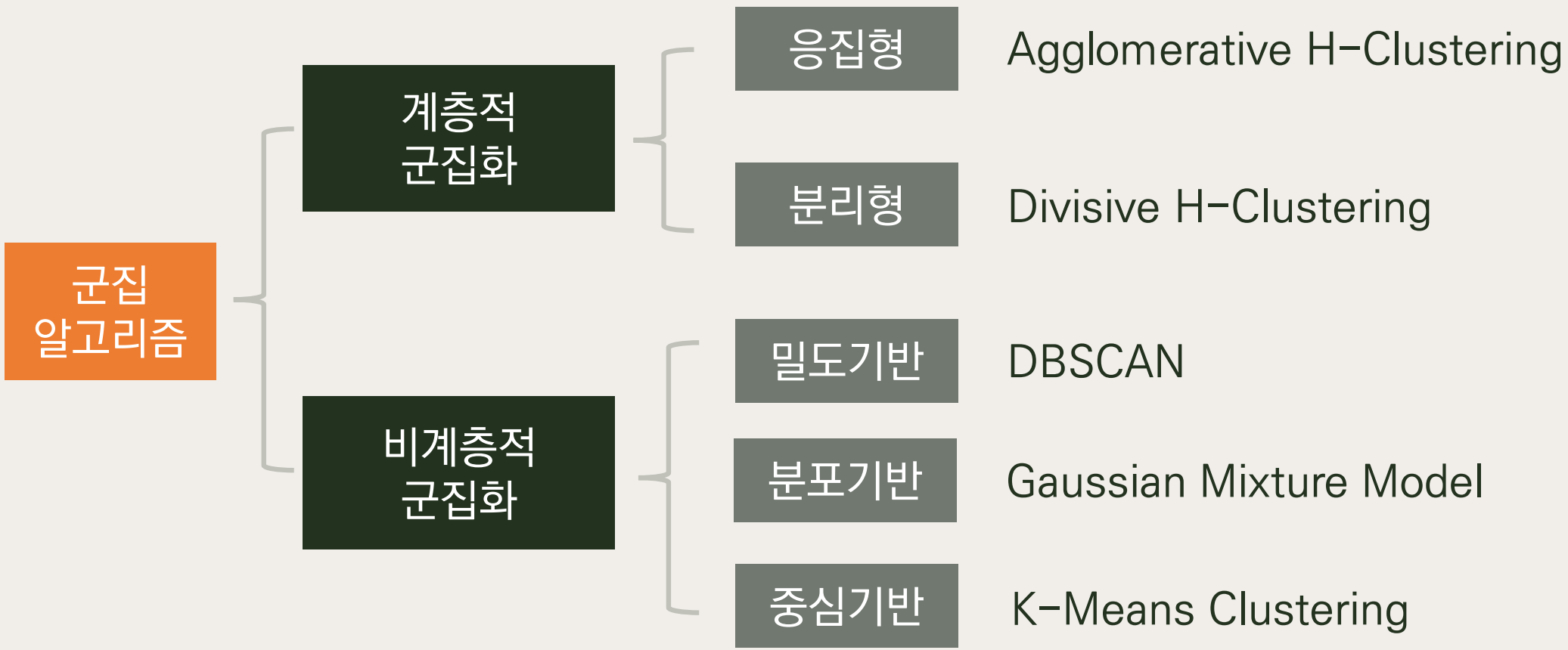
12주차 군집 알고리즘

비타민 10기 4조 노지예 부도현 임청수 한세림

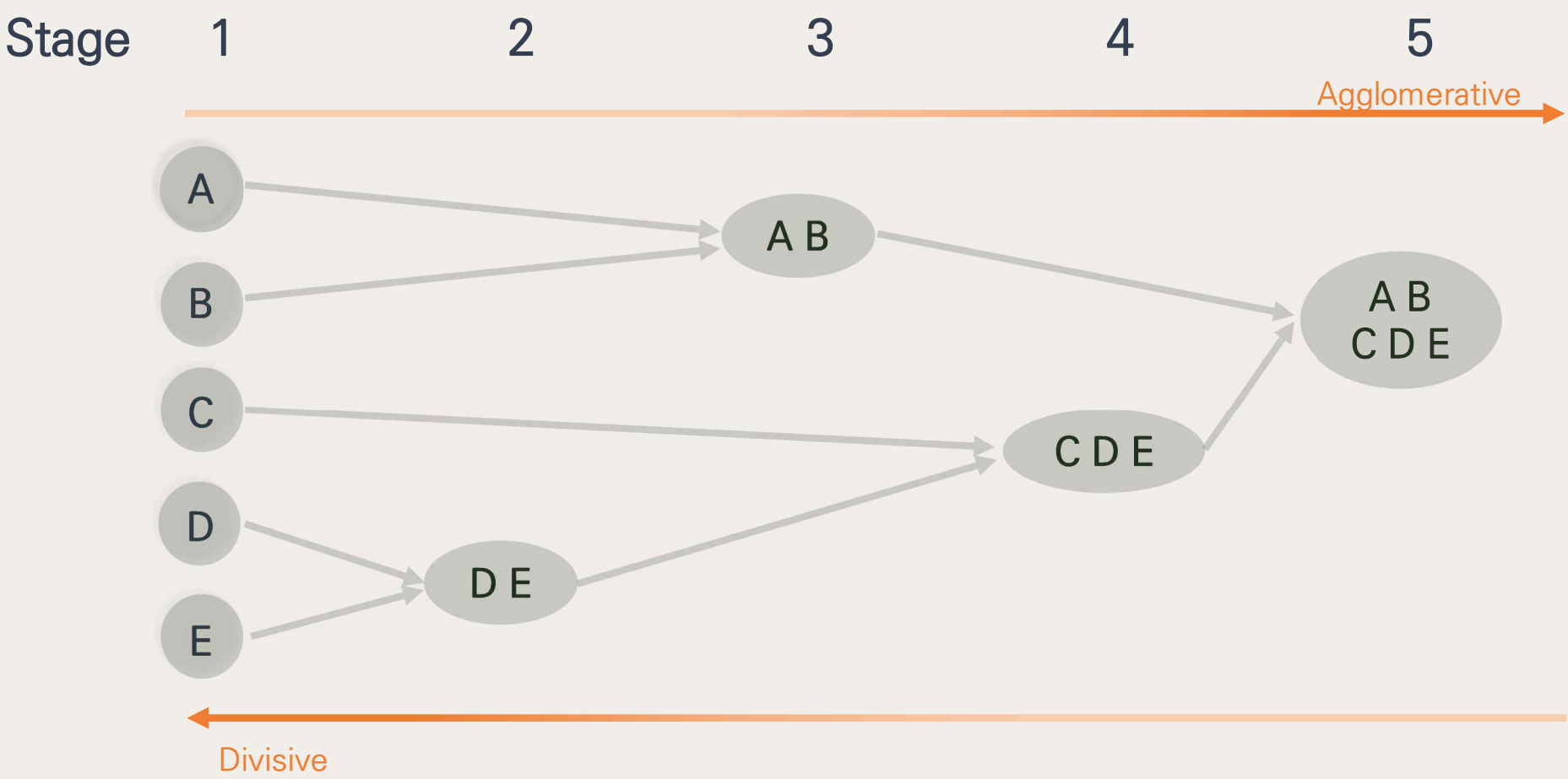
목차

군집 알고리즘(Clustering Algorithm)이란?

- 분류(Classification)가 목적이 아니라 → 지도학습
  분류된 **집단의 특성**을 분석하는 알고리즘 → 비지도학습
- 데이터 사이의 **유사성**을 이용하여 군집화
  → 데이터의 유사성을 판단하는 방법에 따라 알고리즘이 세분화됨


군집 알고리즘
계층적 군집화
비계층적 군집화
응집형
분리형
밀도기반
분포기반
중심기반
Agglomerative H-Clustering
Divisive H-Clustering
DBSCAN
Gaussian Mixture Model
K-Means Clustering

Part 1. 계층적 군집화


Stage
1
2
3
4
5
Agglomerative
A
B
C
D
E
A B
D E
C D E
A B
C D E
Divisive

계층적 군집 알고리즘 H-Clustering

응집형 계층 군집
Agglomerative H-Clustering

개별 데이터를 각각 하나의 군집으로 고려하고,
가장 가까운 거리의 군집끼리 **응집해가면서**
점점 큰 군집을 만들어가는 구조

분리형 계층 군집
Divisive H-Clustering

전체 데이터를 하나의 군집으로 고려하고,
가장 먼 거리의 군집을 **분리해가면서**
점점 작은 군집들을 만들어가는 구조

Part 2. 군집간 거리 측정

계층적 군집화에서 거리란?

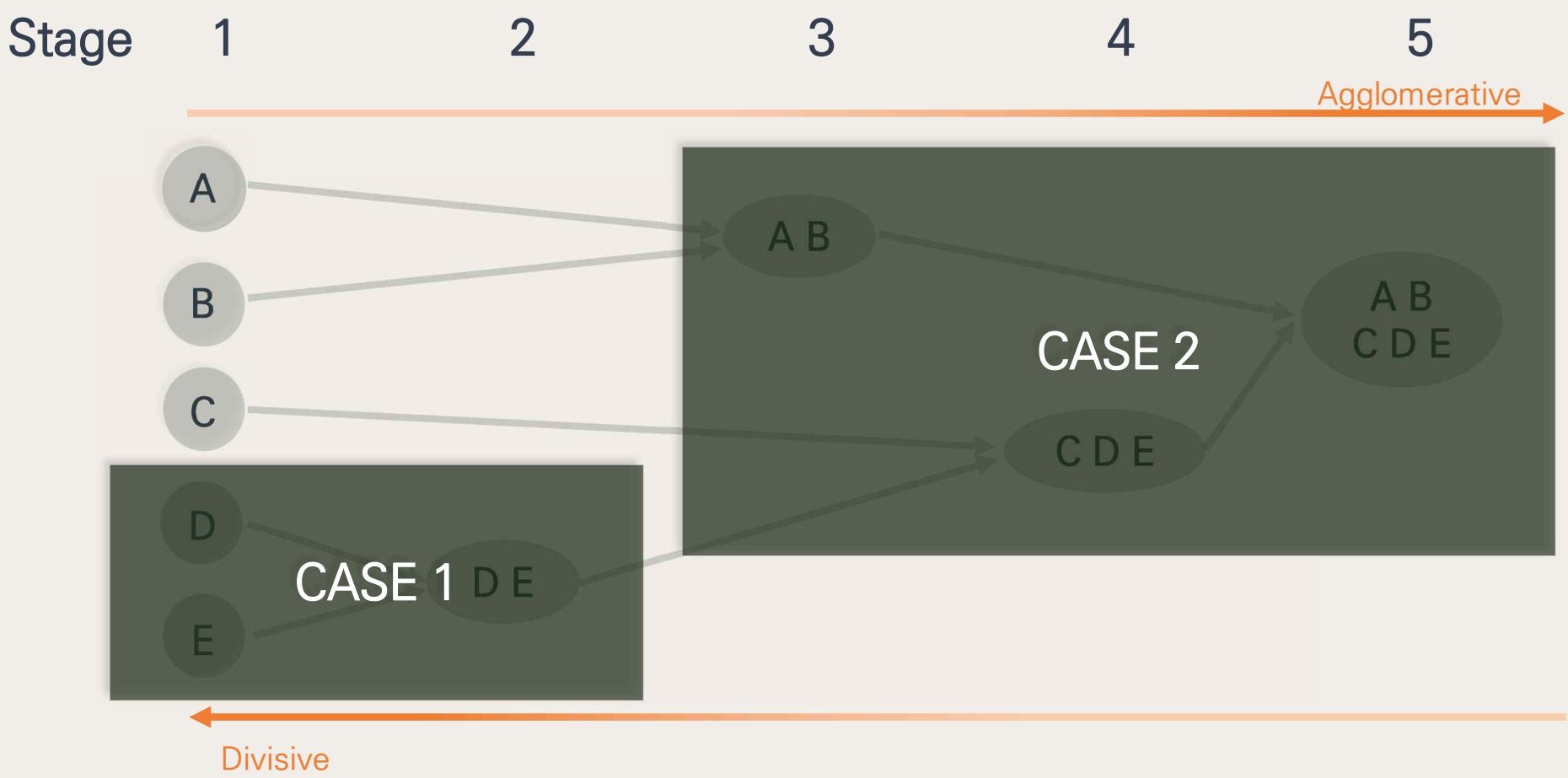
데이터의 비유사성(Dissimilarity)

- 데이터들의 거리 멀다 = 데이터들의 유사성 ↓
- 데이터들의 거리 가깝다 = 데이터들의 유사성 ↑

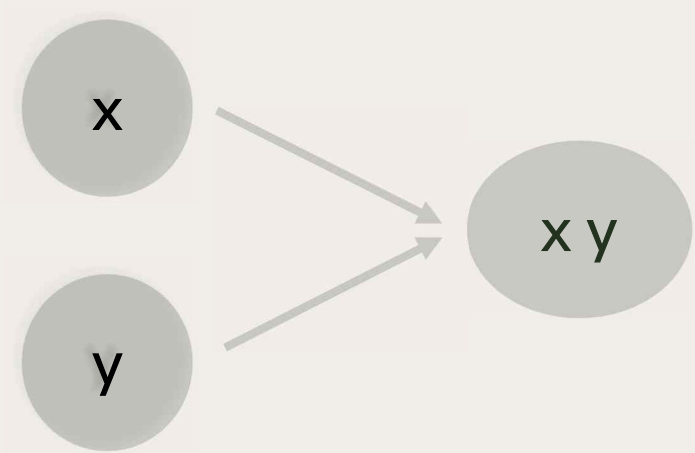
목표

- 군집 내 이질성 (Intra-Cluster Distance) 최소화
- 군집 간 이질성 (Inter-Cluster Distance) 최대화

Part 2 군집간 거리 측정

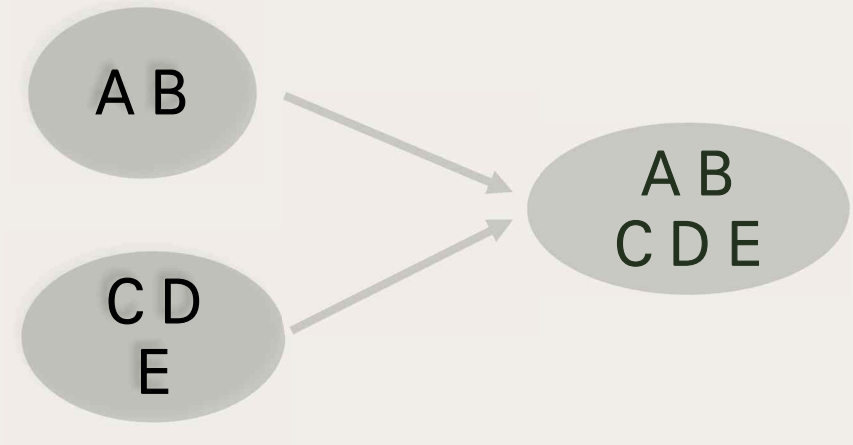


CASE 1. 군집 내 데이터가 1개 일 때



- 유클리드 거리
- 맨하튼 거리
- 표준화 거리
- 민코우스키 거리
- 마할라노비스 거리

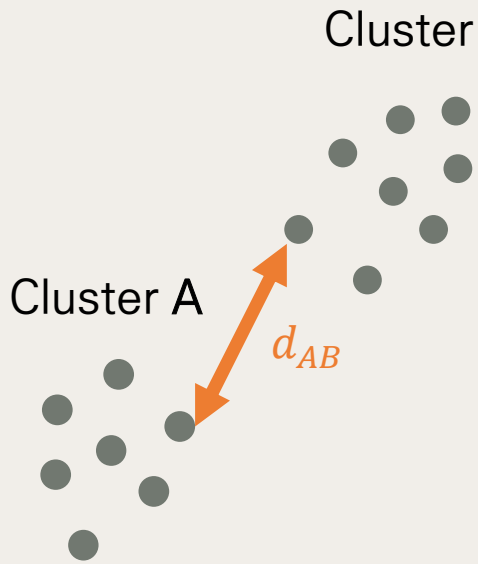
CASE 2. 군집 내 데이터가 2개 이상 일 때



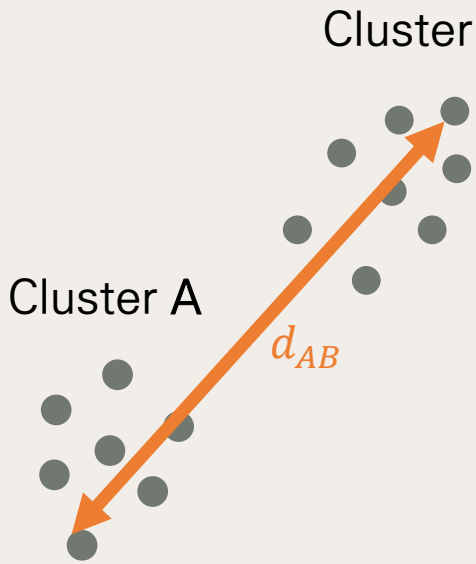
- Single Linkage
- Complete Linkage
- Group Average Linkage
- Ward Linkage
- Centroid Linkage

CASE 2. 군집 내 데이터가 2개 이상 일 때

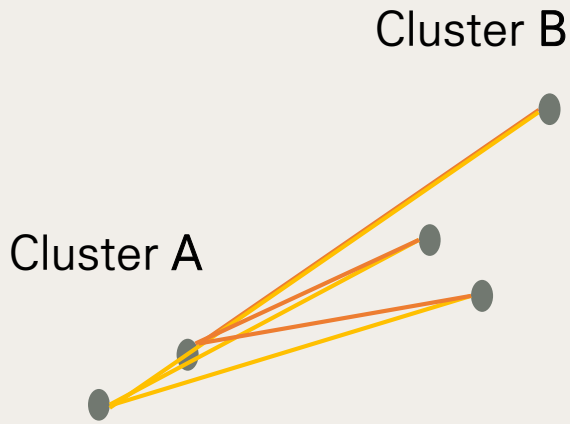
Single Linkage



Complete Linkage

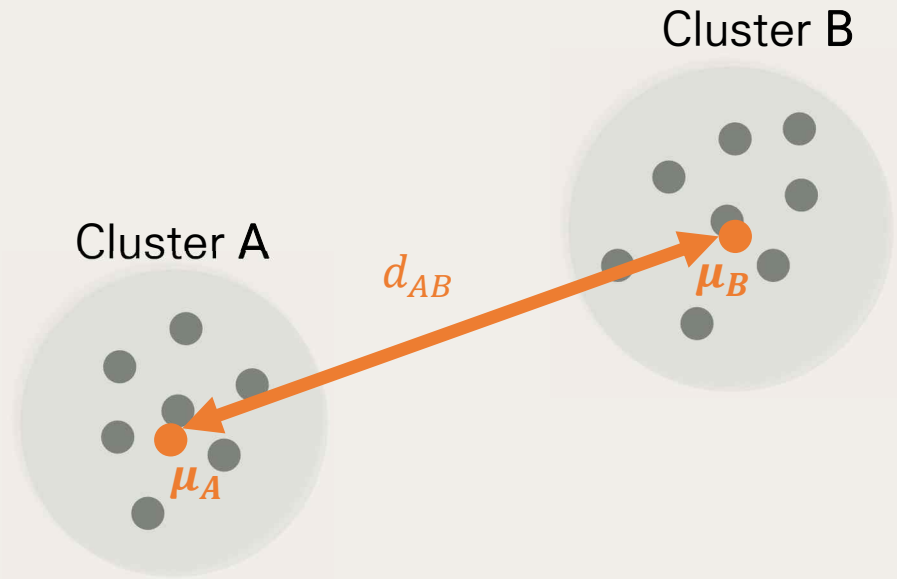


Group Average Linkage

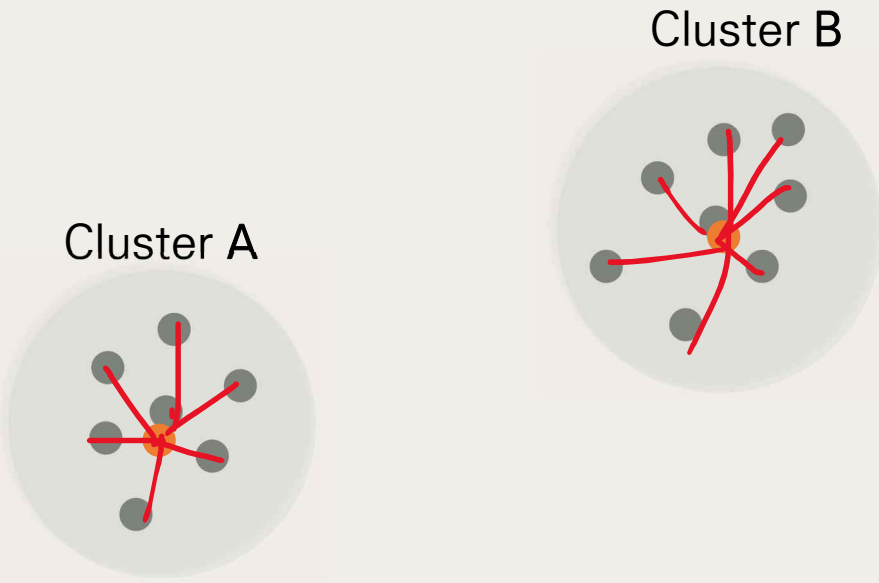


CASE 2. 군집 내 데이터가 2개 이상 일 때

Centroid Linkage

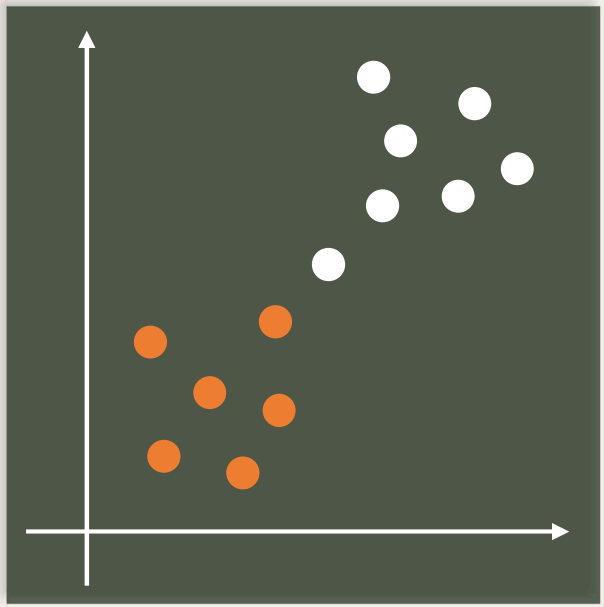


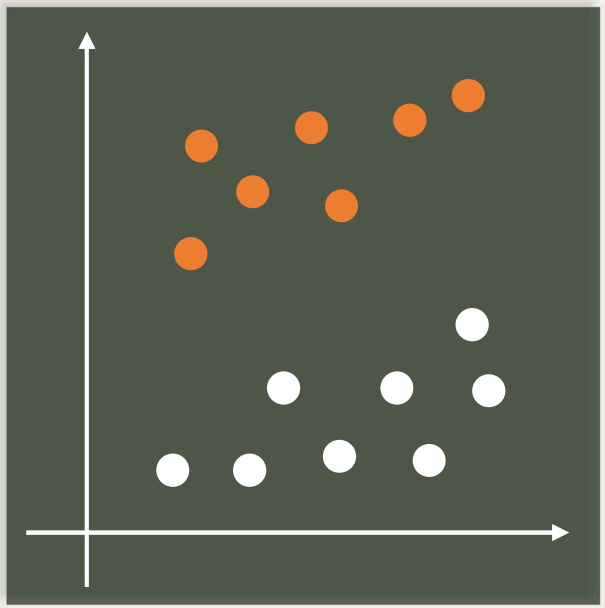
Ward Linkage



CASE 2. 군집 내 데이터가 2개 이상 일 때

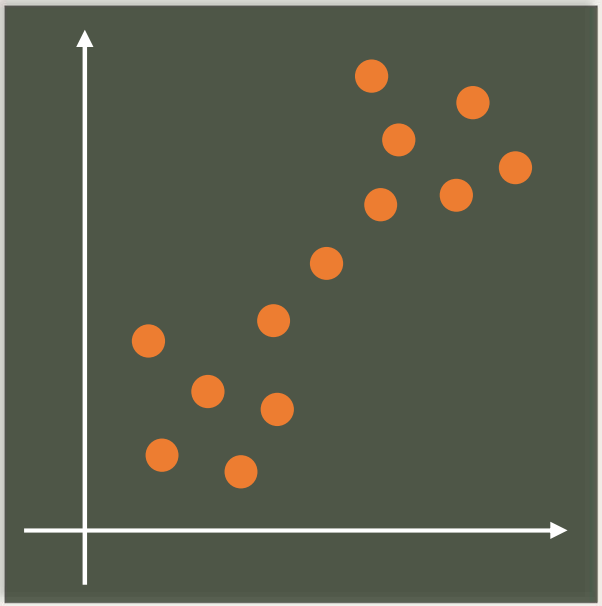
Empirical investigations indicate that no single method could be claimed superior for all types of data.



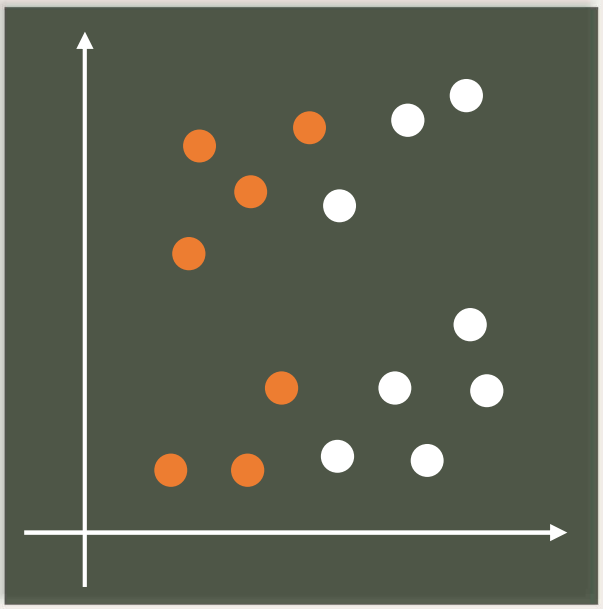


CASE 2. 군집 내 데이터가 2개 이상 일 때

Single Linkage
chaining issue

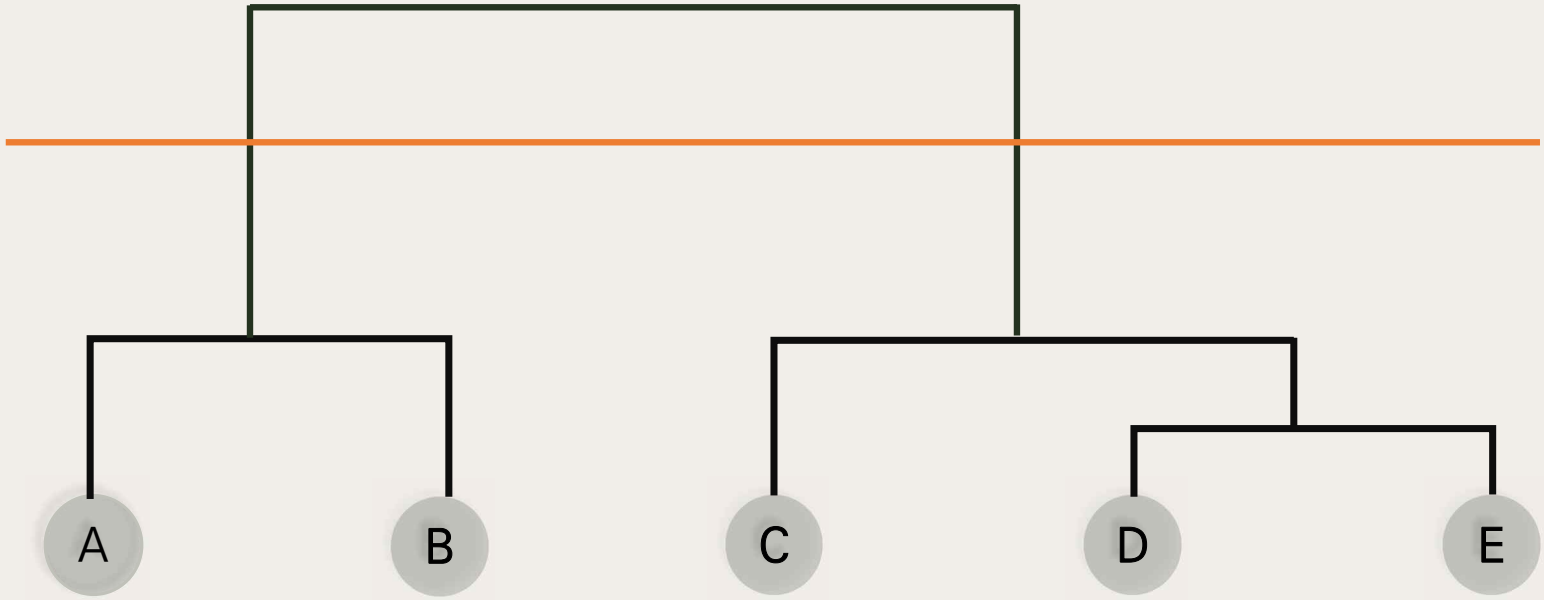


Complete
& Group Average
clusters are spherical



How many? 계층적 군집화

Large changes in dendrogram may indicate a particular number of clusters



Part 2 군집간 거리 측정

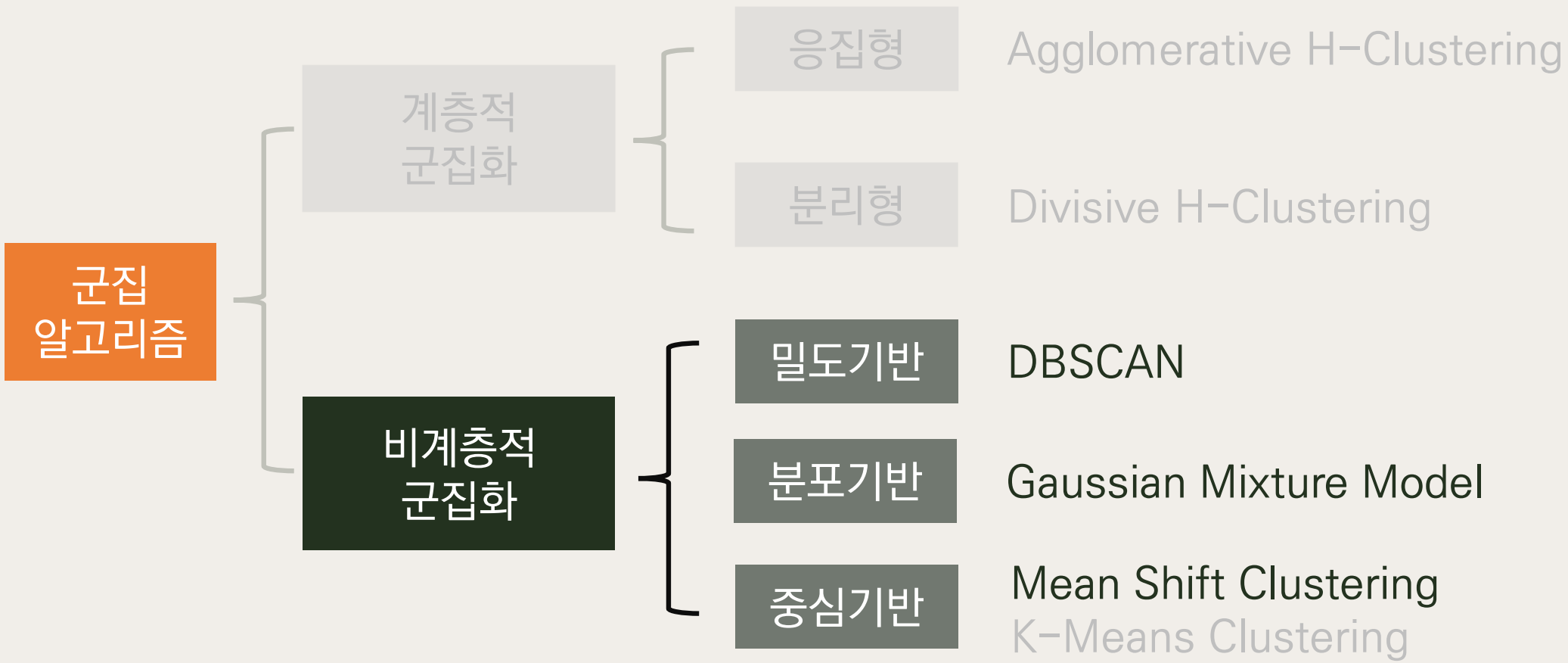
```
# dendrogram 그리기
from scipy.cluster.hierarchy import linkage, dendrogram

link=linkage(X_input,'single')
dendrogram(linked, orientaion='top')
                    # 트리 방향 (위에서부터 내려오는)

# 군집화 수행
from sklearn.cluster import AgglomerativeClustering

agg=AgglomerativeClustering(n_clusters=5,linkage='single')
                            # 군집의 개수를 미리 정해줘야함 (default=2)
                            # 군집 간 거리 계산법 (default='ward')
y_pred = agg.fit_predict(X_input)
```

Part 3. 비계층적 군집화

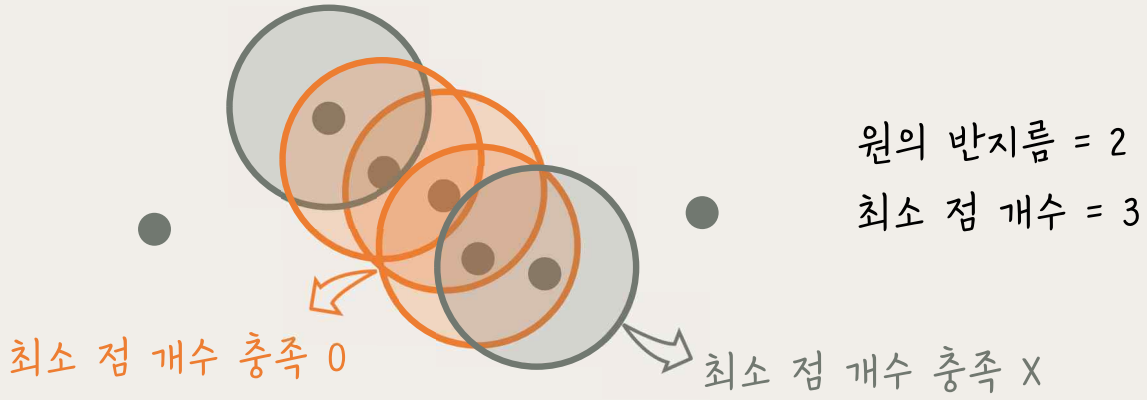

군집
알고리즘
계층적
군집화
응집형
Agglomerative H-Clustering
분리형
Divisive H-Clustering
비계층적
군집화
밀도기반
DBSCAN
분포기반
Gaussian Mixture Model
중심기반
Mean Shift Clustering
K-Means Clustering

밀도기반 DBSCAN

Density-Based Spatial Clustering of Applications with Noise

같은 군집에 속한 데이터들은 **밀도있게 모여 있을 것**이라는 가정에서 출발

→ 밀도가 높은 부분을 클러스터링하는 알고리즘

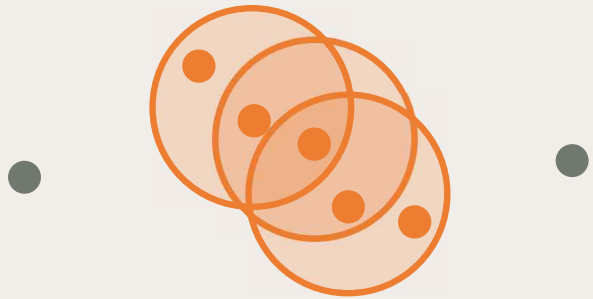


밀도기반 DBSCAN

Density-Based Spatial Clustering of Applications with Noise

같은 군집에 속한 데이터들은 **밀도있게 모여 있을 것**이라는 가정에서 출발

→ 밀도가 높은 부분을 클러스터링하는 알고리즘



최종 클러스터 형성

원의 반지름 = 2

최소 점 개수 = 3

밀도기반 DBSCAN

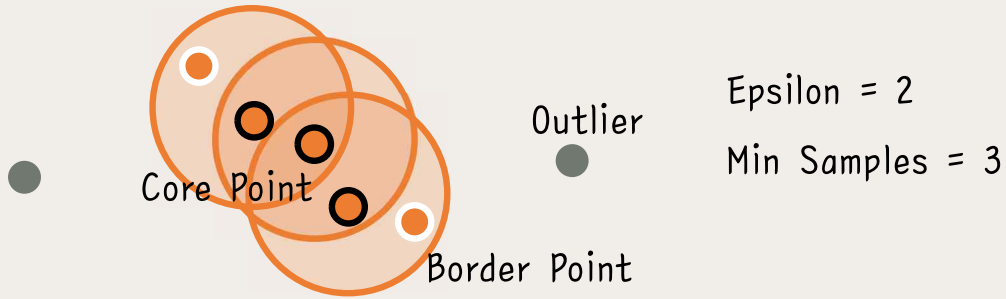
epsilon : 같은 클러스터로 묶일 수 있는 두 데이터(점)의 최대 거리, 원의 반지름

min samples : 원의 반경 내에 존재해야 하는 최소한의 데이터(점) 개수

core point (중심점) : min samples를 만족하는 원의 중심이 되는 데이터

border point (경계점) : min samples를 만족하지는 않지만 클러스터에 속하는 데이터

outlier : 클러스터링되지 못한 데이터



밀도기반 DBSCAN

```
import pandas as pd
from sklearn.cluster import DBSCAN

dbscan = DBSCAN(eps=10, min_samples=3, metric='euclidean')
    # 모델 선언 및 파라미터 설정

prediction = dbscan.fit_pred(X_input)
    # fit과 predict 동시에 가능
    # 군집 번호 저장
```

분포기반 GMM (Gaussian Mixture Model)

데이터가 여러개의 가우시안 분포(Gaussian Distribution) 로 결합되어 있다는 가정에서 출발

→ EM 알고리즘을 활용하여 데이터를 같은 가우시안 분포끼리 클러스터링하는 알고리즘

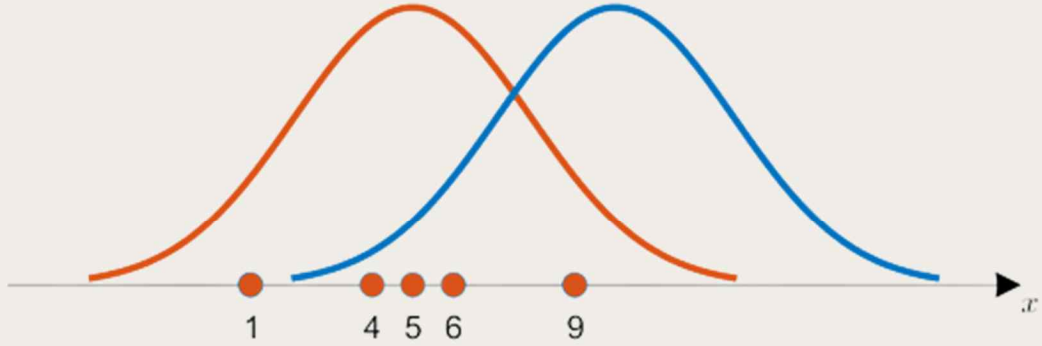
최대우도법 (Maximum Likelihood Estimation)	EM 알고리즘	베이즈 정리

분포기반 GMM (Gaussian Mixture Model)

Recap. 최대우도법(MLE)

관측된 데이터로 확률밀도함수의 모수를 추정하는 방법

- 관측된 데이터 : $\boldsymbol{x} = (x_1, x_2, \dots x_n)$
- 확률밀도함수 : $p(x\,|\theta)$
- 모수 : θ



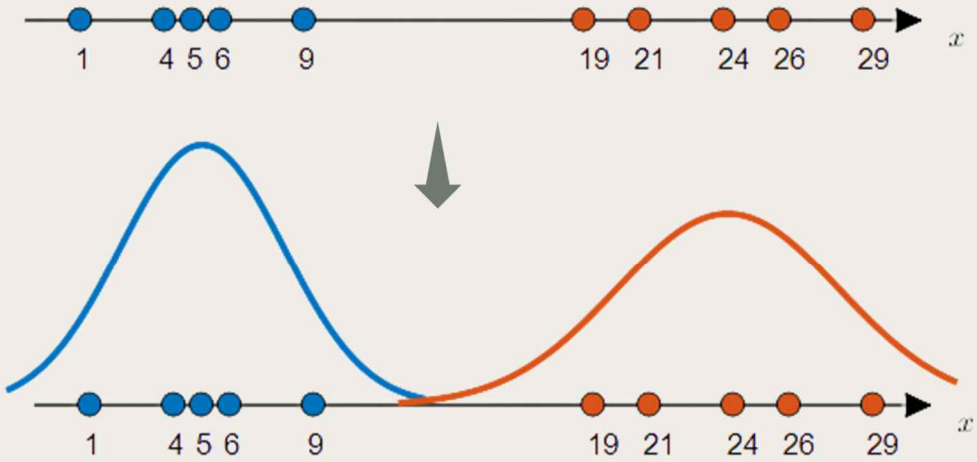
$p(x|\theta) = \prod_{i=1}^{n} p(x_i|\theta)$ → 우도함수

$\Leftrightarrow \log p(x_i|\theta) = \sum_{i=1}^{n} \log p(x_i|\theta)$ → 로그우도함수

로그우도함수를 최대로 만드는 θ(모수) = Maximum Likelihood Estimator

분포기반 GMM (Gaussian Mixture Model)

Recap. 최대우도법(MLE)



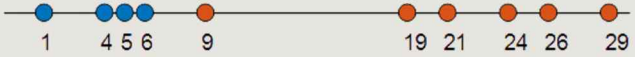
Label 이 주어진 데이터는
MLE를 통해 모수를 추정할 수 있다!

하지만, Label 이 없는 경우에는 ?

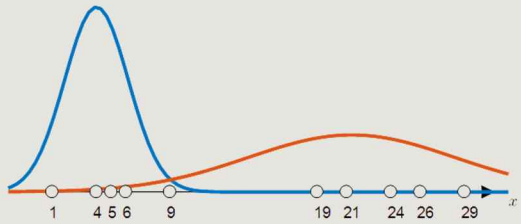
분포기반 GMM (Gaussian Mixture Model)

① 일단, 랜덤하게 분포 생성

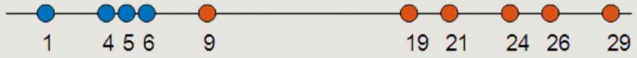
② 분포에 따라 Label 부여



③ MLE로 분포 업데이트



④ Label 업데이트



분포기반 GMM (Gaussian Mixture Model)

EM 알고리즘

① E-step (Expectaion) : 로그우도의 기댓값을 계산하는 과정

② M-step (Maximization) : ML estimation을 통해 모수를 추정하는 과정

(E-step) For each i , j

$$w_j^i := P\left(z^i = j \mid x^i ; \phi , \mu, \Sigma\right)$$

(M-step) Update the parameters:

$$\phi_j := \frac{1}{m}\sum_{i=1}^{m} w_j^i$$

$$\mu_j := \frac{\sum_{i=1}^{m} w_j^i x^i}{\sum_{i=1}^{m} w_j^i}$$

$$\Sigma_j := \frac{\sum_{i=1}^{m} w_j^i (x^i - \mu_j)(x^i - \mu_j)^T}{\sum_{i=1}^{m} w_j^i}$$

- i 데이터 순번
- j Label
- x^i i 번째 데이터
- z^i i 번째 데이터의 Label
- w_j^i i 번째 데이터가 j 그룹일 확률

분포기반 GMM (Gaussian Mixture Model)

EM 알고리즘

① E-step (Expectaion) : 로그우도의 기댓값을 계산하는 과정 → 변수의 label을 찾는 과정

베이즈 정리

새로운 정보를 토대로 어떤 사건이 발생했다는 주장에 대한 신뢰도를 갱신해 나가는 방법

사전확률 (prior) : 어떤 사건이 발생했다는 가설의 신뢰도

$$P(H|E) = \frac{P(E|H)P(H)}{P(E)}$$

사후확률 (posterior)

새로운 정보를 받은 후 갱신된 신뢰도

분포기반 GMM (Gaussian Mixture Model)

EM 알고리즘

① E-step (Expectaion) : 로그우도의 기댓값을 계산하는 과정 → 변수의 label을 찾는 과정

(E-step) For each i , j

$$w_j^i := P\left(z^i = j \mid x^i ; \phi , \mu, \Sigma\right) = \frac{P\left(x^i \middle| z^i=j ;\mu,\Sigma\right) P\left(z^i=j ; \phi\right)}{P\left(x^i;\phi,\mu,\Sigma\right)} = \frac{P\left(x^i \middle| z^i=j ;\mu,\Sigma\right) P\left(z^i=j ; \phi\right)}{\sum P\left(x^i \middle| z^i=k; \mu,\Sigma\right) P\left(z^i=k;\phi\right)}$$

1) xi 라는 데이터가 주어졌고

2) phi, mu, sigma 라는 모수를 통해 각 label에 대한 가우시안 확률분포를 가정했을 때

3) 확률밀도함수 값을 w_ij 라고 하겠다.

4) 그 확률은 베이즈 정리를 통해 계산할 수 있다.

분포기반 GMM (Gaussian Mixture Model)

EM 알고리즘

② M-step (Maximization) : ML estimation을 통해 모수를 추정하는 과정 → 어떤 분포인지 추정하는 과정

(M-step) Update the parameters:

$$\phi_j := \frac{1}{m}\sum_{i=1}^{m} w_j^i$$

$$\mu_j := \frac{\sum_{i=1}^{m} w_j^i x^i}{\sum_{i=1}^{m} w_j^i}$$

$$\Sigma_j := \frac{\sum_{i=1}^{m} w_j^i (x^i-\mu_j)(x^i-\mu_j)^T}{\sum_{i=1}^{m} w_j^i}$$

E-step에서 계산한 w_j^i 값을 이용해 모수를 추정!

MLE를 통해 쉽게 계산할 수 있다

분포기반 GMM (Gaussian Mixture Model)

```
from sklearn.mixture import GaussianMixture
gmm = GaussianMixture( n_components=3, random_state=42)
# 클러스터링
y_pred = gmm.fit_predict(X_input)

# AIC BIC
print(f'AIC : {gmm.aic(X_input)}')
gmm.bic(f'BIC : {gmm.bic(X_input)}')
```

중심기반 Mean Shift Clustering

데이터가 특정 분포를 따르지 않는다는 가정에서 출발 → 비모수적

1) 개별 데이터의 **특정반경** 내에서 주변의 **데이터 분포도 계산**
   → band width　　by. KDE 커널 함수

2) 가장 밀도가 높은 방향으로 중심 이동

3) 업데이트된 중심에서 1,2번 반복 수행
   (더 이상 업데이트 안될 때까지)

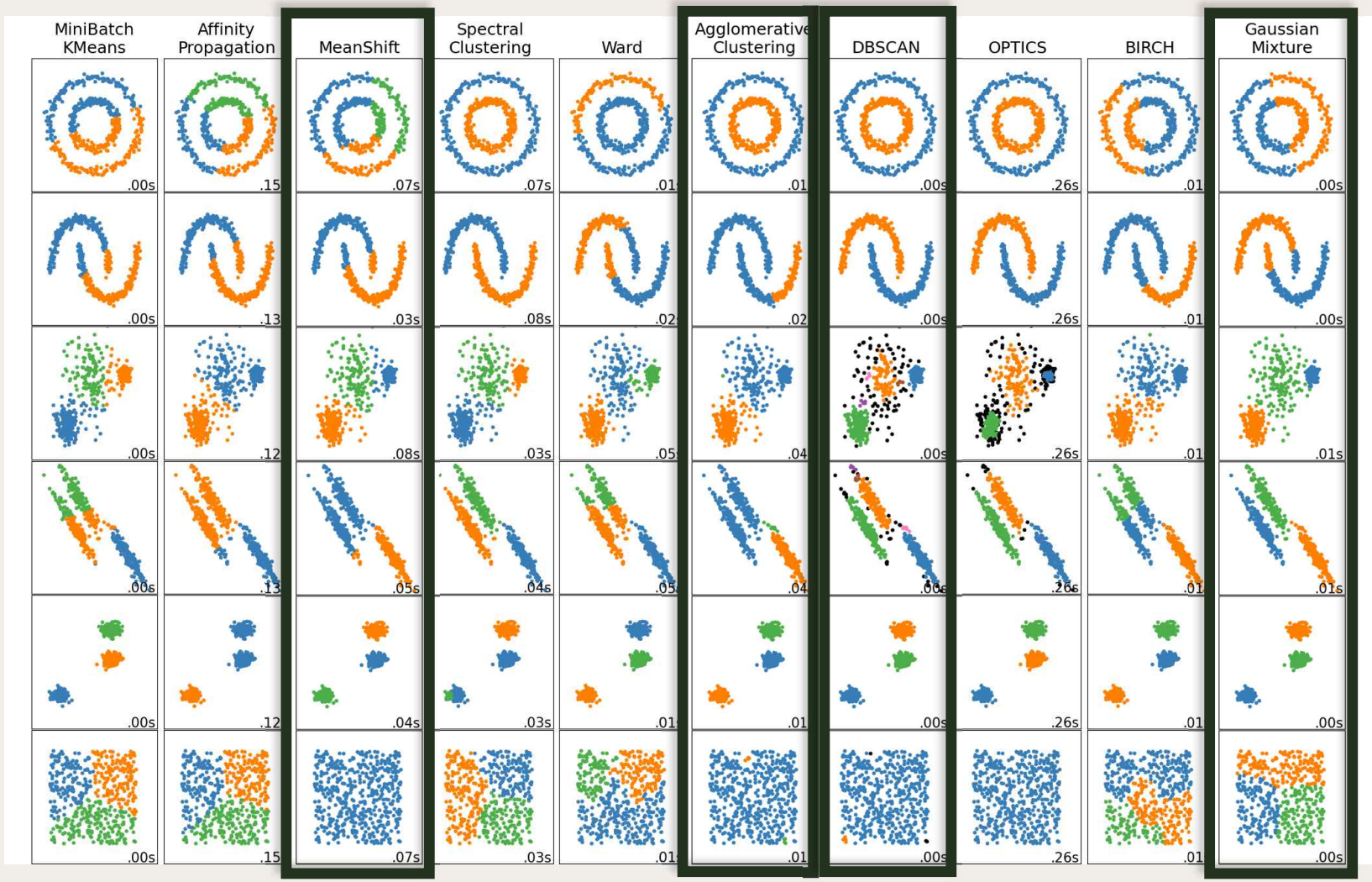
중심기반 Mean Shift Clustering

```
from sklearn.cluster import MeanShift, estimate_bandwidth
# sklearn은 최적의 bandwidth를 계산해줌
bw=estimate_bandwidth(X_input)

ms = MeanShift(bandwidth=bw)
# 클러스터링
y_pred=ms.fit_predict(X_input)
```

Part 4. 군집 알고리즘 비교

Part 4 군집 알고리즘 비교



Part 5. 참고문헌

Part5 참고문헌

Hierarchical Clustering

- *Applied Multivariate Data Analysis 2nd Edition*
- *송주원(고려대학교) – 다변량통계분석*
- http://www.datamarket.kr/xe/board_mXVL91/9807

Clustering Algorithm

- *https://developers.google.com/machine-learning/clustering/clustering-algorithms?hl=en*

DBSCAN

- *유용재(고려대학교)-컴퓨터프로그래밍2*

GMM and EM Algorithm

- *https://angeloyeo.github.io/2021/02/08/GMM_and_EM*

Mean Shift Clustering

- https://scikit-learn.org/stable/modules/generated/sklearn.cluster.MeanShift

감사합니다