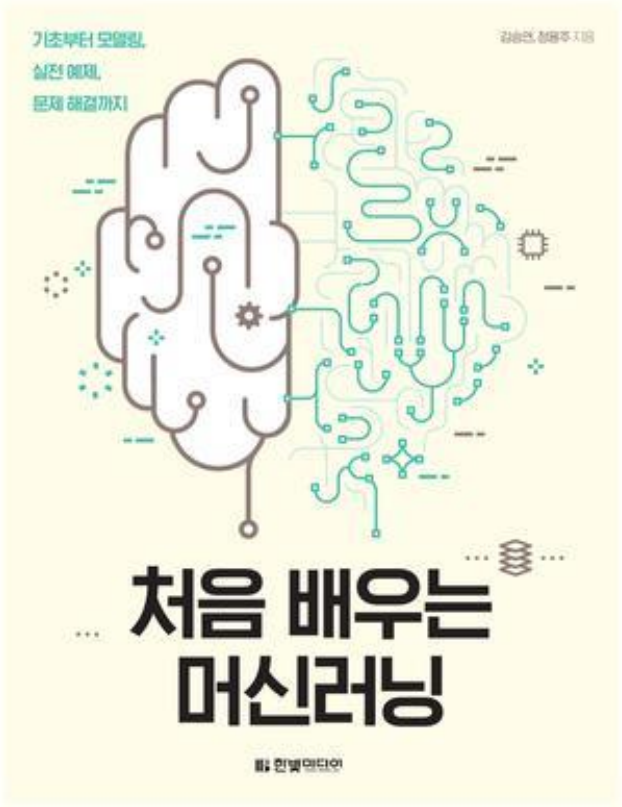
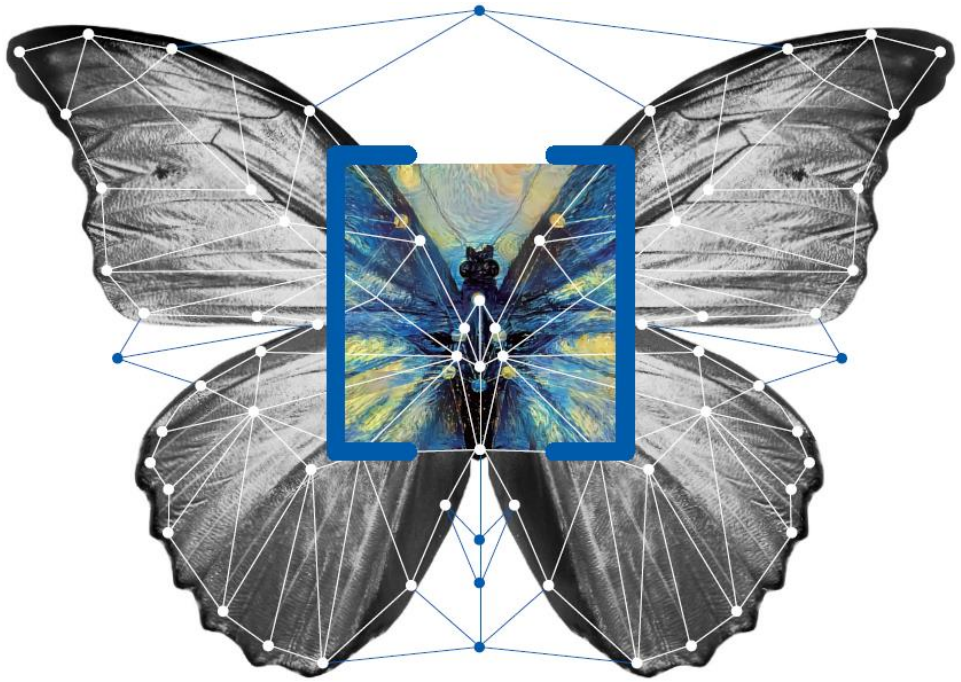




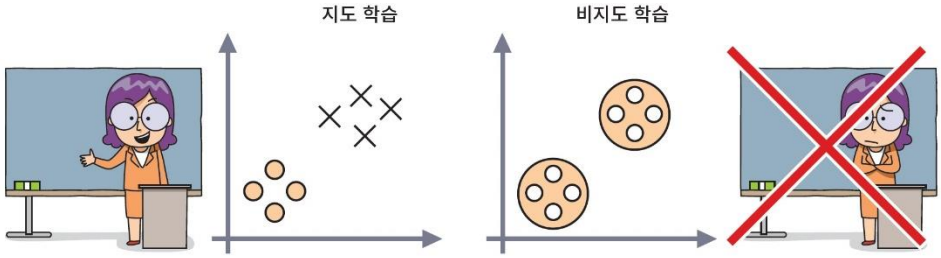
MACHINE LEARNING 기계 학습

군집화

PREVIEW

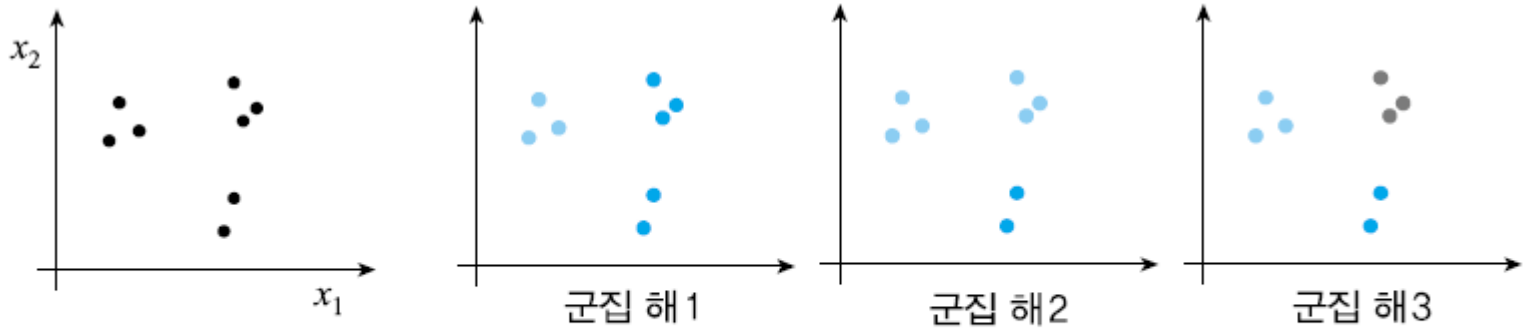
군집화(Clustering)

- 샘플은 부류 정보가 없다. ($X=\{\mathbf{x}_1, \mathbf{x}_2, \ldots, \mathbf{x}_N\}$으로 표기)
- 군집화는 비지도 학습에 해당
- 군집이 몇 개인지 모르는 경우도 많다.
- 군집화를 부류 발견 작업이라고도 부른다.



군집화의 특성

- 주관성: 군집화 결과의 품질은 응용이 처한 상황과 요구 사항에 따라 다름



(a) 샘플 집합 (b) 세 가지 군집화 결과를 어떻게 평가할까?

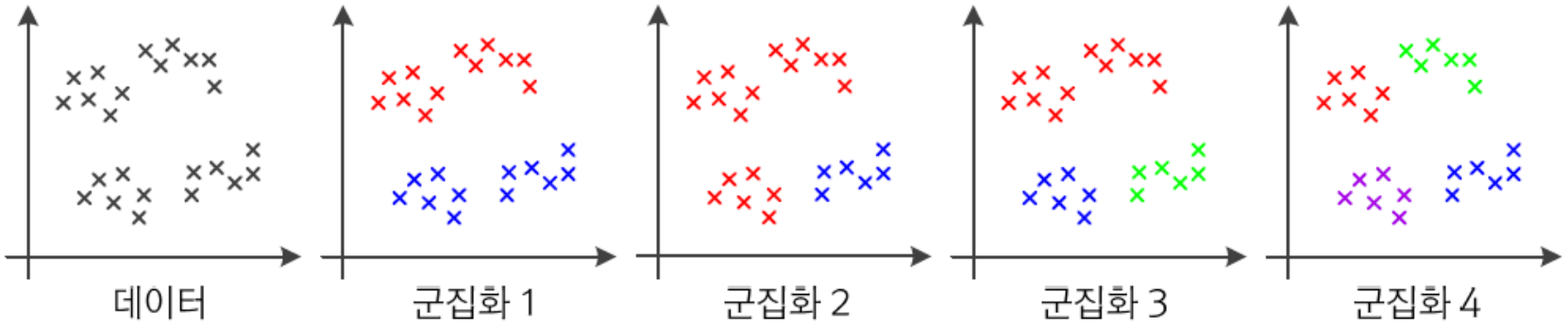
군집화의 주관성

군집화의 정의

- 군집화 문제
  - $\mathbb{X} = \{\mathbf{x}_1, \mathbf{x}_2, \cdots, \mathbf{x}_n\}$에서 다음 식을 만족하는 군집집합 $C = \{c_1, c_2, \cdots, c_k\}$를 찾아내는 작업

$$\left.\begin{array}{l} c_i \neq \emptyset, i = 1,2,\cdots,k \\ \bigcup_{i=1}^{k} c_i = \mathbb{X} \\ c_i \bigcap c_j = \emptyset, i \neq j \end{array}\right\}$$

  - 군집의 개수 k는 주어지는 경우와 자동으로 찾아야 하는 경우가 있음
  - 군집화를 부류 발견 작업이라 부르기도 함
- 군집화의 주관성



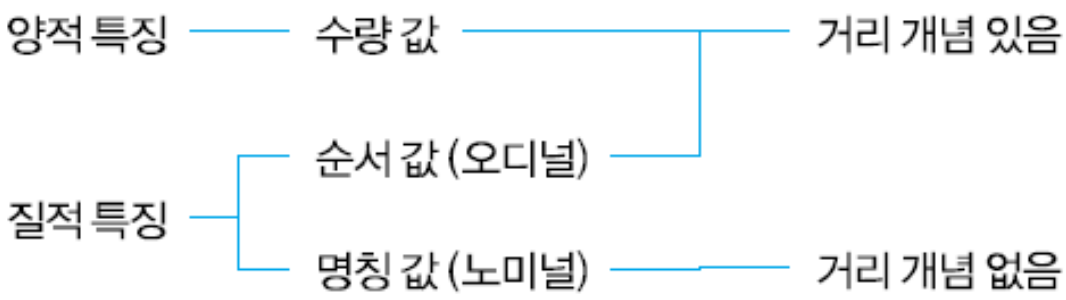
군집화의 주관성

거리와 유사도

군집화가 추구하는 본질적인 목표는 같은 군집 내의 샘플은 서로 가깝고 다른 군집에 속한 샘플 사이의 거리는 멀게 하는 것이다. 따라서 군집화에서 거리 개념은 매우 중요하고 여러 가지 계산 방법이 개발되어 있다. 어떤 계산 방법을 사용하느냐에 따라 군집화 결과는 달라지고 상황에 적합할 수도 있고 그렇지 않을 수도 있다. 따라서 주어진 문제에 적합한 거리 측정 방법을 선택하는 것이 매우 중요하다. 거리와 distance 유사도는 similarity 반대 개념이고 하나를 알면 다른 것은 공식을 이용하여 쉽게 계산할 수 있다.

특징 값의 종류

- 특징 값의 종류는 다양하다. (군집화를 공부하는데 이에 대한 이해가 필요하다.)



특징의 유형

고객 레코드 (샘플)을 12개의 특징으로 표현한다고 하자.

$\mathbf{x}$ = (나이, 직업, 연봉, 성별, 월평균 구매액, 반품 성향, 선호하는 물품 수준, 의류 선호도, 전자제품 선호도, 식품 선호도, 팬시 선호도, DVD 선호도)^T

나이: [1,100]의 정수

직업: [1,10]의 정수 (1 = 회사원, 2 = CEO, 3 = 교사, …)

연봉: [1,5]의 정수 (1 = 2천만원 미만, 2 = 2천~3천만원, …, 5 = 1억 이상)

성별: [0,1]의 정수 (0 = 여자, 1 = 남자)

월평균 구매액: 평균을 계산하여 실수로 표현

반품 성향: [1,4]의 정수 (1 = 년간 반품 횟수 0, 2 = 2회 미만, …)

선호하는 물품 수준: [1,4]의 정수 (1 = 저가, 2 = 보통, 3 = 고가, 4 = 명품)

제품 선호도: [1,5]의 정수 (1 = 구매한 적 없음, …, 5 = 아주 선호)

수량 값을 갖는 특징: 나이, 연봉, 월평균 구매액

순서 값을 갖는 특징: 반품 성향, 선호하는 물품 수준, 제품 선호도

명칭 값을 갖는 특징: 직업, 성별

거리와 유사도 측정

- Minkowski 거리 (10.4)
  - 두 점 $\mathbf{x}_i=(x_{i1},\ldots,x_{id})^T$와 $\mathbf{x}_j=(x_{j1},\ldots,x_{jd})^T$간의 거리 척도
  - p=2면 유클리디언 거리 , p=1이면 도시블록 거리

$$d_{ij}=\left(\sum_{k=1}^{d}\left|x_{ik}-x_{jk}\right|^{p}\right)^{1/p}$$

$$\text{유클리디언 거리 } (p=2) \quad d_{ij}=\sqrt{\sum_{k=1}^{d}(x_{ik}-x_{jk})^2}$$

$$\text{맨하탄 거리 } (p=1) \quad d_{ij}=\sum_{k=1}^{d}\left|x_{ik}-x_{jk}\right|$$

- Hamming 거리
  - 이진 특징 벡터에 적용 가능 (서로 다른 비트의 개수)
  - 예) $(1,0,1,0,0,0,1,1)^T$과 $(1,0,0,1,0,0,1,0)^T$의 해밍 거리는 3

6.3.1 k-평균 알고리즘

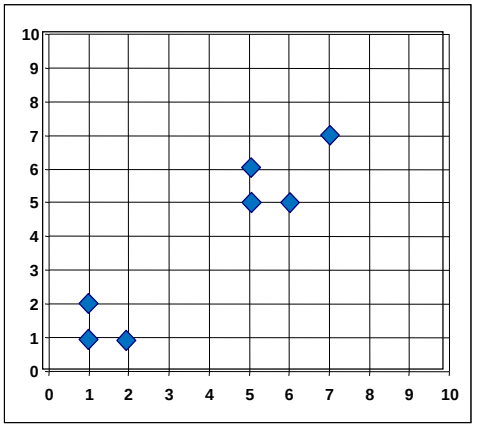
- k-평균 알고리즘의 특성
  - 원리 단순하지만 성능이 좋아 인기 좋음
  - 직관적으로 이해하기 쉽고 구현 쉬움
  - 군집 개수 k를 알려줘야 함

알고리즘 6–1 k-평균

입력: 훈련집합 $\mathbb{X} = \{\mathbf{x}_1, \mathbf{x}_2, \cdots, \mathbf{x}_n\}$, 군집의 개수 k
출력: 군집집합 $C = \{c_1, c_2, \cdots, c_k\}$

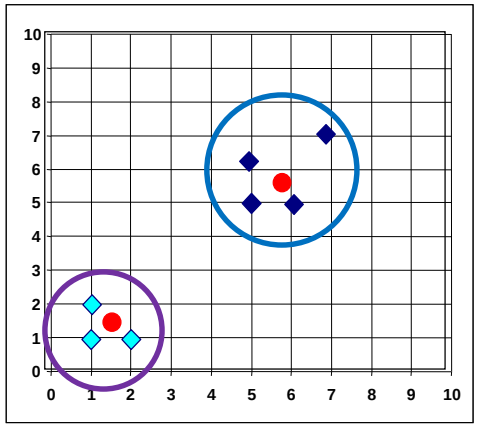
```
1  k개의 군집 중심 Z = {z_1, z_2, ..., z_k}를 초기화한다.
2  while (true)
3     for (i=1 to n)
4           x_i를 가장 가까운 군집 중심에 배정한다.
5     if (라인 3~4에서 이루어진 배정이 이전 루프에서의 배정과 같으면) break
6     for (j=1 to k)
7           z_j에 배정된 샘플의 평균으로 z_j를 대치한다.
8  for (j=1 to k)
9     z_j에 배정된 샘플을 c_j에 대입한다.
```

예제) K-평균 알고리즘

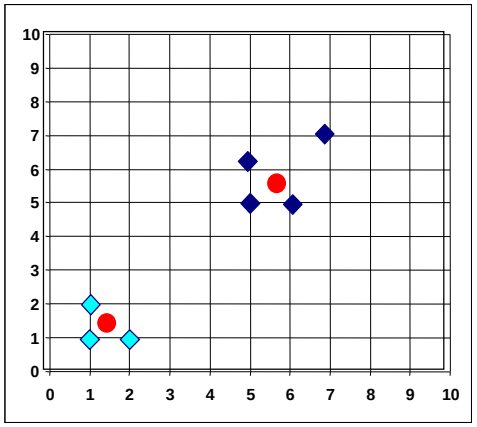


K=2

Arbitrarily choose K object as initial cluster center

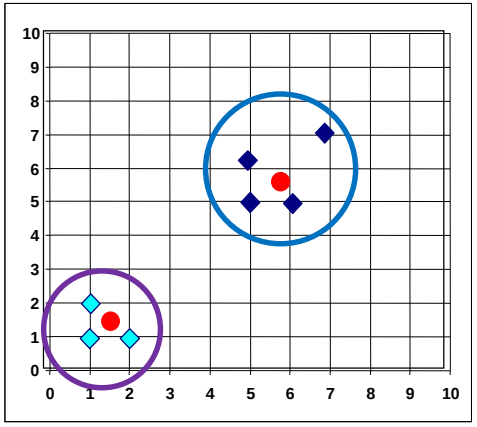


Assign each objects to most similar center



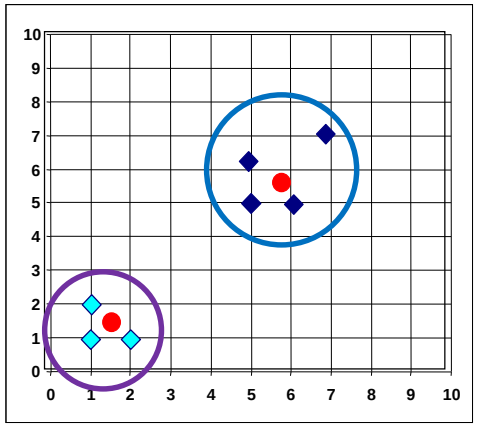
Update the cluster means

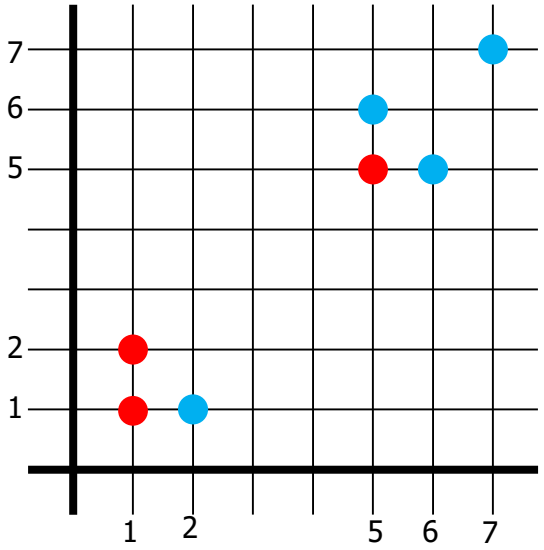
Reassign each object



Update the cluster means

reassign





(1, 1)(2, 1)(1, 2)(5, 5)(6, 5)(5, 6)(7, 7)

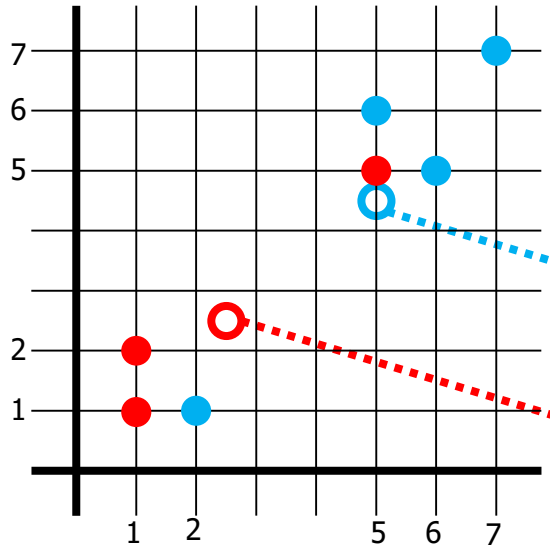
Class A

(1, 1)(1, 2)(5, 5)

Class B

(2, 1)(6, 5)(5, 6)(7, 7)

Assume we randomly partitioned the objects into 2 nonempty subsets as above!



(1, 1)(2, 1)(1, 2)(5, 5)(6, 5)(5, 6)(7, 7)

Class A

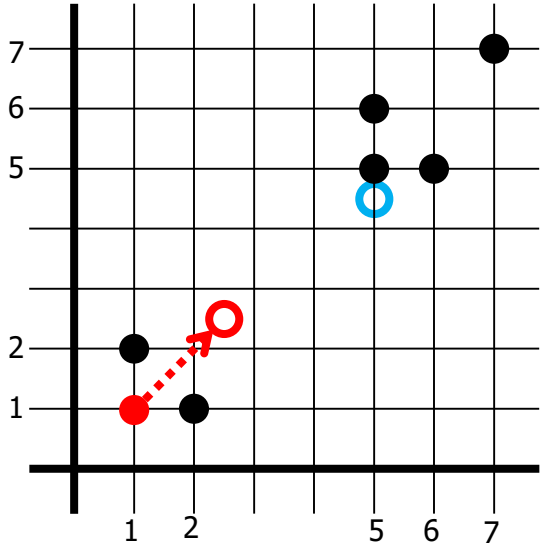
(1, 1)(1, 2)(5, 5)

Class B

(2, 1)(6, 5)(5, 6)(7, 7)

Center : (7/3, 8/3)

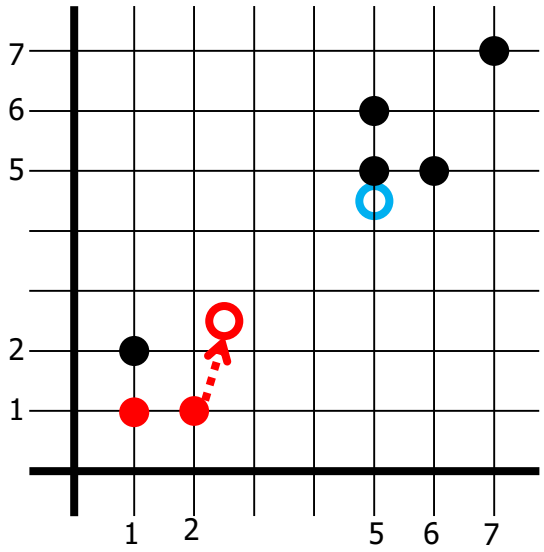
Center : (20/4, 19/4)



(1, 1)(2, 1)(1, 2)(5, 5)(6, 5)(5, 6)(7, 7)

Class A	Class B
(1, 1)	

Center : (7/3, 8/3) Center : (20/4, 19/4)

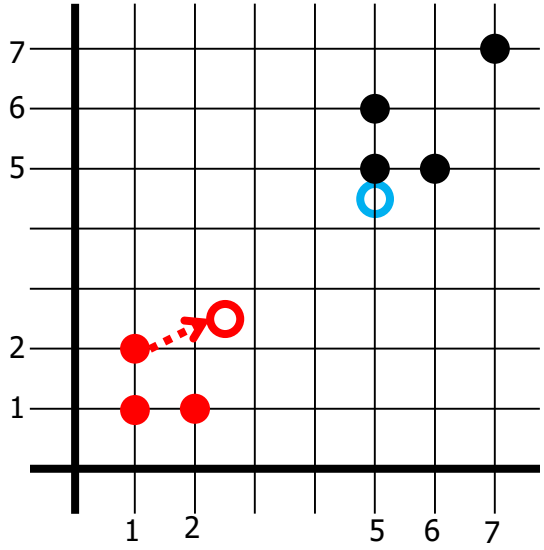


(1, 1) *(2, 1)* (1, 2) (5, 5) (6, 5) (5, 6) (7, 7)

Class A	Class B
(1, 1)(2, 1)	

Center : (7/3, 8/3)

Center : (20/4, 19/4)



(1, 1)(2, 1)*(1, 2)*(5, 5)(6, 5)(5, 6)(7, 7)

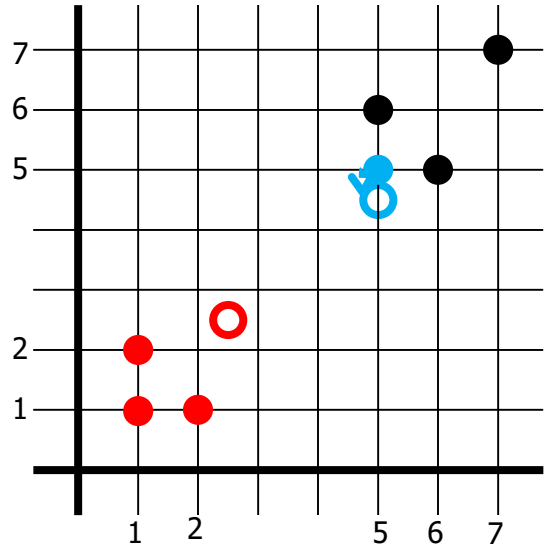
Class A

(1, 1)(2, 1)(1, 2)

Class B

Center : (7/3, 8/3)

Center : (20/4, 19/4)

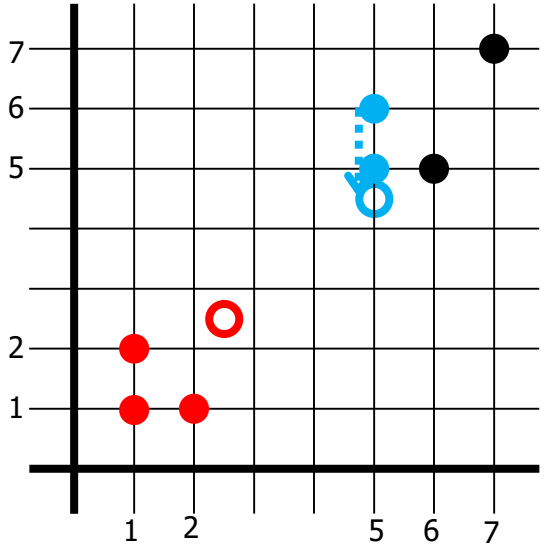


(1, 1) (2, 1) (1, 2) *(5, 5)* (6, 5) (5, 6) (7, 7)

Class A	Class B
(1, 1) (2, 1) (1, 2)	(5, 5)

Center : (7/3, 8/3)

Center : (20/4, 19/4)

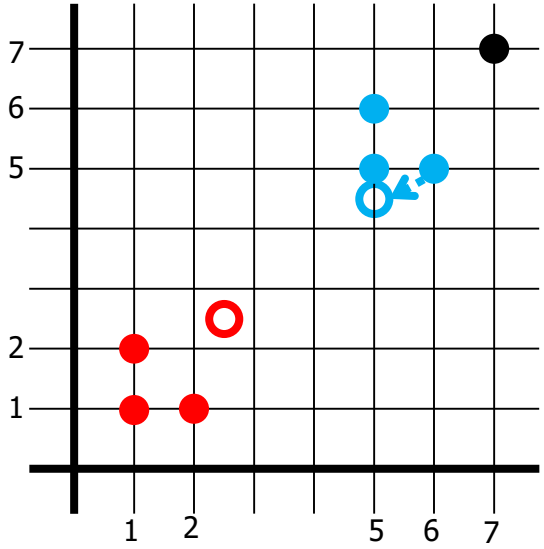


(1, 1) (2, 1) (1, 2) (5, 5) *(6, 5)* (5, 6) (7, 7)

Class A	Class B
(1, 1) (2, 1) (1, 2)	(5, 5) (6, 5)

Center : (7/3, 8/3)

Center : (20/4, 19/4)

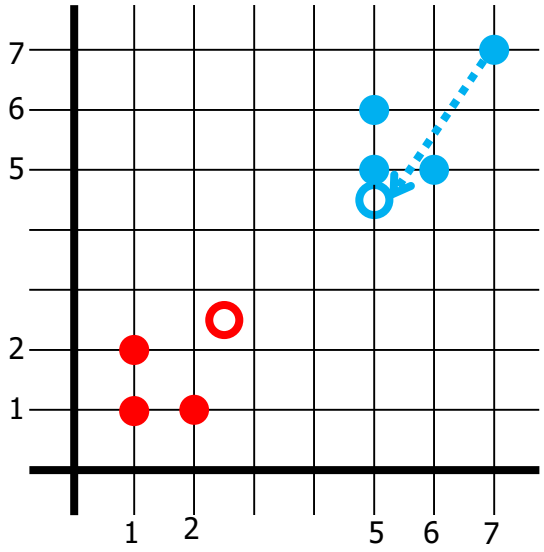


(1, 1) (2, 1) (1, 2) (5, 5) (6, 5) *(5, 6)* (7, 7)

Class A	Class B
(1, 1) (2, 1) (1, 2)	(5, 5) (6, 5) (5, 6)

Center : (7/3, 8/3)

Center : (20/4, 19/4)

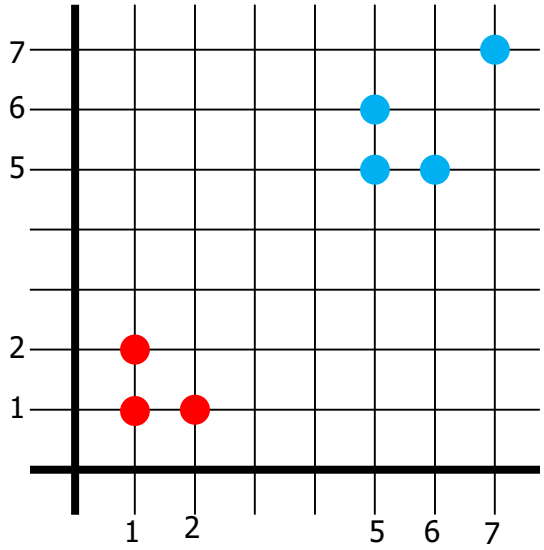


(1, 1) (2, 1) (1, 2) (5, 5) (6, 5) (5, 6) *(7, 7)*

Class A	Class B
(1, 1) (2, 1) (1, 2)	(5, 5) (6, 5) (5, 6) (7, 7)

Center : (7/3, 8/3)

Center : (20/4, 19/4)



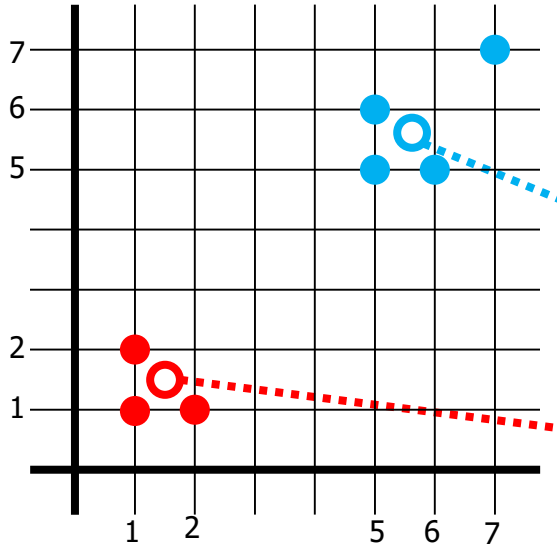
(1, 1)(2, 1)(1, 2)(5, 5)(6, 5)(5, 6)(7, 7)

Class A

(1, 1)(2, 1)(1, 2)

Class B

(5, 5)(6, 5)(5, 6)(7, 7)



(1, 1)(2, 1)(1, 2)(5, 5)(6, 5)(5, 6)(7, 7)

Class A

(1, 1)(2, 1)(1, 2)

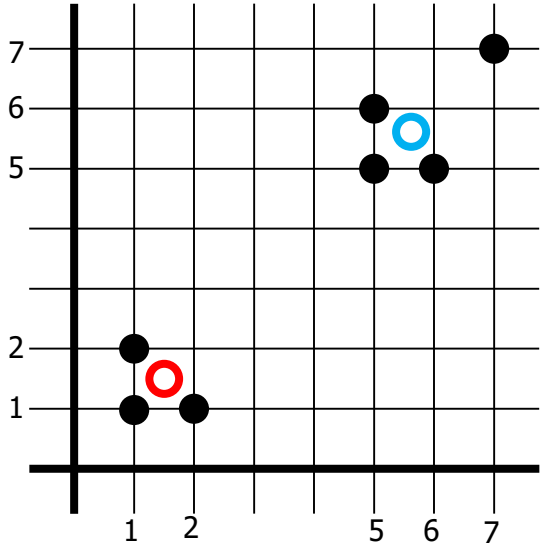
Class B

(5, 5)(6, 5)(5, 6)(7, 7)

Center : (4/3, 4/3)

Center : (23/4, 23/4)

Update the cluster means

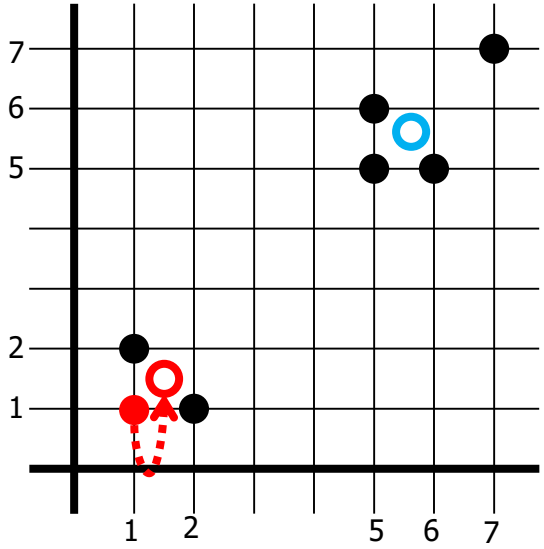


(1, 1) (2, 1) (1, 2) (5, 5) (6, 5) (5, 6) (7, 7)

Class A	Class B
(1, 1)	

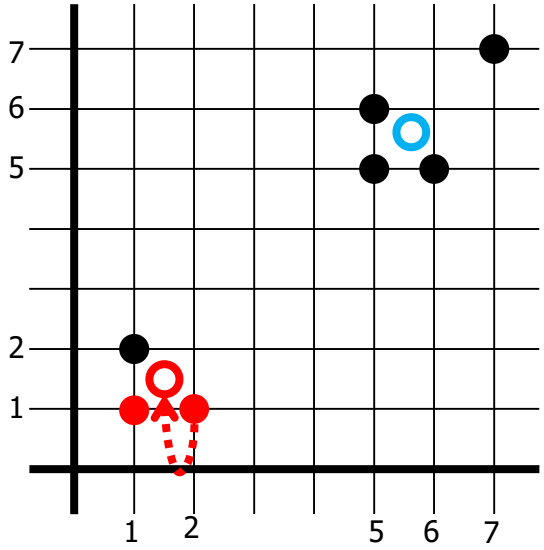
Center : (4/3, 4/3)

Center : (23/4, 23/4)



(1, 1)(2, 1)(1, 2)(5, 5)(6, 5)(5, 6)(7, 7)

Class A	Class B
(1, 1)	
Center : (4/3, 4/3)	Center : (23/4, 23/4)

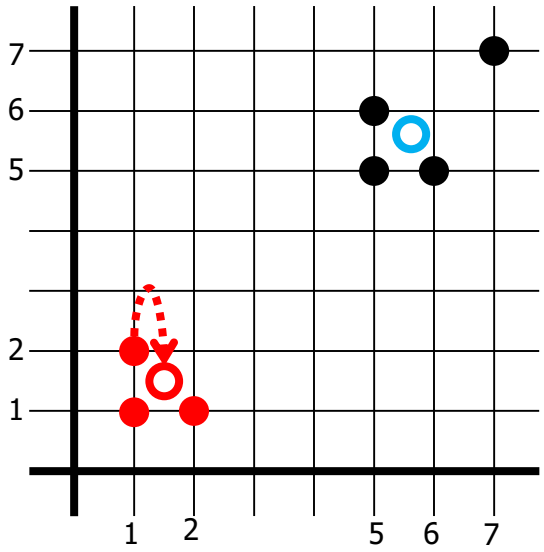


(1, 1) *(2, 1)* (1, 2) (5, 5) (6, 5) (5, 6) (7, 7)

Class A	Class B
(1, 1)(2, 1)	

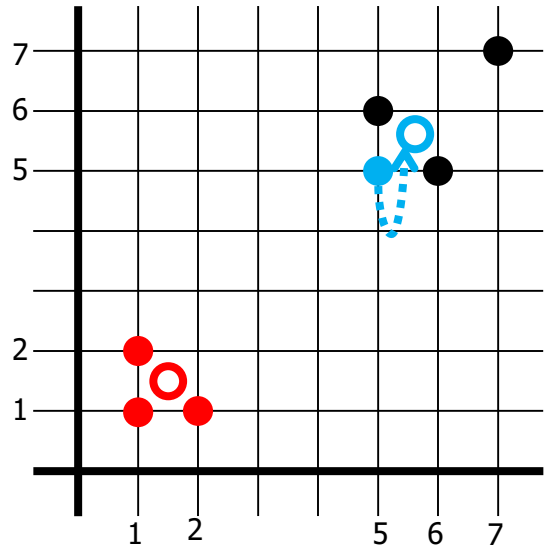
Center : (4/3, 4/3)

Center : (23/4, 23/4)



(1, 1)(2, 1)*(1, 2)*(5, 5)(6, 5)(5, 6)(7, 7)

Class A	Class B
(1, 1)(2, 1)(1, 2)	
Center : (4/3, 4/3)	Center : (23/4, 23/4)

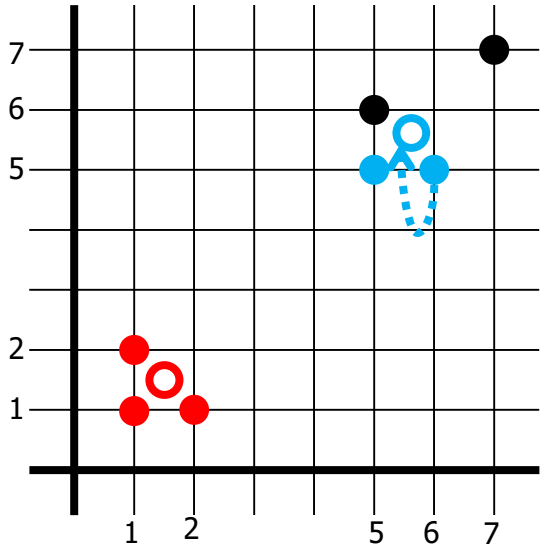


(1, 1) (2, 1) (1, 2) *(5, 5)* (6, 5) (5, 6) (7, 7)

Class A	Class B
(1, 1) (2, 1) (1, 2)	(5, 5)

Center : (4/3, 4/3)

Center : (23/4, 23/4)

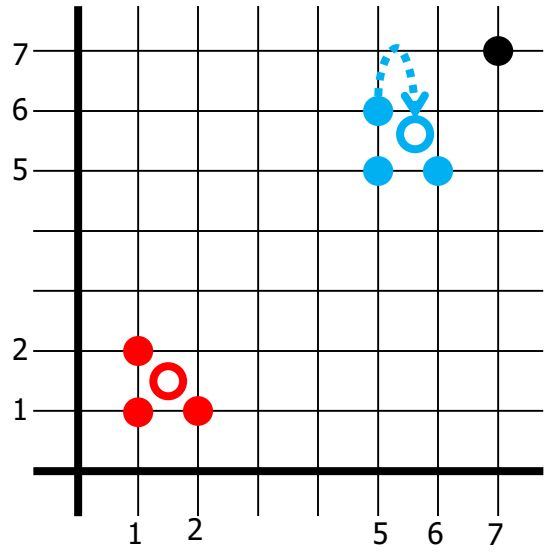


(1, 1) (2, 1) (1, 2) (5, 5) *(6, 5)* (5, 6) (7, 7)

Class A	Class B
(1, 1) (2, 1) (1, 2)	(5, 5) (6, 5)

Center : (4/3, 4/3)

Center : (23/4, 23/4)

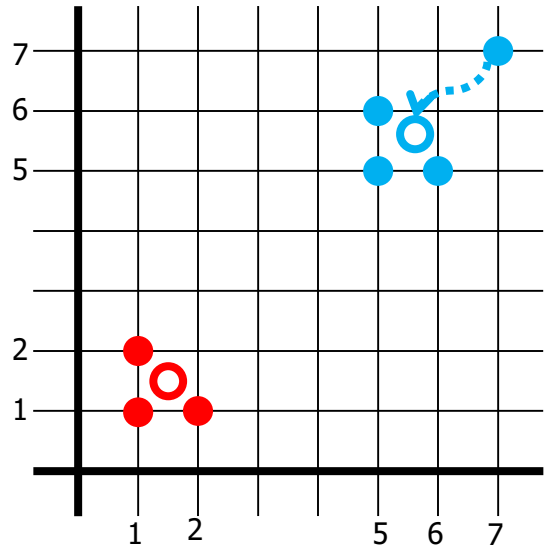


(1, 1)(2, 1)(1, 2)(5, 5)(6, 5)*(5, 6)*(7, 7)

Class A	Class B
(1, 1)(2, 1)(1, 2)	(5, 5)(6, 5)(5, 6)

Center : (4/3, 4/3)

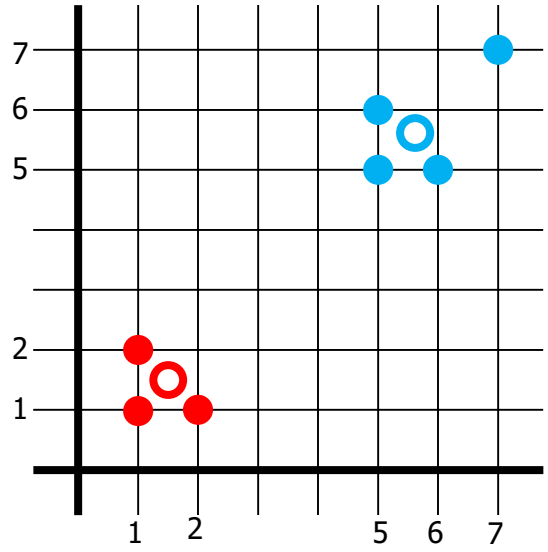
Center : (23/4, 23/4)



(1, 1)(2, 1)(1, 2)(5, 5)(6, 5)(5, 6) *(7, 7)*

Class A	Class B
(1, 1)(2, 1)(1, 2)	(5, 5)(6, 5)(5, 6)(7, 7)

Center : (4/3, 4/3) Center : (23/4, 23/4)



(1, 1)(2, 1)(1, 2)(5, 5)(6, 5)(5, 6)(7, 7)

Class A	Class B
(1, 1)(2, 1)(1, 2)	(5, 5)(6, 5)(5, 6)(7, 7)

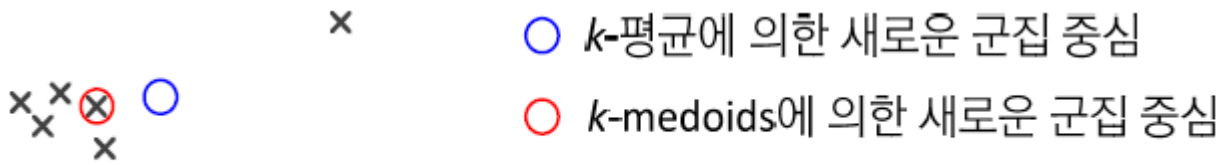
Center : (4/3, 4/3) Center : (23/4, 23/4)

No change ⇒ STOP

K-평균 알고리즘

k-평균과 *k*-medoids

- *k*-평균은 [알고리즘 6-1]의 라인 7에서 샘플의 평균으로 군집 중심을 갱신
- *k*-medoids는 대표를 뽑아 뽑힌 대표로 군집 중심을 갱신(*k*-평균에 비해 잡음에 둔감)



***k*-평균과 *k*-medoids가 군집 중심을 갱신하는 과정**

최적화 문제로 해석

- *k*-평균은 다음 식의 목적함수를 최소화하는 알고리즘 (Z는 군집 중심)
- 행렬 **A**는 군집 배정 정보를 나타내는 $k*n$ 행렬(i번째 샘플이 j번째 군집에 배정되었다면 a_{ji}는 1, 그렇지 않으면 0)
- Dist(X, Z)는 x와 z 사이의 거리를 측정하는 함수

$$J(Z,\mathbf{A}) = \sum_{i=1}^{n}\sum_{j=1}^{k} a_{ji}\,dist(\mathbf{x}_i,\mathbf{z}_j) \qquad (6.2)$$

k-평균 알고리즘

예제 6-1 *k*-평균의 동작

[그림 6-5]는 훈련집합이 7개의 샘플을 가진 n=7인 예를 보여 준다. 좌표는 다음과 같다.

$$\mathbf{x}_1 = \begin{pmatrix}18\\5\end{pmatrix}, \mathbf{x}_2 = \begin{pmatrix}20\\9\end{pmatrix}, \mathbf{x}_3 = \begin{pmatrix}20\\14\end{pmatrix}, \mathbf{x}_4 = \begin{pmatrix}20\\17\end{pmatrix}, \mathbf{x}_5 = \begin{pmatrix}5\\15\end{pmatrix}, \mathbf{x}_6 = \begin{pmatrix}9\\15\end{pmatrix}, \mathbf{x}_7 = \begin{pmatrix}6\\20\end{pmatrix}$$

군집의 개수 k=3이라 하자. 맨 왼쪽 그림은 초기 군집 중심을 보여 준다. [알고리즘 6-1]의 라인 3~4는 7개 샘플을 아래와 같이 배정할 것이다.

$$\{\mathbf{x}_1\}\text{은 } \mathbf{z}_1,\ \{\mathbf{x}_2\}\text{은 } \mathbf{z}_2,\ \{\mathbf{x}_3, \mathbf{x}_4, \mathbf{x}_5, \mathbf{x}_6, \mathbf{x}_7\}\text{은 } \mathbf{z}_3$$

이 배정을 행렬 **A**로 표현하면 다음과 같다.

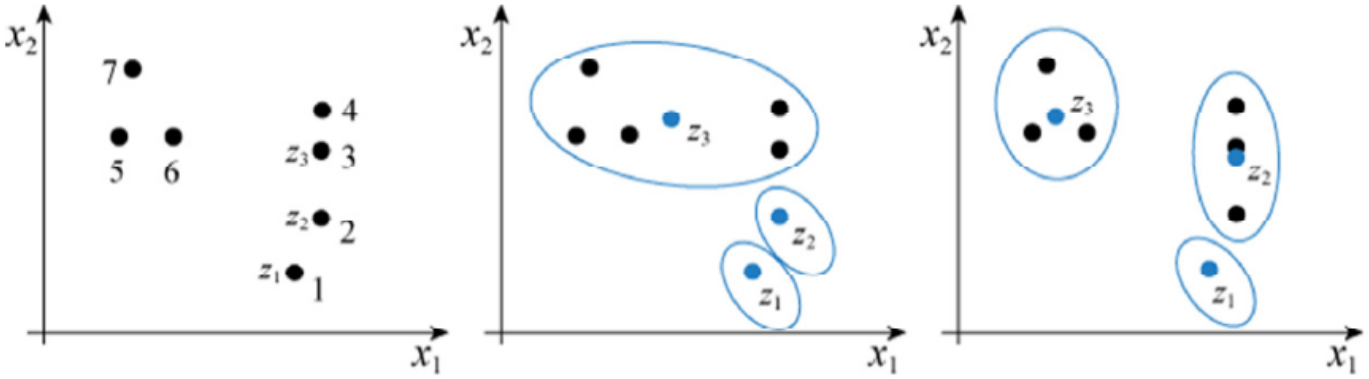
$$\mathbf{A} = \begin{pmatrix}1 & 0 & 0 & 0 & 0 & 0 & 0\\0 & 1 & 0 & 0 & 0 & 0 & 0\\0 & 0 & 1 & 1 & 1 & 1 & 1\end{pmatrix}$$

k-평균 알고리즘

[그림 6-5]의 가운데 그림은 새로 계산한 군집 중심이다. $\mathbf{z}_1 = (18, 5)^{\mathrm{T}}$, $\mathbf{z}_2 = (20, 9)^{\mathrm{T}}$, $\mathbf{z}_3 = (12, 16.2)^{\mathrm{T}}$ 이고, 식 (6.2)에 대입하면 $J = 244.8$이 된다. 이때 거리함수 dist로 식 (1.7)의 유클리디언 거리를 사용한다.

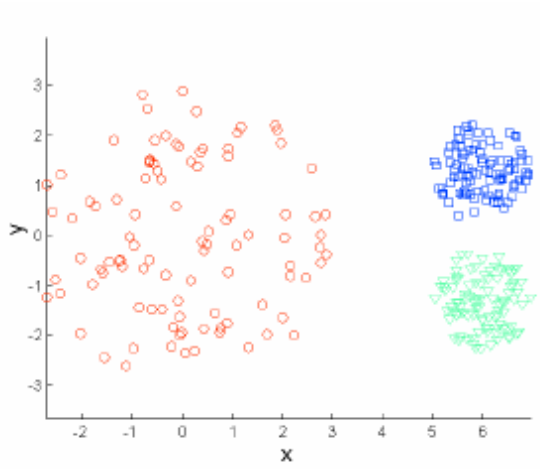
두 번째 루프를 실행하면 행렬 **A**는 아래와 같이 바뀐다. 군집 중심은 $\mathbf{z}_1 = (18, 5)^{\mathrm{T}}$, $\mathbf{z}_2 = (20, 13.333)^{\mathrm{T}}$, $\mathbf{z}_3 = (6.667, 16.667)^{\mathrm{T}}$이다. 이것을 식 (6.2)에 대입하면 $J = 58.00$이 된다. [그림 6-5]의 맨 오른쪽 그림은 두 번째 루프 수행 후의 상황이다.

$$\mathbf{A} = \begin{pmatrix} 1 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 1 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 1 & 1 \end{pmatrix}$$

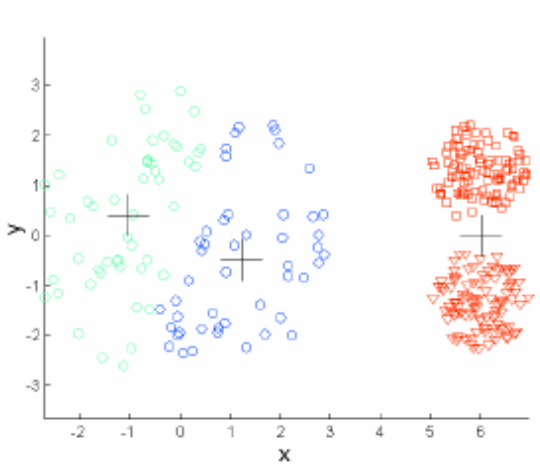


k-평균의 동작 예제

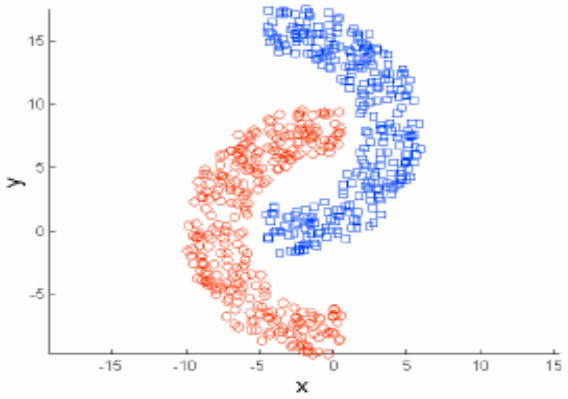
K-평균 알고리즘의 문제점



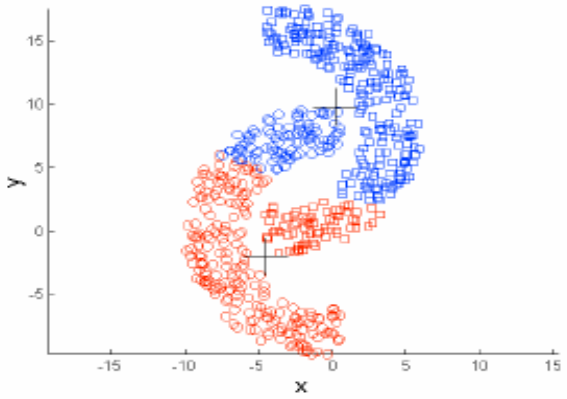
Original Points



K-means (3 Clusters)



Original Points



K-means (2 Clusters)

계층(Hierarchical) 접근 방법

- 샘플 각각이 군집이 되어 출발, 유사한 군집을 모으는 작업을 반복
  - 출력은 군집화 결과를 트리로 표현한 덴드로그램

알고리즘 [10.1] 응집 계층 군집화

입력: 샘플 집합 $X = \{\mathbf{x}_1, \mathbf{x}_2, \cdots, \mathbf{x}_N\}$

출력: 덴드로그램

알고리즘:

1. $C_0 = \{c_1 = \{\mathbf{x}_1\}, c_2 = \{\mathbf{x}_2\}, \cdots, c_N = \{\mathbf{x}_N\}\}$; // 각 샘플이 하나의 군집
2. **for** $(t = 1$ **to** $N-1)$ {
3. C_{t-1}의 모든 군집 쌍 (c_i, c_j)를 조사하여 아래를 만족하는 쌍 (c_p, c_q)를 찾아라.

$$D(c_p, c_q) = \min_{c_i, c_j \in C_{t-1}} D(c_i, c_j)$$ // 가장 가까운 쌍을 찾는 조건

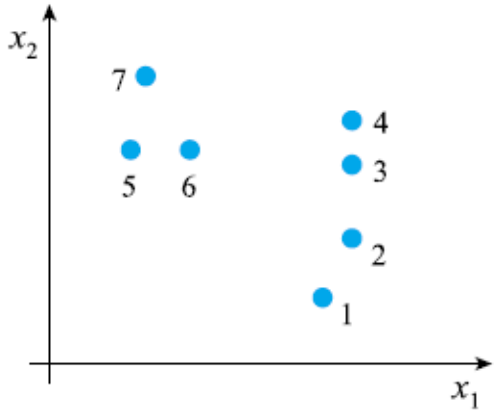
4. $c_r = c_p \cup c_q$; // 두 군집을 하나로 합쳐라.
5. $C_t = (C_{t-1} - c_p - c_q) \cup c_r$; // 두 군집을 제거하고 새로운 군집을 추가하라.
6. }

응집 계층 군집화

일곱 개의 샘플이 주어진 상황

$\mathbf{x}_1 = (18,5)^T,\ \mathbf{x}_2 = (20,9)^T,\ \mathbf{x}_3 = (20,14)^T,\ \mathbf{x}_4 = (20,17)^T,\ \mathbf{x}_5 = (5,15)^T,\ \mathbf{x}_6 = (9,15)^T,$

$\mathbf{x}_7 = (6,20)^T$



군집화 예제

(라인 1의) 초기화에 의해,

$$C_0 = \{c_1 = \{\mathbf{x}_1\},\ c_2 = \{\mathbf{x}_2\},\ c_3 = \{\mathbf{x}_3\},\ c_4 = \{\mathbf{x}_4\},\ c_5 = \{\mathbf{x}_5\},\ c_6 = \{\mathbf{x}_6\},\ c_7 = \{\mathbf{x}_7\}\}$$

루프를 반복하면, (거리 척도는 유클리언 거리와 $D_{\min}$을 가정)

$$C_1 = \{c_1 = \{\mathbf{x}_1\},\ c_2 = \{\mathbf{x}_2\},\ c_3 = \{\mathbf{x}_3,\mathbf{x}_4\},\ c_4 = \{\mathbf{x}_5\},\ c_5 = \{\mathbf{x}_6\},\ c_6 = \{\mathbf{x}_7\}\}$$

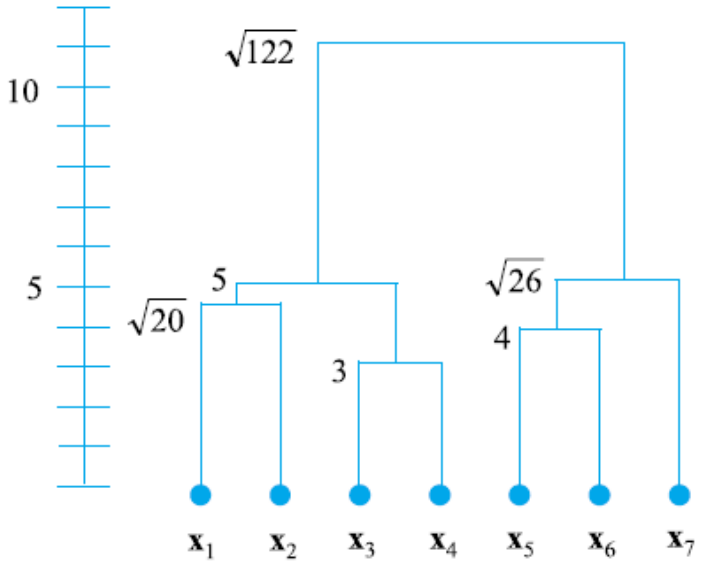
$$C_2 = \{c_1 = \{\mathbf{x}_1\},\ c_2 = \{\mathbf{x}_2\},\ c_3 = \{\mathbf{x}_3,\mathbf{x}_4\},\ c_4 = \{\mathbf{x}_5,\mathbf{x}_6\},\ c_5 = \{\mathbf{x}_7\}\}$$

$$C_3 = \{c_1 = \{\mathbf{x}_1,\mathbf{x}_2\},\ c_2 = \{\mathbf{x}_3,\mathbf{x}_4\},\ c_3 = \{\mathbf{x}_5,\mathbf{x}_6\},\ c_4 = \{\mathbf{x}_7\}\}$$

$$C_4 = \{c_1 = \{\mathbf{x}_1,\mathbf{x}_2,\mathbf{x}_3,\mathbf{x}_4\},\ c_2 = \{\mathbf{x}_5,\mathbf{x}_6\},\ c_3 = \{\mathbf{x}_7\}\}$$

$$C_5 = \{c_1 = \{\mathbf{x}_1,\mathbf{x}_2,\mathbf{x}_3,\mathbf{x}_4\},\ c_2 = \{\mathbf{x}_5,\mathbf{x}_6,\mathbf{x}_7\}\}$$

$$C_6 = \{c_1 = \{\mathbf{x}_1,\mathbf{x}_2,\mathbf{x}_3,\mathbf{x}_4,\mathbf{x}_5,\mathbf{x}_6,\mathbf{x}_7\}\}$$

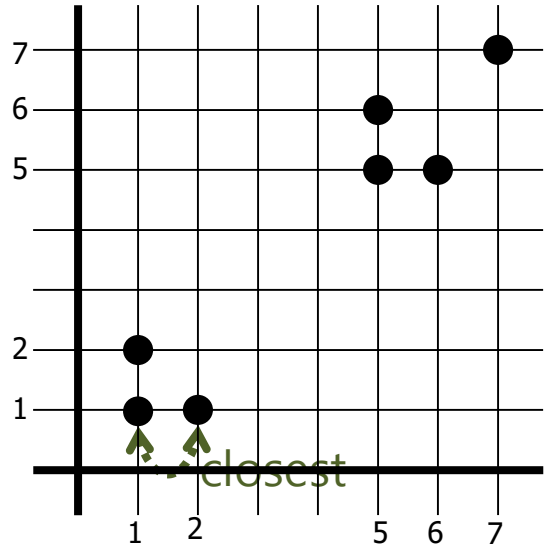


단일 연결 알고리즘이 만드는 덴드로그램

연필을 들고 직접 계산해 보자.
직접 해보는 것의 힘은 생각보다 크다.

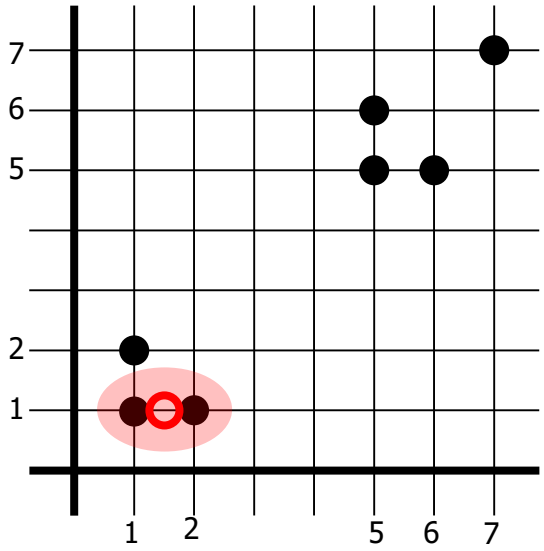
$D_{\min}$대신 $D_{\max}$를 사용하면 어떻게 될까?

An Example of Centroid-based Clustering Algorithm



(1, 1) (2, 1) (1, 2) (5, 5) (6, 5) (5, 6) (7, 7)

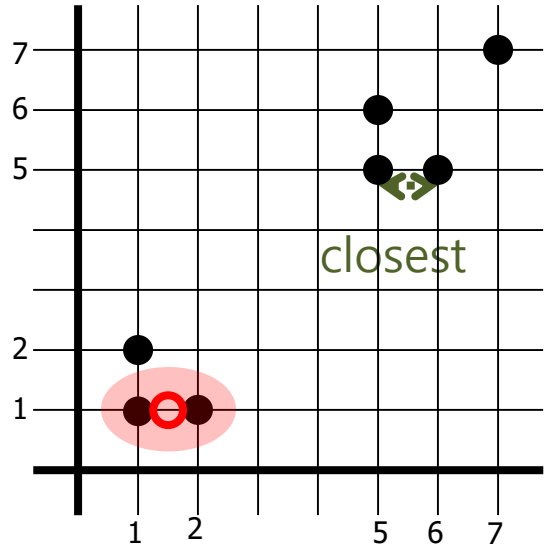
An Example of Centroid-based Clustering Algorithm



(1, 1) (2, 1) (1, 2) (5, 5) (6, 5) (5, 6) (7, 7)

Center : (3/2, 1)

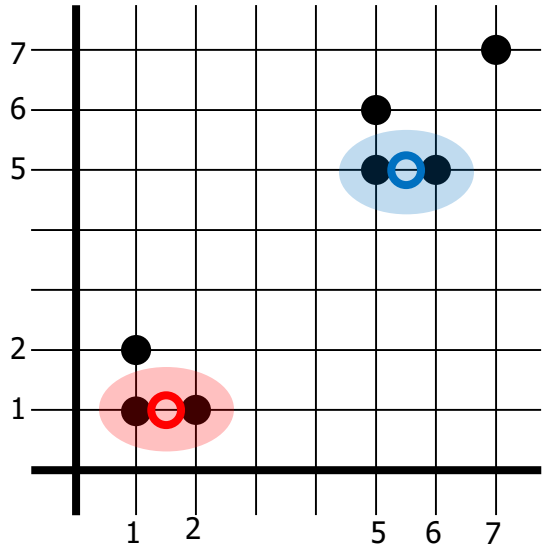
An Example of Centroid-based Clustering Algorithm



(1, 1) (2, 1) (1, 2) (5, 5) (6, 5) (5, 6) (7, 7)

Center : (3/2, 1)

An Example of Centroid-based Clustering Algorithm

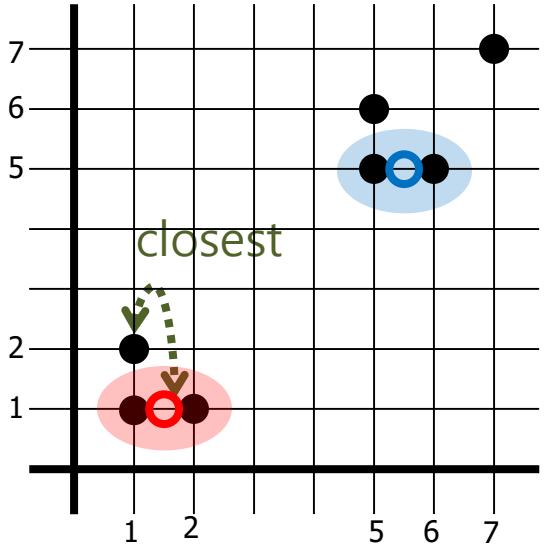


(1, 1) (2, 1) (1, 2) (5, 5) (6, 5) (5, 6) (7, 7)

Center : (3/2, 1)

Center : (11/2, 5)

An Example of Centroid-based Clustering Algorithm

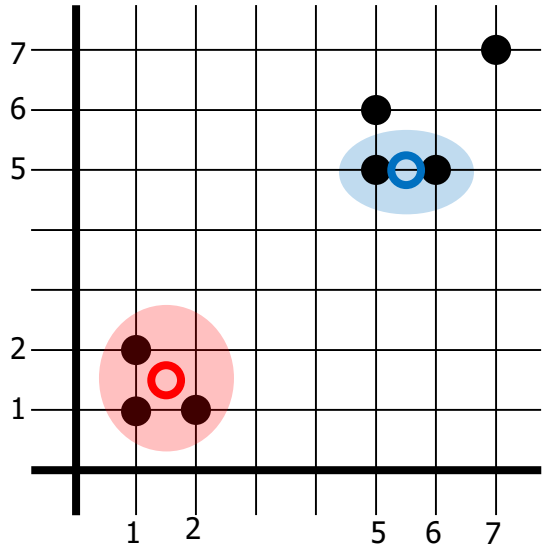


(1, 1) (2, 1) (1, 2) (5, 5) (6, 5) (5, 6) (7, 7)

Center : (3/2, 1)

Center : (11/2, 5)

An Example of Centroid-based Clustering Algorithm

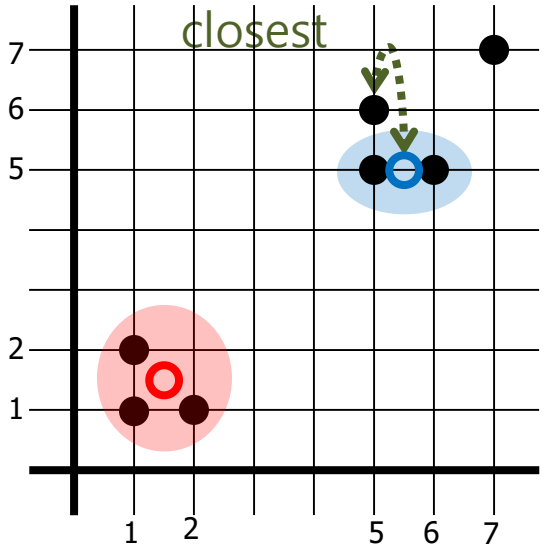


(1, 1) (2, 1) (1, 2) (5, 5) (6, 5) (5, 6) (7, 7)

Center : (4/3, 4/3)

Center : (11/2, 5)

An Example of Centroid-based Clustering Algorithm

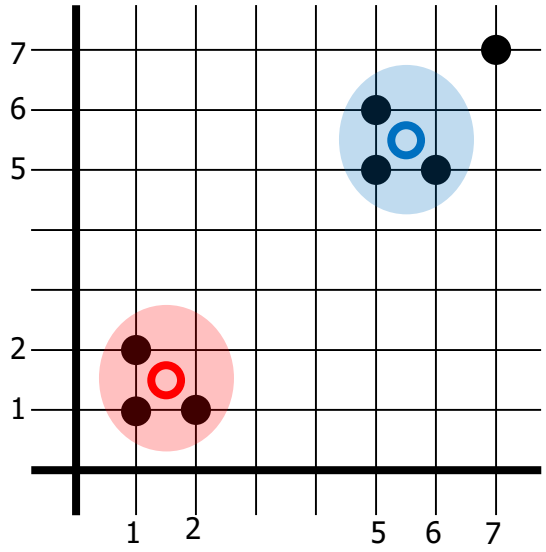


(1, 1) (2, 1) (1, 2) (5, 5) (6, 5) (5, 6) (7, 7)

Center : (4/3, 4/3)

Center : (11/2, 5)

An Example of Centroid-based Clustering Algorithm

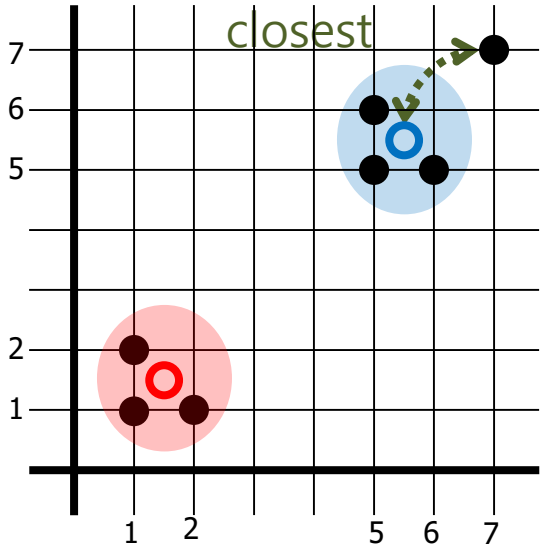


(1, 1) (2, 1) (1, 2) (5, 5) (6, 5) (5, 6) (7, 7)

Center : (4/3, 4/3)

Center : (16/3, 16/3)

An Example of Centroid-based Clustering Algorithm

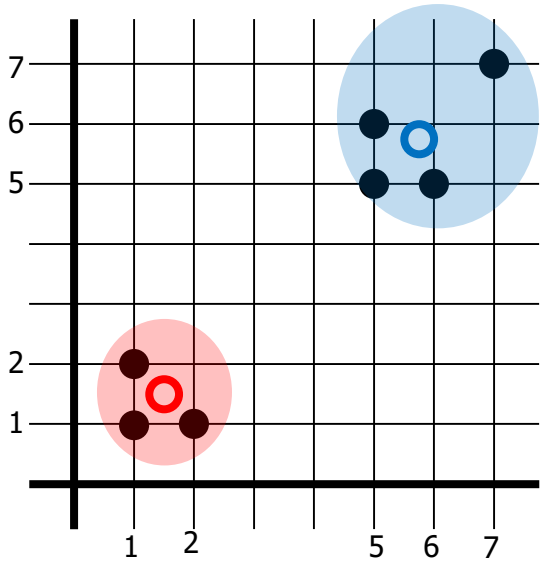


(1, 1) (2, 1) (1, 2) (5, 5) (6, 5) (5, 6) (7, 7)

Center : (4/3, 4/3)

Center : (16/3, 16/3)

An Example of Centroid-based Clustering Algorithm

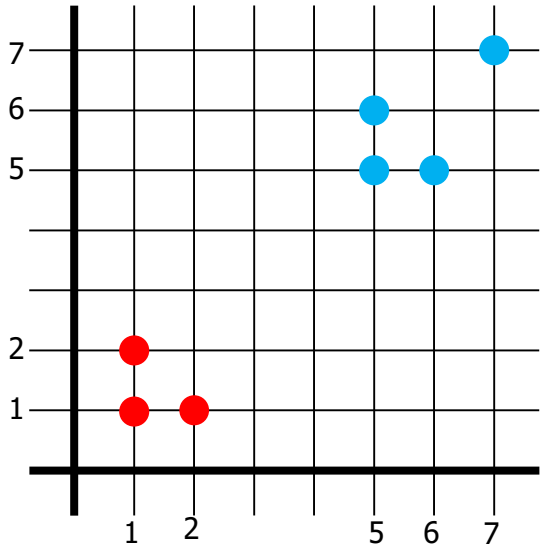


(1, 1) (2, 1) (1, 2) (5, 5) (6, 5) (5, 6) (7, 7)

Center : (4/3, 4/3)

Center : (23/4, 23/4)

An Example of Centroid-based Clustering Algorithm



(1, 1)(2, 1)(1, 2)(5, 5)(6, 5)(5, 6)(7, 7)

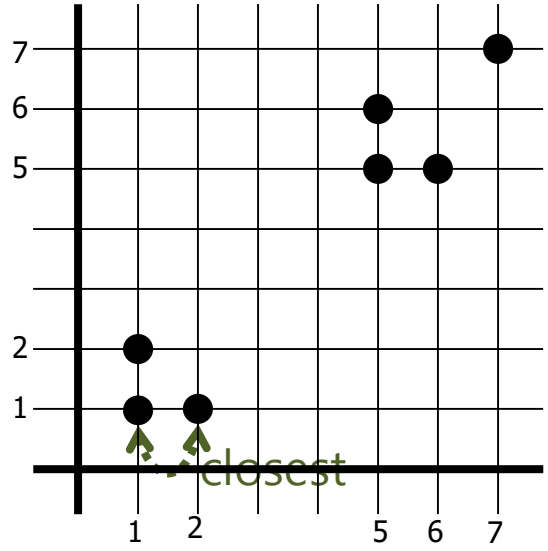
Class A

(1, 1)(2, 1)(1, 2)

Class B

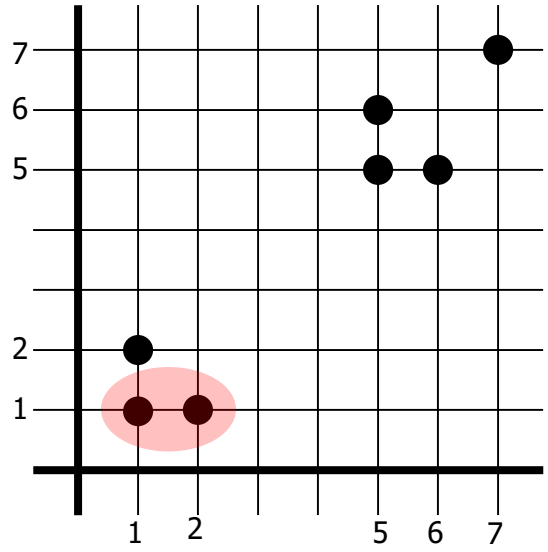
(5, 5)(6, 5)(5, 6)(7, 7)

An Example of Minimum-spanning Tree Clustering Algorithm



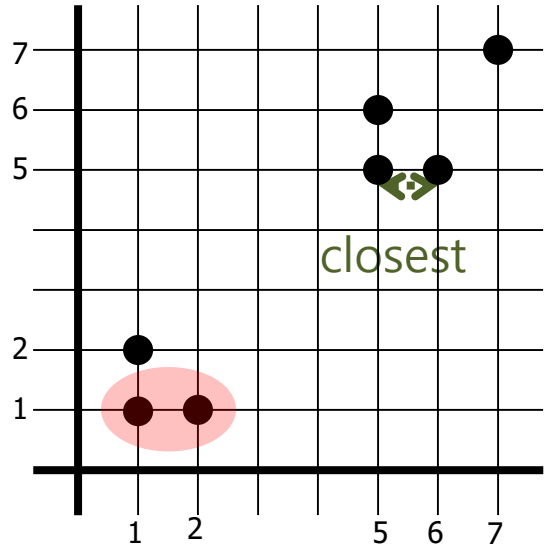
(1, 1) (2, 1) (1, 2) (5, 5) (6, 5) (5, 6) (7, 7)

An Example of Minimum-spanning Tree Clustering Algorithm



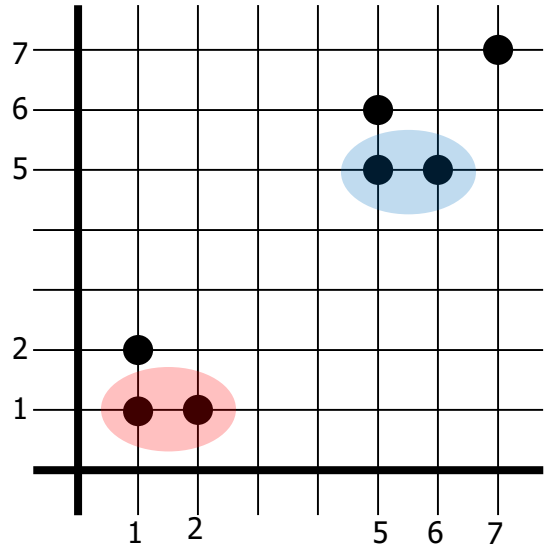
(1, 1) (2, 1) (1, 2) (5, 5) (6, 5) (5, 6) (7, 7)

An Example of Minimum-spanning Tree Clustering Algorithm



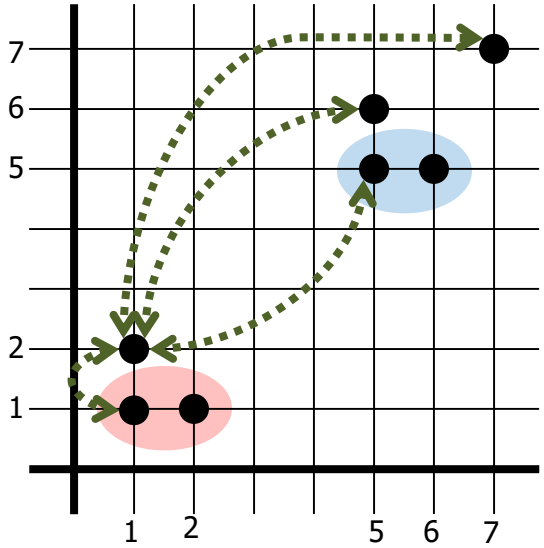
(1, 1) (2, 1) (1, 2) (5, 5) (6, 5) (5, 6) (7, 7)

An Example of Minimum-spanning Tree Clustering Algorithm



(1, 1) (2, 1) (1, 2) (5, 5) (6, 5) (5, 6) (7, 7)

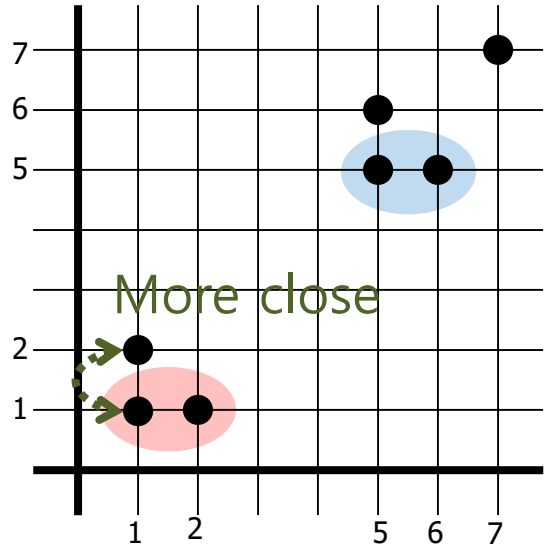
An Example of Minimum-spanning Tree Clustering Algorithm



(1, 1) (2, 1) (1, 2) (5, 5) (6, 5) (5, 6) (7, 7)

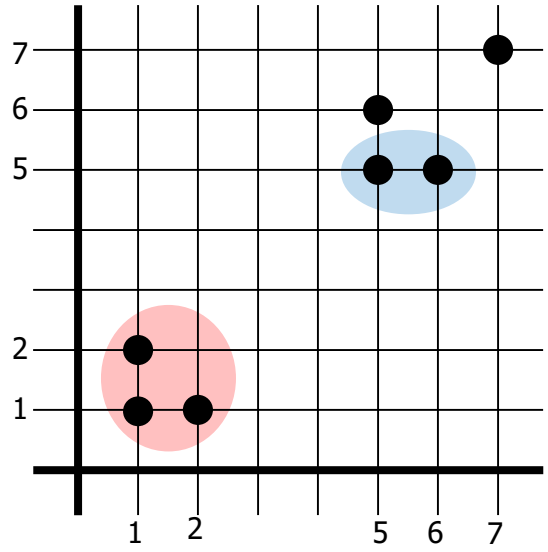
Compare each distance from closest item in clusters

An Example of Minimum-spanning Tree Clustering Algorithm



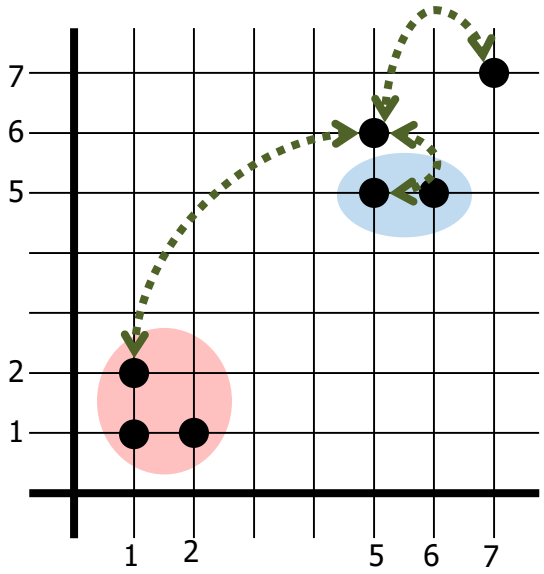
(1, 1) (2, 1) (1, 2) (5, 5) (6, 5) (5, 6) (7, 7)

An Example of Minimum-spanning Tree Clustering Algorithm



(1, 1) (2, 1) (1, 2) (5, 5) (6, 5) (5, 6) (7, 7)

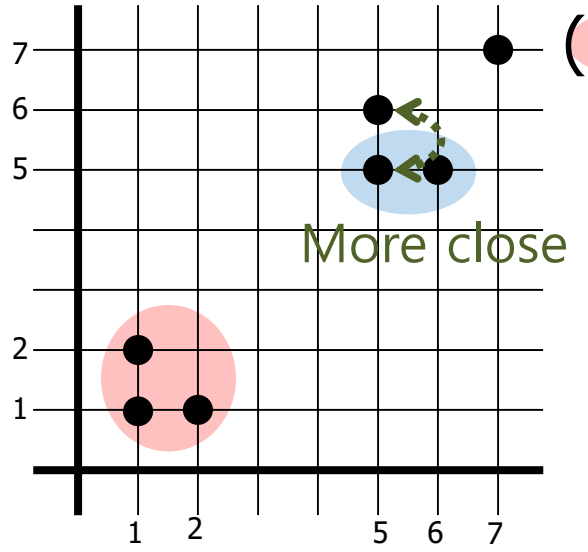
An Example of Minimum-spanning Tree Clustering Algorithm



(1, 1) (2, 1) (1, 2) (5, 5) (6, 5) (5, 6) (7, 7)

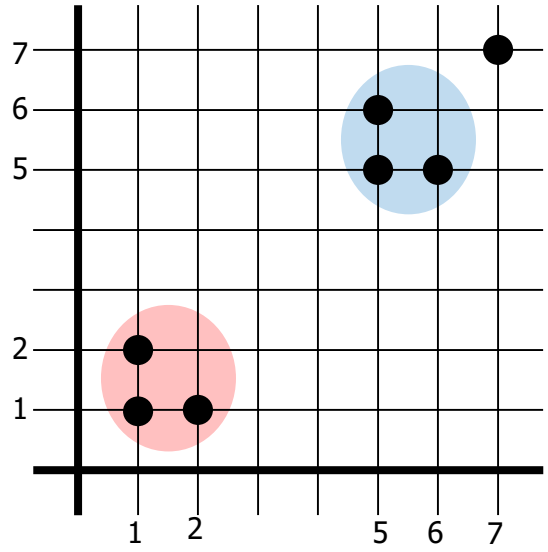
Compare each distance from closest item in clusters

An Example of Minimum-spanning Tree Clustering Algorithm



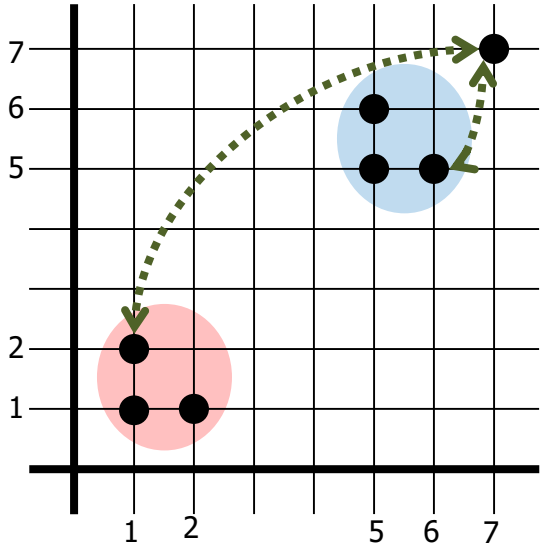
(1, 1) (2, 1) (1, 2) (5, 5) (6, 5) (5, 6) (7, 7)

An Example of Minimum-spanning Tree Clustering Algorithm



(1, 1) (2, 1) (1, 2) (5, 5) (6, 5) (5, 6) (7, 7)

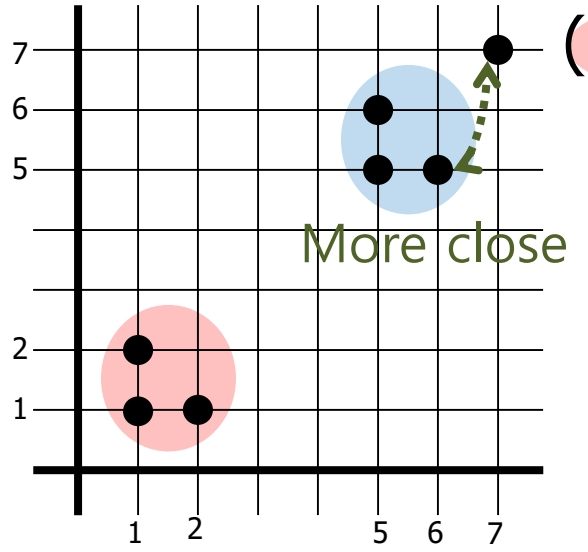
An Example of Minimum-spanning Tree Clustering Algorithm



(1, 1) (2, 1) (1, 2) (5, 5) (6, 5) (5, 6) (7, 7)

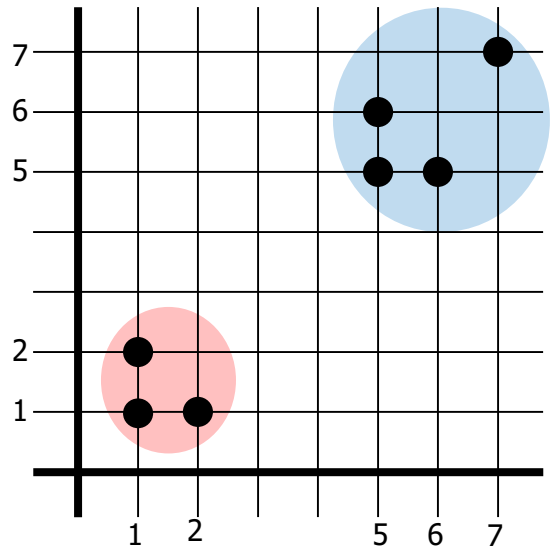
Compare each distance from closest item in clusters

An Example of Minimum-spanning Tree Clustering Algorithm



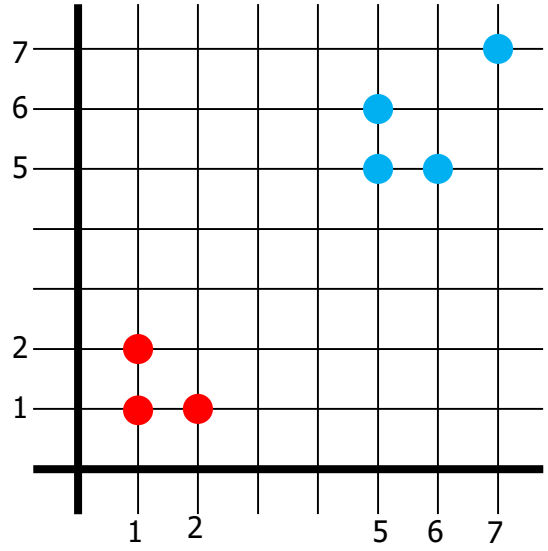
(1, 1) (2, 1) (1, 2) (5, 5) (6, 5) (5, 6) (7, 7)

An Example of Minimum-spanning Tree Clustering Algorithm



(1, 1) (2, 1) (1, 2) (5, 5) (6, 5) (5, 6) (7, 7)

An Example of Minimum-spanning Tree Clustering Algorithm



(1, 1)(2, 1)(1, 2)(5, 5)(6, 5)(5, 6)(7, 7)

Class A

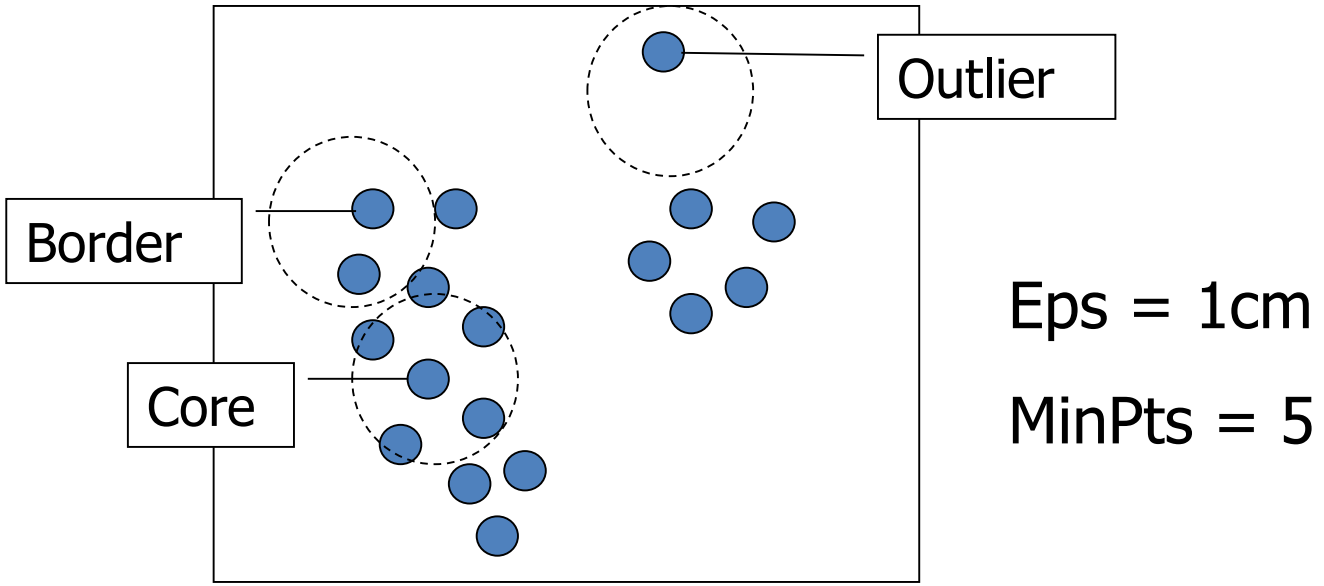
(1, 1)(2, 1)(1, 2)

Class B

(5, 5)(6, 5)(5, 6)(7, 7)

밀도 기반 군집화

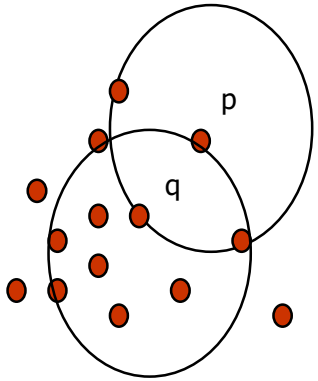
- 밀도 모델(Density-based, DBSCAN)
  - K-Means처럼 데이터의 분포(평균과의 거리가 얼마나 떨어져 있는지로 군집을 결정)를 통해 군집을 정하는 것이 아니라, 데이터들의 밀도를 이용한다
  - 같은 군집내의 데이터들은 밀도가 높게 위치해 있을 것이라는 가정
  - 주변 데이터들의 밀도를 이용해 군집을 생성해 나가는 방식
  - 2개의 파라미터가 필요
    - 주변 공간에 대한 정의 (반경): Eps
    - 반경 이내의 최소 개수 : MinPts



밀도 기반 군집화

용어 정리

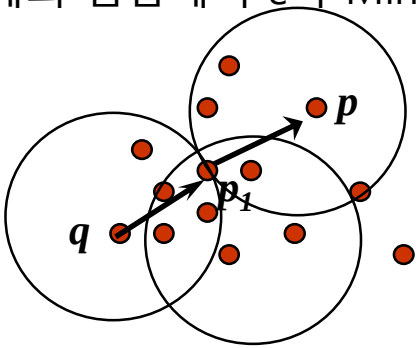
- 주어진 객체의 반경 ε 내의 이웃을 그 객체의 ε-neighbourhood라 부름
- 객체의 ε-neighbourhood 가 적어도 객체의 최소한의 개수, MinPts를 가지고 있으면 그 객체를 중심객체(core object)라고 부름
- 객체의 집합 D가 주어졌을 때, p가 q의 ε-neighbourhood 안에 q가 중심객체이면, 객체 p를 객체 q로부터 직접 밀도 도달가능(directly density-reachable)이라고 부름



MinPts = 5

Eps = 1 cm

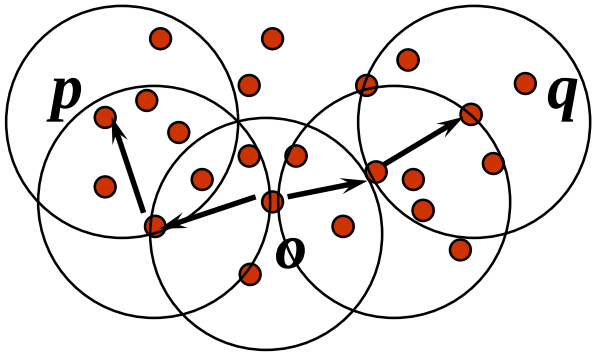
- P_{i+1}이 ε과 MinPts에 관하여 P_i로부터 직접 밀도 도달 가능 객체일 때, p_1, …, p_n, p_1=q and p_n=p의 연쇄가 존재하면 객채 p는 객체의 집합에서 ε과 MinPts에 관하여 객체 q로부터 밀도 도달가능(density reachable)이다.



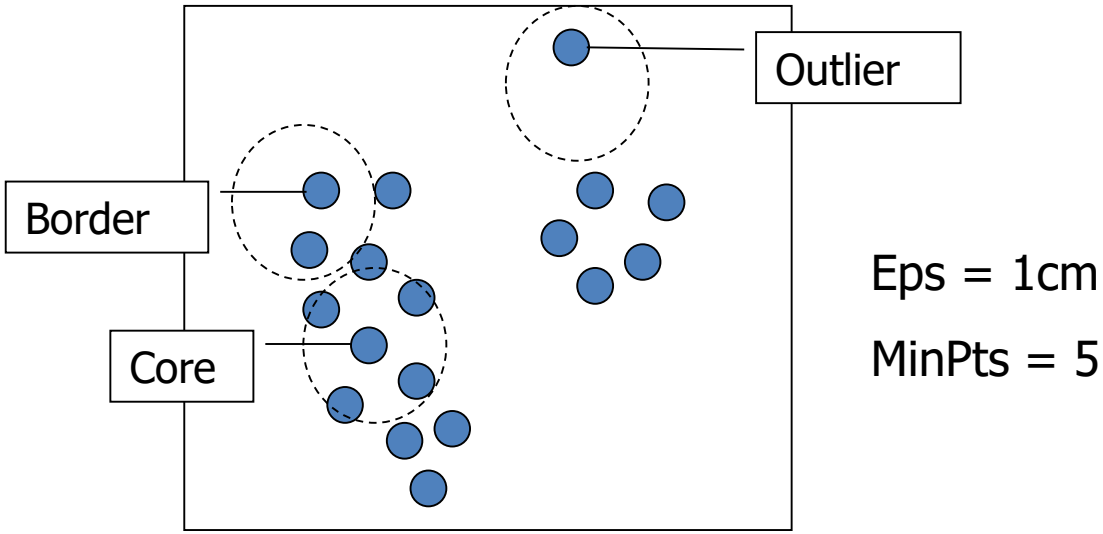
밀도 기반 군집화

용어 정리

- P와 q 모두 ε과 MinPts의 관점에서 o으로부터 밀도 도달가능성이 되는 객체 o ∈ D가 존재한다면 객체들의 집합 D에 ε과 MinPts에 관점에서 객체 p는 객체 q에 밀도 연결 (density connected) 되어 있다



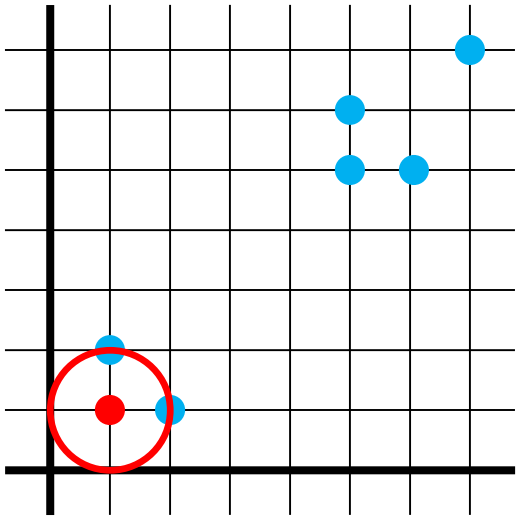
- 밀도 기반 군집은 밀도 도달가능에 대하여 최대인 밀도 연결 객체들의 집합



Algorithm

1. Arbitrary select a point p
2. Retrieve all points density-reachable from p w.r.t. Eps and MinPts.
3. If p is a core point, a cluster is formed.
4. If p is a border point, no points are density-reachable from p and DBSCAN visits the next point of the database.
5. Continue the process until all of the points have been processed

An Example of DBSCAN



(1,1) start : Core Point
(1,2) ,(2,1)

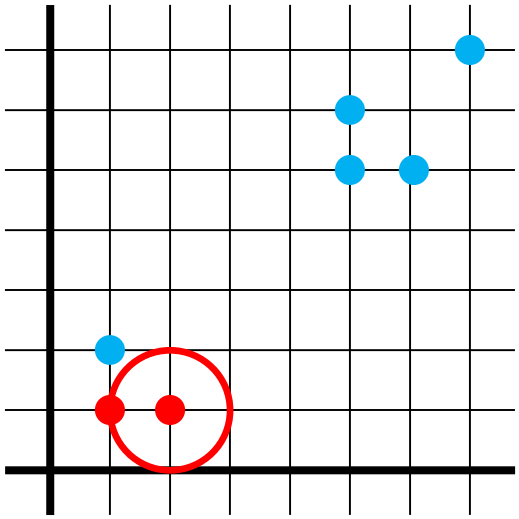
Class A

(1,1)

Unclassed

(1,2)(2,1)(5,5)
(5,6)(6,5)(7,7)

An Example of DBSCAN



(1,2) point : Border Point

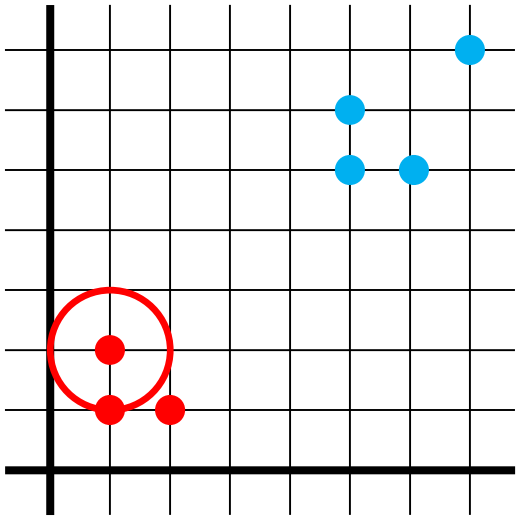
Class A

(1,1)(1,2)

Unclassed

(2,1)(5,5)(5,6)
(6,5)(7,7)

An Example of DBSCAN



(2,1) point : Border Point

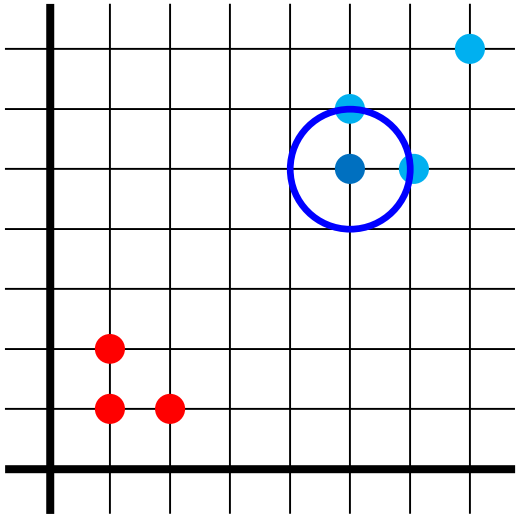
Class A

(1,1)(1,2)
(2,1)

Unclassed

(5,5)(5,6)
(6,5)(7,7)

An Example of DBSCAN



(5,5) point : Core Point

Class A

(1,1)(1,2)
(2,1)

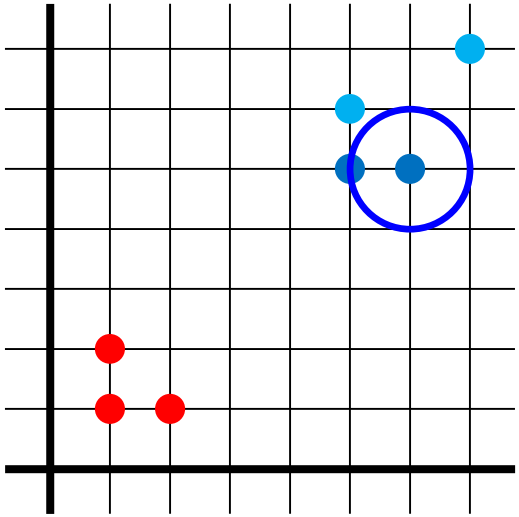
Class B

(5,5)

Unclassed

(5,6)
(6,5)(7,7)

An Example of DBSCAN



(6,5) point : Border Point

Class A

(1,1)(1,2)
(2,1)

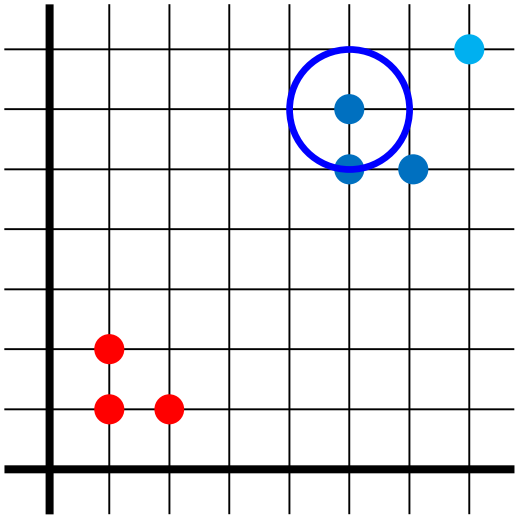
Class B

(5,5)(6,5)

Unclassed

(6,5)(7,7)

An Example of DBSCAN



(5,6) point : Border Point

Class A

(1,1)(1,2)
(2,1)

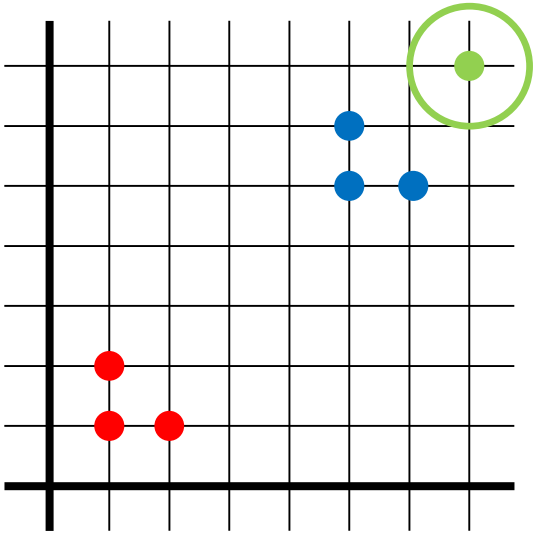
Class B

(5,5)(6,5)
(5,6)

Unclassed

(7,7)

An Example of DBSCAN



(5,6) point : Border Point

Class A

(1,1)(1,2)
(2,1)

Class B

(5,5)(6,5)
(5,6)

Class C

(7,7)

→ Outlier