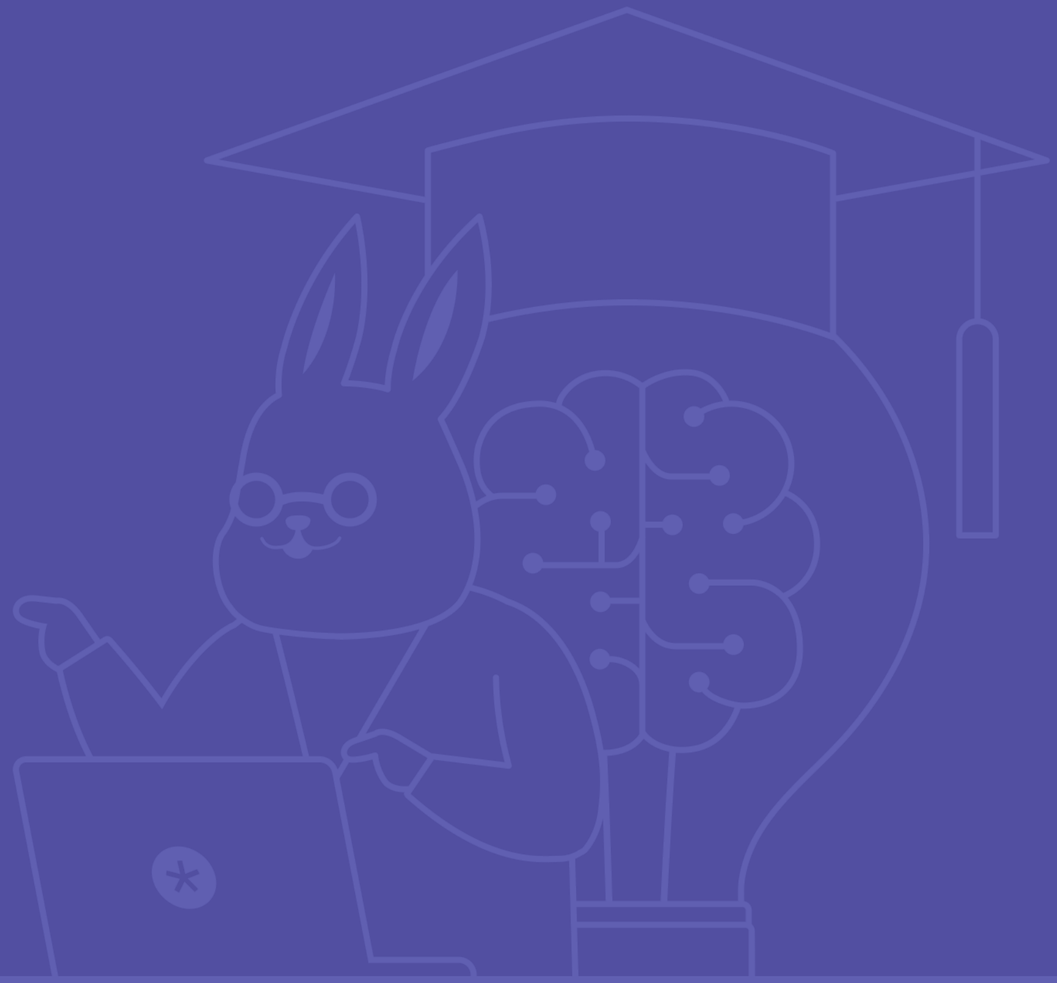
2020 AI College

7장 비지도 학습 - 클러스터링

정민수 강사



/* elice */

Contents

Target

클러스터링을 이해한다.

클러스터링이 무엇이고 어떤 것을 목표로 하는지 이해한다.

여러가지 클러스터링 알고리즘에 대해 살펴본다

K-means, KNN, GMM, HC 등 주로 사용되는 클러스터링 알고리즘에 대하여 살펴본다.

01

클러스터링(Clustering)

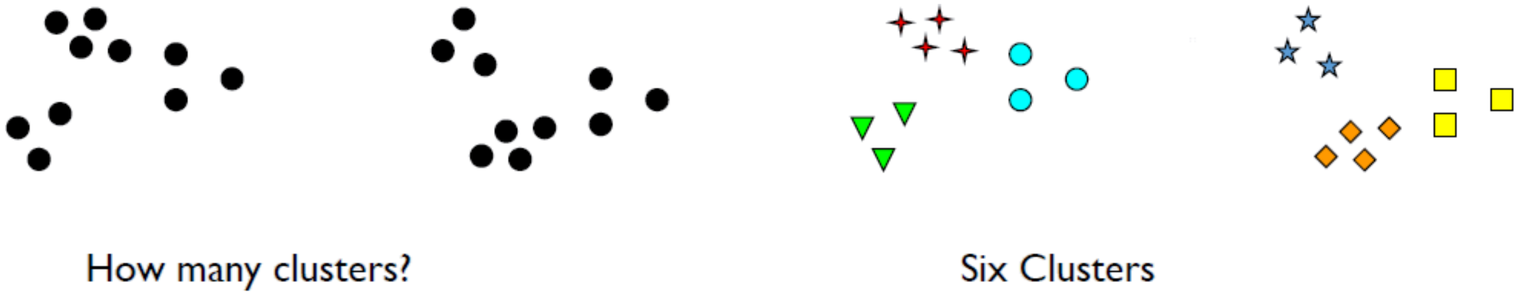
01 클러스터링

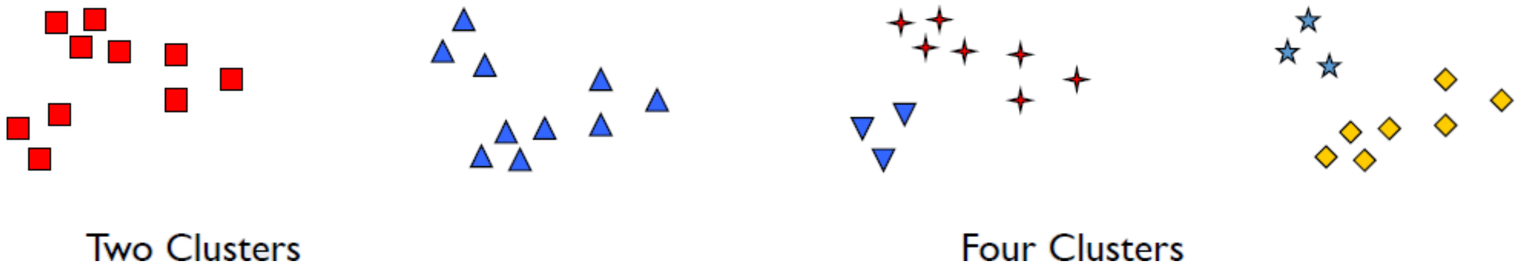
클러스터링이란?

- 클러스터링
  - 각 데이터에 대한 class 정보가 없는 상태에서 데이터의 분포 혹은 특성을 기반으로 class를 유추하는 기법
  - 데이터 간 거리, 주변 밀도 등의 여러 기준에 따라 데이터들을 그룹화하여 각 데이터가 속한 그룹을 유추한다.
- VS 분류
  - 분류 문제는 각 데이터의 정답 class가 주어지는 지도 학습(Supervised Learning)을 통해 모델을 학습
  - 클러스터링은 각 데이터의 정답 class가 없는 비지도 학습(Unsupervised Learning)을 통해 모델을 학습

클러스터링의 타당성 평가

- 비지도학습인 클러스터링 기법은 정답이 없기 때문에 일반적인 머신러닝 알고리즘처럼 단순 정확도(Accuracy) 등의 지표로 평가할 수 없음
- 그림과 같이 최적의 군집 개수를 정답 없이 알아내기 쉽지 않음

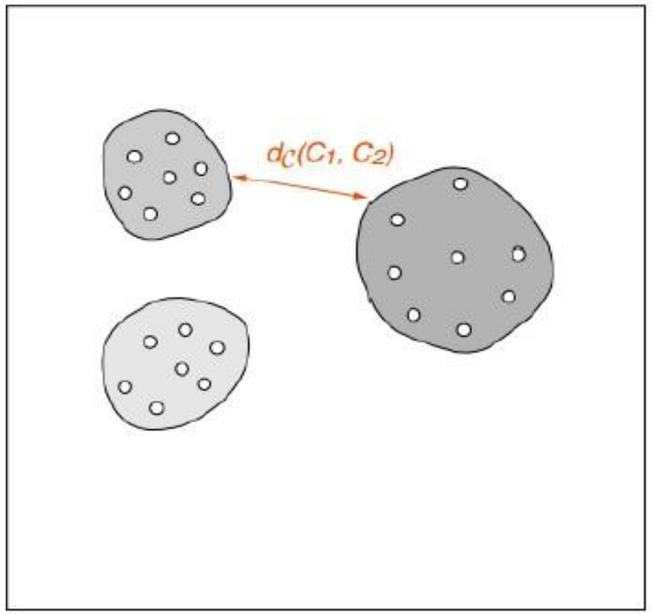




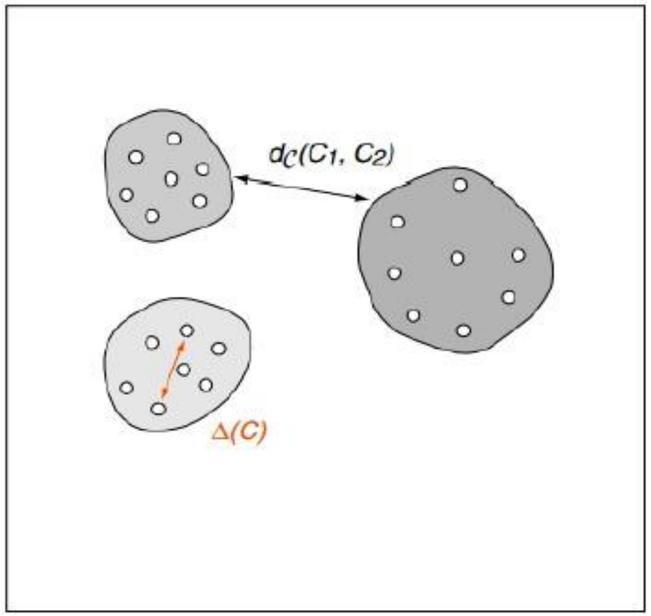
타당성 지표

- 군집 간 거리
- 군집의 지름
- 군집의 분산

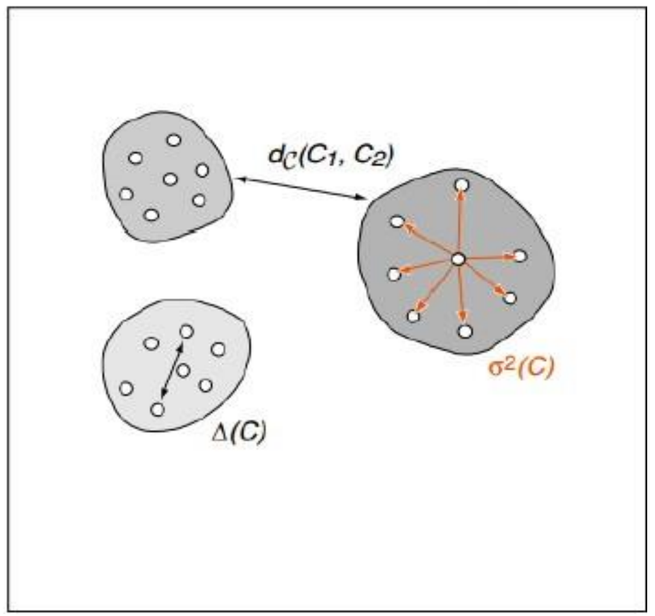
Distance between two clusters



Diameter of a cluster



Scatter within a cluster (SSE)

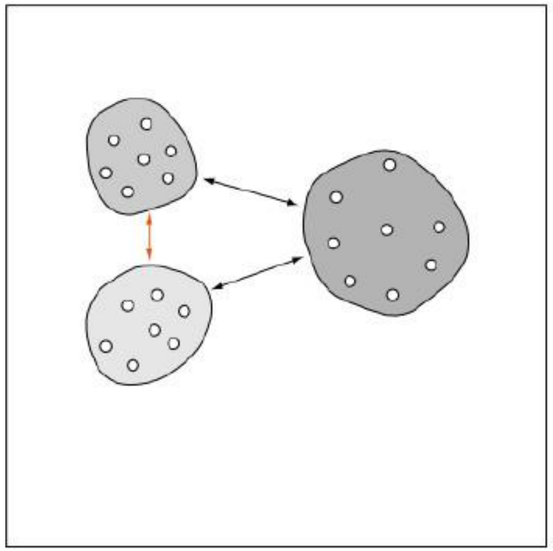


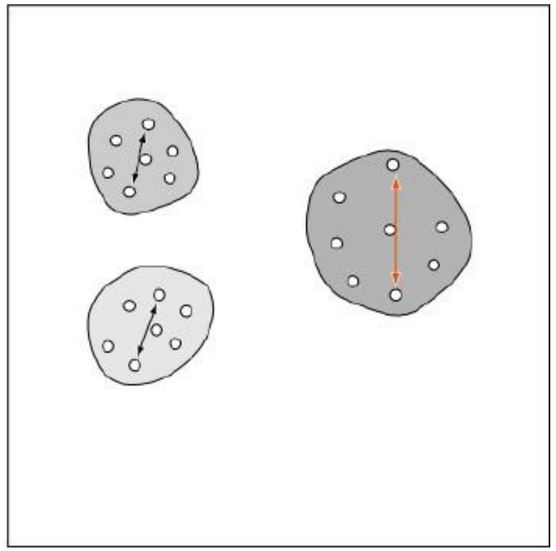
/* elice */

타당성 지표

- Dunn Index
  - 군집 간 거리의 최소값(하단 좌측)을 분자, 군집 내 요소 간 거리의 최대값(하단 우측)을 분모로 하는 지표

$$I(C) = \frac{\min_{i \neq j} \{d_c(C_i, C_j)\}}{\max_{1 \leq l \leq k} \{\triangle(C_l)\}}$$





타당성 지표

- Dunn Index
  - 군집 내 요소 간 거리 Δ_i

1. 최대 거리를 계산 #

$$\Delta_i = \max_{x,y \in C_i} d(x,y)$$

2. 모든 쌍 사이의 평균 거리 계산 #

$$\Delta_i = \frac{1}{|C_i|(|C_i|-1)} \sum_{x,y \in C_i, x \neq y} d(x,y)$$

3. 평균으로부터 모든 점의 거리 계산 #

$$\Delta_i = \frac{\sum_{x \in C_i} d(x,\mu)}{|C_i|}, \mu = \frac{\sum_{x \in C_i} x}{|C_i|}$$

02

K-means

K-means란?

- K-means는 EM 알고리즘을 기반으로 함

- EM알고리즘은 Expectation 스텝과 Maximization 스텝으로 나뉨
- 이를 수렴할 때까지 반복
- 동시에 해를 찾기 어려운 문제를 풀 때 많이 사용되는 방법론

- E step: 각 군집 중심의 위치를 구함
- M step: 각 점이 어떤 클러스터에 속해야 하는지 멤버십 갱신

K-means 알고리즘

- 입력
  - 클러스터 수 : K
  - 트레이닝 셋 : $x^{(1)}, x^{(2)}, \cdots, x^{(m)} \quad \in \mathbb{R}^n$

- 알고리즘

K개의 무게중심 $\mu_1, \mu_2, \cdots, \mu_K \quad \in \mathbb{R}^n$을 랜덤하게 초기화

Repeat

i는 1부터 m까지

$c^{(i)} := x^{(i)}$에서 가장 가까운 클러스터 인덱스

k는 1부터 K까지

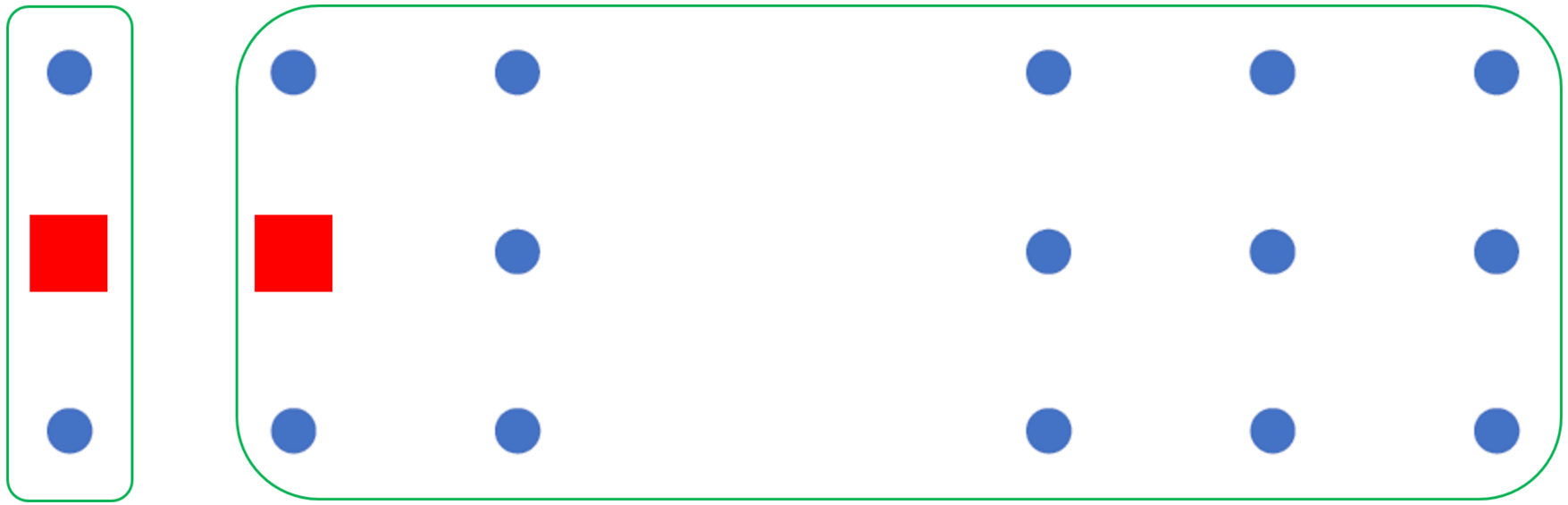
$\mu_K :=$ 클러스터 k에 속해있는 점들의 평균

K-means 학습 과정

- 클러스터 수는 2로 가정
- 군집의 무게중심(빨간 네모)을 랜덤하게 초기화하여 지정

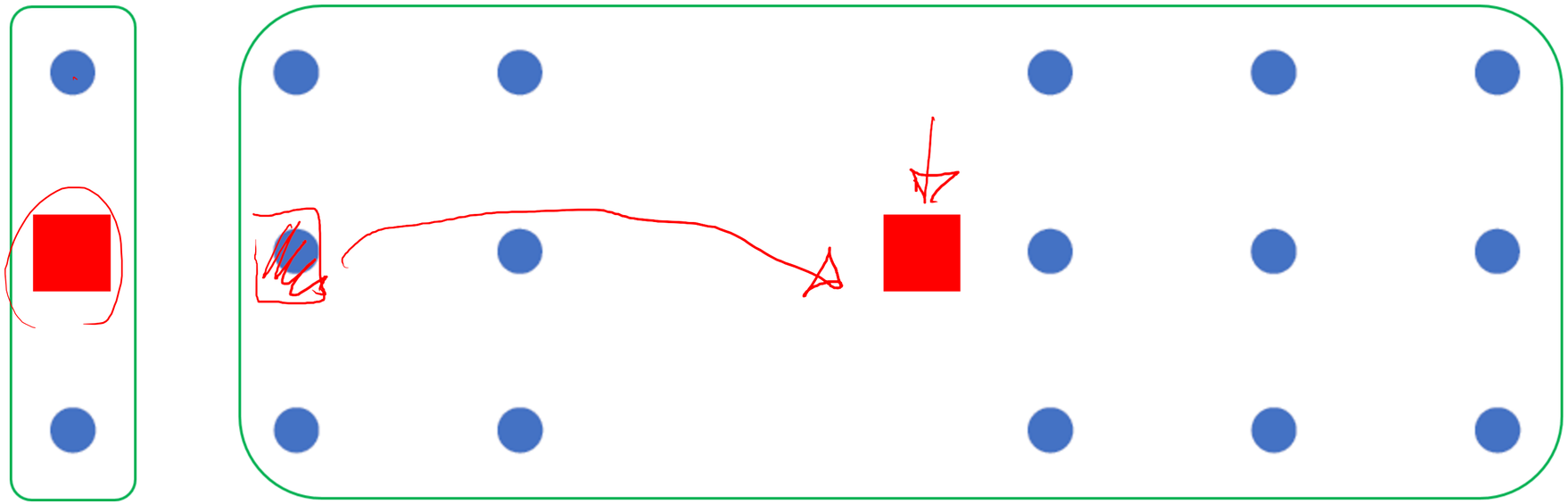
K-means 학습 과정

- E Step
  - 모든 점들(파란색 점)을 가장 가까운 무게중심을 기준으로 클러스터링



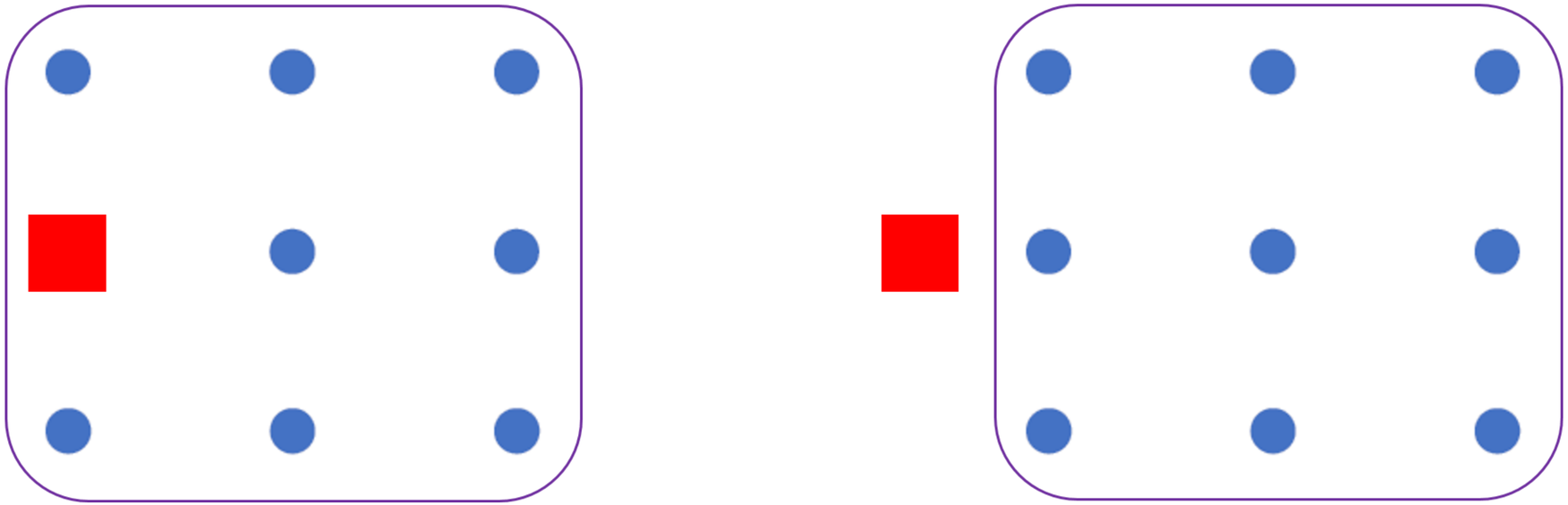
K-means 학습 과정

- M Step
  - 클러스터의 무게중심 업데이트



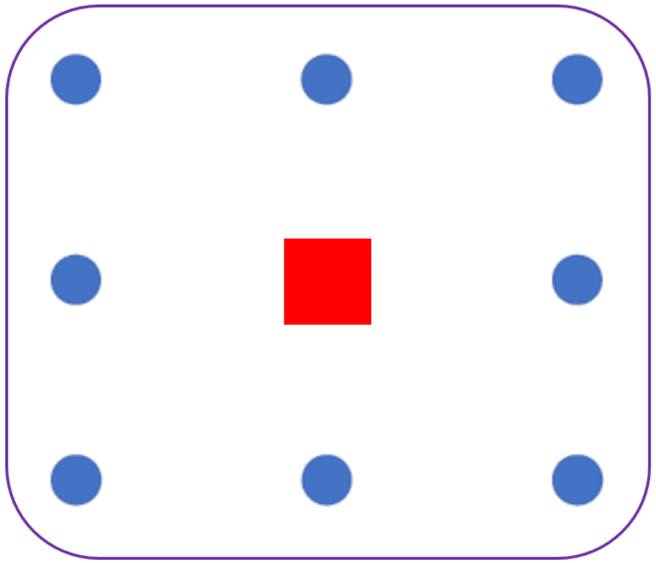
K-means 학습 과정

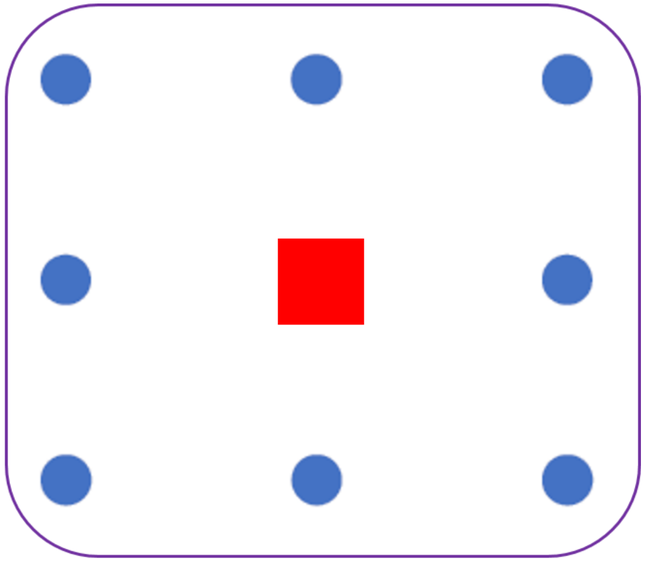
- E Step (2회차)
  - 모든 점들(파란색 점)을 가장 가까운 무게중심을 기준으로 클러스터링



K-means 학습 과정

- M Step (2회차)
  - 클러스터의 무게중심 업데이트
  - 수렴하거나 지정한 Max Iteration 수가 끝나면 학습 종료





K-means 알고리즘

Given an initial set of k means $m_1^{(1)},\ldots,m_k^{(1)}$ (see below), the algorithm proceeds by alternating between two steps:[7]

Assignment step: Assign each observation to the cluster with the nearest mean: that with the least squared Euclidean distance.[8] (Mathematically, this means partitioning the observations according to the Voronoi diagram generated by the means.)

$$S_i^{(t)} = \left\{x_p : \left\|x_p - m_i^{(t)}\right\|^2 \leq \left\|x_p - m_j^{(t)}\right\|^2 \ \forall j, 1 \leq j \leq k\right\},$$

where each x_p is assigned to exactly one $S^{(t)}$, even if it could be assigned to two or more of them.

Update step: Recalculate means (centroids) for observations assigned to each cluster.

$$m_i^{(t+1)} = \frac{1}{\left|S_i^{(t)}\right|} \sum_{x_j \in S_i^{(t)}} x_j$$

K-means의 Loss Function

- **K-means 알고리즘의 Loss Function**

$$J(c^{(1)},\ldots,c^{(m)},\mu_1,\ldots,\mu_K) = \frac{1}{m}\sum_{i=1}^{m}||x^{(i)} - \mu_{c^{(i)}}||^2$$

$x^{(i)}$: i번째 데이터

$c^{(i)}$: $x^{(i)}$가 있는 그룹

μ_i : i번째 그룹의 무게중심

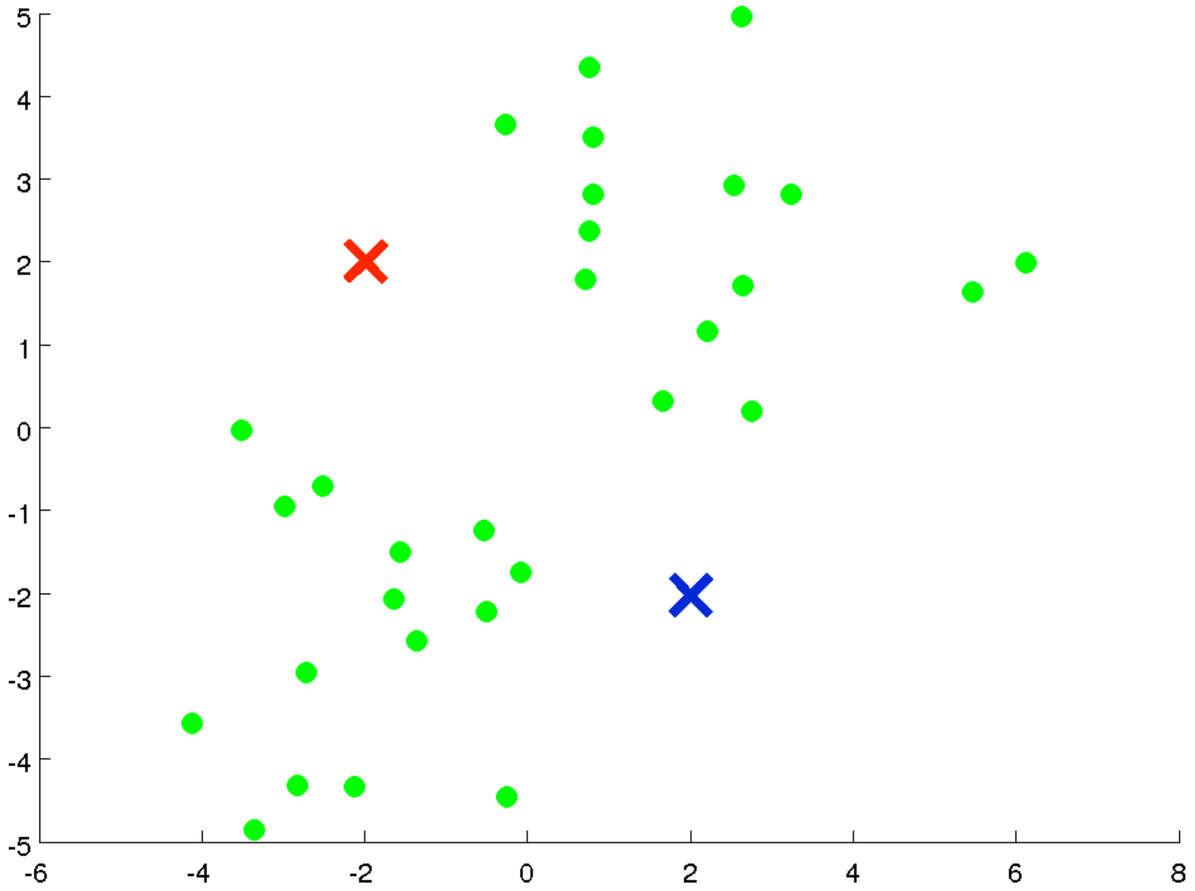
K-means의 Loss Function

- **K-means 알고리즘의 Loss Function**

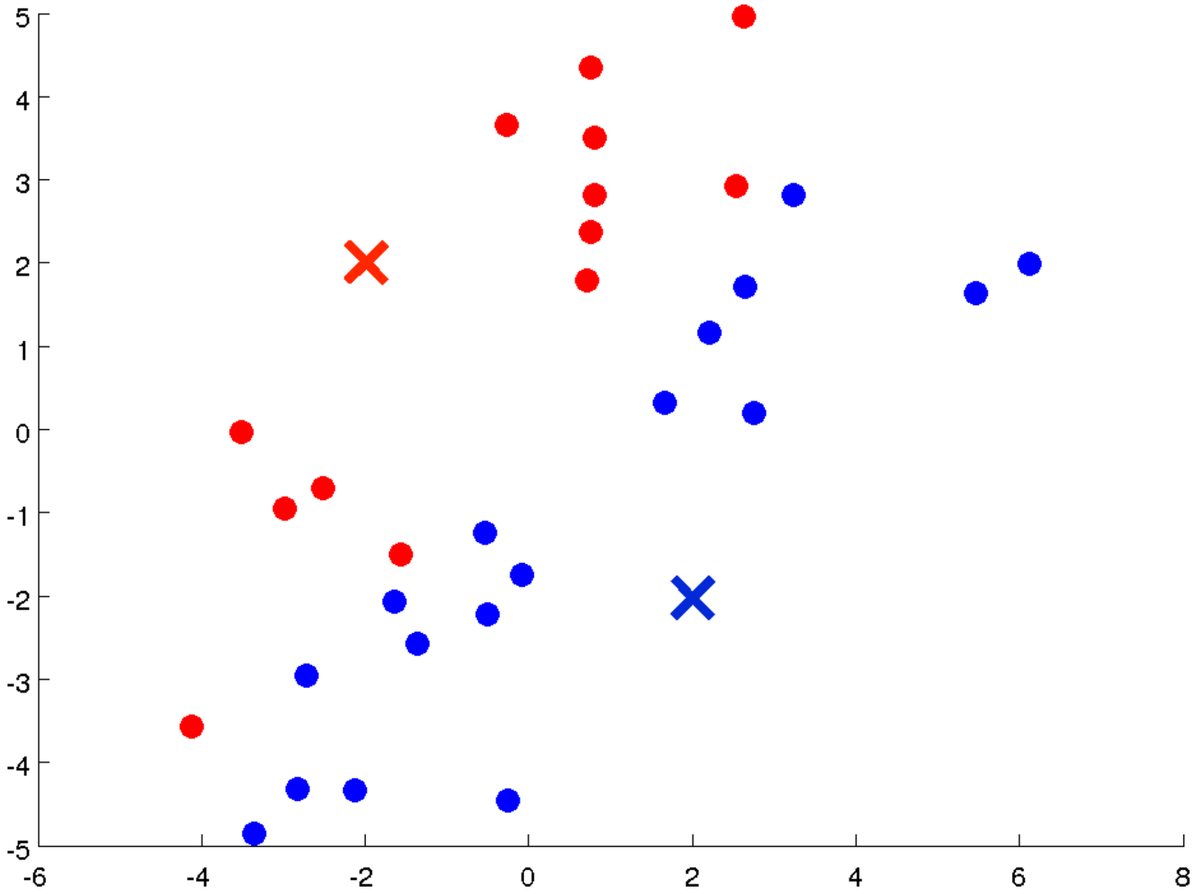
$$J(c^{(1)}, \ldots, c^{(m)}, \mu_1, \ldots, \mu_K) = \frac{1}{m} \sum_{i=1}^{m} ||x^{(i)} - \mu_{c^{(i)}}||^2$$

- **E step**
  $c^{(1)}$, $c^{(2)}$, …, $c^{(i)}$에 대해서 최소화
- **M step**
  $\mu_1, \mu_2, \cdots, \mu_K$에 대해서 최소화

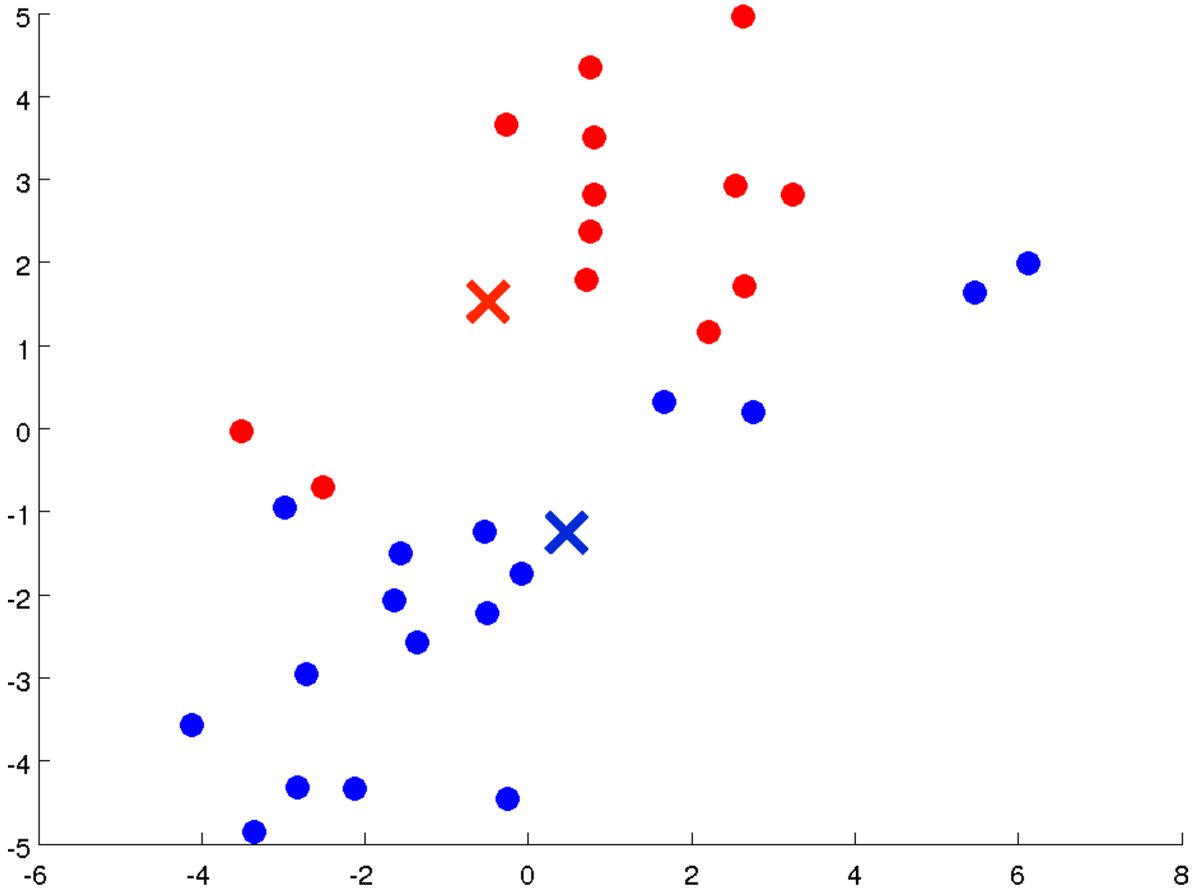
K-means 예시



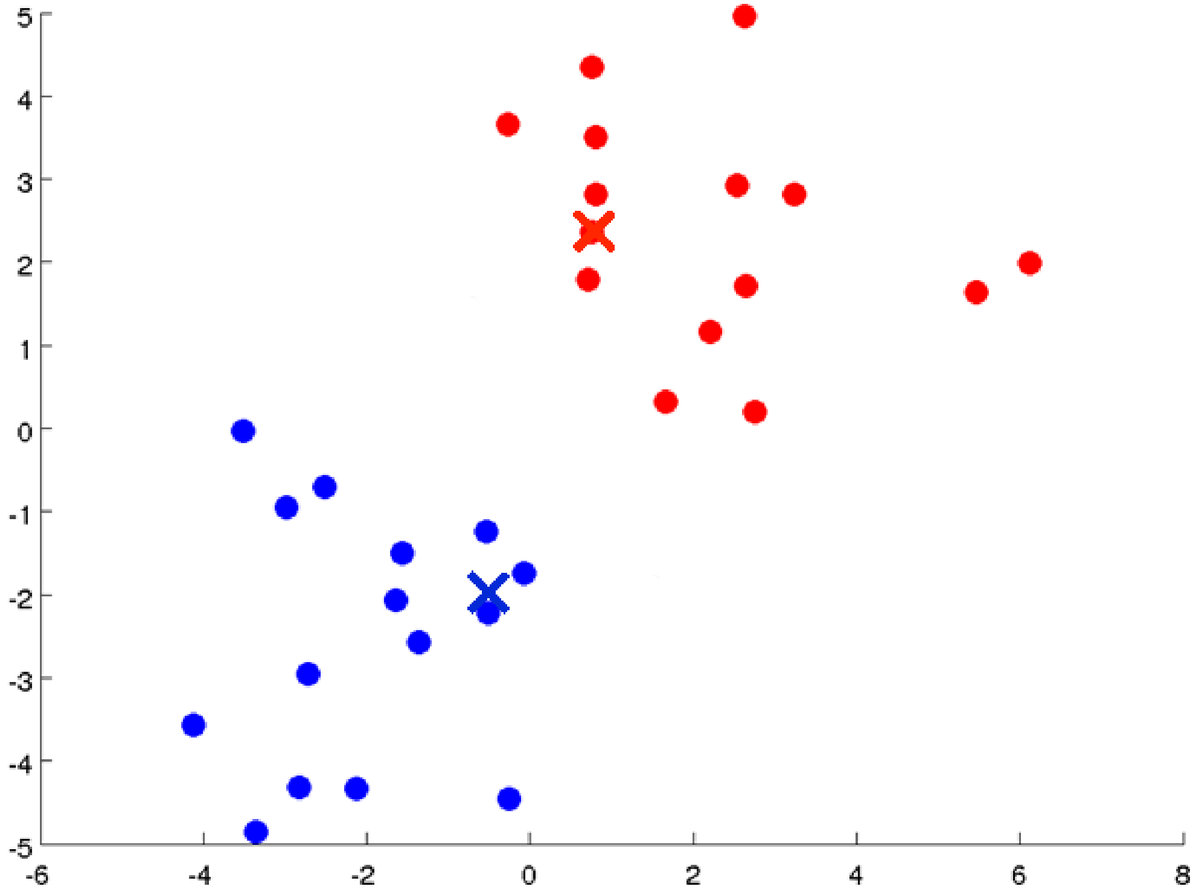
K-means 예시



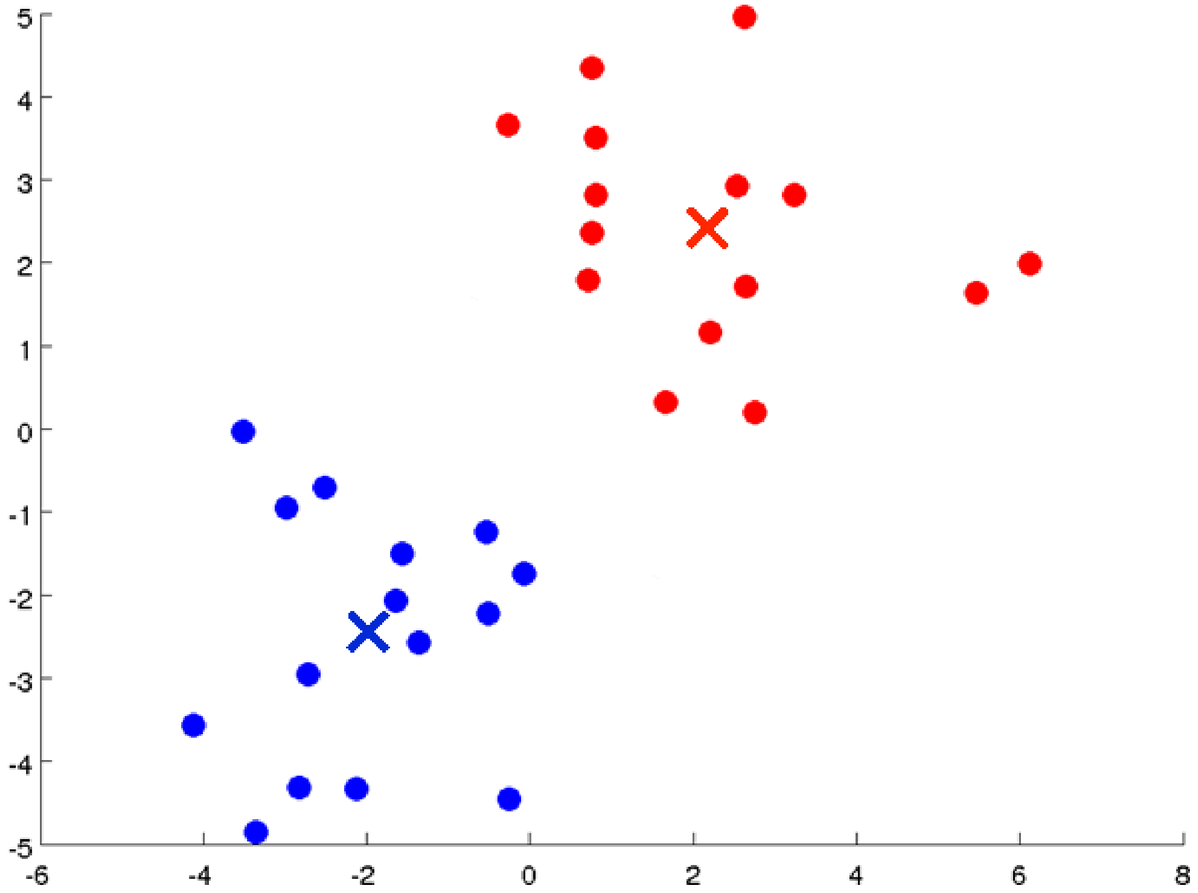
K-means 예시



K-means 예시

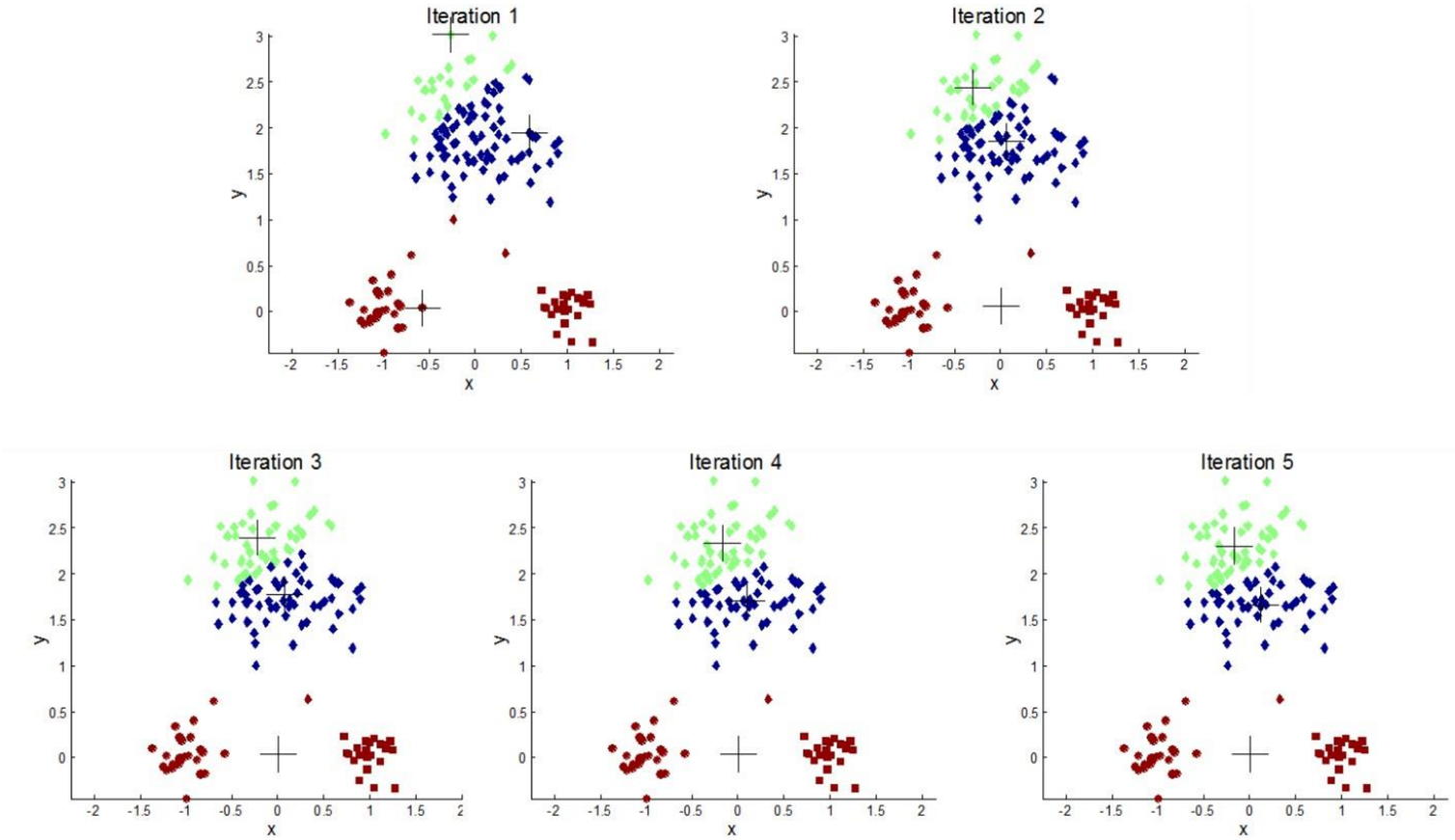


K-means 예시



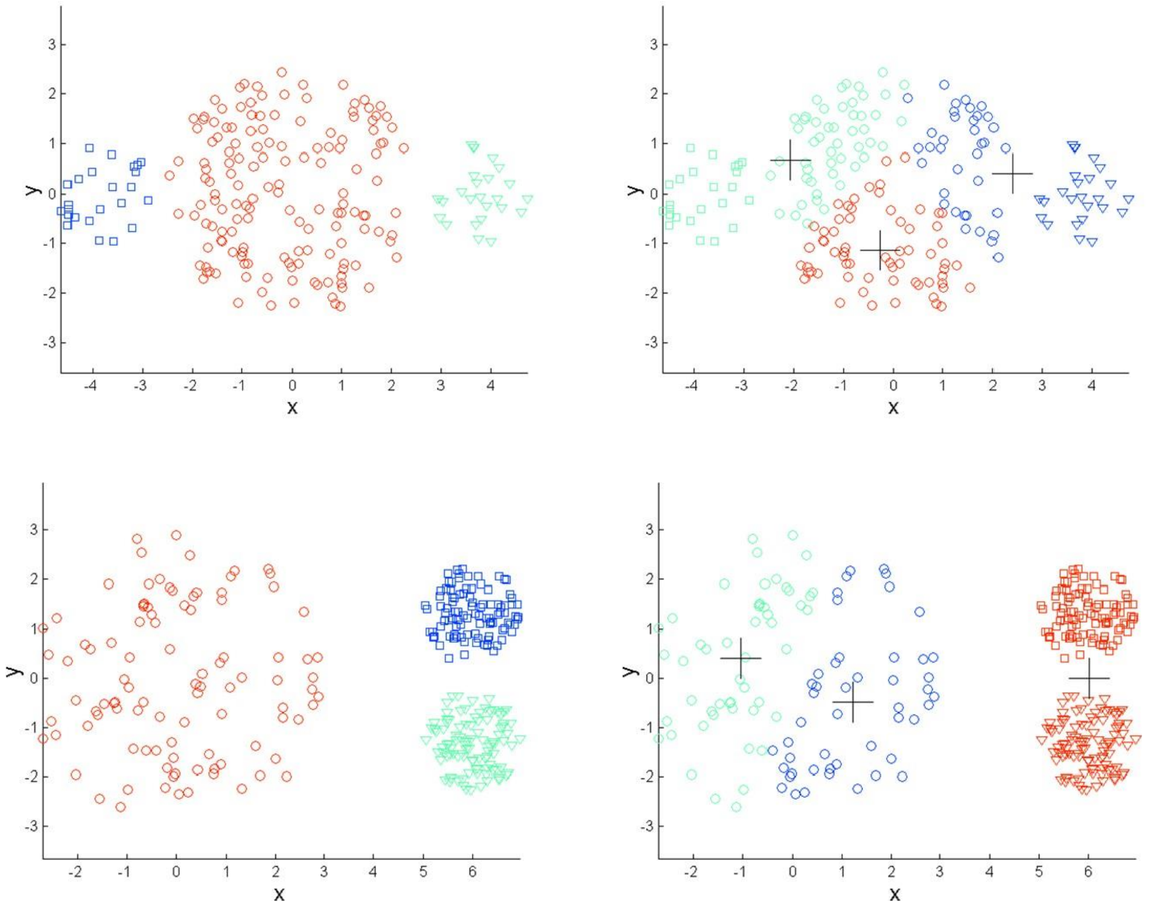
Local Minima 문제

- 군집의 무게중심 초기값 위치에 따라 최적의 결과가 나오지 않을 수 있음



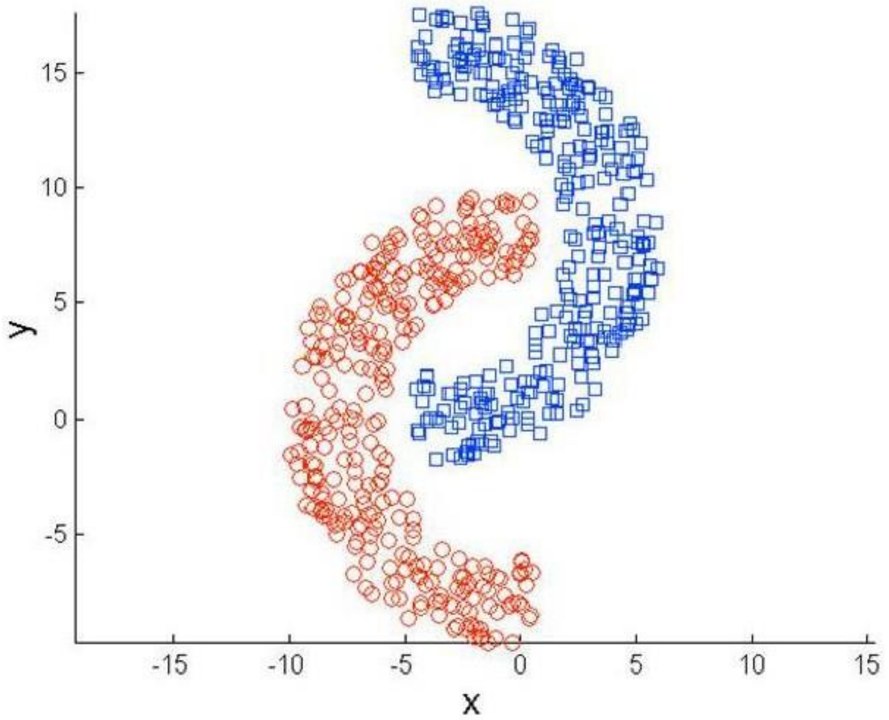
Local Minima 문제

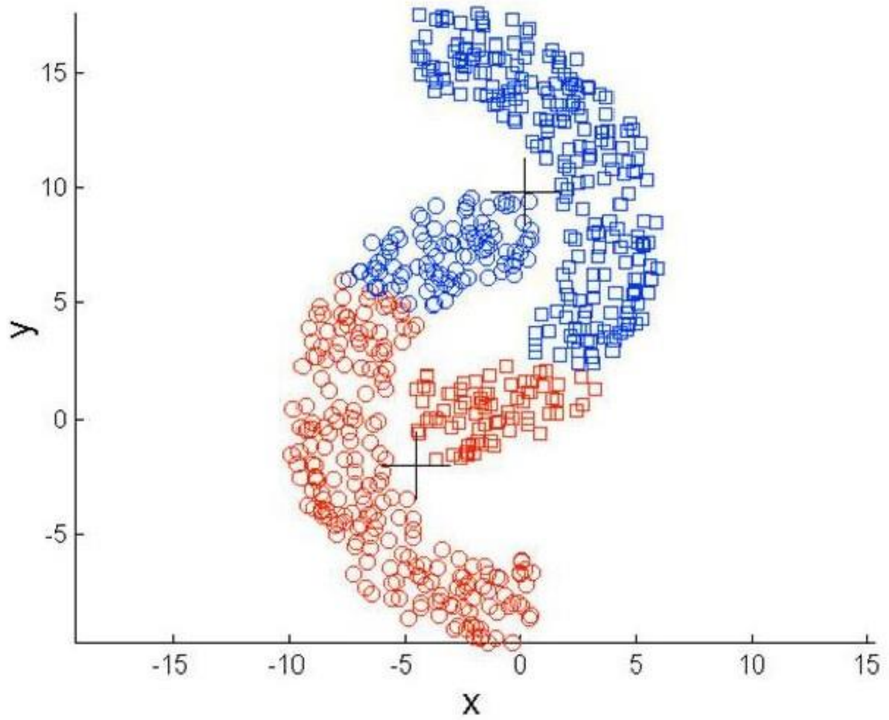
- 군집의 크기나 밀도가 다를 경우에도 의도치 않은 결과가 나올 수 있음



Local Minima 문제

- 데이터 분포가 특이한 경우에도 원하는 결과가 나오지 않을 수 있음

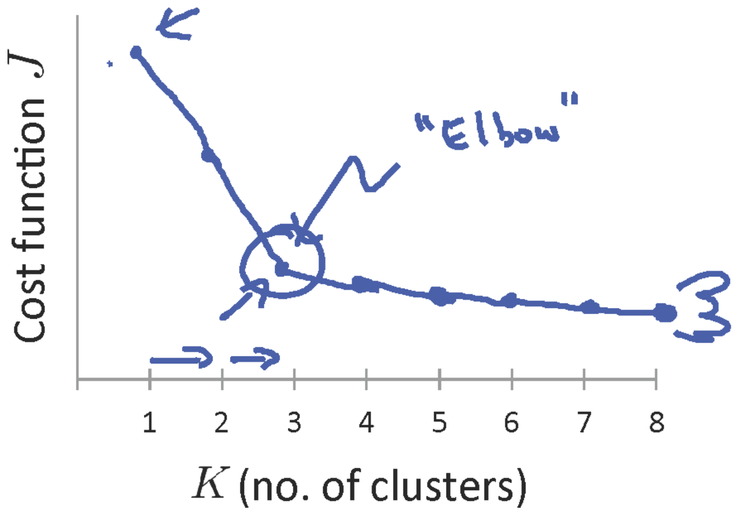


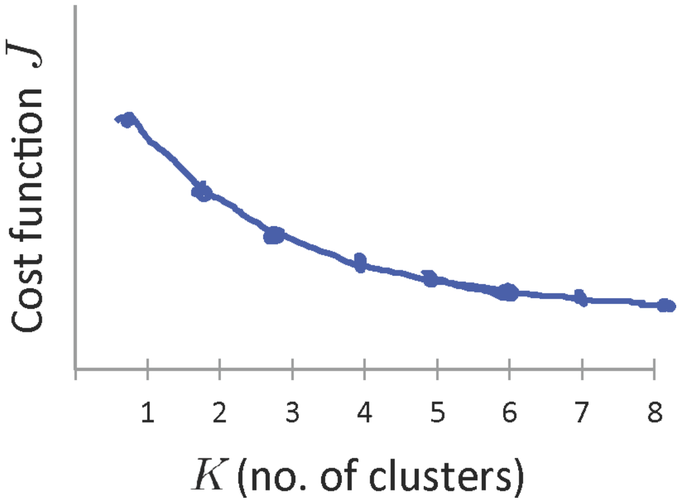


클러스터 개수 정하기

- K-means 알고리즘에서 하이퍼 파라미터인 최적의 클러스터 개수 K를 어떻게 정할까?

- Elbow Method
  다양한 K에 대하여 시도해보고 꺾이는 점이 있는 경우 해당 지점을 최적의 K라고 판단



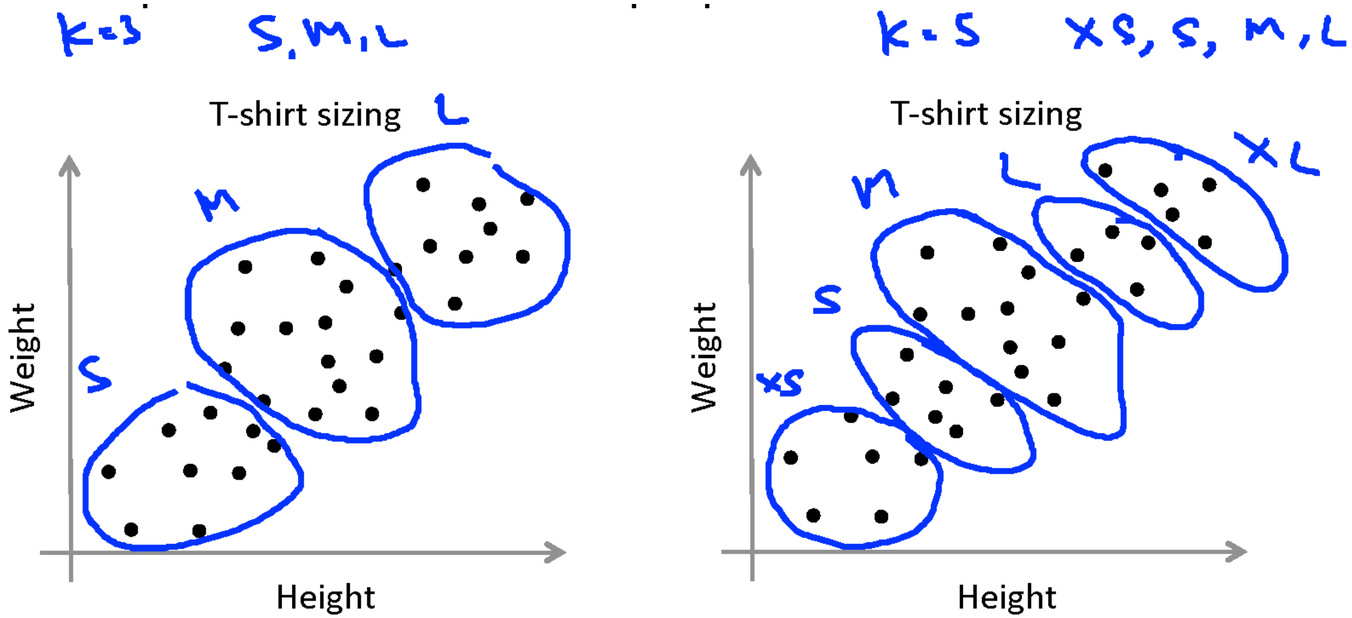


클러스터 개수 정하기

- K-means 알고리즘에서 하이퍼 파라미터인 최적의 클러스터 개수 K를 어떻게 정할까?

- 사전 지식 기반
  티셔츠 사이즈 등과 같은 경우 사전이 이미 어떤 사이즈가 있는지 알고 있으므로 개수를 정하기 수월하다.

E.g.

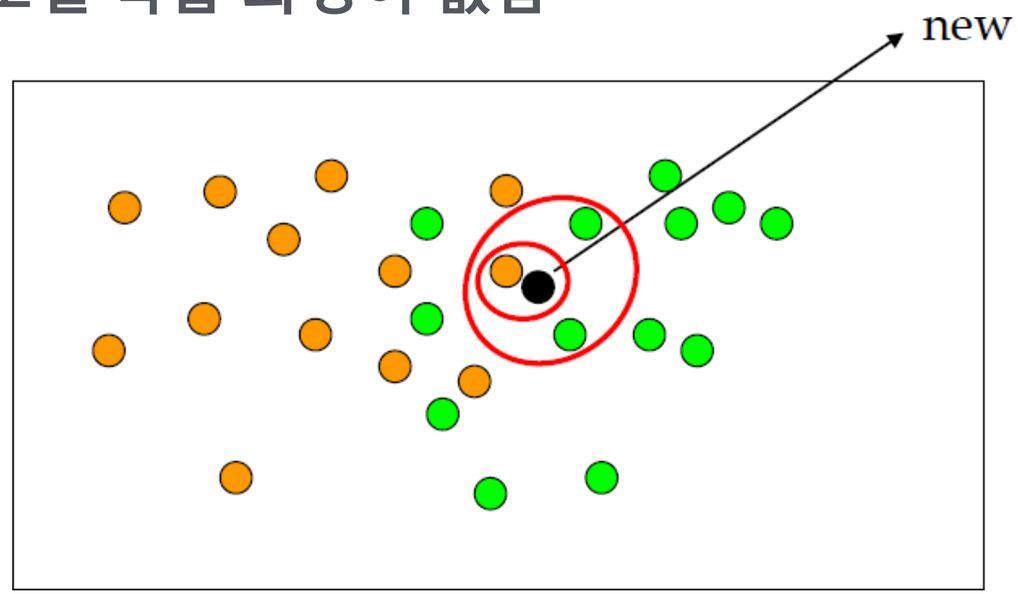


03

KNN(K-Nearest Neighbor)

KNN(K-Nearest Neighbor)

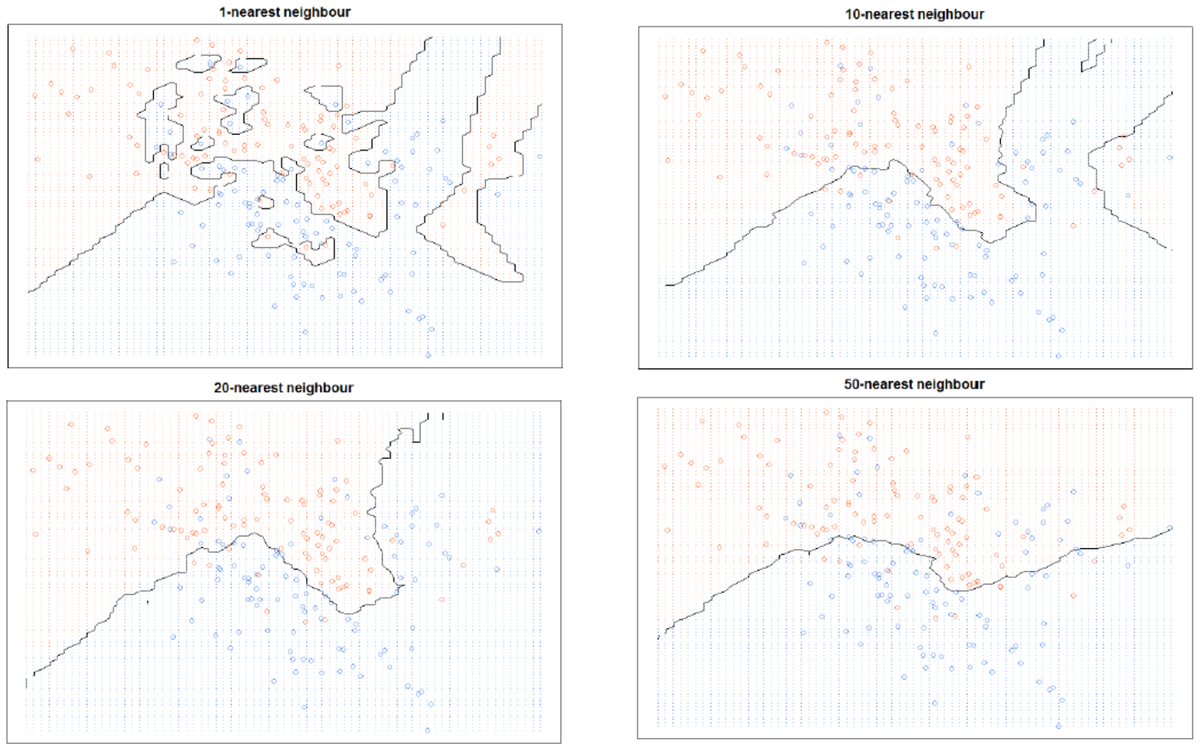
- 새로운 데이터가 주어졌을 때 기존 데이터 가운데 가장 가까운 k개 이웃의 정보로 새로운 데이터를 예측하는 방법론 (Supervised)
- 레이지(lazy) 모델 : 모델 학습 과정이 없음



- 그림에서 검은색 점의 범주 정보는 주변 이웃들을 가지고 추론
- k가 1이면 오렌지색, k가 3라면 녹색으로 분류

KNN(K-Nearest Neighbor)

- 입력
  탐색할 이웃 수(k), 거리 측정 방법
- k가 작을 경우 오버피팅, 반대로 매우 클 경우 언더피팅이 일어나는 경향이 있음

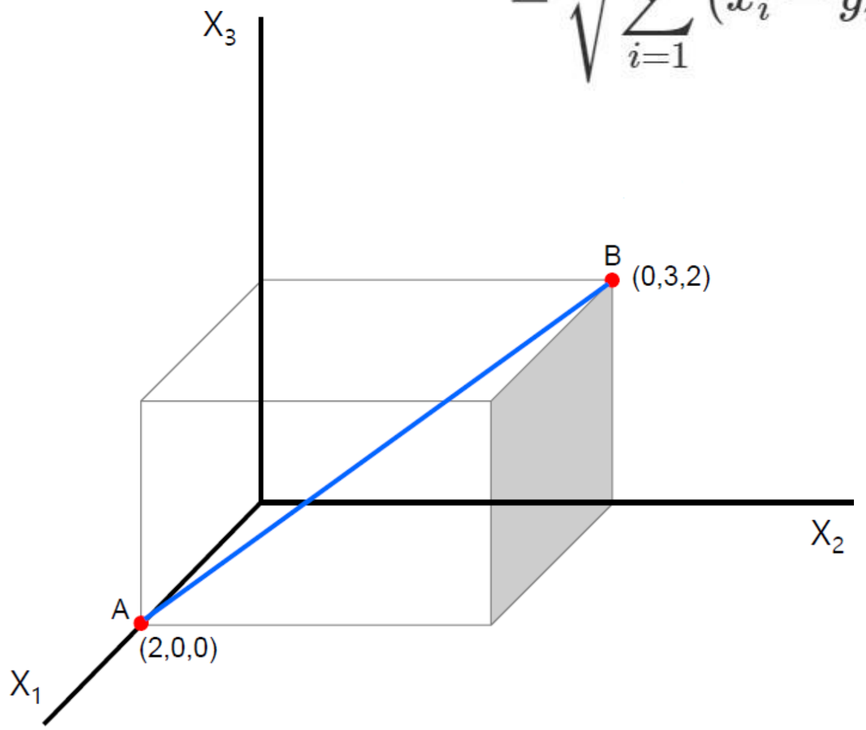


KNN의 거리 지표

- 유클리드 거리(Euclidean distance)

$$d_{(X,Y)} = \sqrt{(x_1 - y_1)^2 + \ldots + (x_n - y_n)^2}$$

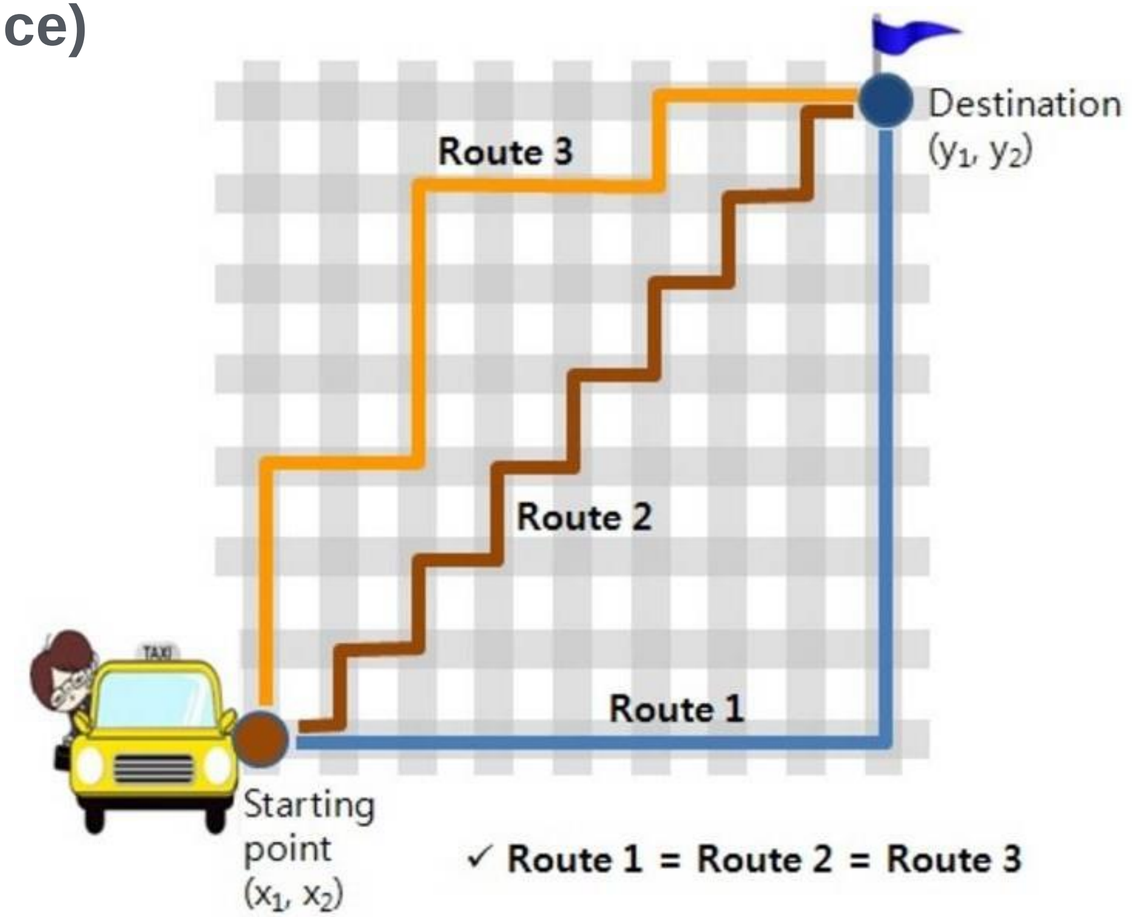
$$= \sqrt{\sum_{i=1}^{n} (x_i - y_i)^2}$$



KNN의 거리 지표

- 맨해튼 거리(Manhattan distance)

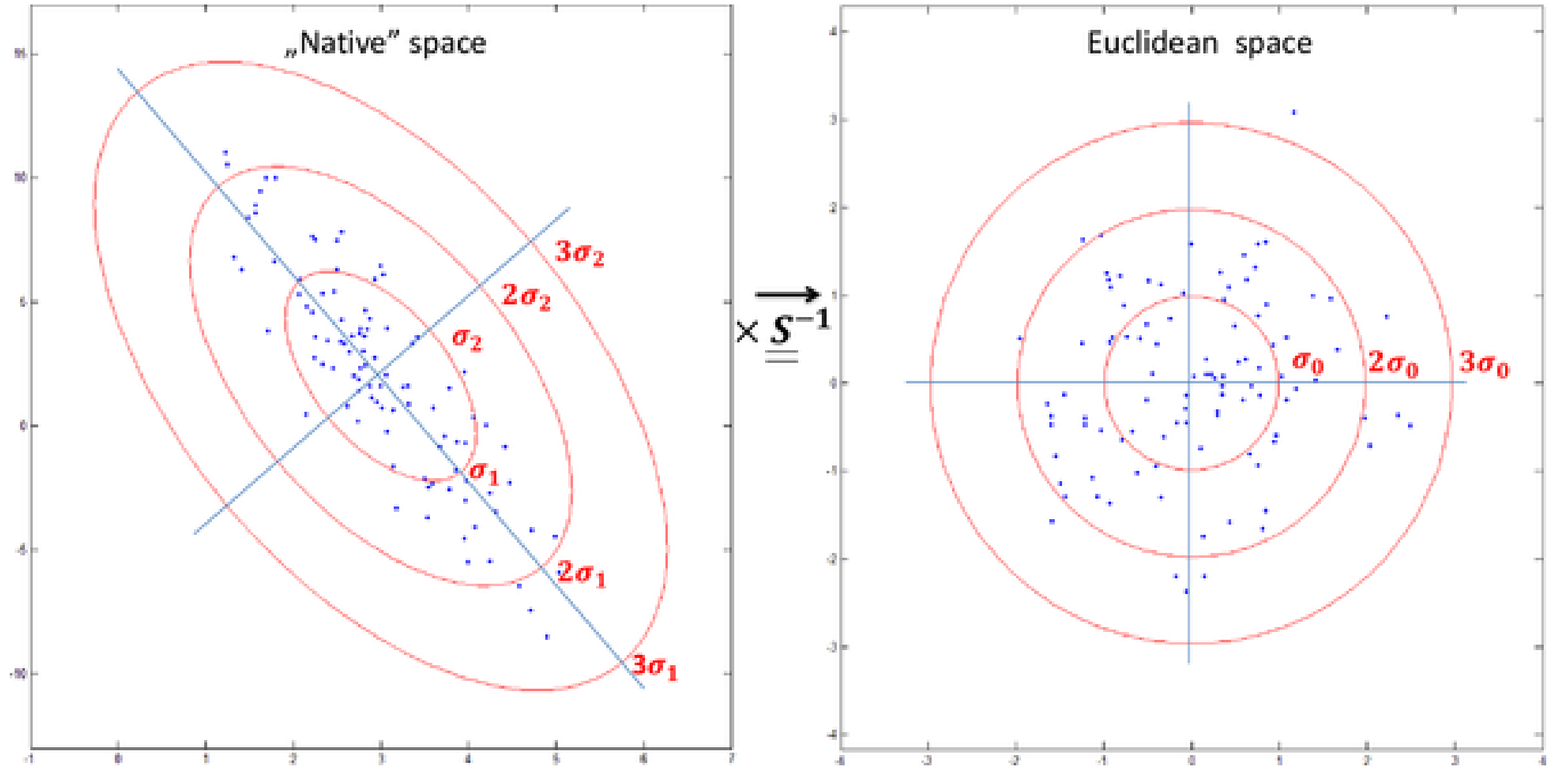
$$d_{Manhattan(X,Y)} = \sum_{i=1}^{n} |x_i - y_i|$$



KNN의 거리 지표

- 마할라노비스 거리(Mahalanobis distance)

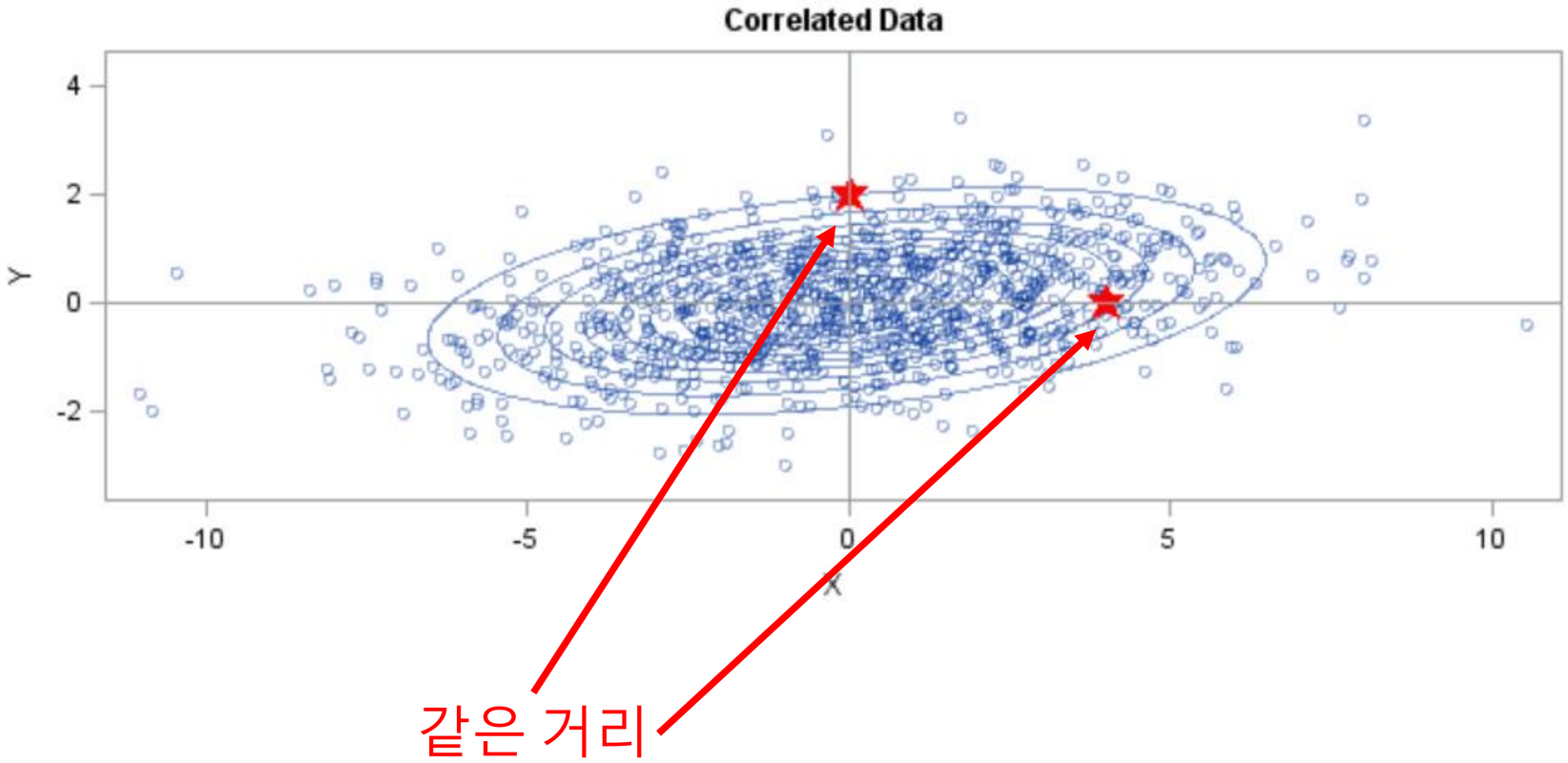
$$d(\vec{x},\vec{y}) = \sqrt{(\vec{x}-\vec{y})^T S^{-1}(\vec{x}-\vec{y})}.$$ S : covariance matrix



/* elice */

KNN의 거리 지표

- 마할라노비스 거리(Mahalanobis distance)



04

GMM(Gaussian Mixture Model)

Gaussian Distribution

- Gaussian 분포(정규 분포)

$$f(x) = \frac{1}{\sigma\sqrt{2\pi}} e^{-\frac{1}{2}\left(\frac{x-\mu}{\sigma}\right)^2}$$

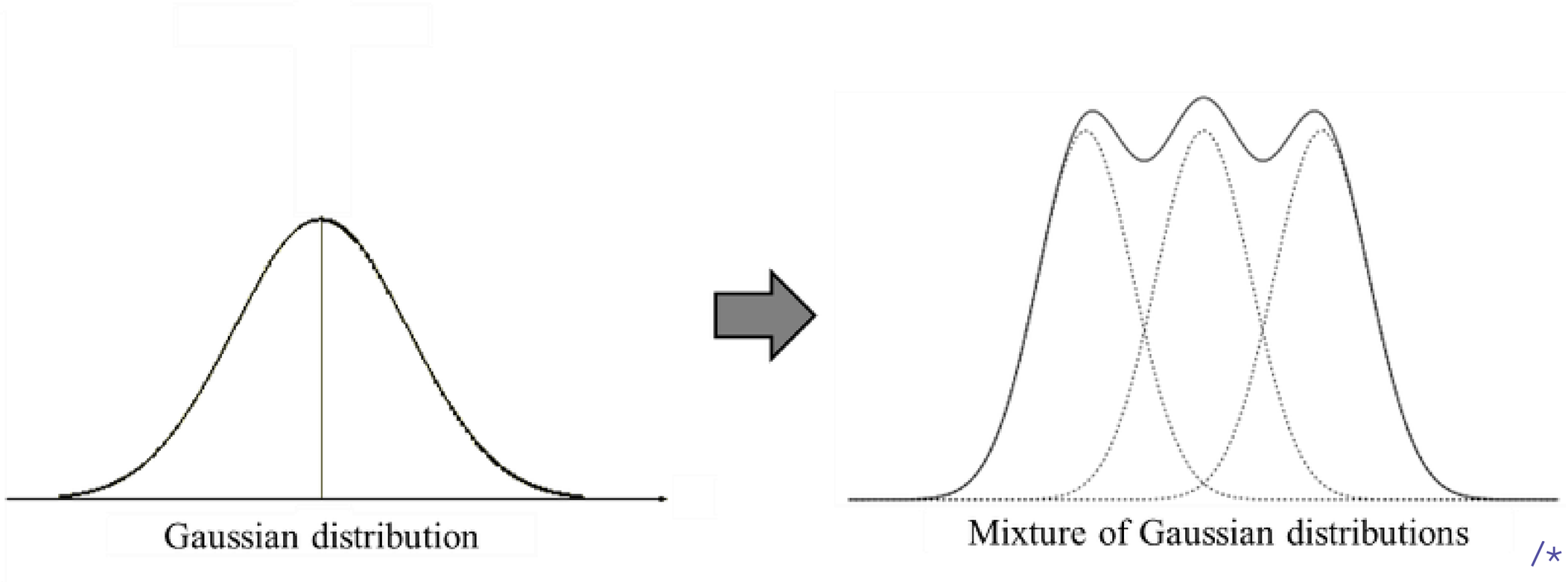
μ : 평균

σ : 분산

- 평균이 0이고 분산이 1이면 표준 정규분포

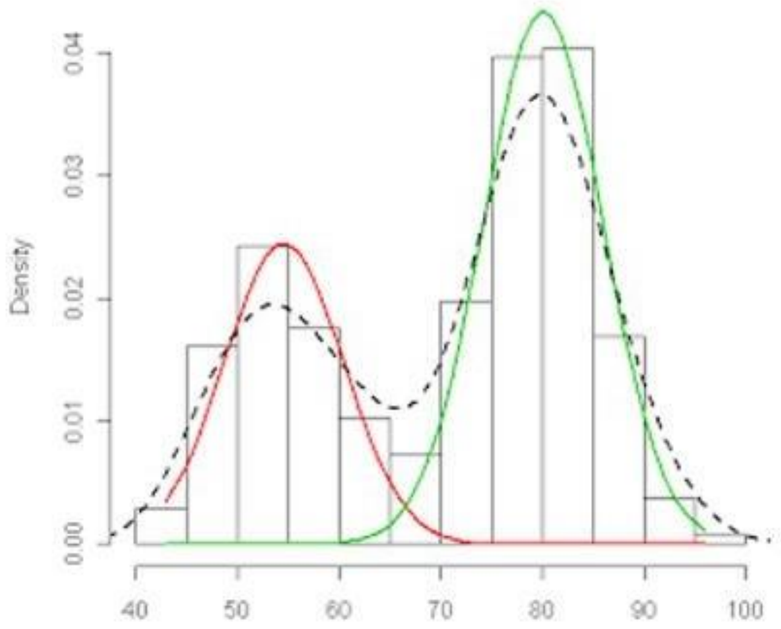
GMM?

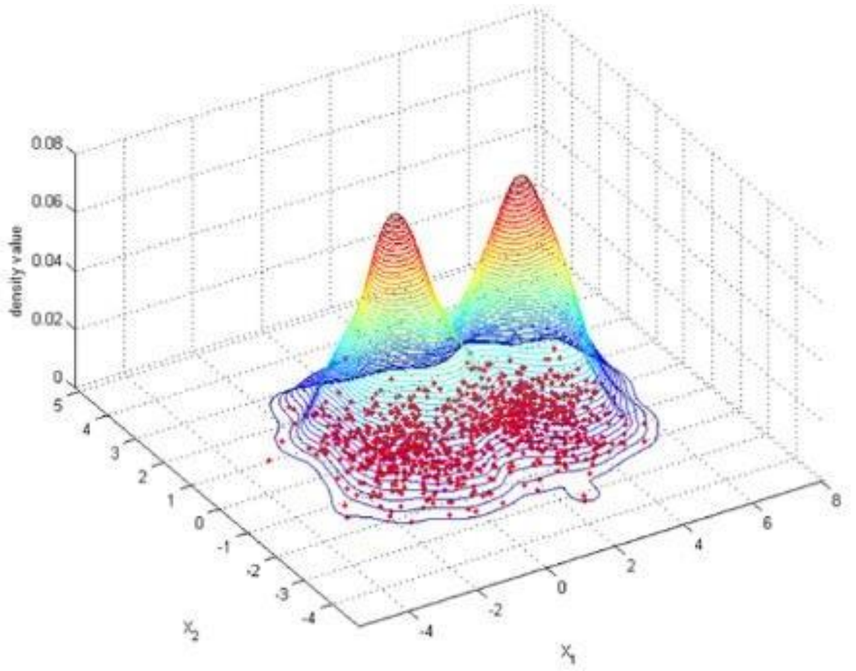
- 전체 데이터의 확률분포가 여러 개의 정규분포의 합으로 이루어져 있다고 가정하고 각 분포에 속할 확률이 높은 데이터끼리 클러스터링 하는 방법



GMM?

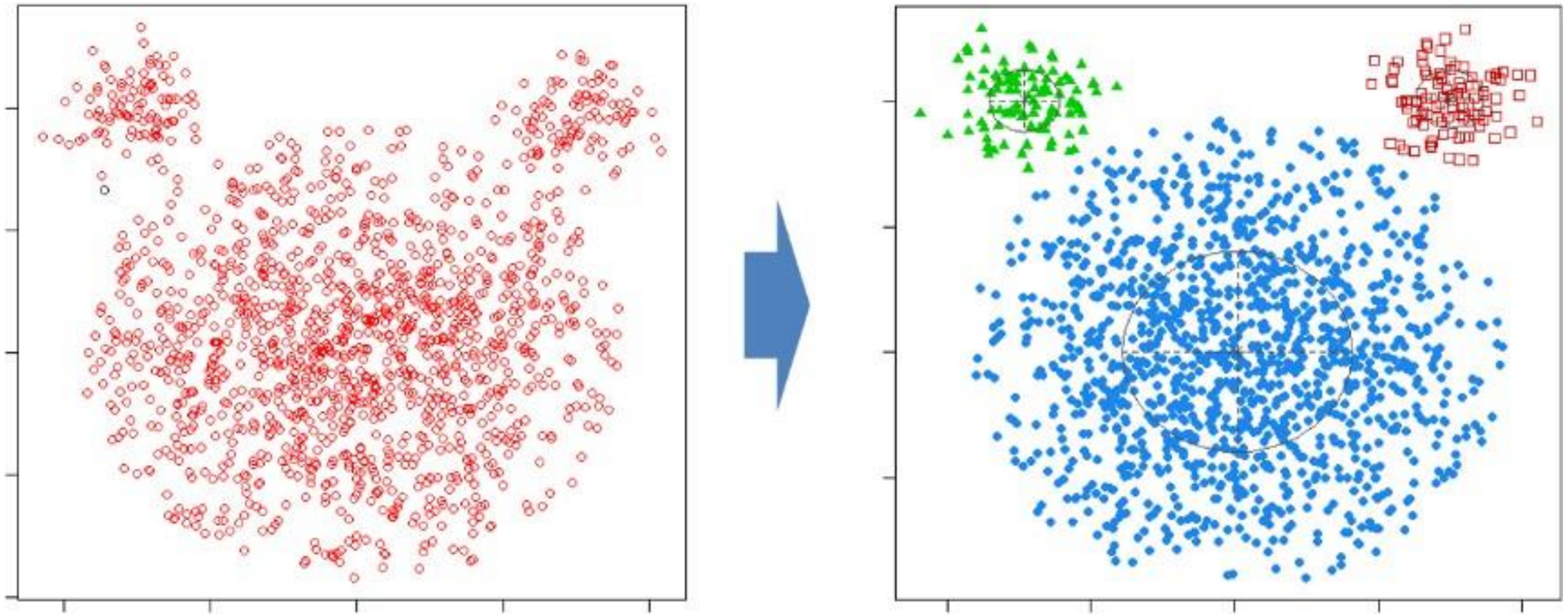
- 다변수 정규분포(Multivariate Gaussian Distribution)를 가정할 수도 있다.





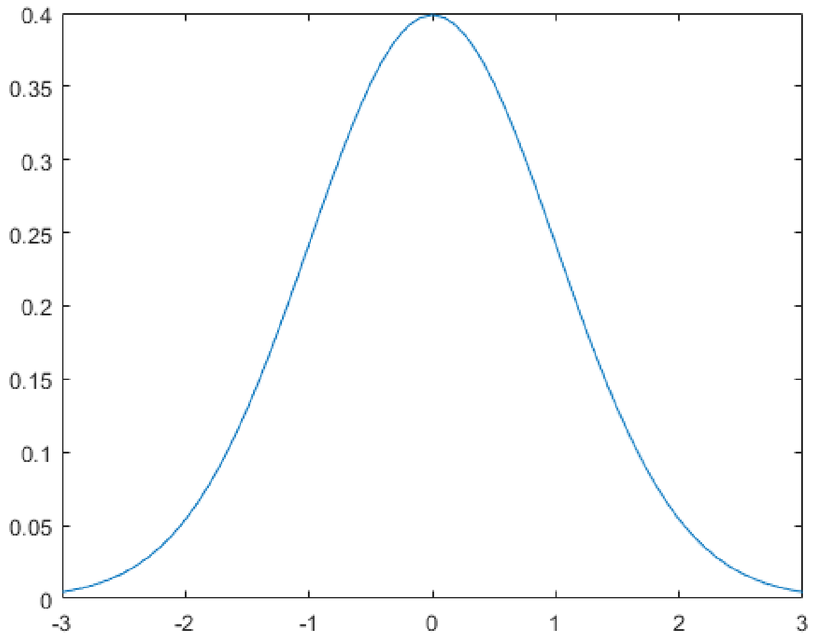
GMM?

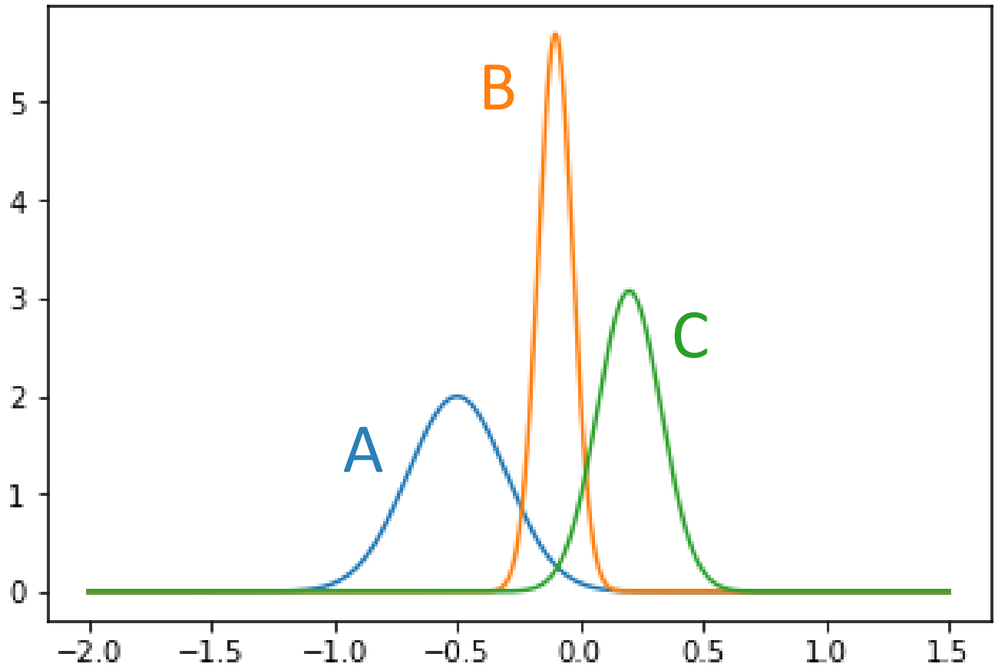
- K-means 클러스터링 알고리즘으로 잘 묶이지 않았던 데이터에 대하여 잘 동작하는 모습을 보인다.



GMM 동작 과정

- 데이터들이 여러 개의 정규분포에서 나왔다고 가정
- 만약 A, B, C 라는 세 종류의 정규분포에서 데이터들이 생성 되었다고 한다면?

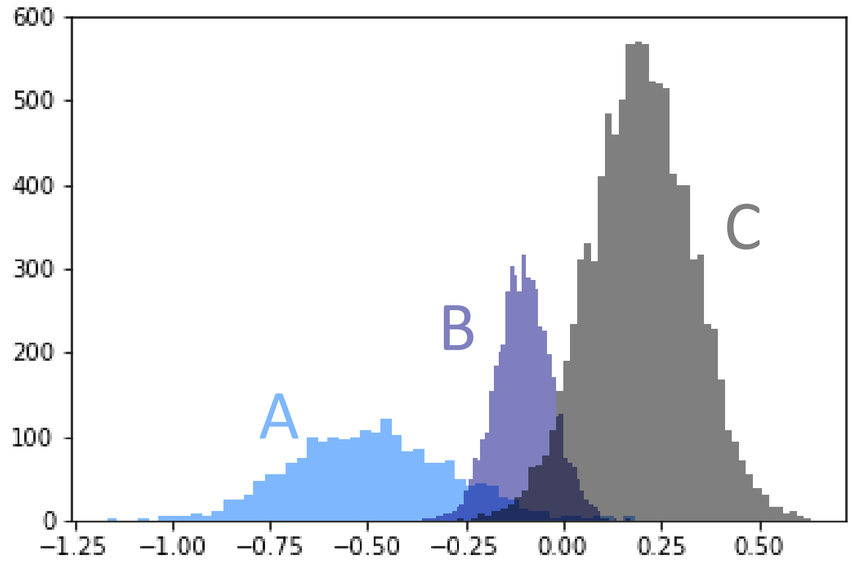


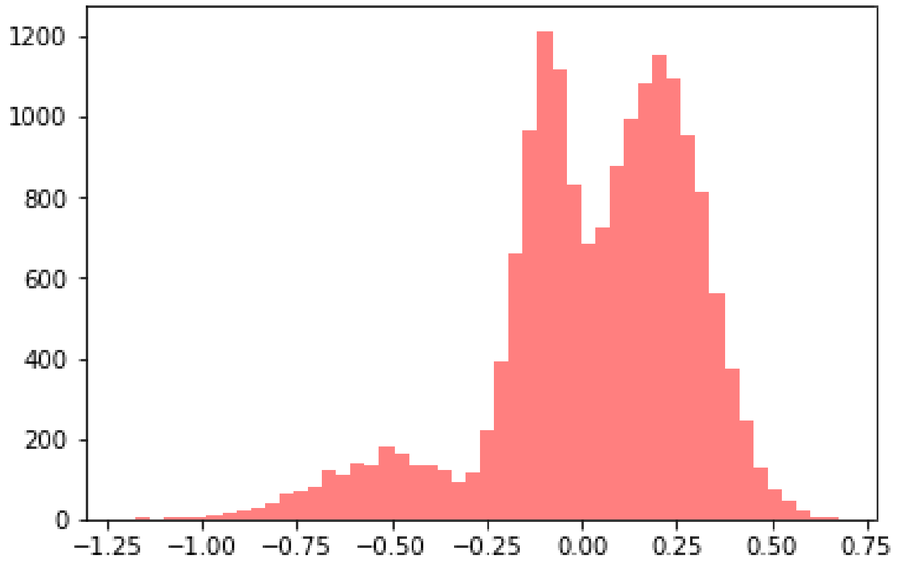


/* elice */

GMM 동작 과정

- 각 정규분포 A, B, C에서 2,000개, 5,000개, 10,000개 씩 임의로 뽑은 뒤 분포를 그려볼 수 있다. (왼쪽 그림)
- 하지만 실제 우리가 보는 것은 오른쪽 그림
- 오른쪽 그림에 GMM을 적용하여 3개의 정규분포의 평균, 분산, 임의의 데이터가 확률적으로 어디에 속해 있는지(Weight) 를 구해야 함.

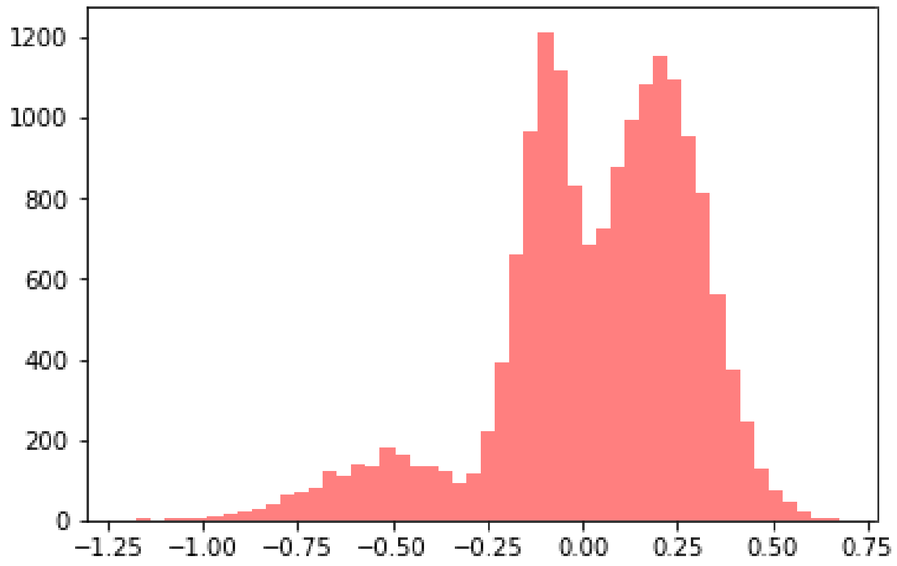


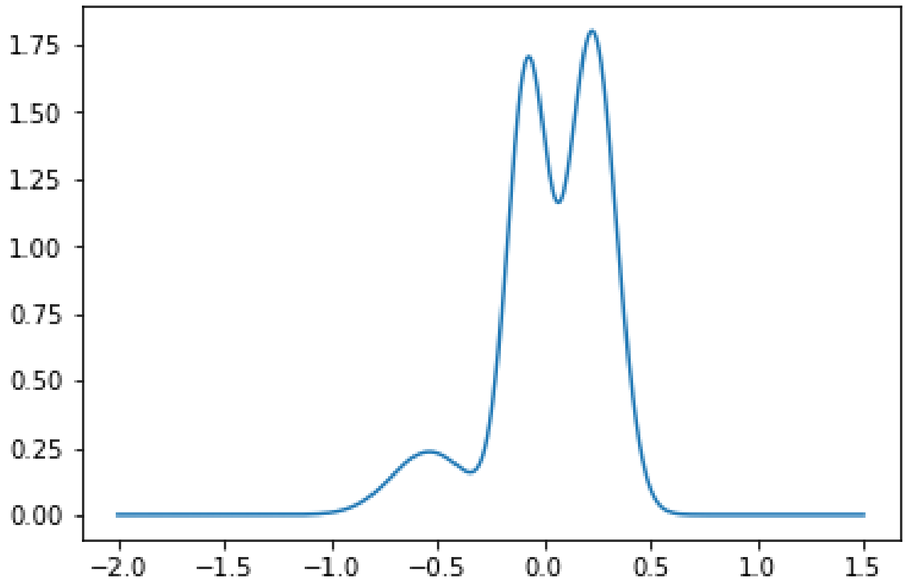


/* elice */

GMM 동작 과정

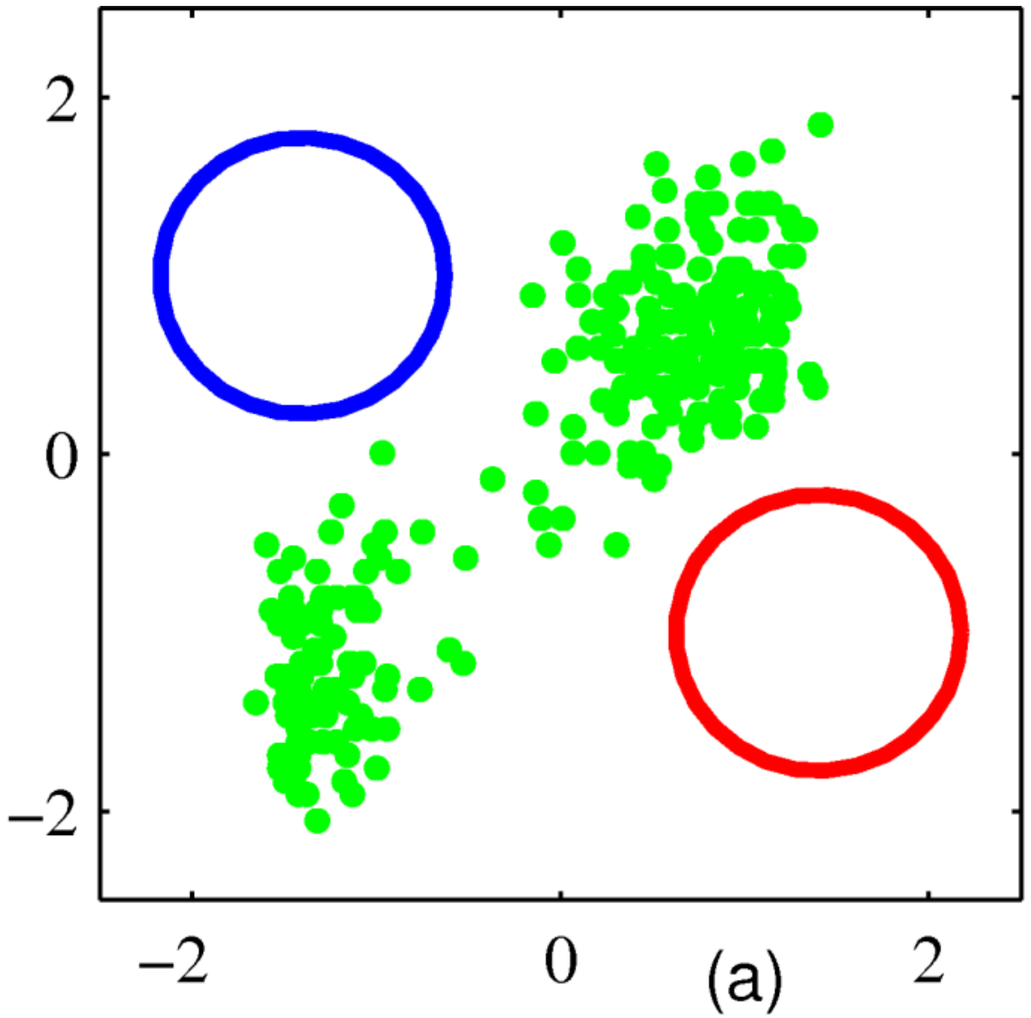
- 3개의 정규분포의 평균, 분산, 임의의 데이터가 확률적으로 어디에 속해 있는지(Weight) 에 대한 값들을 **모수**라고 부름
- Weight : 3개의 정규분포 중 데이터가 확률적으로 어디에 속해 있는가 (잠재변수)
- Expectation Maximization (EM) 알고리즘을 이용하여 구한다.



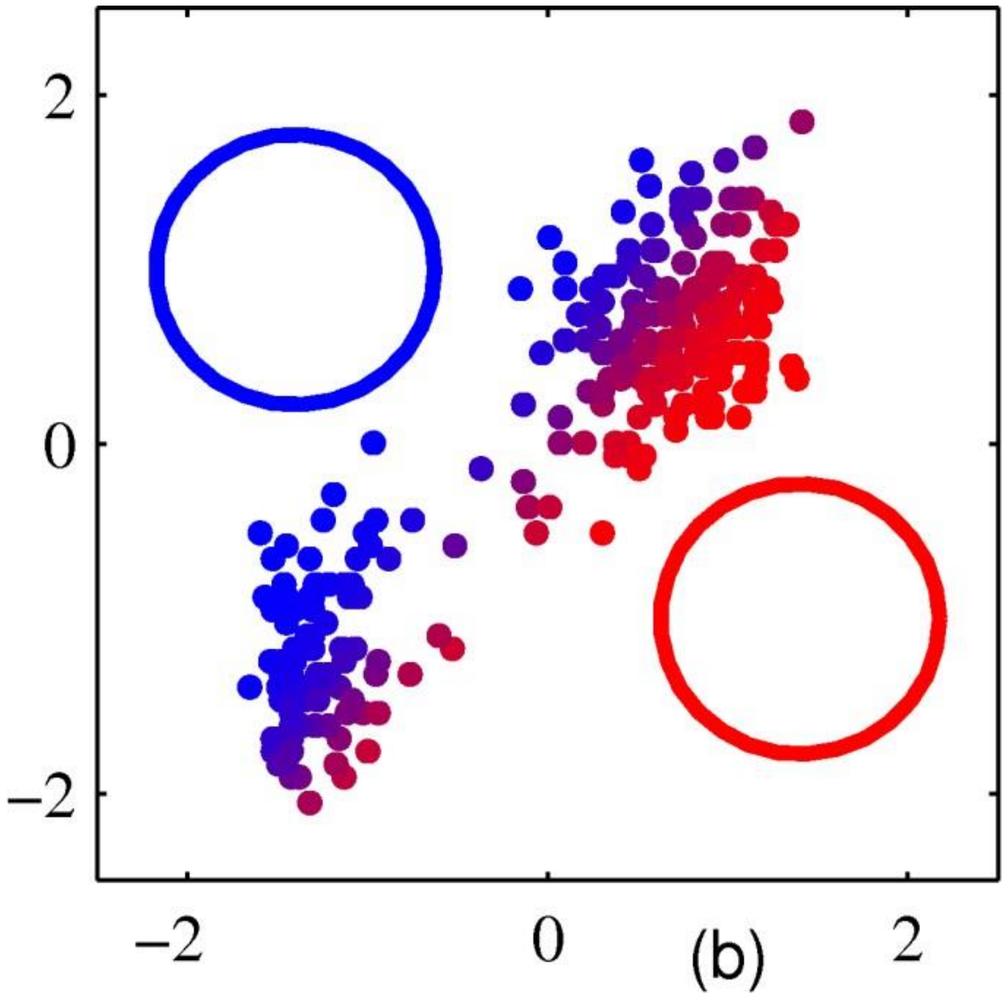


/* elice */

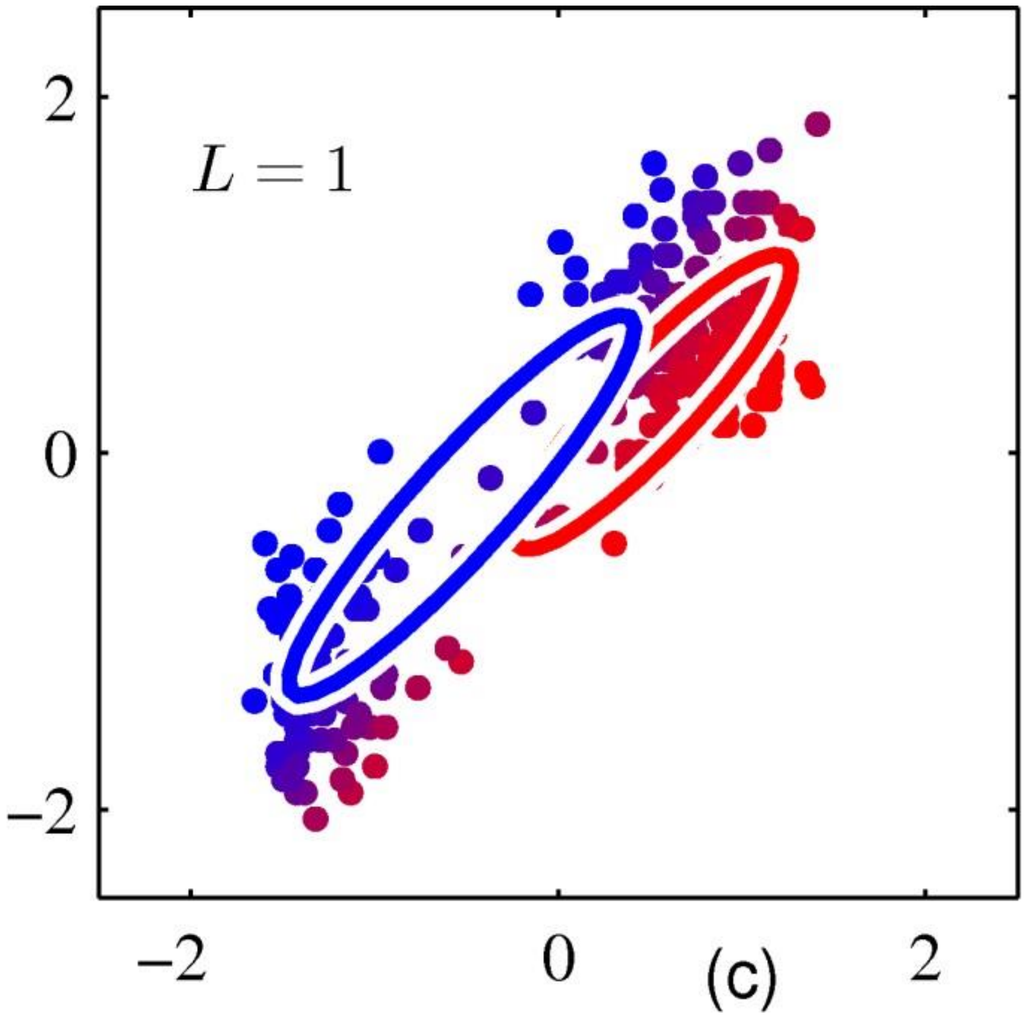
GMM 동작 과정



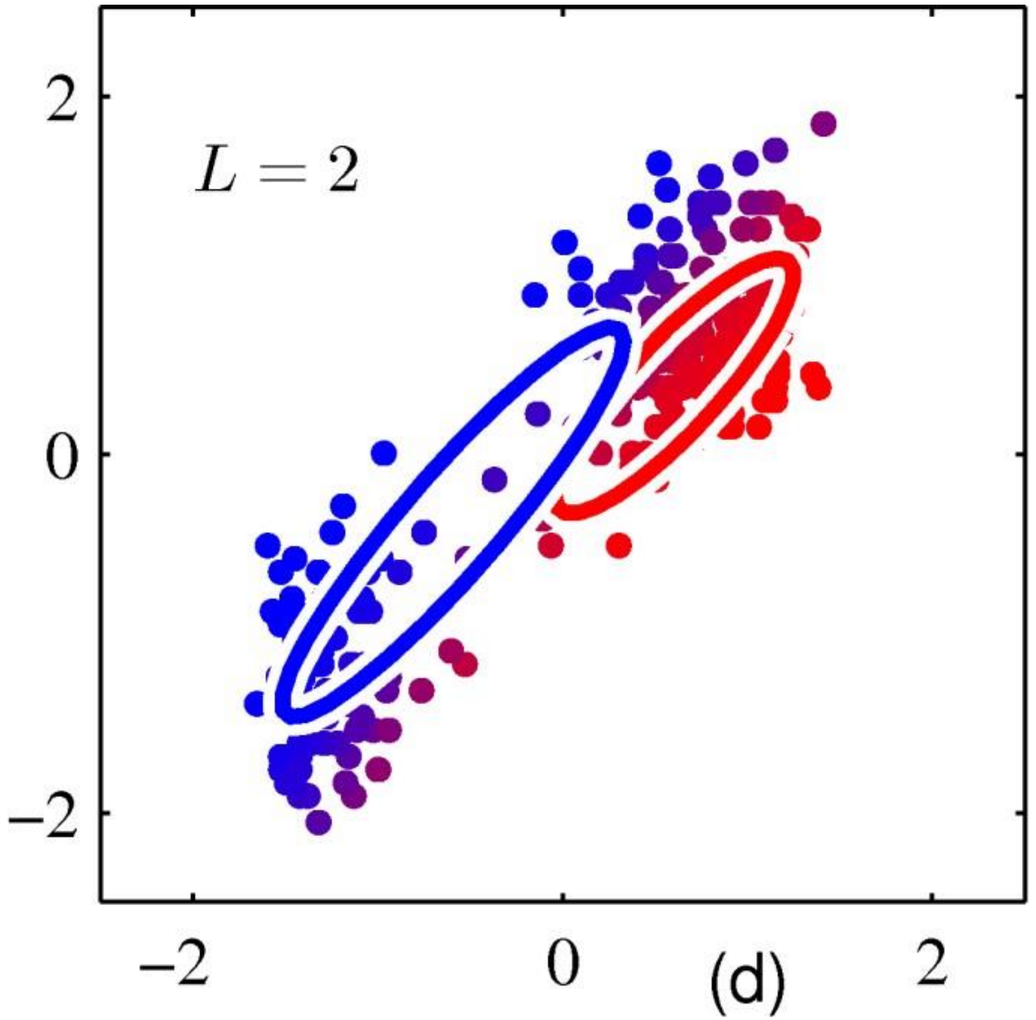
GMM 동작 과정



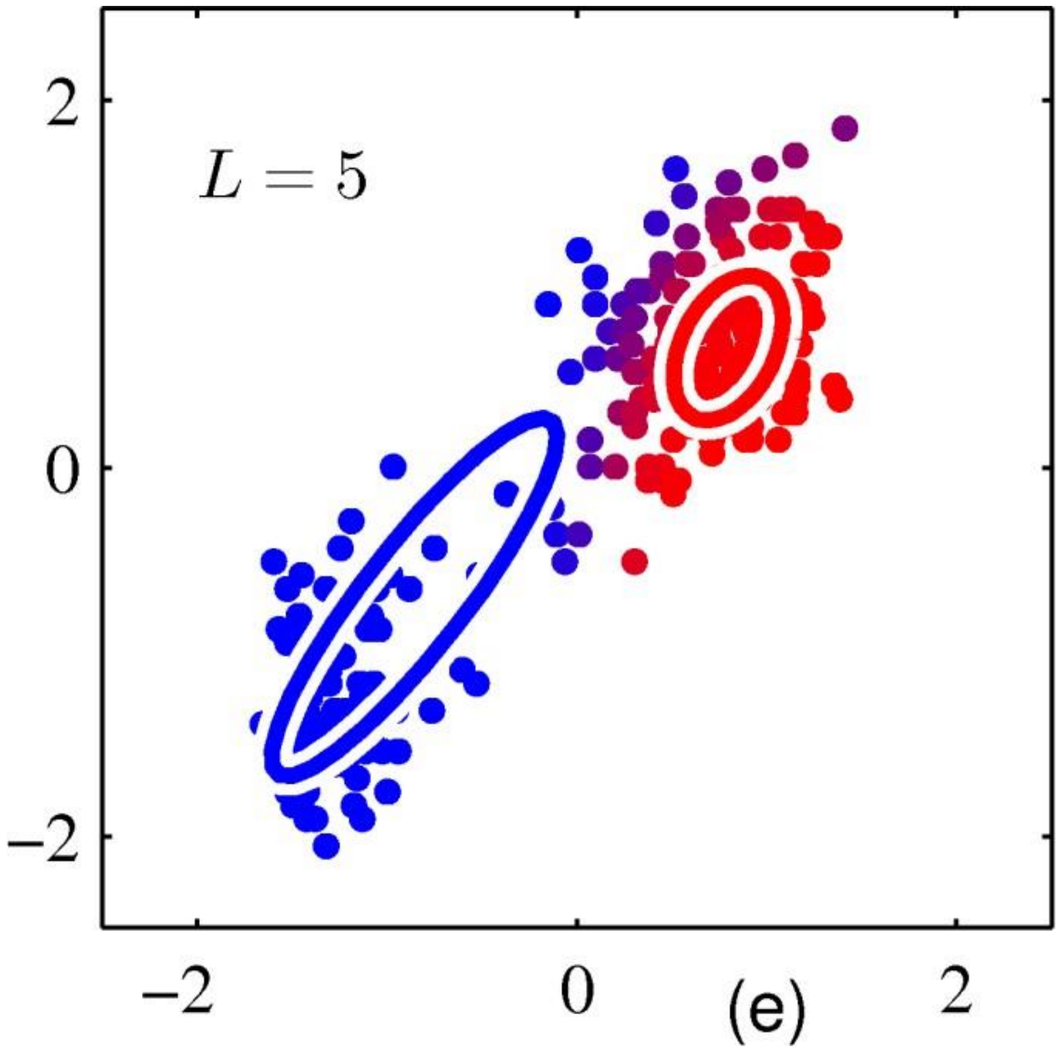
GMM 동작 과정



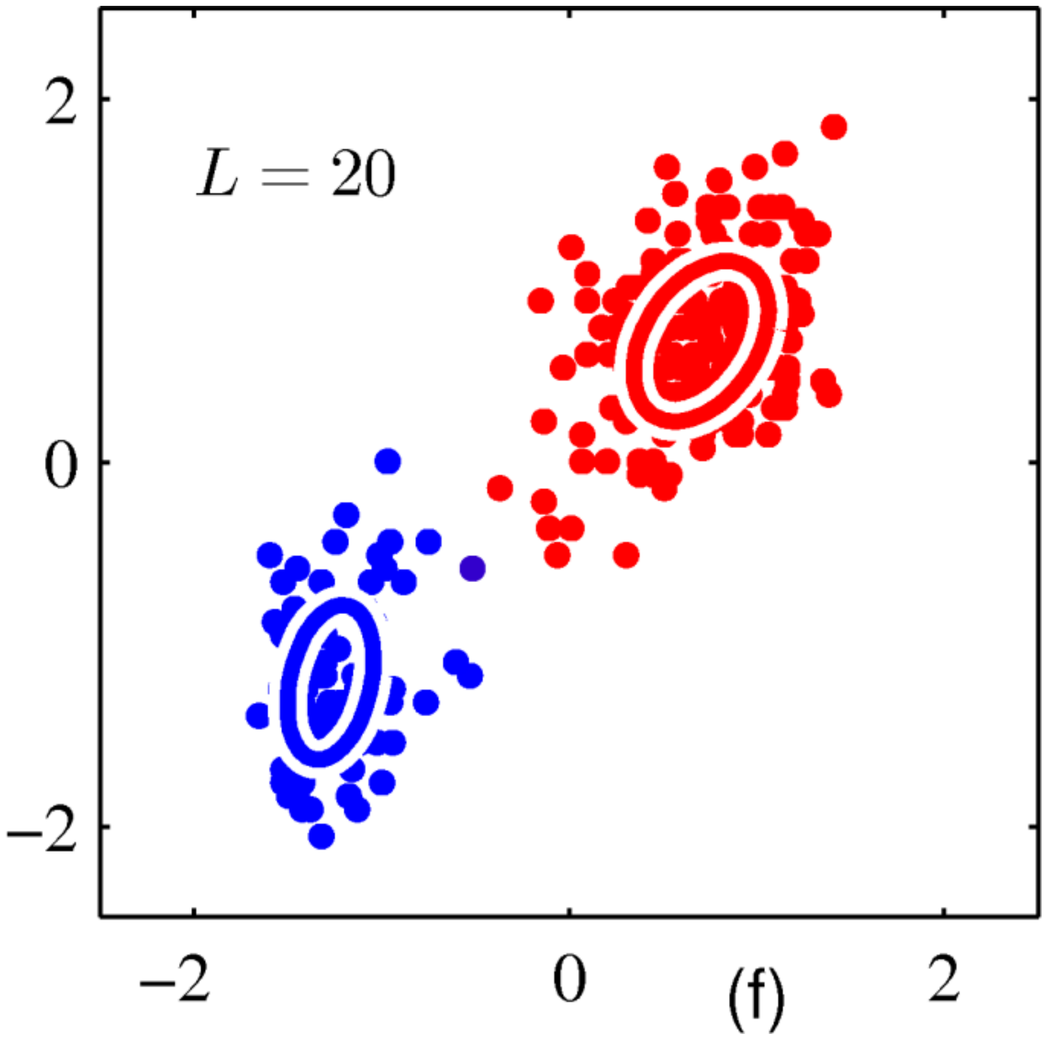
GMM 동작 과정



GMM 동작 과정



GMM 동작 과정

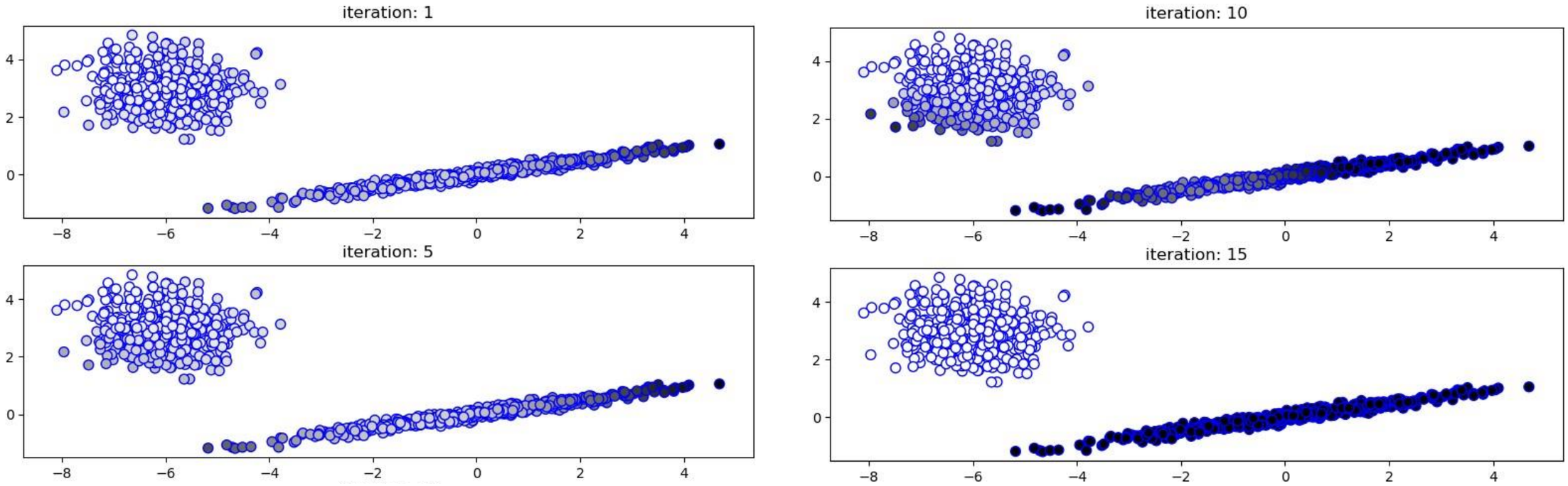


클러스터 구분 방법

- 어떤 Gaussian 분포에서 나온 데이터인지 예측
- EM 알고리즘을 통해 모수(평균, 분산)와 responsibility 를 예측
- 데이터에 대해 responsibility 값이 가장 큰 카테고리를 찾아낸다.
- 데이터가 각 정규분포에서 가지는 확률 값이 가장 큰 분포에 할당

클러스터 구분 방법

- iteration이 증가함에 따라 분류가 진행됨을 확인할 수 있음



모수 추정

Randomly initialize π, μ, Σ
for $t = 1 : T$ **do**
  // E-step
  for $n = 1 : N$ **do**
    for $k = 1 : K$ **do**
      $\gamma(z_{nk}) = \frac{\pi_k N(x_n|\mu_k, \Sigma_k)}{\sum_{j=1}^{K} \pi_j N(x_n|\mu_j, \Sigma_j)}$
    end
  end
  // M-step
  for $k = 1 : K$ **do**
    $\mu_k = \frac{\sum_{n=1}^{N} \gamma(z_{nk}) x_n}{\sum_{n=1}^{N} \gamma(z_{nk})}$
    $\Sigma_k = \frac{\sum_{n=1}^{N} \gamma(z_{nk})(x_n - \mu_k)(x_n - \mu_k)^T}{\sum_{n=1}^{N} \gamma(z_{nk})}$
    $\pi_k = \frac{1}{N} \sum_{n=1}^{N} \gamma(z_{nk})$
  end
end

Input : a given data $X = \{x_1, x_2, \ldots, x_n\}$
Output : $\pi = \{\pi_1, \pi_2, \ldots, \pi_K\}$,
$\mu = \{\mu_1, \mu_2, \ldots, \mu_K\}$,
$\Sigma = \{\Sigma_1, \Sigma_2, \ldots, \Sigma_K\}$

/* elice */

Class 분류

Algorithm 2: GMM classification

Input : a given data $X = \{x_1, x_2, ..., x_n\}$,
$\pi = \{\pi_1, \pi_2, ..., \pi_K\}$,
$\mu = \{\mu_1, \mu_2, ..., \mu_K\}$,
$\Sigma = \{\Sigma_1, \Sigma_2, ..., \Sigma_K\}$

Output : class labels $y = \{y_1, y_2, ..., y_N\}$ for X

1 **for** $n = 1 : N$ **do**

2 $\quad y_n = \arg\max_k \gamma(z_{nk})$

3 **end**

(부록) 일반적인 EM 알고리즘

- **Latent variable?**

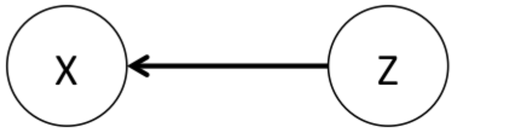
  우리가 원래 알고있는 Random Variable이 아닌 관측 불가능한 임의로 설정한 hidden variable

- **일반적인 EM 알고리즘?**

  X의 Log Likelihood의 식이 복잡하여 MLE를 통해 모수 계산이 어려운 경우 사용하는 기법으로 계산이 쉽도록 도와주는 latent variable Z를 가정한 후 Expectation 단계와 Maximization 단계를 반복해가며 수렴시켜 MLE를 계산하는 기법

(부록) 일반적인 EM 알고리즘

- 아래 그림과 같은 Grapical model을 고려해보자



Hidden variable

- 이때 우리가 관측할 수 있는 random variable은 paramter θ로 parameterized 되어있는 X 하나이고, Z은 우리가 관측할 수 없는 hidden variable이라고 가정한다.
- 만약 marginal distribution p(X|θ)를 직접 계산하는 것이 매우 어려울 경우 X의 maximum likelihood를 계산하고 싶다면 어떻게 해야 할까?

(부록) 일반적인 EM 알고리즘

- X 의 maximum likelihood는 다음과 같이 표현된다.

$$\max_{\theta} p(\mathbf{X}|\theta) = \sum_{\mathbf{Z}} p(\mathbf{X}, Z|\theta).$$

- 위의 식에서 Z는 discrete variable이라고 정의
- joint distribution p(X, Z|θ)의 계산이 쉽도록 Z를 가정
  - Z는 우리 마음대로 정할 수 있는 latent variable

만약 우리가 latent variable Z2의 marginal distribution을 q(Z)q(Z)라고 정의한다면, 앞에서 살펴본 log-likelihood를 다음과 같이 decompose할 수 있다.

(부록) 일반적인 EM 알고리즘

- log-likelihood를 다음과 같이 decompose

$$\ln p(\mathbf{X}|\theta) = \mathcal{L}(q,\theta) + \mathrm{KL}(q\|p)$$

- q(Z) : latent variable Z의 marginal distribution

만약 우리가 latent variable Z2의 marginal distribution을 q(Z)q(Z)라고 정의한다면, 앞에서 살펴본 log-likelihood를 다음과 같이 decompose할 수 있다.

(부록) 일반적인 EM 알고리즘

$$\ln p(\mathbf{X}|\theta) = \mathcal{L}(q,\theta) + \mathrm{KL}(q\|p)$$

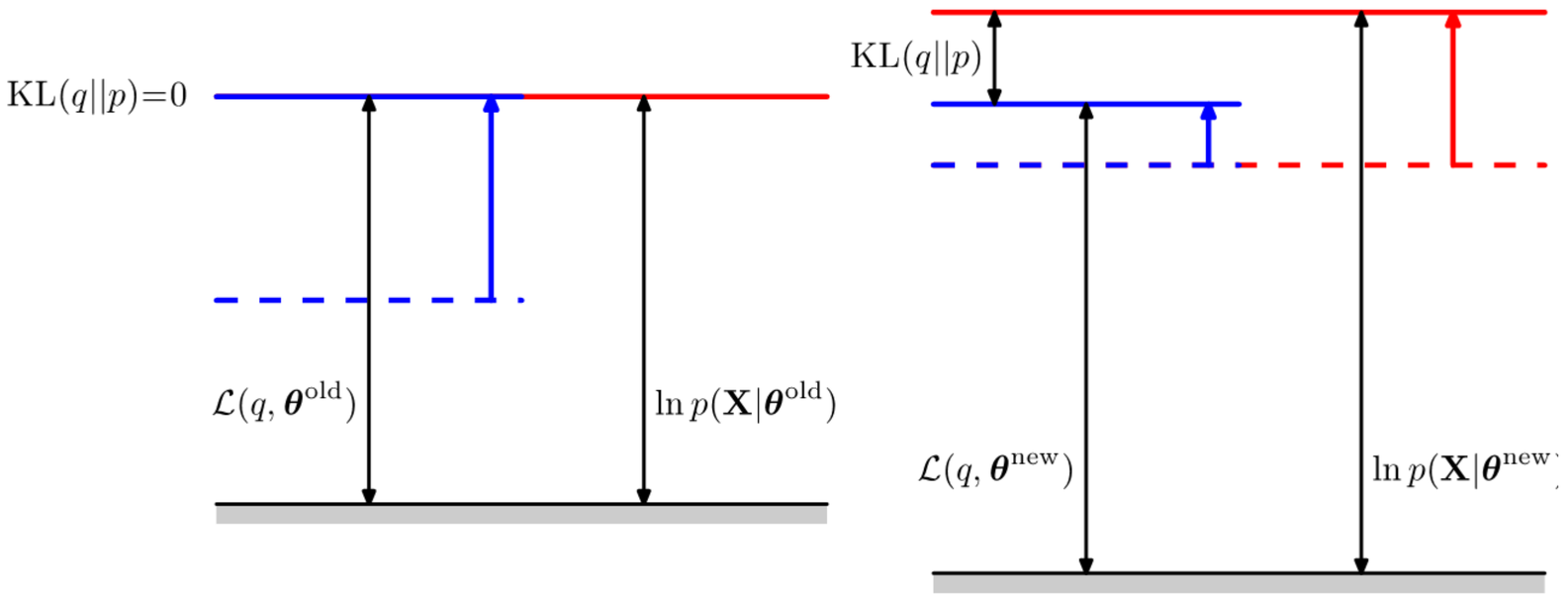
- $$\mathcal{L}(q,\theta) = \sum_{\mathbf{Z}} q(\mathbf{Z}) \ln \frac{p(\mathbf{X},\mathbf{Z}|\theta)}{q(\mathbf{Z})} \text{ and } \mathrm{KL}(q\|p) = -\sum_{\mathbf{Z}} q(\mathbf{Z}) \ln \frac{p(\mathbf{Z}|\mathbf{X},\theta)}{q(\mathbf{Z})}$$

- L(q,θ) : q(Z)의 function
- KL(q‖p) : q, p의 KL divergence

- 왼쪽은 X, Z의 joint distribution, 오른쪽은 conditional distribution으로 표현
  - **p(X|θ)를 직접 계산하지 않아도 됨!**

(부록) 일반적인 EM 알고리즘

- KL divergence는 반드시 0보다 크거나 같기 때문에 L(q,θ)이 log-likelihood의 lower bound가 된다.
- 따라서 lower bound가 maximum이 되는 θ와 q(Z)의 값을 찾고, 그에 해당하는 log-likelihood의 값을 찾는 알고리즘을 설계하는 것이 가능



(부록) 일반적인 EM 알고리즘

- 만약 θ와 q(Z)를 jointly optimize하는 문제가 어렵다면?

  둘 중 한 variable을 고정해두고 나머지를 update한 다음, 나머지 variable을 같은 방식으로 update하는 과정을 반복하여 수렴 시키면 됨. (EM 알고리즘 핵심 아이디어)
- EM 알고리즘은 E-step과 M-step 두 가지 단계로 구성되며 각각의 step에서는 θ와 q(Z)를 번갈아 가면서 한 쪽은 고정한채 나머지를 update한다.
- E-step과 M-step을 수렴할 때까지 혹은 지정된 iteration 수만큼 반복

(부록) 일반적인 EM 알고리즘

- E step
- 현재 우리가 가지고 있는 parameter θ의 값을 θ_{old}라고 정의
- 먼저 θ_{old} 값을 고정해두고 L(q,θ)의 값을 최대로 만드는 q(Z)의 값을 찾는 과정
- KL-divergence를 0으로 만들고, lower bound와 likelihood의 값을 일치시킴
  - log-likelihood $lnp(X|\theta_{old})$는 q(Z) 값과 전혀 관계가 없음
  - 따라서 L(q,θ)를 최대로 만드는 조건은 KL divergence가 0이 되는 상황
  - KL divergence는 $q(Z) = p(Z|X, \theta_{old})$일 때 0이 되기 때문에 q(Z)에 posterior distribution $p(Z|X, \theta_{old})$을 대입하는 것으로 해결

(부록) 일반적인 EM 알고리즘

- M step
- q(Z)를 고정하고 log-likelihood를 가장 크게 만드는 θ_{new}를 찾는 optimization 문제를 푸는 단계
- θ가 log-likelihood에 직접 영향을 미치기 때문에 log-likelihood 자체가 증가

- θ_{old}가 θ_{new}로 바뀌었기 때문에 E-step에서 구했던 p(Z)로 더 이상 KL-divergence가 0이 되지 않음
- 따라서 다시 E-step을 진행시켜 KL-divergence를 0으로 만들고, log-likelihood의 값을 M-step을 통해 키우는 과정을 계속 반복

05

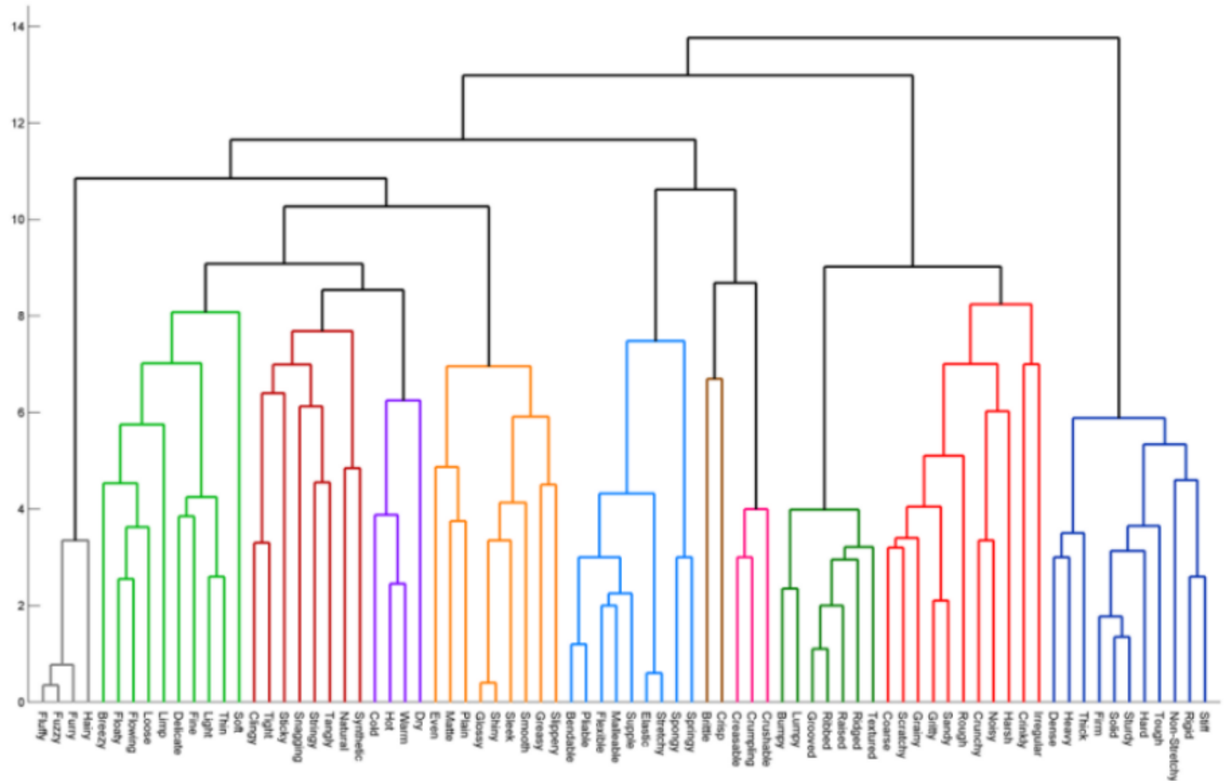
계층 클러스터링(Hierarchical Clustering)

계층 클러스터링

- 계층적 트리 모형을 이용해 개별 개체들을 순차적, 계층적으로 유사한 개체 내지 그룹과 통합하여 군집화를 수행하는 알고리즘
- 클러스터의 수를 사전에 정하지 않아도 학습 수행 가능

계층 클러스터링

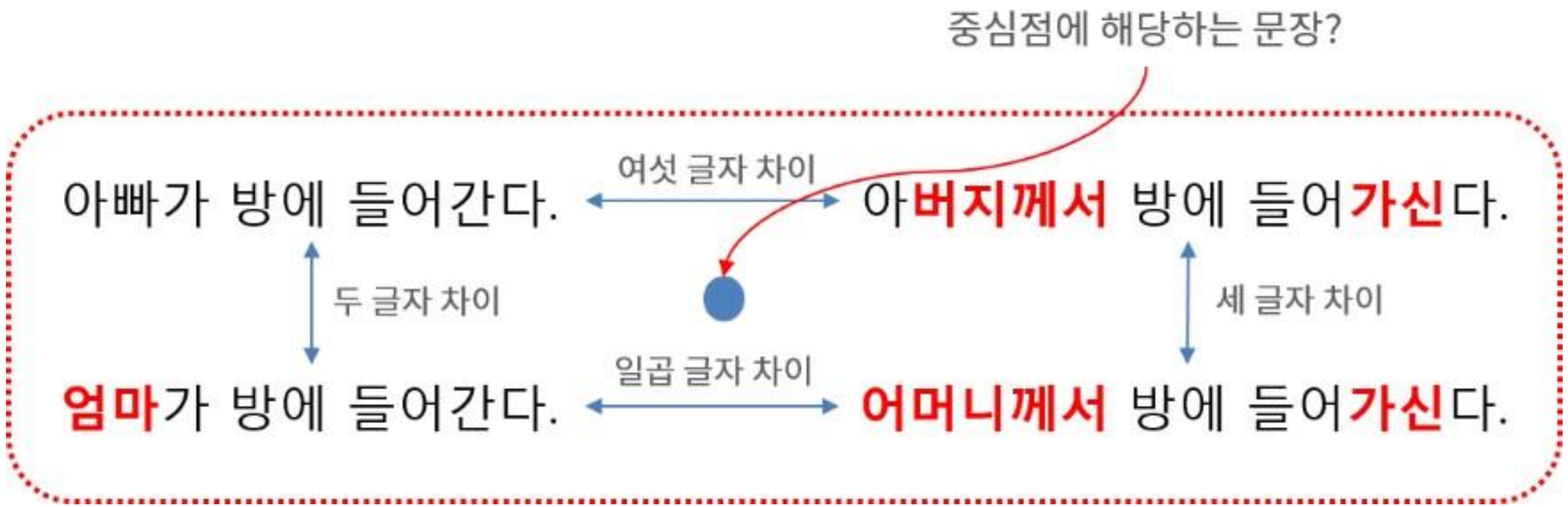
- 개체들이 결합되는 순서를 나타내는 트리형태의 구조인 덴드로그램(Dendrogram)을 생성한 후 적절한 수준에서 트리를 자르면 클러스터링이 완료됨.



/* elice */

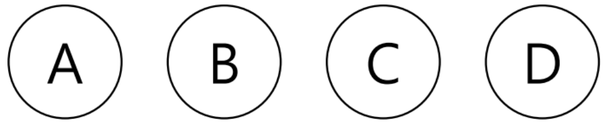
계층 클러스터링의 강점

- 데이터 각 쌍에 대해서는 유사도를 측정할 수 있지만 평균이나 분산 등을 구할 수 없는 경우에 사용할 수 있는 기법



학습 방법

- 계층 클러스터링을 수행하려면 모든 개체들 간 거리(distance)나 유사도(similarity)가 이미 계산되어 있어야 한다.

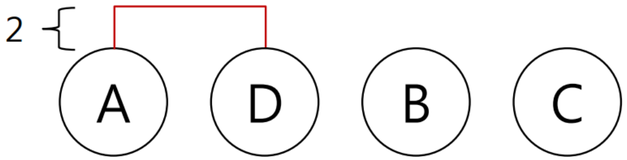


	A	B	C	D
A		20	7	2
B			10	25
C				3
D				

05 계층 클러스터링

학습 방법

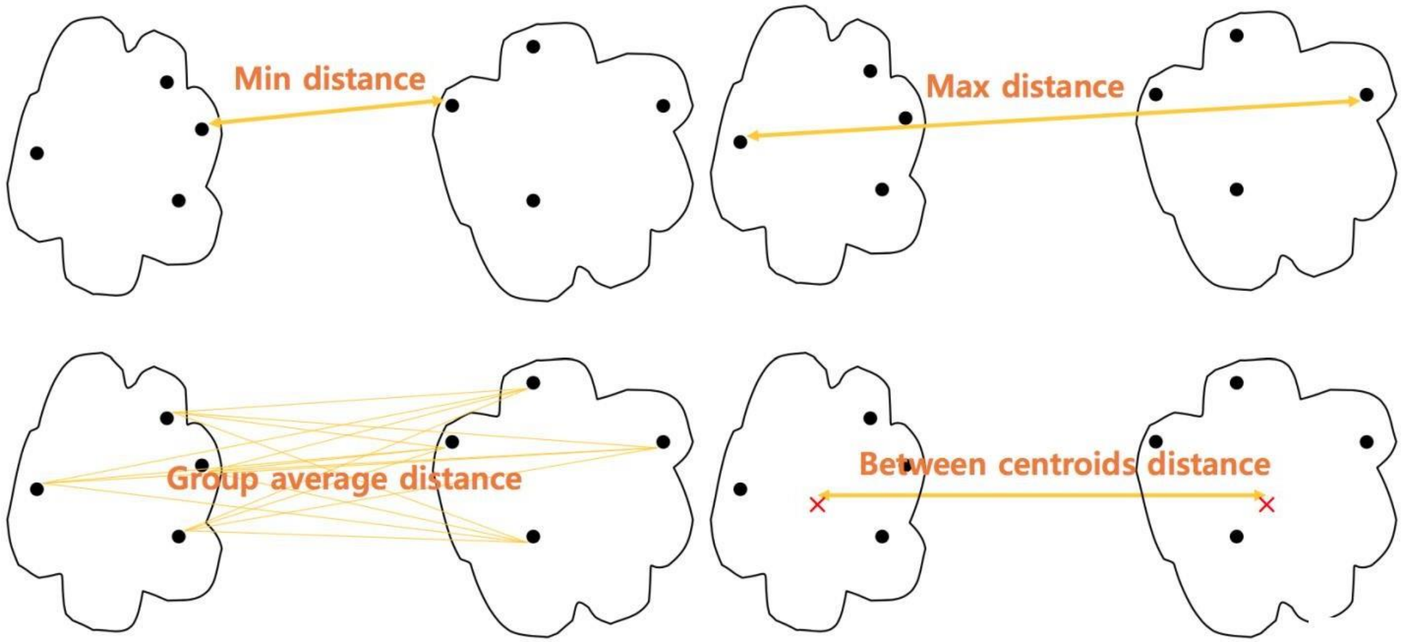
- 거리가 가장 짧은 것이 2이고 이에 해당하는 개체는 A와 D이므로 먼저 A와 D를 하나의 클러스터로 엮음
- 덴드로그램의 높이는 관측치간 거리(2)가 되도록 함



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

학습 방법

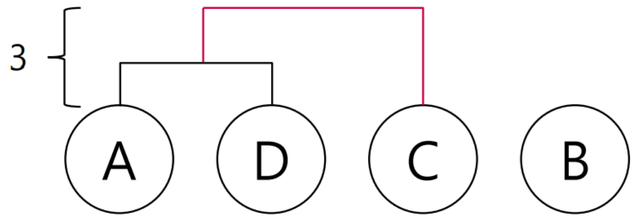
- 여기서 A와 D를 한 군집으로 엮었으니 거리행렬을 바꿔주어야 한다.
  - 즉 개체-개체 거리를 군집-개체 거리로 계산해야 함
- 군집-개체, 혹은 군집-군집 간 거리 계산 방법은 여러가지 선택지가 존재



05 계층 클러스터링

학습 방법

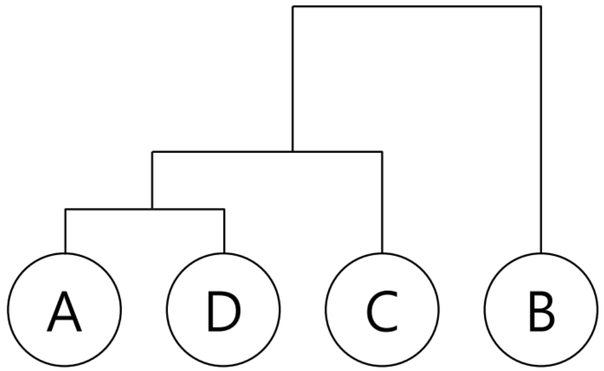
- 거리행렬 업데이트
- AD와 C가 가장 인접



	AD	B	C	
AD		20	3	
B			10	
C				

학습 방법

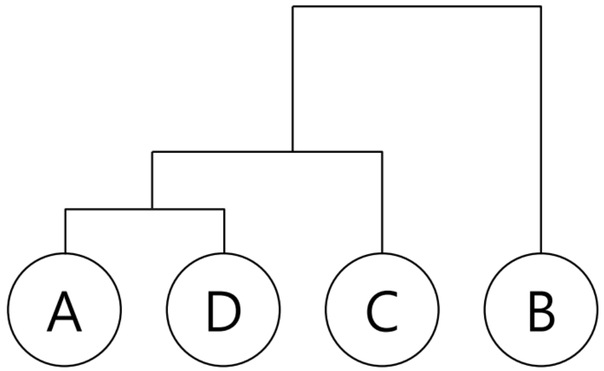
- 더 이상 연결할 노드가 없으면 학습 종료



	AD CB			
AD CB				

학습 방법

- 더 이상 연결할 노드가 없으면 학습 종료



	AD CB			
AD CB				

06

알고리즘 간 비교

06 알고리즘 간 비교

K-means vs GMM vs 계층 클러스터링

	사용되는 유사도 측정 방식	하이퍼 파라미터	장점	단점 (한계점)
k 평균	좌표 기반 거리 구하기 (유클리드, 마할라노비스, 맨하탄)	• 군집 개수 • 종료 조건	• 알고리즘이 쉽고 직관적임 • 분산 시스템을 이용한 대용량 데이터 처리 가능 • 각 유형의 특징 파악 용이	• 아웃라이어에 민감 • 유형별 데이터의 분산이 비슷하고 구형으로 분포되어 있지 않으면 결과가 좋지 못함
GMM	좌표 기반 거리 구하기 (유클리드, 마할라노비스, 맨하탄)	• 군집 개수 • 종료 조건	• 확률 분포의 차이를 고려하여 군집을 묶는 방식이기 때문에 k 평균 알고리즘에 비해 좀 더 통계적으로 엄밀한 결과를 얻을 수 있음	• 계산량이 많기 때문에 대량의 데이터에 사용하기 어려움 • 유형들의 분포가 정규 분포와 차이가 크다면 결과가 좋지 못함
계층 클러스터링	모든 유사도 측정 방식 사용 가능	• 군집 개수 • 군집 간의 거리 측정 방법	• 개체간의 거리는 구할 수는 있지만 군집의 평균을 구할 수 없는 데이터에 대해서도 적용 가능	• 계산량이 많기 때문에 대량의 데이터에 사용하기 어려움

Credit

/* elice */

코스 매니저

콘텐츠 제작자
정민수

강사
정민수

감수자

디자인

Contact

TEL
070-4633-2015

WEB
https://elice.io

E-MAIL
contact@elice.io