1

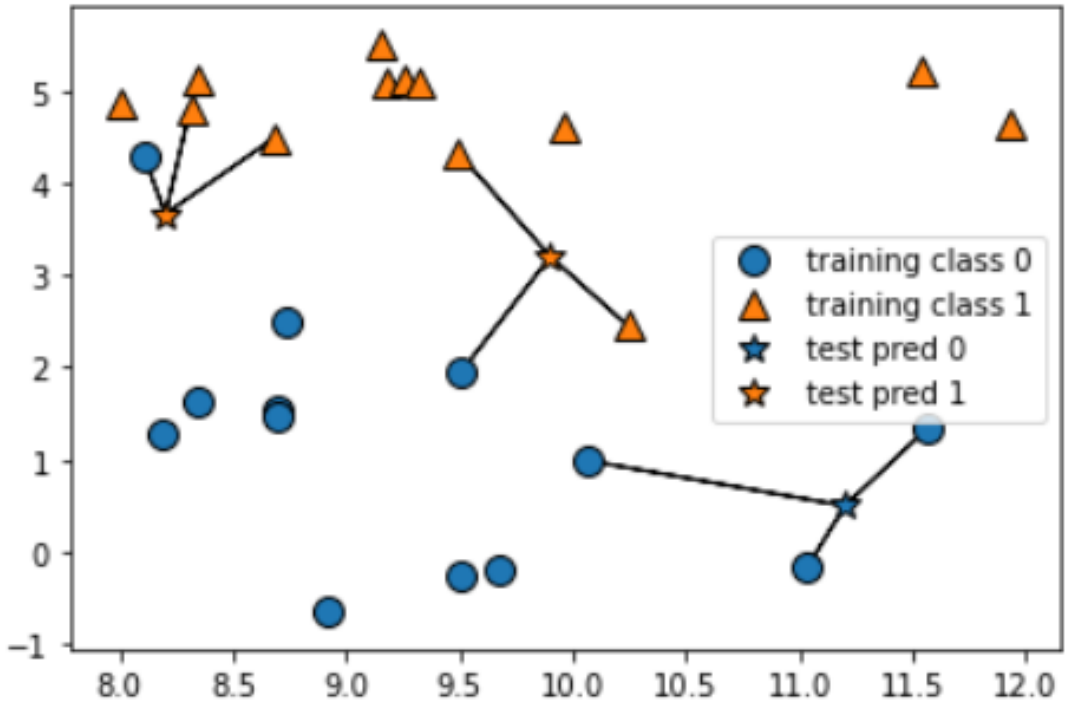


1-NN

Unit 01 | KNN

<Classification>

K = 3



3-NN

Unit 01 | KNN

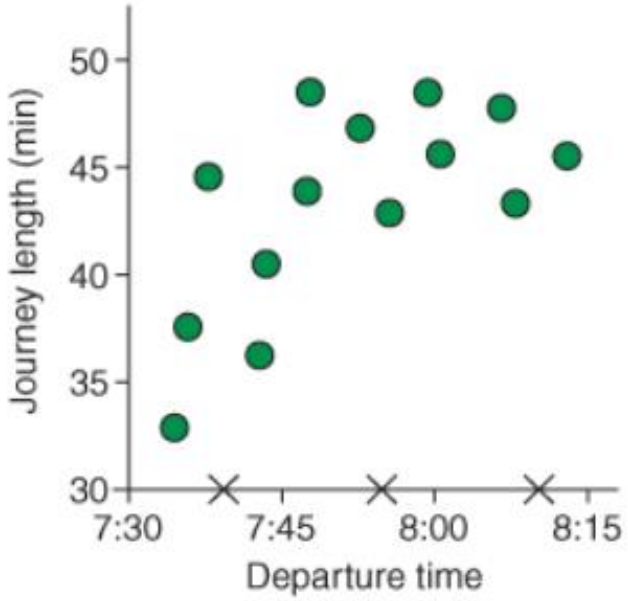
<Classification>

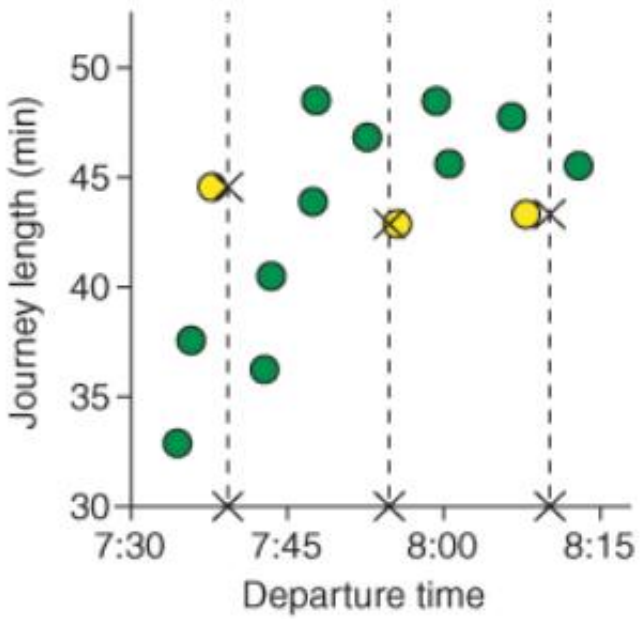
1. 새로운 데이터가 들어오면
2. 모든 데이터들과의 거리를 구해서
3. 가장 가까운 K개의 데이터를 선택하고
4. 이 K개 데이터의 클래스를 확인해 다수의 데이터가 속한 클래스를 찾는다.
5. 이 클래스를 새로운 데이터의 클래스로 할당한다.

Unit 01 | KNN

<Regression>

K = 1



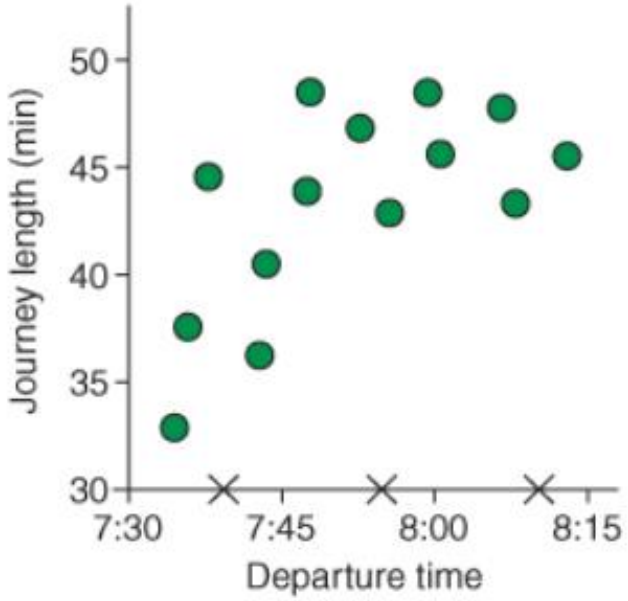


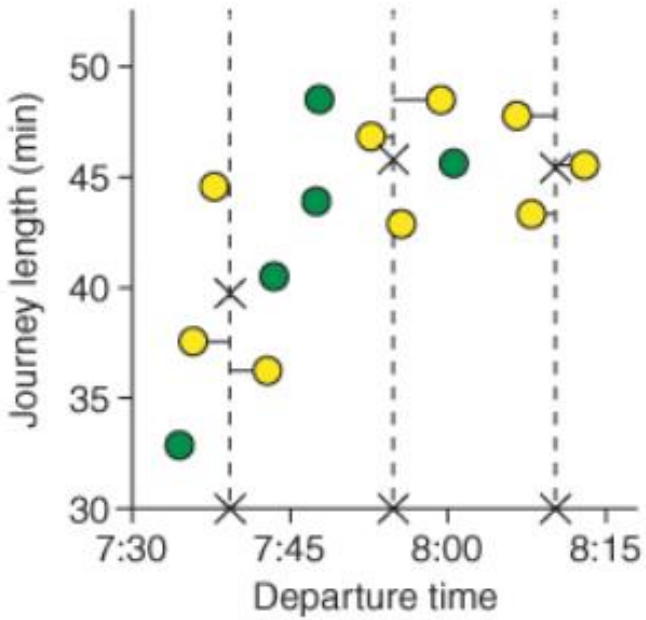
1-NN

Unit 01 | KNN

<Regression>

K = 3





3-NN

Unit 01 | KNN

<Regression>

1. 새로운 데이터가 들어오면
2. 모든 데이터들과의 거리를 구해서
3. 가장 가까운 K개의 데이터를 선택하고
4. 이 K개 데이터의 평균값을 계산한다.
5. 이 값을 새로운 데이터의 예측값으로 지정한다.

Unit 01 | KNN

1) 가장 가까운 이웃 ? – “Distance Measures”

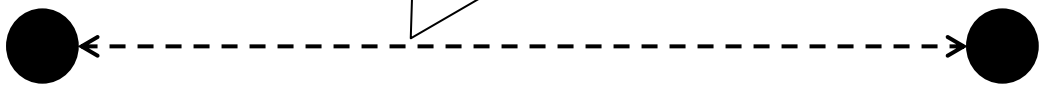
2) 최적의 K 값 ? - “K”

Contents

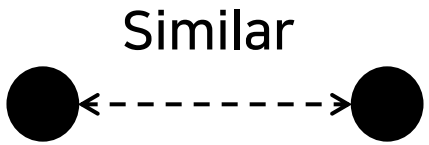
Unit 02 | (Hyperparameter) Distance Measures

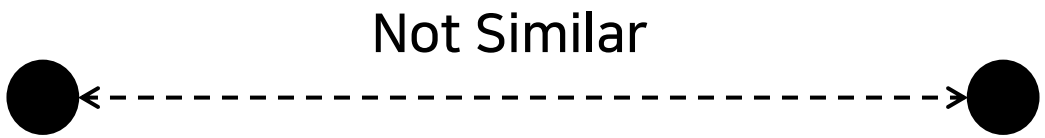
두 데이터가 얼마나 유사한가?

=> 두 데이터 사이의 '거리'를 측정하자!



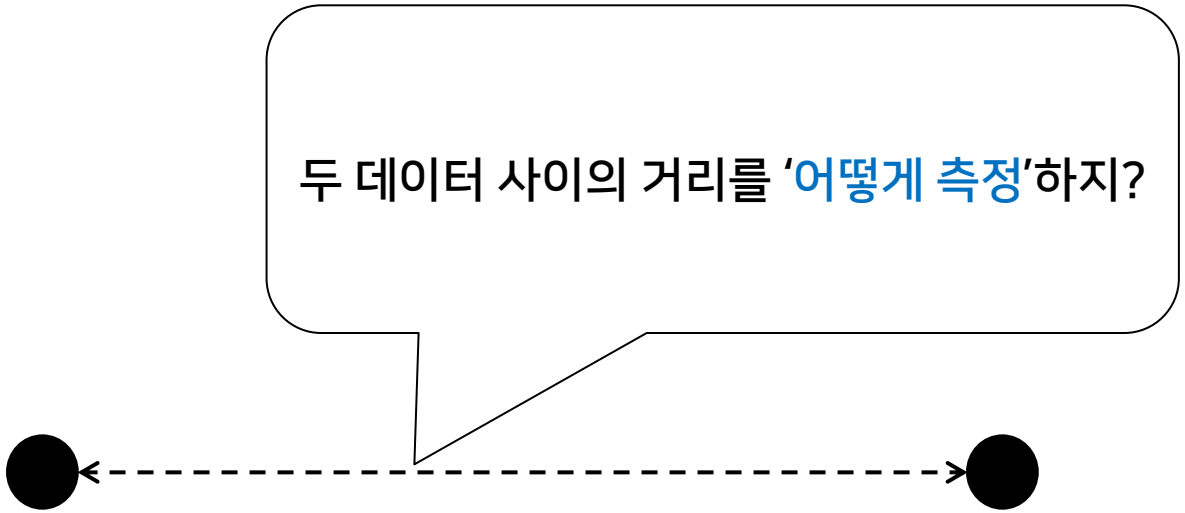
Unit 02 | (Hyperparameter) Distance Measures





거리가 가까울수록 유사도가 높다고 할 수 있다

Unit 02 | (Hyperparameter) Distance Measures



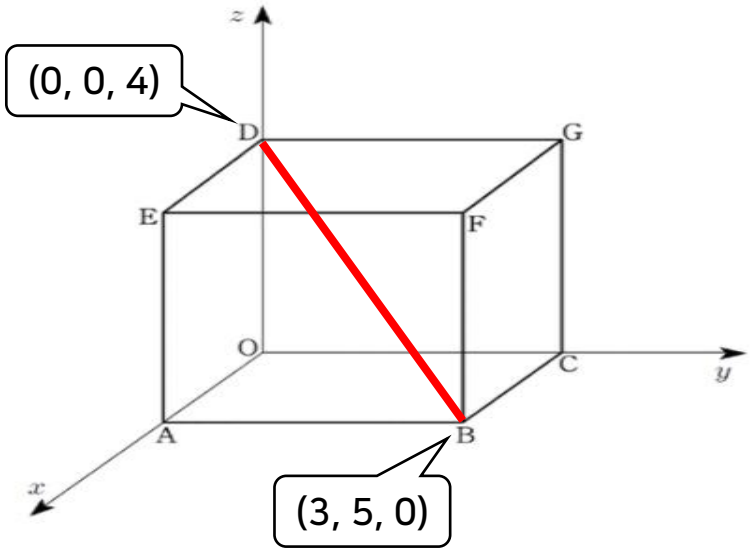
Unit 02 | (Hyperparameter) Distance Measures

Unit 02 | (Hyperparameter) Distance Measures

Distance Measure 1 : Euclidean Distance

- 가장 흔히 사용하는 거리 척도
- 두 점 사이의 최단거리(직선거리)를 의미
- L2-Norm

$$X = (x_1, x_2, \ldots, x_n)$$
$$Y = (y_1, y_2, \ldots, y_n)$$
$$d_{euclidean}(X, Y) = \sqrt{\sum_{i=1}^{n}(x_i - y_i)^2}$$



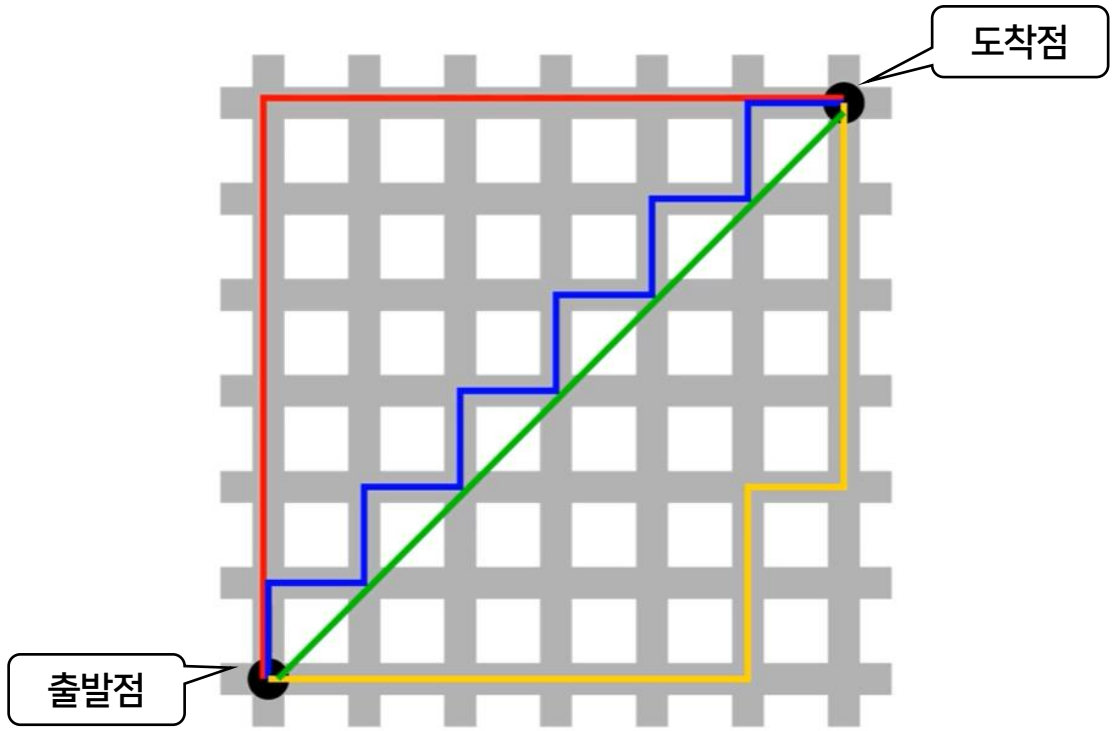
$$d_{(B,D)} = \sqrt{(0-3)^2 + (0-5)^2 + (4-0)^2} = \sqrt{50}$$

Unit 02 | (Hyperparameter) Distance Measures

Distance Measure 2 : Manhattan Distance

- 유클리드 거리 다음으로 많이 사용하는 거리 척도
- 한 번에 대각선이 아닌 한 축 방향으로만 움직일 수 있다고 할 때 두 점 사이의 거리 (= x 좌표와 y 좌표 차이의 총합)
- L1-Norm

$$X = (x_1, x_2, \ldots, x_n)$$
$$Y = (y_1, y_2, \ldots, y_n)$$
$$d_{manhattan}(X, Y) = \sum_{i=1}^{n} |x_i - y_i|$$



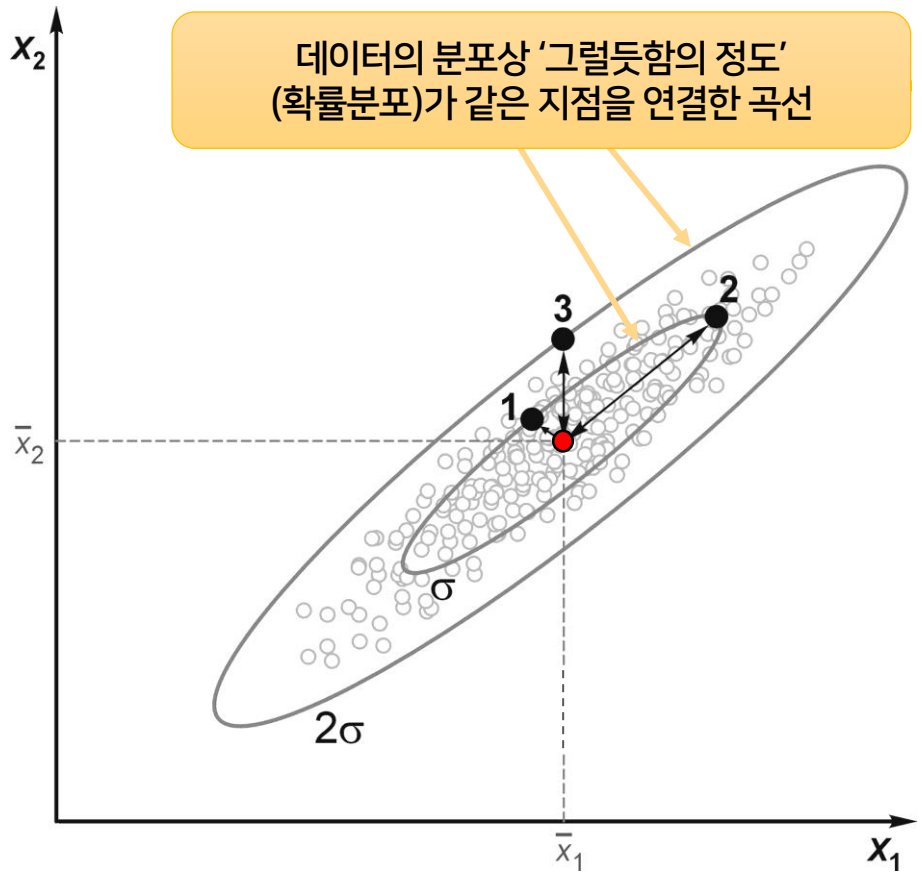
Unit 02 | (Hyperparameter) Distance Measures

Distance Measure 3 : Mahalanobis Distance

- 데이터가 평균과 표준편차를 고려했을 때 얼마나 중심에서 멀리 떨어져 있는지를 측정 (밀도를 고려한 거리 척도)
- 두 변수 사이에 상관관계가 거리에 영향을 미치기 때문에 변수들 간에 상관관계가 존재하는 경우 사용하면 유용

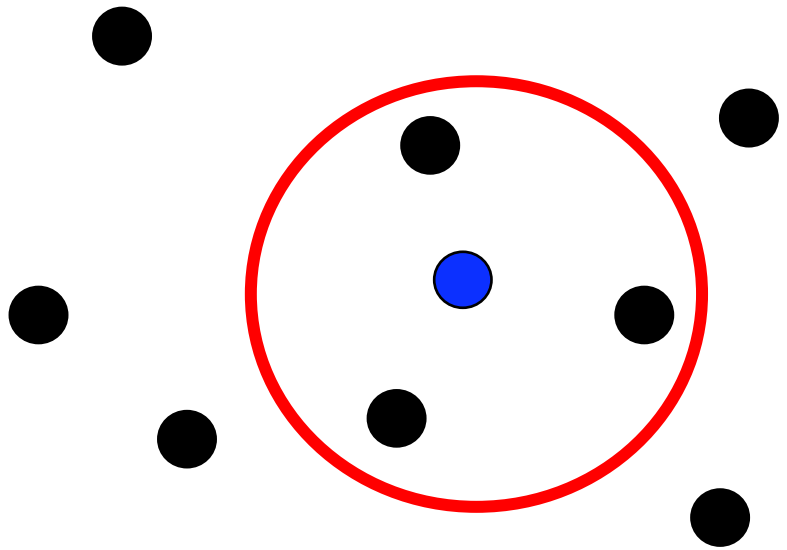
유클리드 거리 $d_{E1} < d_{E3} < d_{E2}$

마할라노비스 거리 $d_{M1} = d_{M2} < d_{M3}$



Contents

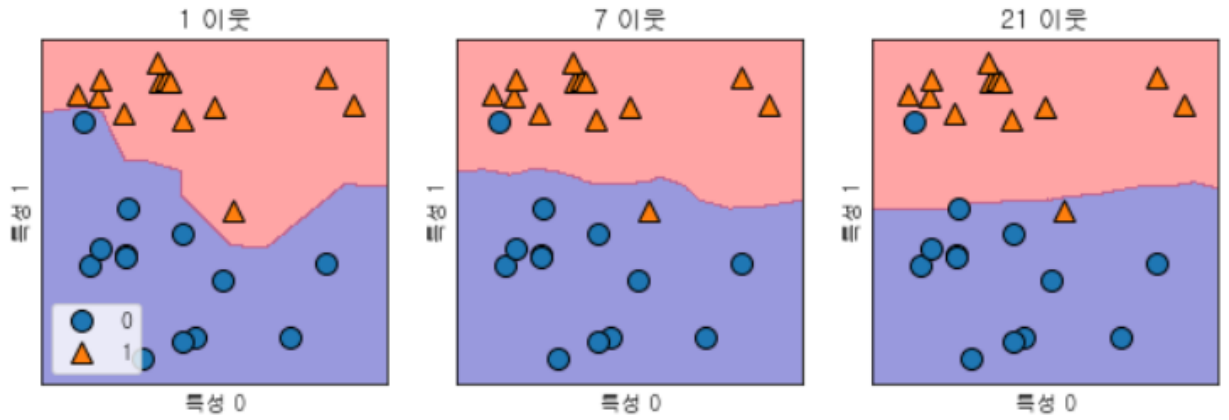
Unit 03 | (Hyperparameter) K



최적의 K값(이웃의 수)의 선택 방법?

Unit 03 | (Hyperparameter) K

K값에 따른 결정 경계 (decision boundary)



이웃 적게 사용
모델이 복잡(일반화 어렵다☹)
=> overfitting

이웃 많이 사용
모델이 단순(KNN 의미가 없다☹)
=> underfitting

적절한 K를 찾는 것이 중요!

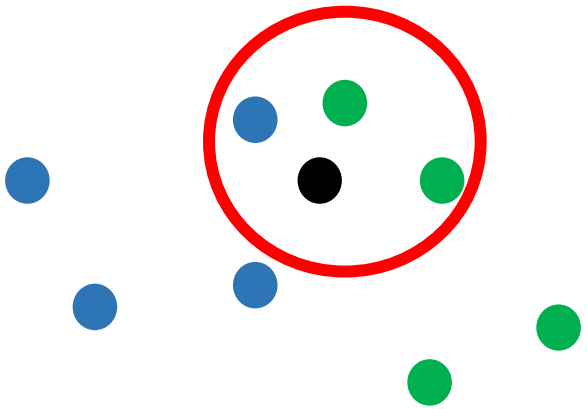
Unit 03 | (Hyperparameter) K

보통 K는 홀수로 지정

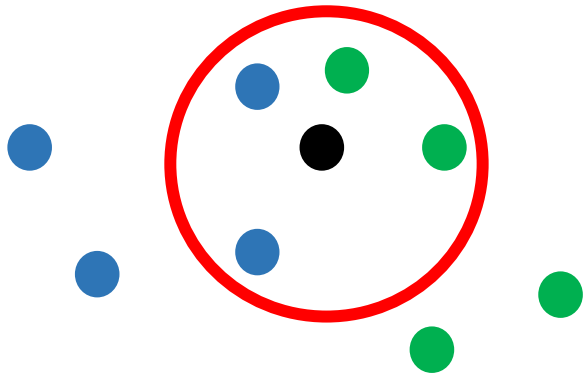
-> 짝수의 경우 동점이 발생할 수 있기 때문

(!주의!) 무조건 K를 홀수로 설정해야 한다고 생각하면 안 됨!
-> 클래스의 수가 3개인 경우 K=3으로 설정한다면?

K = 3 (홀수)



K = 4 (짝수)



Unit 03 | (Hyperparameter) K

(참고) K > 1 분류에서 동점이 발생한다면?

- Binary Classification (이진 분류)
  : K를 홀수로 설정

- Multi-class Classification (다중 분류)
  : 동점이 없어질 때까지 K값을 줄이기
  : 그래도 효과 없으면 1NN 분류기 사용 -> 다른 알고리즘 쓰기 추천…☺

Unit 03 | (Hyperparameter) K

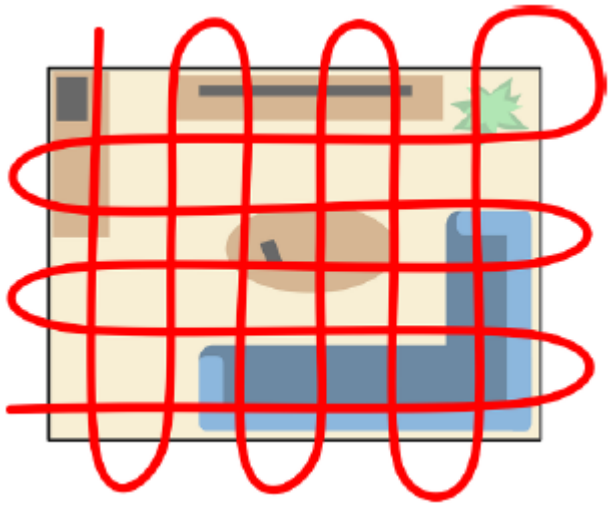
<최적의 K값 선택 방법 정리>

- 최적의 K를 결정하는 일반적인 규칙은 없음.
- 데이터에 따라 적절한 K값이 달라지기도 하는데, 데이터에 노이즈가 거의 없고 아주 잘 구조화된 데이터의 경우 K값이 작을수록 잘 동작함.
- 보통 K는 1 ~20 사이의 값으로 설정
- 동점이 나오는 경우를 막기 위해 보통 홀수를 사용

Unit 03 | (Hyperparameter) K

Grid Search (그리드 서치)

- 격자(Grid) 무늬로 Hyperparameter를 탐색(Search)
- 모든 parameter의 경우의 수에 대해 cross-validation 결과가 가장 좋은 parameter를 고르는 방법
  - 장점 : 주어진 공간 내에서 가장 좋은 결과를 얻을 수 있다.
  - 단점 : 시간이 정말 오래 걸린다.

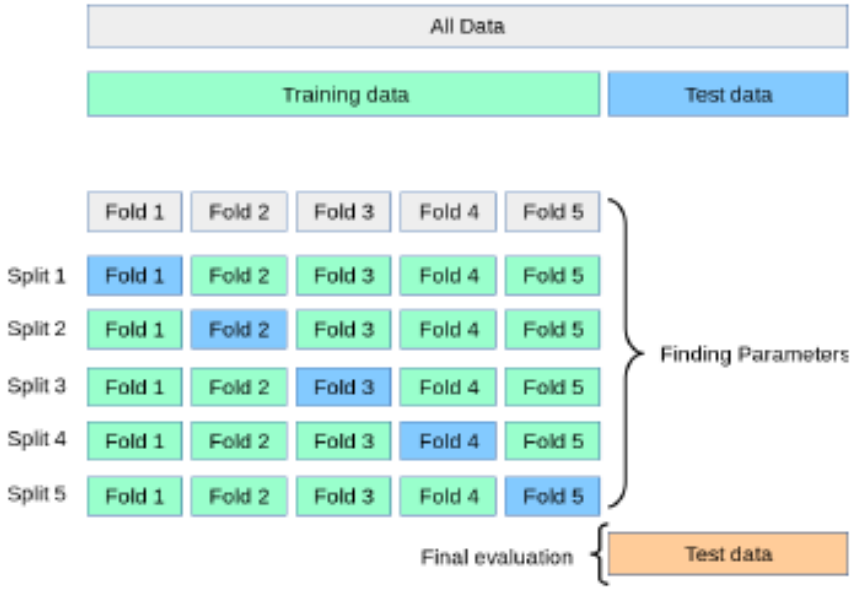


Unit 03 | (Hyperparameter) K

K-fold cross-validation (K겹 교차 검증)

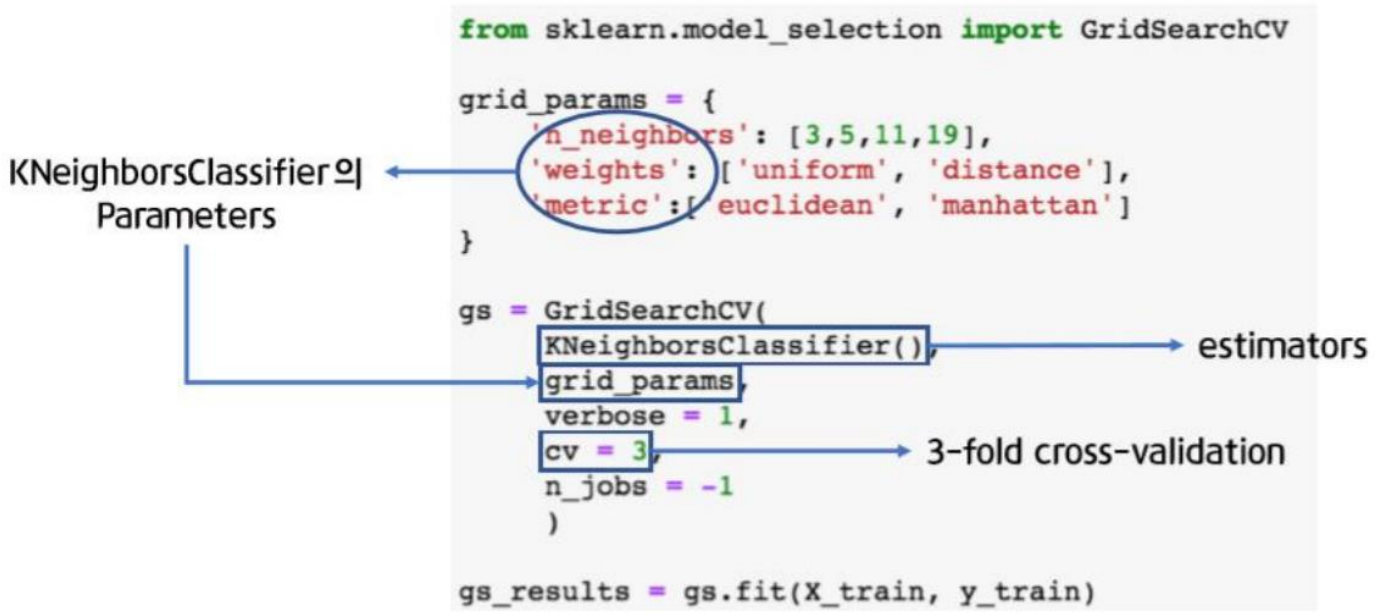
- 모든 데이터가 최소 한 번은 테스트(validation) 셋으로 쓰이도록 함
- (ex) 5-fold cv 에서 총 5개의 성능 평가지표가 생기게 되는데, 보통 이 값들의 평균으로 모델의 성능을 평가

!교차 검증을 통한 성능 평가의 목적!
- 더 좋은 모델을 선택하기 위해
- Hyperparameter tuning을 위해



Unit 03 | (Hyperparameter) K

Grid Search & K-fold cross-validation




```
from sklearn.model_selection import GridSearchCV

grid_params = {
    'n_neighbors': [3,5,11,19],
    'weights': ['uniform', 'distance'],
    'metric': ['euclidean', 'manhattan']
}

gs = GridSearchCV(
    KNeighborsClassifier(),
    grid_params,
    verbose = 1,
    cv = 3,
    n_jobs = -1
    )

gs_results = gs.fit(X_train, y_train)
```

Contents

Unit 04 | KNN 고려사항

1. **Distance 기반 알고리즘**
   - 변수들의 단위 (Scale)에 민감 Feature Scaling
   - categorical은? One-hot encoding

2. **다수 클래스 , 평균 보다 더 좋은 방법?** Weighted KNN

Unit 04 | KNN 고려사항

1. Feature Scaling

Min-Max Normalization

- 데이터를 일반적으로 0~1 사이의 값으로 변환

$$X = \frac{x - x_{min}}{x_{max} - x_{min}}$$

Standardization

- 표준화를 사용하여 데이터의 평균이 0, 표준편차가 1이 되도록 변환
- 통계학에서는 z-score라고도 부름

$$X = \frac{x - x_{mean}}{x_{std}}$$

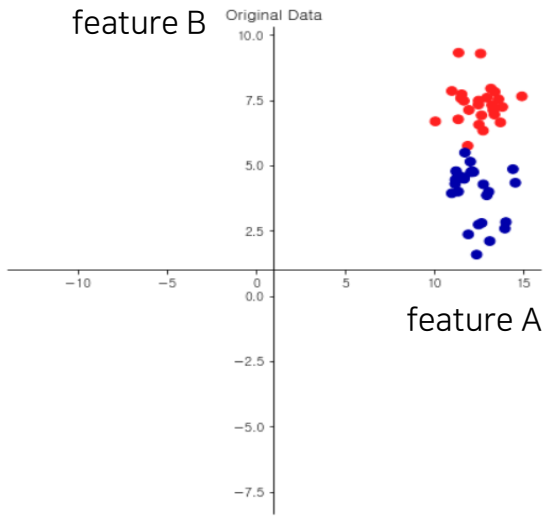
(!주의!)

- train 데이터와 test 데이터의 scale을 따로 조정하면 안 됨.
- train 데이터의 scale을 조정하고자 구한 정규화 parameter(최대최소, 평균/표준편차)등을 기억하여 사용하여 test 데이터도 변환해야 함.

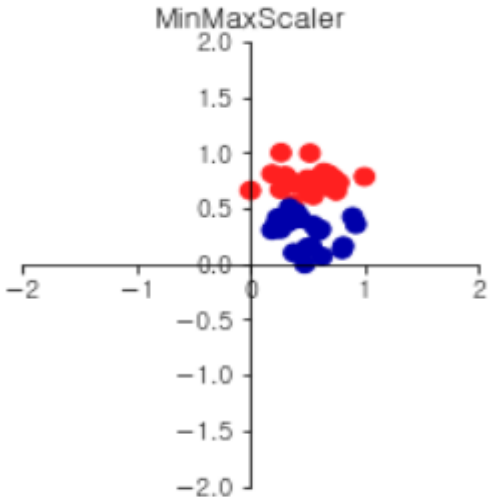
Unit 04 | KNN 고려사항

1. Feature Scaling

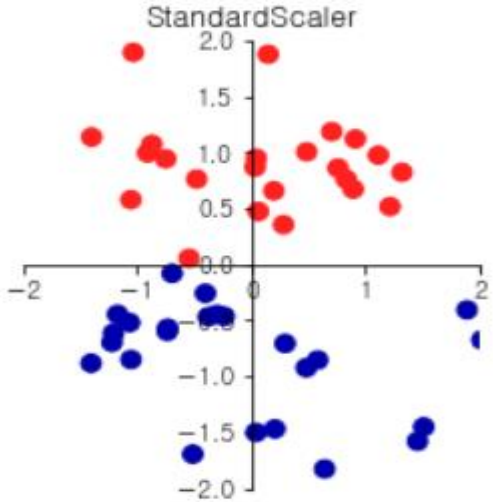
Original data



Min-Max Normalization



Standardization

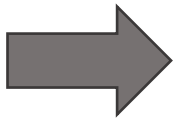


Unit 04 | KNN 고려사항

2. One-Hot encoding

- categorical 값을 feature로 만든 후 1 또는 0으로 지정하는 방법
- 즉, 1개만 Hot(1)이고 나머지는 Cold(0)
- KNN은 거리 기반 -> input에 numerical 와야 함.

color
red
green
blue
red



color_red	color_blue	color_green
1	0	0
0	0	1
0	1	0
1	0	0

Unit 04 | KNN 고려사항

2. One-Hot encoding

color
red
green
blue
red

color_red	color_blue	color_green
1	0	0
0	0	1
0	1	0
1	0	0

color
1
2
3
1

Q. 왜 이렇게 구분하면 안 되는 거지?

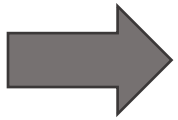
-> 잘못된 관계 형성될 수 있음
red + green = blue (??)

Unit 04 | KNN 고려사항

3. Weighted KNN

- 단순히 다수 클래스, 평균으로 예측하지 않고 거리에 따라서 영향력을 달리 주고 싶을 때 사용

재료	음식종류	거리
딸기	과일	1
오이	채소	2
자몽	과일	3
상추	채소	4
아스파라거스	채소	5



New data의 범주

: 채소

Unit 04 | KNN 고려사항

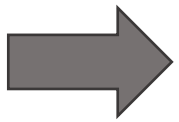
3. Weighted KNN

주의 : 이해를 쉽게 하기 위해 유사도를 '1/거리'로 정의

재료	음식종류	거리	1/거리 = 유사도	가중치
딸기	과일	1	1	0.44
오이	채소	2	0.5	0.22
자몽	과일	3	0.33	0.15
상추	채소	4	0.25	0.11
아스파라거스	채소	5	0.2	0.08

P(new data = 과일) = 0.44 + 0.15 = 0.59

P(new data = 채소) = 0.22 + 0.11 + 0.08 = 0.41



New data의 범주

: 과일

Contents

Unit 05 | KNN 장단점

Good

1. 이해하기 매우 쉬운 모델이다.
2. 예측을 하는 시점에서 모든 기존 데이터와의 거리를 계산하기 때문에 예측 전에 모델을 따로 학습시킬 필요가 없다.
3. 많이 조정하지 않아도 (데이터가 충분히 많으면) 보통 좋은 성능을 보인다.
4. 더 복잡한 알고리즘을 적용해보기 전에 시도해 볼 수 있는 좋은 시작점이다.

Unit 05 | KNN 장단점

Bad

1. 하나의 데이터를 예측할 때마다 전체 데이터와의 거리를 계산하기 때문에 연산 속도가 다른 알고리즘에 비해 느리다.
2. 자연어 처리처럼 높은 차원의 희소한 데이터를 다루는 경우에는 성능이 낮다.

15기 정규세션

ToBig's 14기 김민경

Clustering

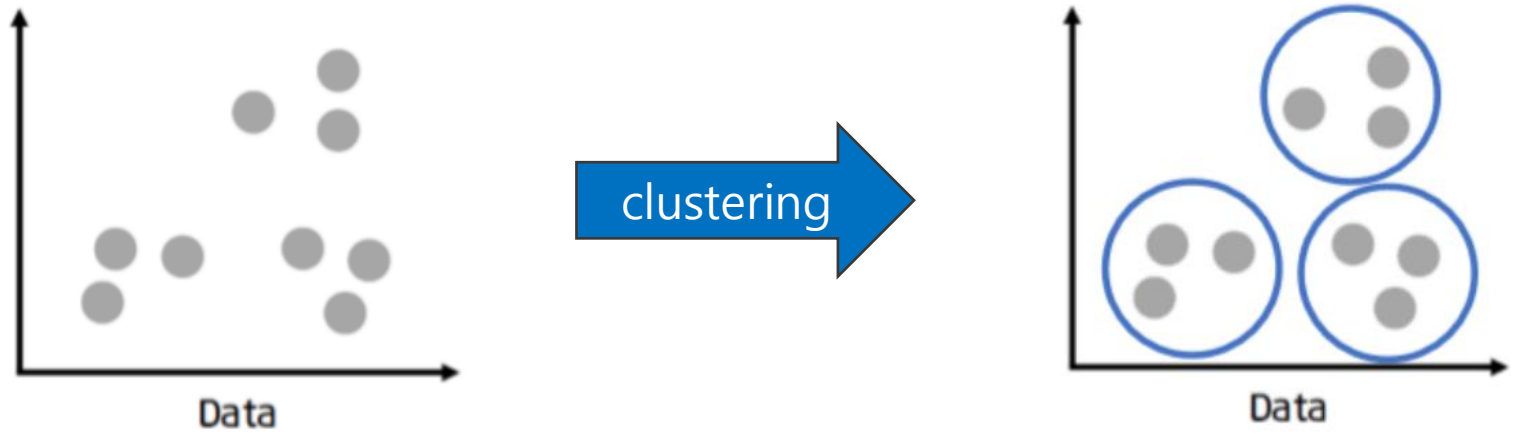
Contents

Unit 01 | Clustering

(!주의!)
- 비지도 학습을 할 때에는 일반적으로 데이터를 적절하게 scaling해야 함.
- categorical 데이터 One-Hot encoding 해주기

Clustering (군집화)

유사한 속성을 갖는 데이터를 묶어 전체 데이터를 몇 개의 군집(그룹)으로 나누는 것

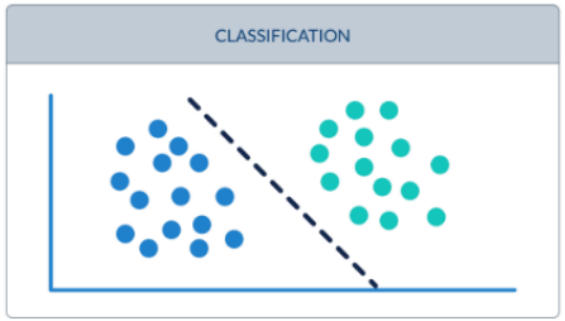


Unit 01 | Clustering

Classification vs Clustering

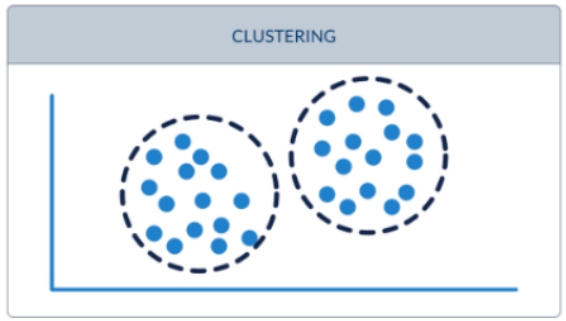
Classification (Supervised)

- 소속 집단의 정보를 이미 알고 있는 상태에서 비슷한 집단으로 묶는 방법
- 즉, label이 있는 데이터를 나누는 방법



Clustering (Unsupervised)

- 소속 집단의 정보가 없고, 모르는 상태에서 비슷한 집단으로 묶는 방법
- 즉, label이 없는 data를 나누는 방법



Unit 01 | Clustering

Clustering의 적용

- 데이터들로부터 유의미한 그룹들을 찾음
- 주로 데이터의 경향성을 파악

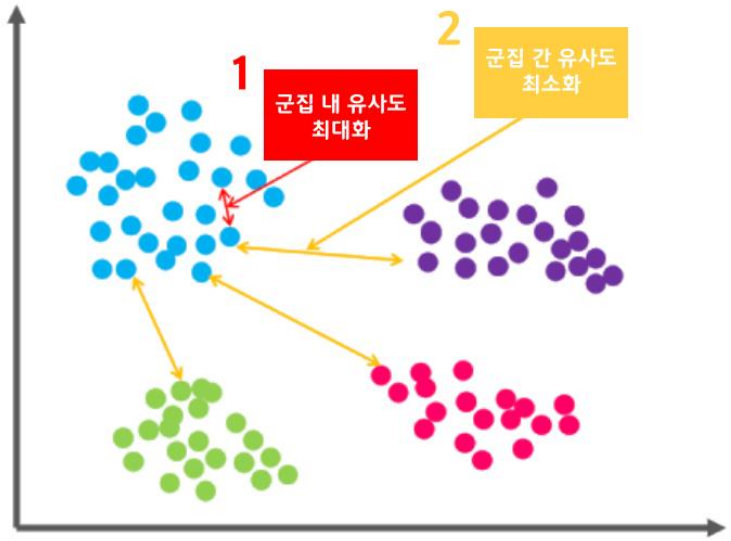
  - 구해진 그룹들을 직접 사용
  - 예측을 위한 회귀나 분류 모델의 input 또는 target feature로 사용

Unit 01 | Clustering

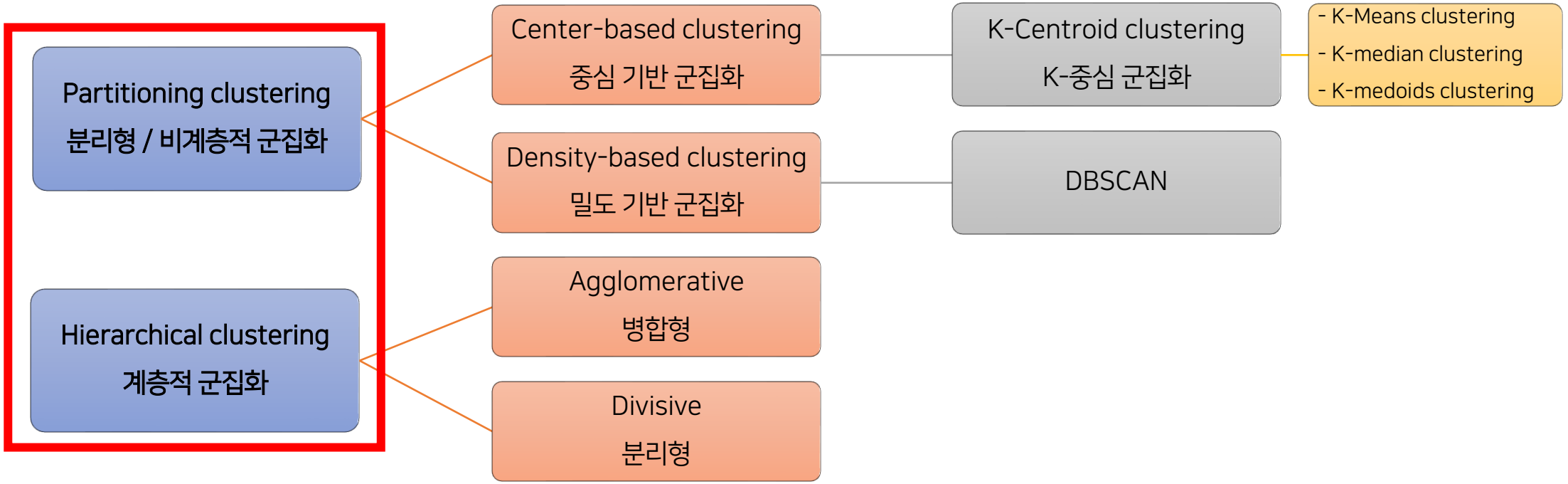
Good Clustering!

1. **Maximizes the similarity within a group**
   : 군집 내 응집도 (cohesion) 최대화

2. **Maximizes the difference between groups**
   : 군집 간 분리도 (separation) 최대화



Unit 01 | Clustering

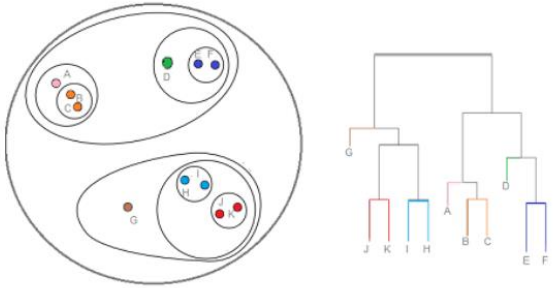


Unit 01 | Clustering

Clustering 방법

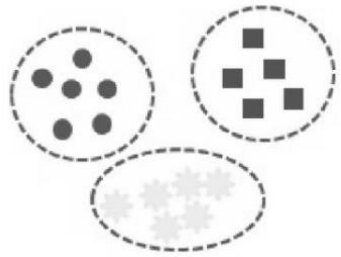
Hierarchical clustering (계층적 군집화)

- 개체들을 가까운 집단부터 차근차근 묶어나가는 방식
- 군집화 결과 뿐만 아니라 유사한 개체들이 결합되는 절차까지
- Agglomerative / Divisive



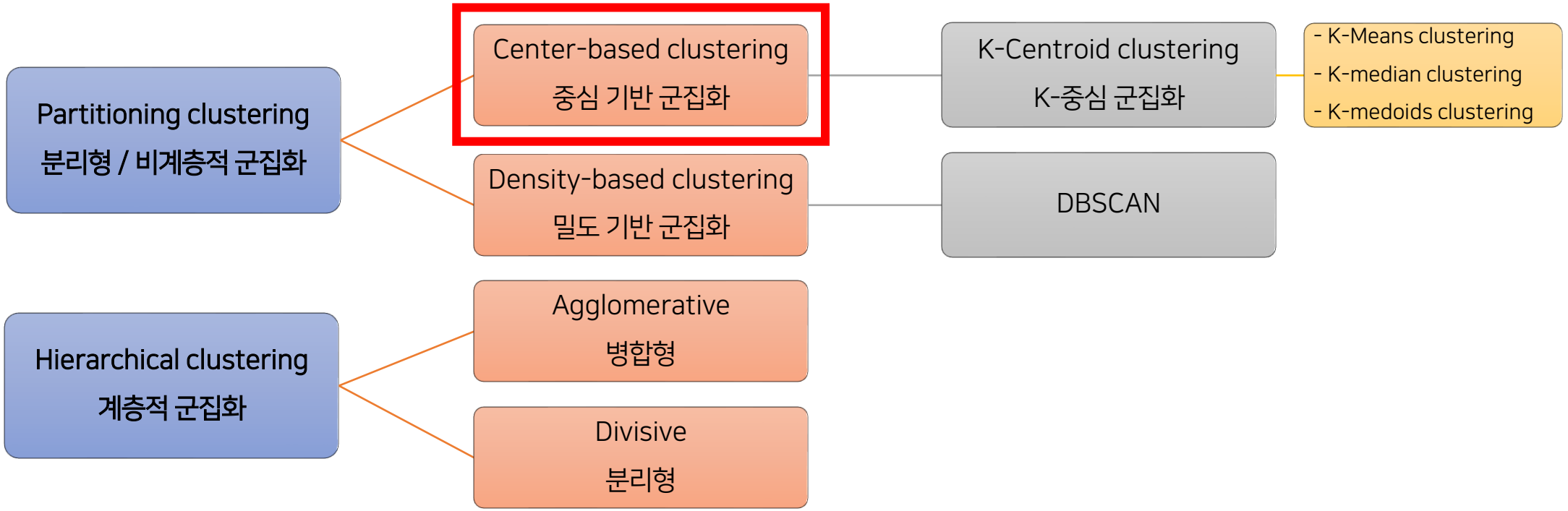
Partitioning clustering (분리형 / 비계층적 군집화)

- 전체 데이터의 영역을 특정 기준에 의해 동시에 구분
- K-Centroid / DBSCAN



Contents

Unit 02 | K-Means Clustering

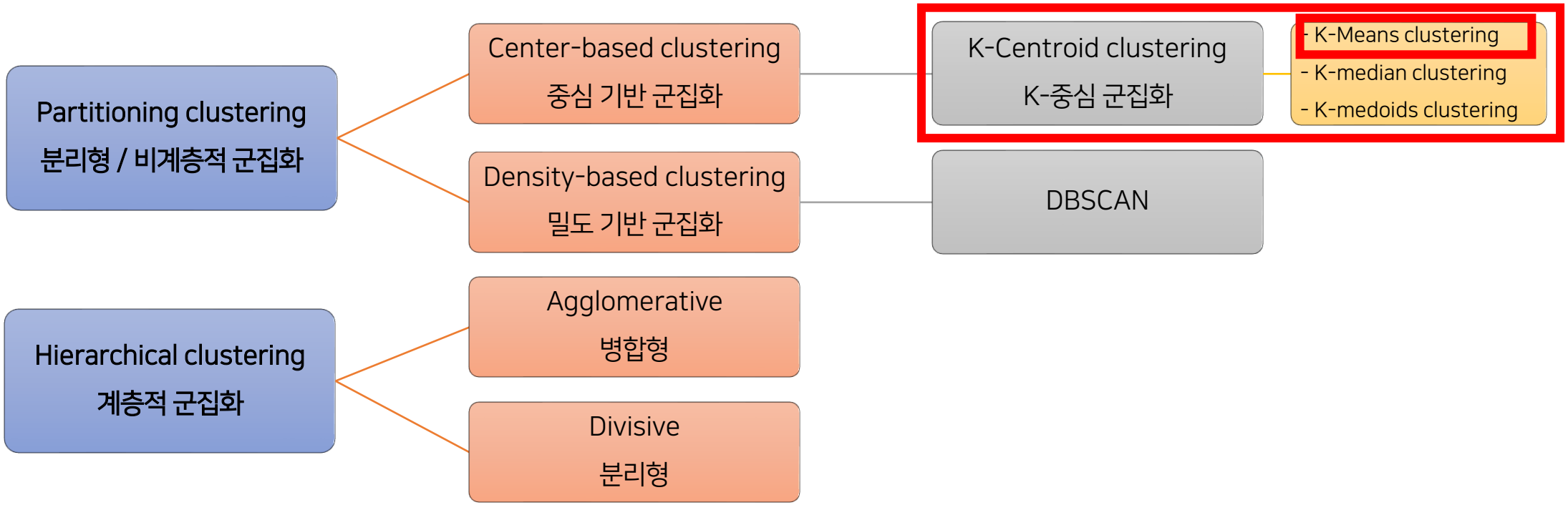


Unit 02 | K-Means Clustering

Partitioning clustering (분리형 / 비계층적 군집화)

1. Center-based clustering (중심 기반)
   - '동일한 군집에 속하는 데이터는 어떠한 중심을 기준으로 분포할 것이다.'라는 가정을 기반
2. Density-based clustering (밀도 기반)
   - '동일한 군집에 속하는 데이터는 서로 근접하게 분포할 것이다.'라는 가정을 기반

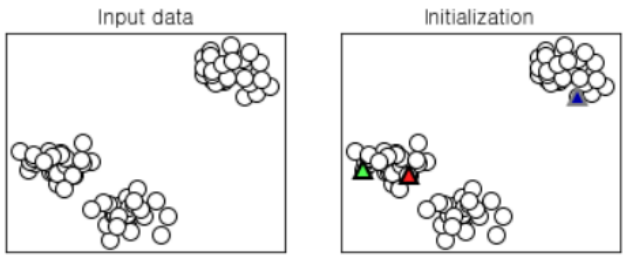
Unit 02 | K-Means Clustering



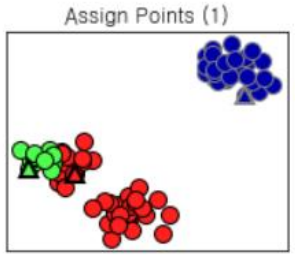
Unit 02 | K-Means Clustering

K-Means Clustering (K-평균 군집)

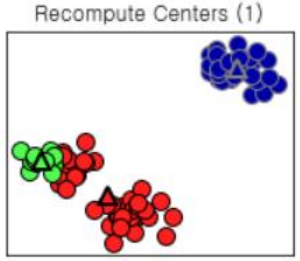
① 초기 중심(centroid)으로 할 K개의 데이터를 임의로 선택한다.



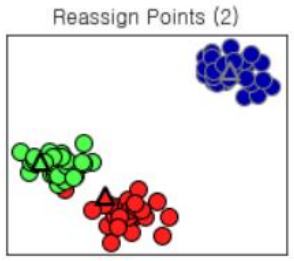
② 각 데이터를 가장 가까운 군집 중심(centroid)에 할당한다.



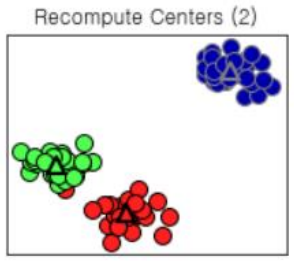
③ 각 군집 내의 데이터들의 평균을 계산하여 군집 중심(centroid)을 update한다.



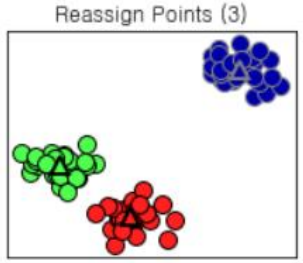
④ 군집 중심(centroid)의 변화가 없을 때(또는 최대 반복수)까지 ②, ③ 과정을 반복한다.



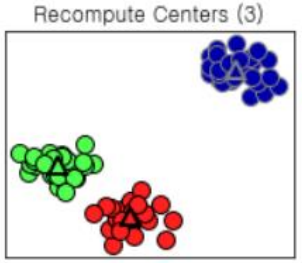
②



③



②

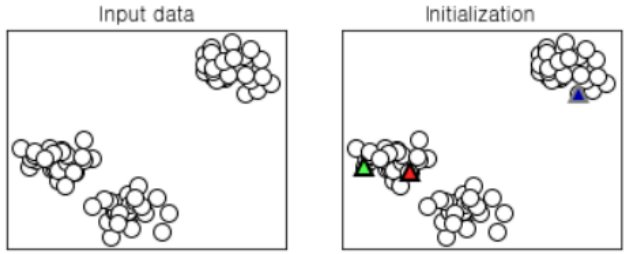


③

- 각 군집은 하나의 중심(centroid)을 가짐
- 사전에 군집의 수, K가 정해져야 함.
- 유클리드 거리 기반 알고리즘

Unit 02 | K-Means Clustering

① 초기 중심(centroid)으로 할 K개의 데이터를 임의로 선택한다.



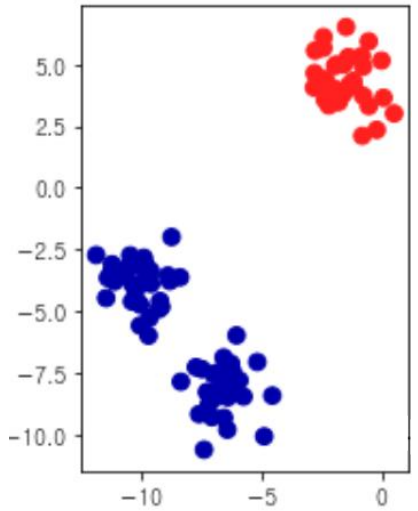
1. K(군집의 수)는 몇 개로 지정하지?

2. 초기 중심(centroid)으로 할 데이터를 선택하는 기준은?

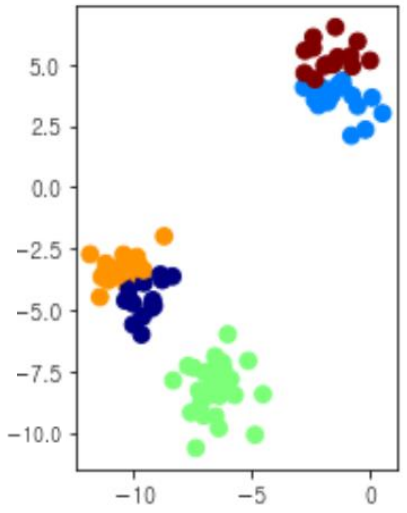
Unit 02 | K-Means Clustering

1. K(군집의 수)는 몇 개로 지정하지?

Elbow Method **Silhouette Coefficient**




```
kmeans = KMeans(n_clusters=2)
kmeans.fit(X)
```




```
kmeans = KMeans(n_clusters=5)
kmeans.fit(X)
```

Unit 02 | K-Means Clustering

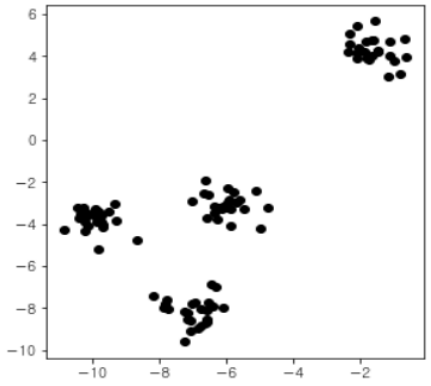
Elbow Method

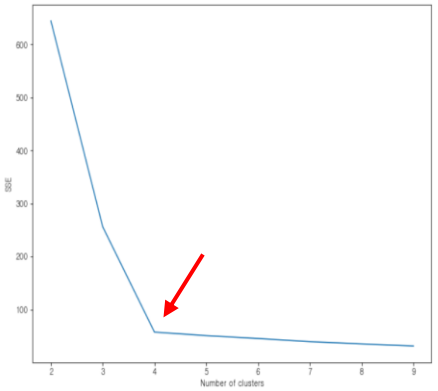
- '군집 내 편차제곱합'(WSS)이 최소가 되도록 군집의 중심을 결정해 나가는 방법
- WSS의 총합이 급격하게 감소하기 시작하는 k를 선택
- 그래프가 꺾이는 모양이 팔꿈치 같아 Elbow Method라고 불림
- 뚜렷하게 구분된 군집이 없는 데이터에서는 눈에 띄는 Elbow point를 찾기 어렵다는 단점이 있음

$$WSS = \sum_{x=C}(x - \mu_C)^2$$

x : 데이터

μ_C : 군집의 중심. K-Means의 경우 군집 안 데이터들의 '평균'이 됨.



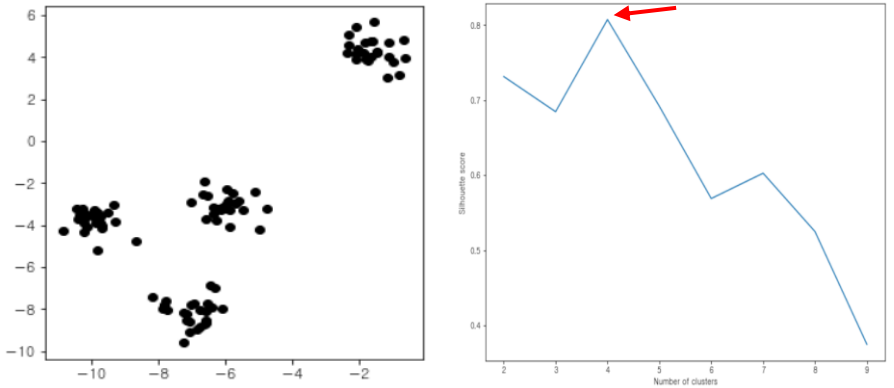


Unit 02 | K-Means Clustering

Silhouette Coefficient

- 실루엣 계수는 군집 안의 데이터가 자신이 속한 군집 안의 다른 데이터와 얼마나 유사하며, 다른 군집에 속한 데이터와 얼마나 차이가 나는지 측정
- -1~1사이의 값을 가지며 1에 가까울 수록 적절한 군집화가 되었다고 판단

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




```
from sklearn.metrics import silhouette_score

kmeans = KMeans(n_clusters=4)
kmeans.fit(X)

silhouette_score(X, kmeans.labels_)
```

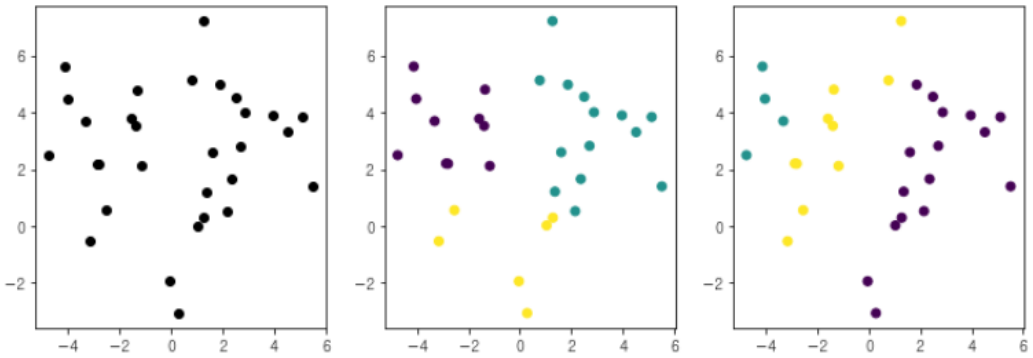
```
0.8071639409492641
```

a : 데이터 x와 동일한 군집 내의 나머지 데이터들과의 평균 거리 - **군집 내 응집도** (cohesion)

b : 데이터 x와 가장 가까운 군집 내의 모든 데이터들 간의 평균 거리 - **군집 간 분리도** (separation)

Unit 02 | K-Means Clustering

2. 초기 중심(centroid)으로 할 데이터를 선택하는 기준은?



<random_state에 따른 차이>

가장 기본적인 방법은 랜덤 초기화

하지만 랜덤 초기 중심 설정의 위험을 피하고자 다양한 연구 존재

- 반복적으로 수행하여 가장 여러 번 나타나는 군집 사용
- 전체 데이터 중 일부만 샘플링하여 계층적 군집화를 수행한 뒤 초기 군집 중심 설정
- 데이터 분포의 정보를 사용하여 초기 중심 설정

  (초기 군집 중심을 가능한 멀리 떨어지도록 만드는 K-Means++ 알고리즘 사용)

Unit 02 | K-Means Clustering

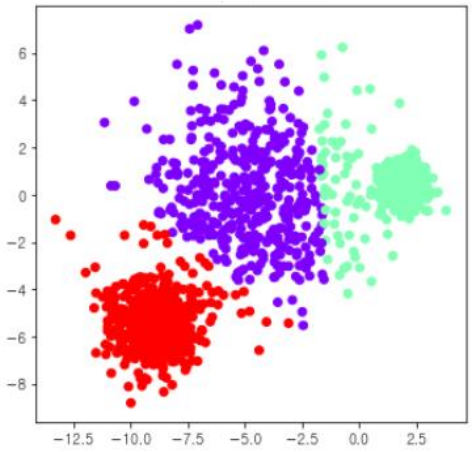
K-Means의 기본 가정

1. 군집을 구 형태로 간주 (ex. 원, 타원)
2. 모든 feature는 동일한 scale을 가짐 (feature scaling 필요)
3. 군집들의 크기가 대체로 비슷함 (노이즈에 취약)

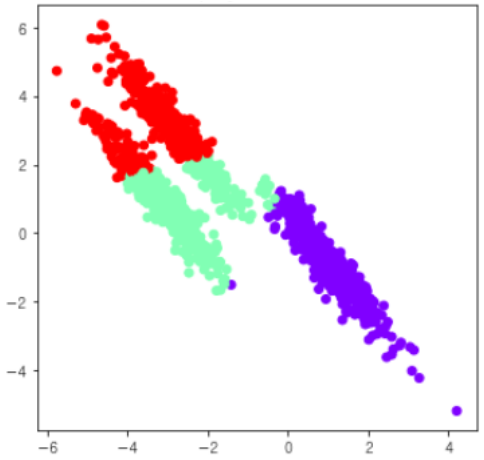
Unit 02 | K-Means Clustering

K-Means Clustering의 한계

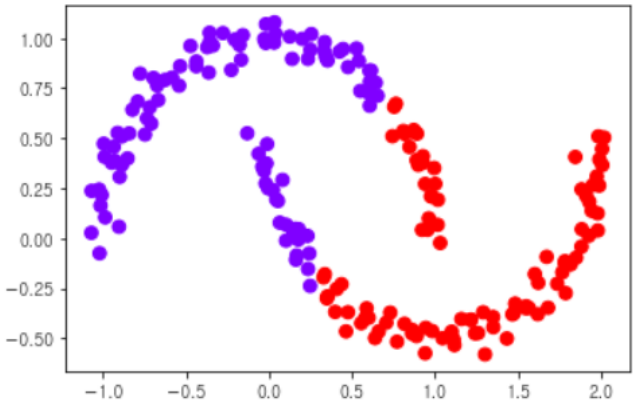
1) 밀도가 다른 군집을 잘 구분하지 못함



2) 원형이 아닌 군집을 잘 구분하지 못함

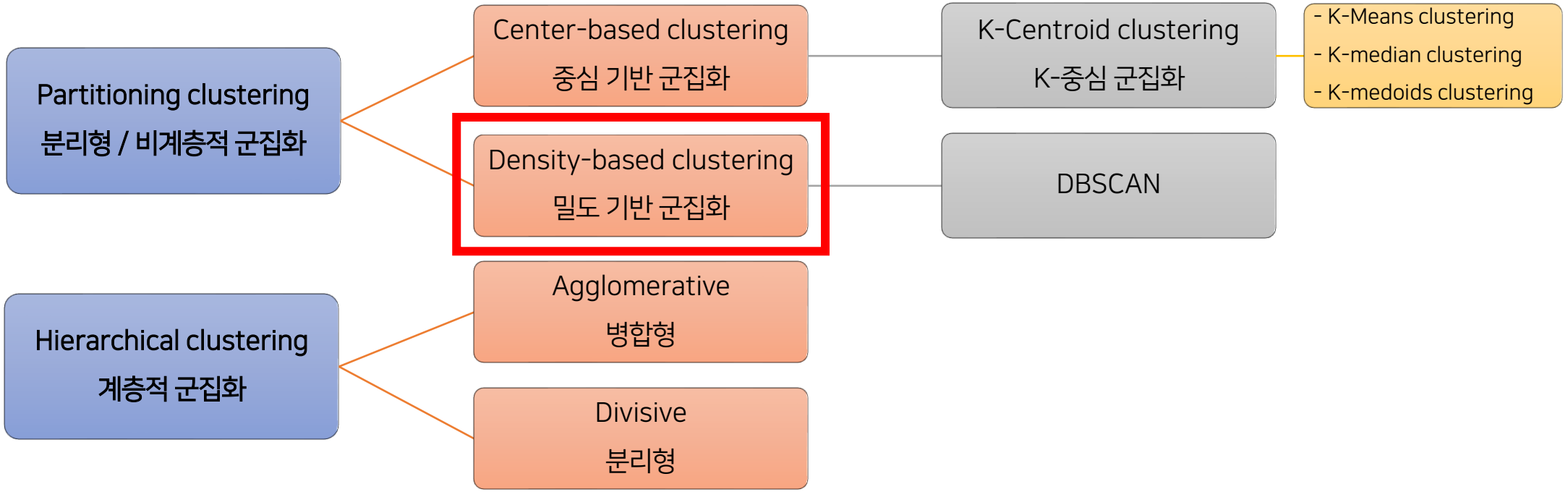


3) 복잡한 형상의 군집을 잘 구분하지 못함



Contents

Unit 03 | DBSCAN



Unit 03 | DBSCAN

Partitioning clustering (분리형 / 비계층적 군집화)

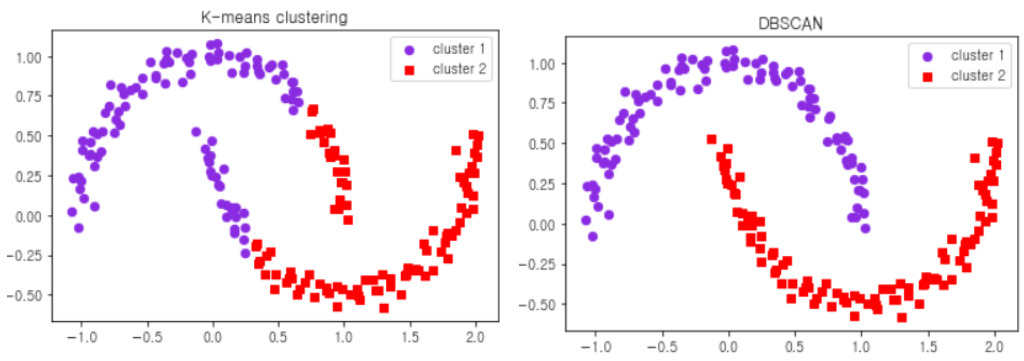
1. Center-based clustering (중심 기반)
   - '동일한 군집에 속하는 데이터는 어떠한 중심을 기준으로 분포할 것이다.'라는 가정을 기반

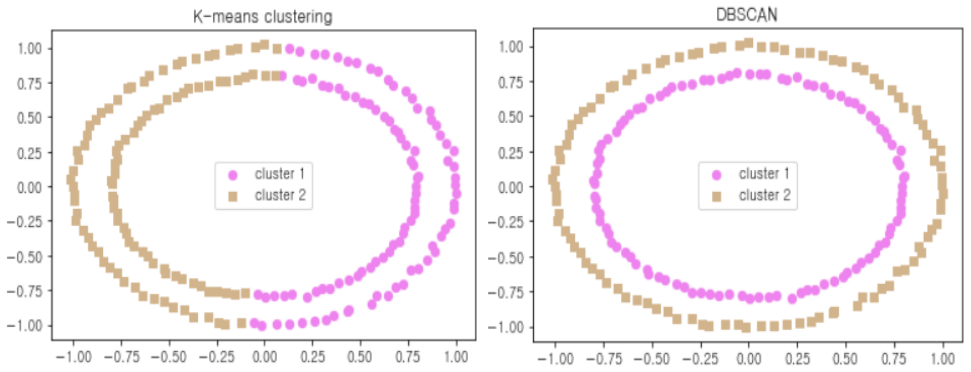
2. Density-based clustering (밀도 기반)
   - '동일한 군집에 속하는 데이터는 서로 근접하게 분포할 것이다.'라는 가정을 기반

Unit 03 | DBSCAN

DBSCAN (Density-Based Spatial Clustering of Applications with Noise)

- 군집의 개수를 미리 지정할 필요 없음 (자동으로 최적의 군집개수 찾음)
- 군집 모양에 대한 어떠한 가정도 하지 않음 (<-> K-Means는 원형 가정)
- 데이터의 밀집 지역에 한 군집을 구성하며 비교적 비어있는 지역을 경계로 다른 군집과 구분됨
- 복잡한 형상도 찾을 수 있으며, 어떤 클래스에도 속하지 않는 데이터를 구분할 수 있음
- 클러스터링을 수행하는 동시에 노이즈 데이터도 분류할 수 있기 때문에 이상치에 의해 클러스터링 성능이 하락하는 현상을 완화





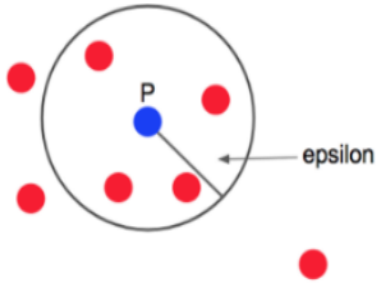
Unit 03 | DBSCAN

DBSCAN (Density-Based Spatial Clustering of Applications with Noise)

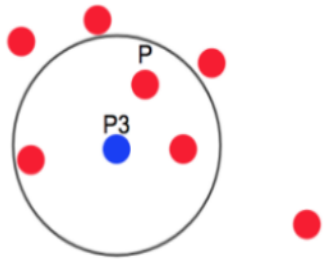
- eps(epsilon) : 한 데이터가 주변의 얼마만큼 떨어진 거리를 같은 군집이라고 할 것인가? (주변 거리)
- min_samples : 적어도 한 군집에 몇 개의 데이터가 있어야 군집이라고 할 것인가? (최소 데이터 수)
- metrics : eps에서 거리 측정 방식 (default : euclidean)

=> 거리 eps 내에 데이터가 min_samples개 이상 있으면 '하나의 군집'으로 인식

min_samples = 4

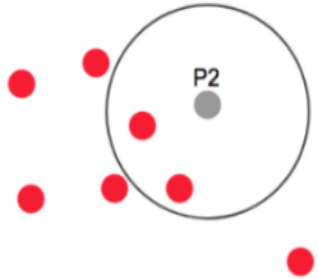


P3 : 핵심 포인트



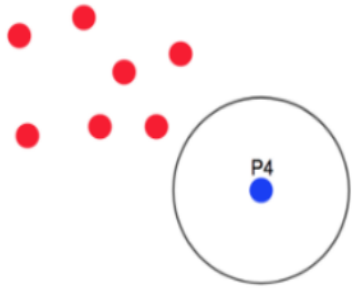
eps 거리 내에 데이터가 (자신 포함)
4개 이상 있는 포인트

P2 : 경계 포인트



핵심 포인트를 이웃으로 갖고 있지만
eps 거리 내에 데이터가 4개 보다 적은 포인트

P4 : 잡음 포인트

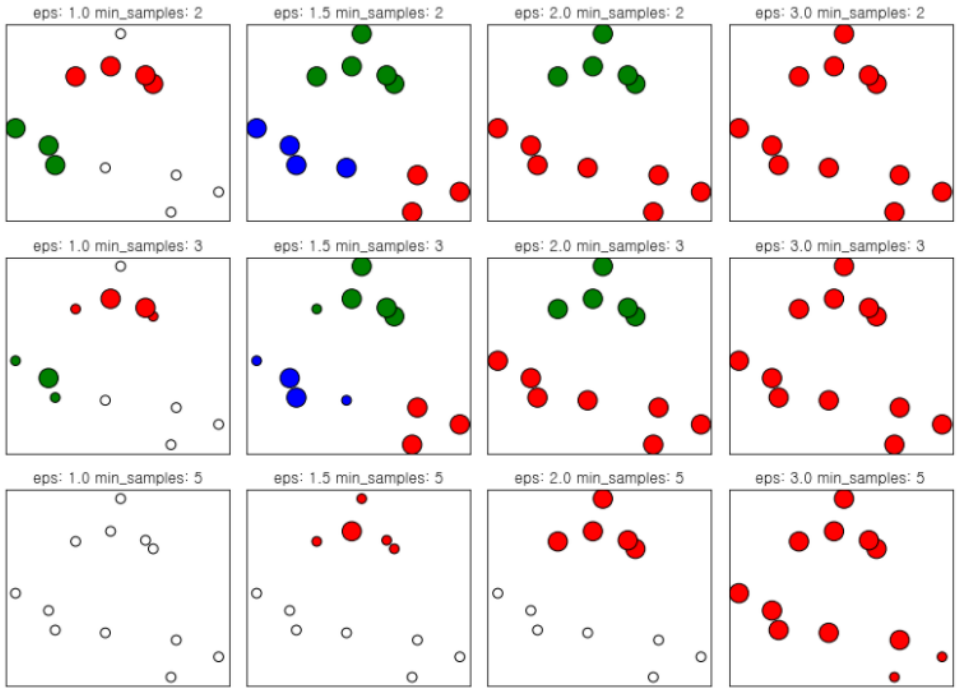


핵심 포인트를 이웃으로 갖고 있지 않고
eps 거리 내에 데이터가 4개 보다 적은 포인트

Unit 03 | DBSCAN

eps 증가 : 하나의 군집에 더 많은 데이터가 포함 (= 군집의 크기 증가)

min_samples 증가 : 핵심 포인트 줄고 잡음 포인트 증가



- 흰색 아닌 큰 원 : 핵심 포인트
- 흰색 아닌 작은 원 : 경계 포인트
- 흰색 원 : 잡음 포인트

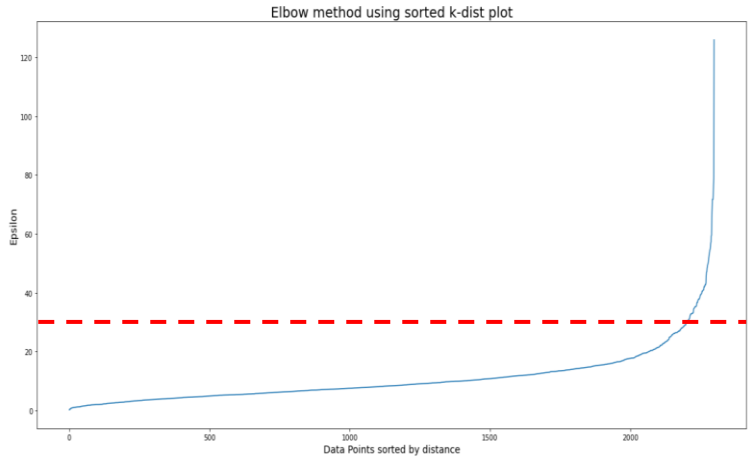
Unit 03 | DBSCAN

1. min_samples 선택

- 데이터의 차원보다 크거나 같게
- 일반적으로 데이터 차원(d)의 2배, 즉 min_samples ≈2 * d를 사용
- 전체 데이터 수(n)를 감안해서 min_samples = ln(n)을 사용하기도 함
- 실제로 도메인 지식에 크게 의존함.

2. eps 선택

- min_samples 선택하면 Elbow Method로 시각화해서 선택 가능



Unit 03 | DBSCAN

작동 방식

1. 랜덤으로 데이터 포인트를 선택
2. 그 포인트에서 eps 거리안의 모든 포인트를 찾음
    - 2-1 eps 거리 안에 있는 데이터 수가 min_samples보다 적다면 어떤 클래스에도 속하지 않는 잡음 포인트로 레이블
    - 2-2 eps 거리 안에 있는 데이터 수가 min_samples보다 많으면 핵심 포인트로 레이블하고 새로운 클러스터 레이블 할당
3. 2-2의 핵심 포인트의 eps거리안의 모든 이웃을 살핌
    - 3-1 만약 어떤 클러스터에도 아직 할당되지 않았다면 바로 전에 만든 클러스터 레이블을 할당
    - 3-2 만약 핵심 포인트면 그 포인트의 이웃을 차례로 확인
4. eps 거리안에 더이상 핵심 포인트가 없을 때까지 진행

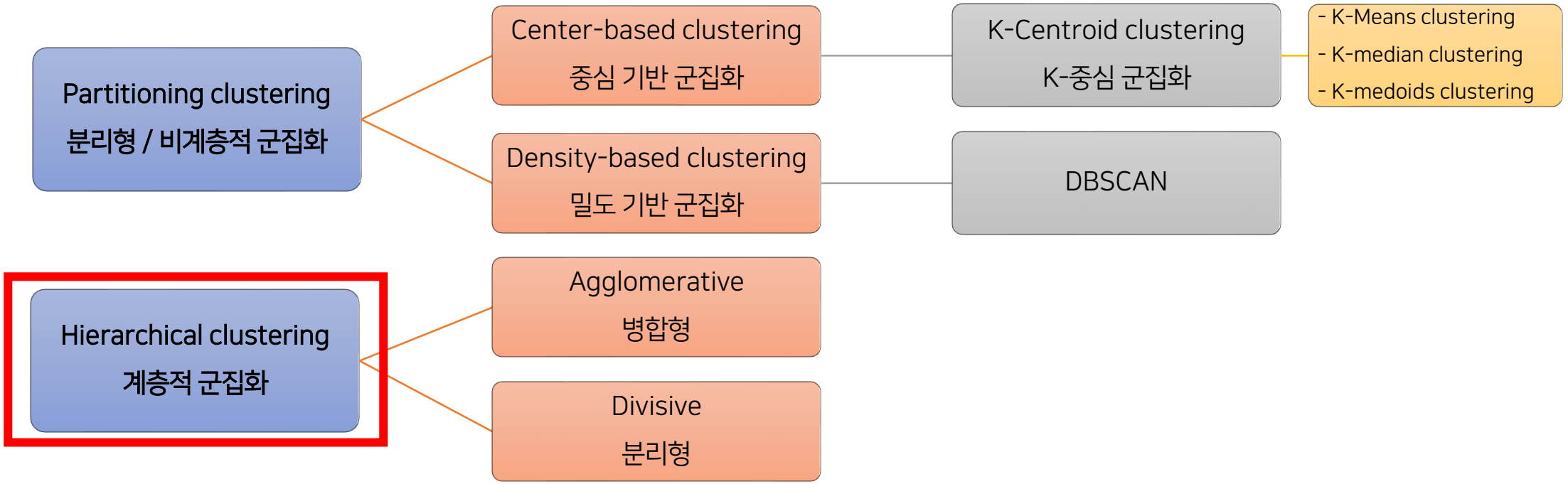
Unit 03 | DBSCAN

DBSCAN의 한계

- 사전에 데이터에 대한 충분한 이해도를 갖고 있지 않다면 eps와 min_samples의 값을 정하기 어려움
- 연산량이 많아 K-Means에 비해 속도가 느림
- 차원의 저주 문제
  - 차원 수가 낮은 데이터는 문제가 되지 않지만 고차원 데이터로 갈수록 학습 데이터 양이 급증해 많은 연산이 필요
  - 유클리드 거리 사용하는 모든 모델의 공통적인 단점

Contents

Unit 04 | Hierarchical Clustering

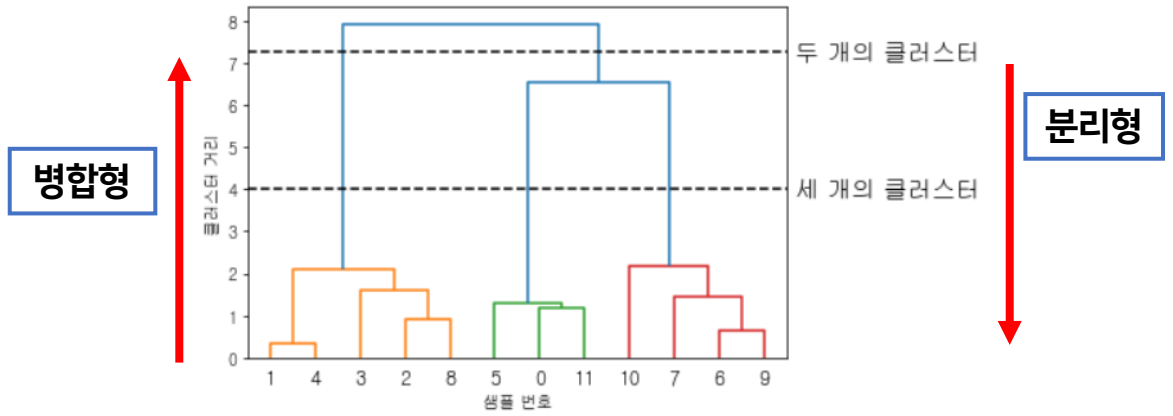


Unit 04 | Hierarchical Clustering

Hierarchical clustering (계층적 군집화)

- 개체들을 가까운 집단부터 차근차근 묶어나가는 방식
- 사전에 군집의 수를 미리 정해주지 않아도 되는 장점
- 덴드로그램으로 군집화 과정을 시각화하기 쉬움

  (덴드로그램 생성 후 적절한 수준에서 자르면 그에 해당하는 군집화 결과 생성)
- Agglomerative(병합형) / Divisive(분리형)

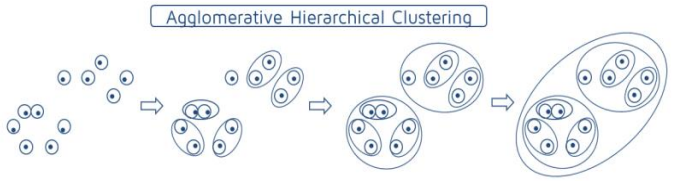


<덴드로그램>

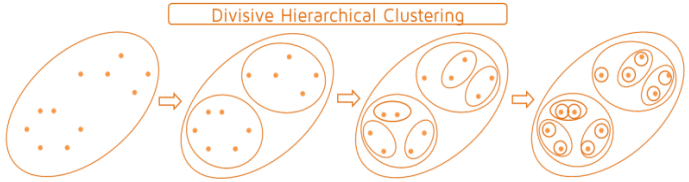
(!주의!)

매번 한 쌍의 군집만 비교하므로 2^n개의 군집만 생성 가능

(군집화 전에 군집 수가 2^n개가 아니라는 점을 알고 있다면 K-Means 추천)

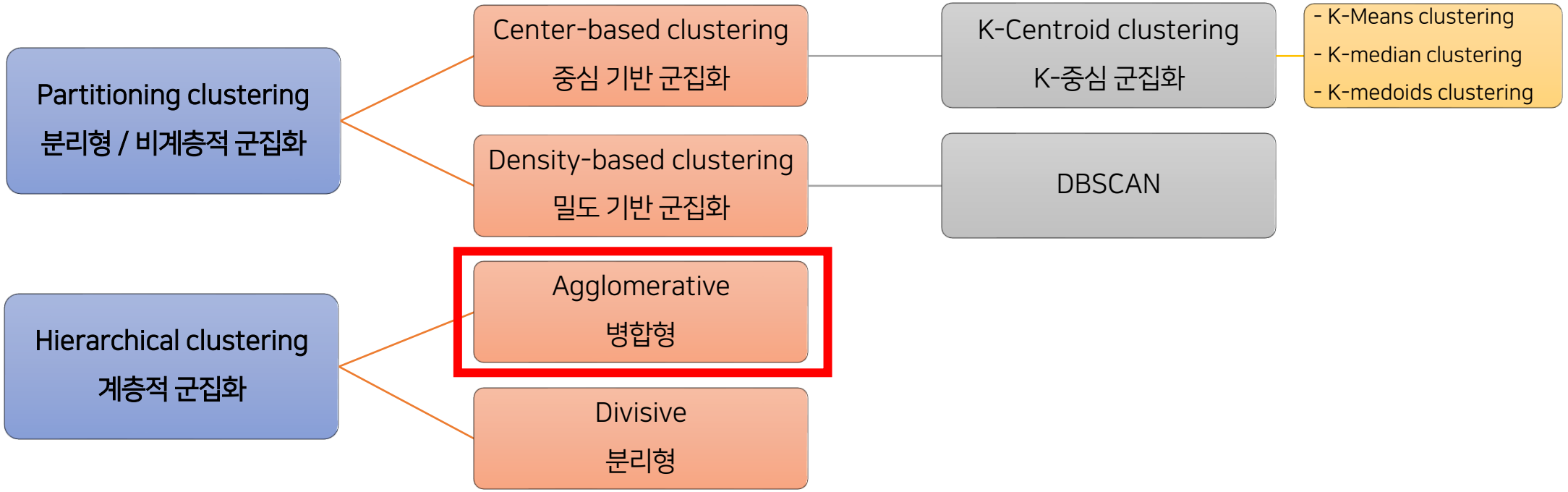


병합형
Bottom-up



분리형
Top-down

Unit 04 | Hierarchical Clustering



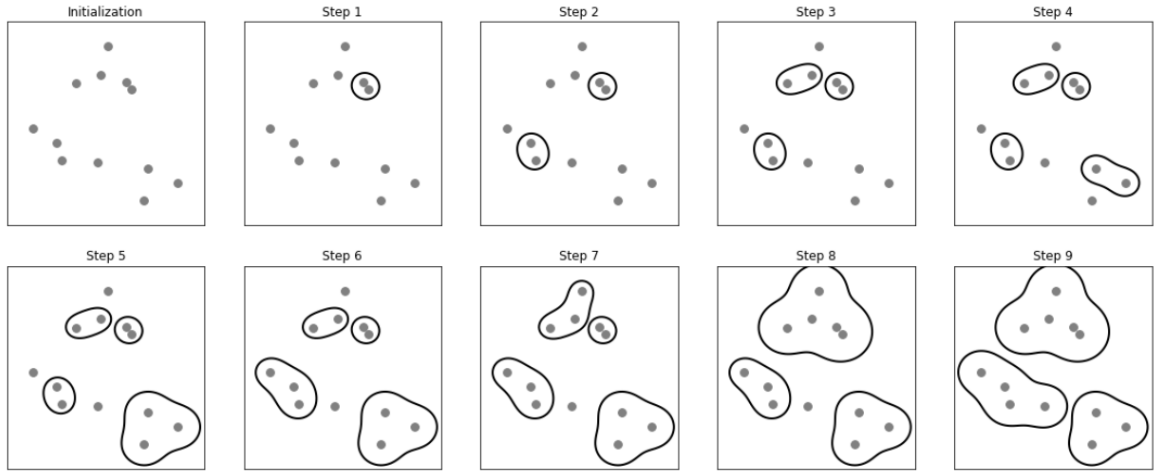
Unit 04 | Hierarchical Clustering

Agglomerative Hierarchical Clustering (병합형 계층적 군집화)

① 하나의 데이터를 하나의 군집으로 지정한다.

② 과정 ①의 군집들에 대해 가장 유사도가 높은 군집 둘을 하나로 합친다. (default : Euclidean)

③ 과정 ②에서 생성된 군집들에 대해 다시 같은 과정을 반복한다.



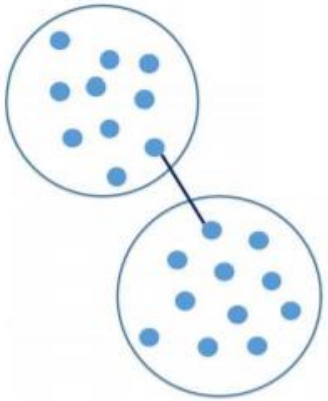
Unit 04 | Hierarchical Clustering

유사도가 높은 군집?

-> 두 개의 군집을 합칠 때 군집 간 유사도(거리) 측정 방법 종류가 많다!

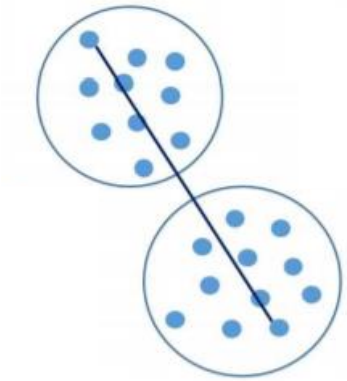
1) Single linkage (단일 연결법)

두 군집에 속하는 데이터들의 거리 중에 '가장 짧은 거리'를 군집 사이의 거리로 간주



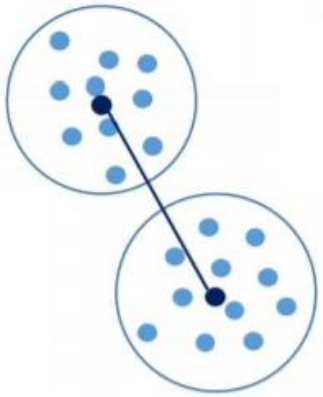
2) Complete linkage (완전 연결법)

두 군집에 속하는 데이터들의 거리 중에 '가장 먼 거리'를 군집 사이의 거리로 간주



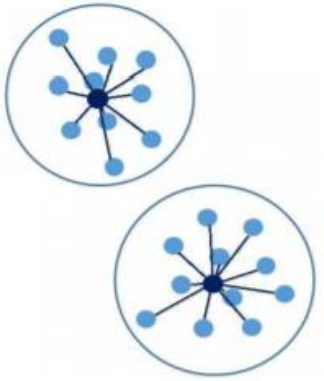
3) Average linkage (평균 연결법)

두 군집에 속하는 데이터들의 거리 '평균'을 군집 사이의 거리로 간주



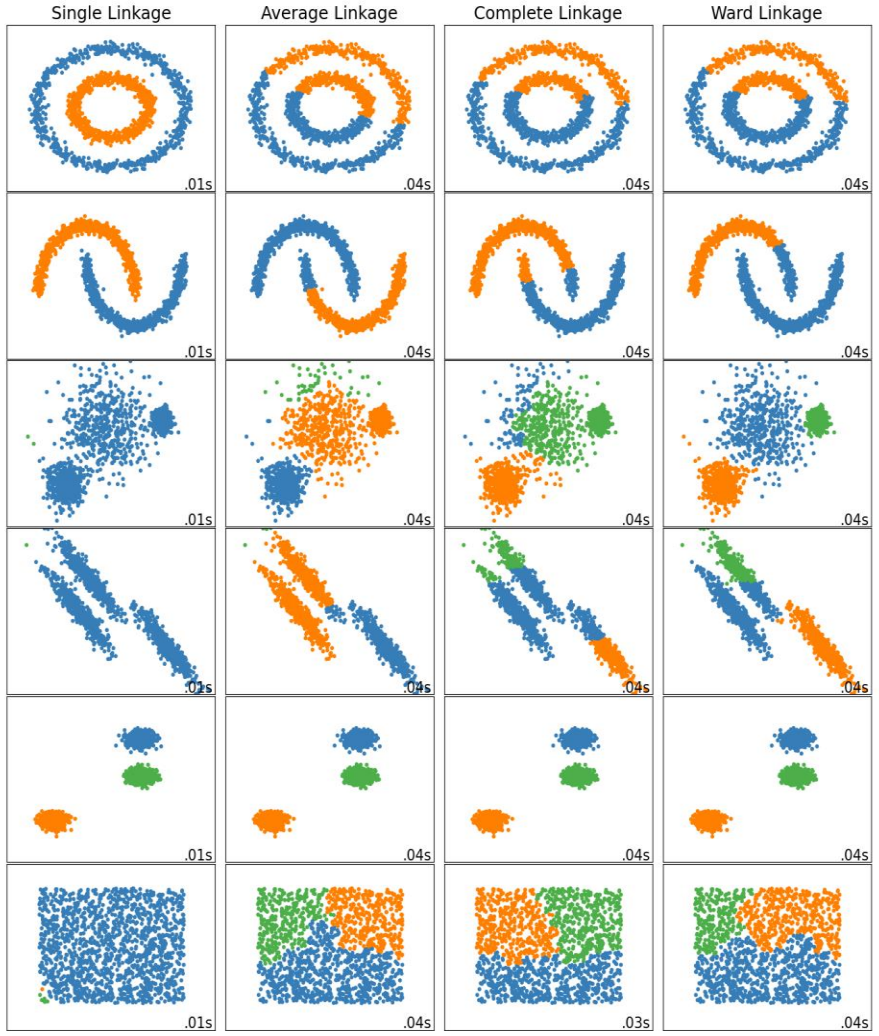
4) Ward linkage (와드 연결법)

군집 내 편차제곱합(WSS)에 기반한 방법
비슷한 크기의 군집끼리 묶어주는 경향



http://contents.kocw.net/KOCW/document/2015/chungbuk/najonghwa1/13.pdf

Unit 04 | Hierarchical Clustering



https://scikit-learn.org/stable/modules/clustering.html

Unit 04 | Hierarchical Clustering

Hierarchical Clustering의 한계

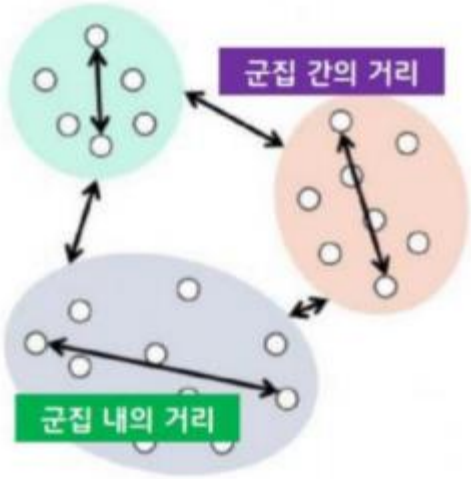
- 대규모 데이터(수백만 개)에는 적용이 불가하며 적당한 크기의 데이터(수만 개)의 경우에도 비용이 많이 든다.
  (대부분 상대적으로 데이터 크기가 작은 문제에 적용되는 알고리즘)

Contents

Unit 05 | 모델평가

1. Dunn's Index

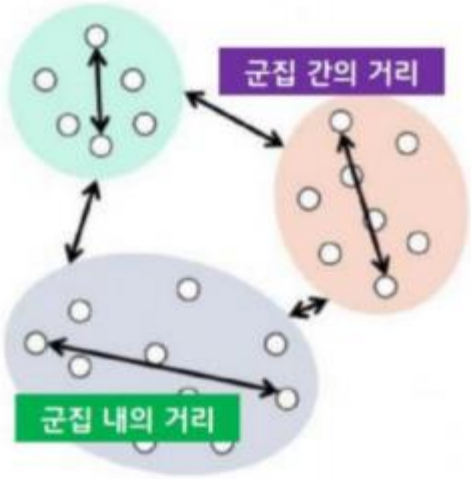
$$DI = \frac{\text{군집과 군집 사이의 거리 중 최솟값}}{\text{군집 내 객체 간 거리 중 최댓값}}$$



Unit 05 | 모델평가

1. Dunn's Index

$$DI = \frac{\text{군집과 군집 사이의 거리 중 최솟값}}{\text{군집 내 객체 간 거리 중 최댓값}}$$



군집과 군집 사이의 거리가 클수록,
군집 내 객체 간 거리가 작을수록 좋은 모델 -> DI가 큰 모델

2. Silhouette Coefficient

- 실루엣 계수는 군집 안의 데이터가 자신이 속한 군집 안의 다른 데이터와 얼마나 유사하며, 다른 군집에 속한 데이터와 얼마나 차이가 나는지 측정
- -1~1사이의 값을 가지며 1에 가까울 수록 적절한 군집화가 되었다고 판단

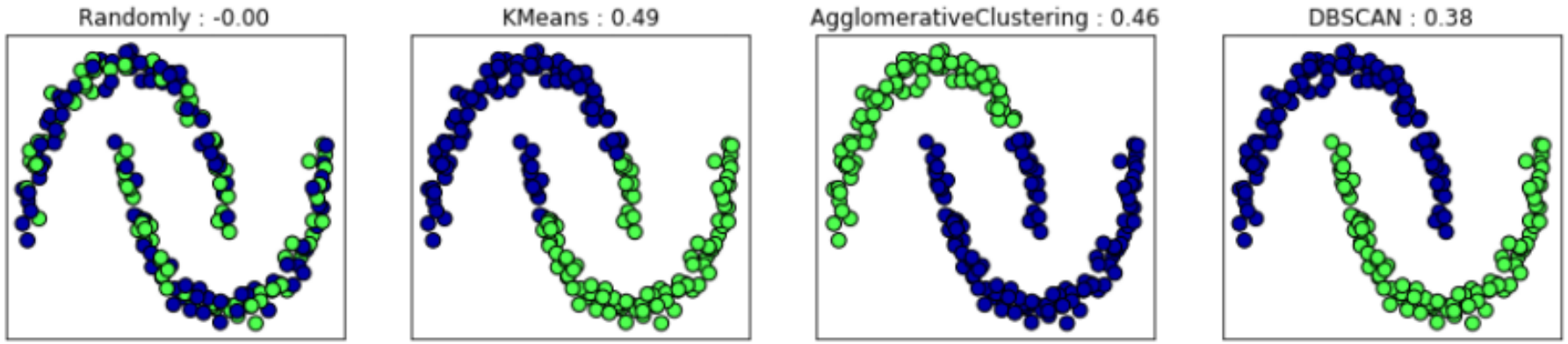
$$s = \frac{b-a}{max(a,b)}$$

a : 데이터 x와 동일한 군집 내의 나머지 데이터들과의 평균 거리 - **군집 내 응집도** (cohesion)

b : 데이터 x와 가장 가까운 군집 내의 모든 데이터들 간의 평균 거리 - **군집 간 분리도** (separation)

Unit 05 | 모델평가

2. Silhouette Coefficient



눈으로 보기엔 DBSCAN의 결과가 더 낫지만 K-Means의 실루엣 계수가 가장 높음
-> 모양이 복잡할 때는 평가가 잘 들어맞지 않음 (원형 클러스터에서 값이 더 높게 나옴)

Reference

참고자료

- 투빅스 13기 김민정님 강의 자료
- 투빅스 3기 박희경님 강의 자료

- https://woolulu.tistory.com/49
- https://srcole.github.io/assets/presentations/cogs108/clustering.html
- http://178.217.173.109/video_lessons/ENGLISH/MACHINE_LEARNING/ENGLISH/pdf/3.pdf
- https://livebook.manning.com/book/machine-learning-for-mortals-mere-and-otherwise/chapter-12/17
- https://wiserloner.tistory.com/1082
- https://m.blog.naver.com/bestinall/221760380344
- 책 파이썬을 활용한 머신러닝 쿡북
- 책 파이썬 라이브러리를 활용한 머신러닝 (변역개정판)

과제

1 KNN으로 Hyperparameter Tuning 이해하기

1) Preprocessing / EDA
2) GridSearchCV or K-fold CV
3) Evaluation

데이터 : blackfriday | Kaggle

2 Clustering 해보기

1) Preprocessing / EDA
2) Clustering (수업시간에 배운 세 가지 방법 + α)
3) Evaluation

데이터 : Mall Customer Segmentation Data | Kaggle

들어주셔서 감사합니다.