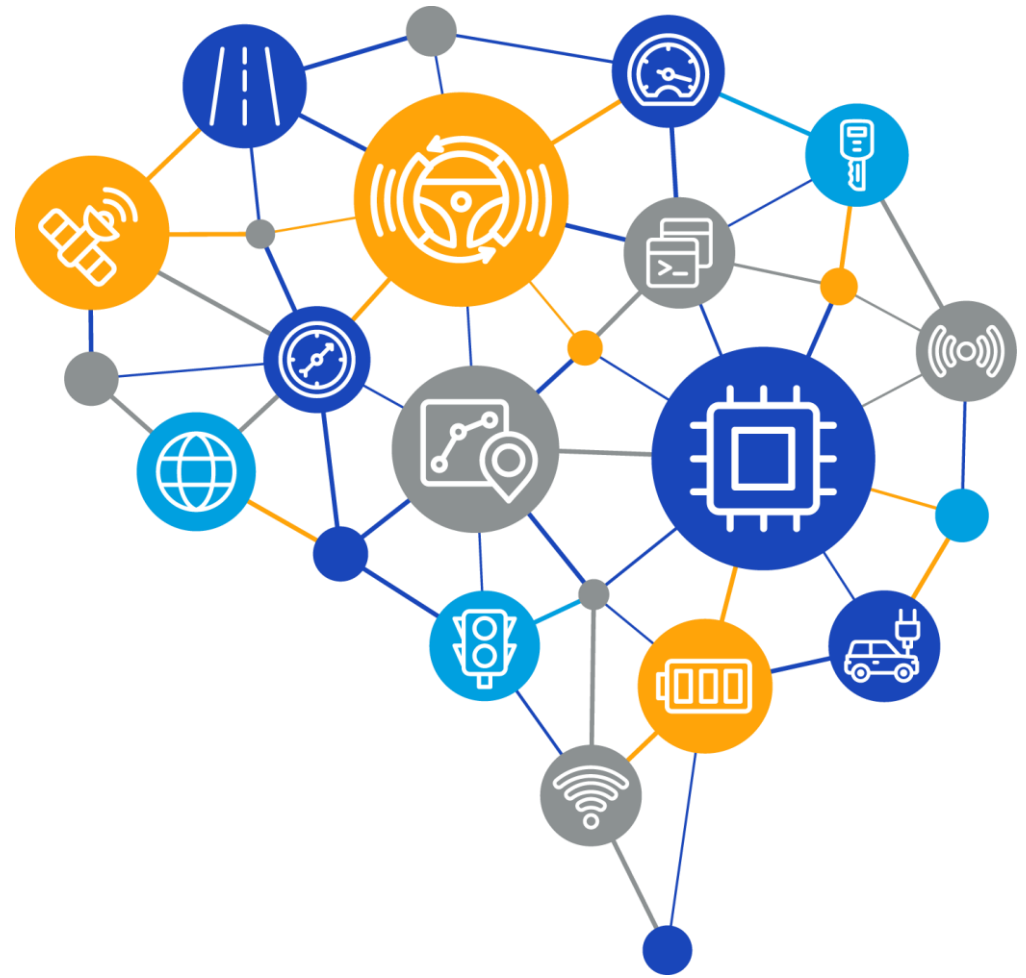


머신러닝 비지도학습

Machine Learning
Supervisor learning

2022.07

강환수 교수



AI Experts Who Lead The Future

CONTENTS

- 01 | 지도학습과 비지도학습
- 02 | 군집화 기법: k-평균 군집화
- 03 | 밀도 기반 클러스터링
- 04 | 계층적 군집화
- 05 | 군집화 평가
- 06 | 과적합

AI Experts
Who Lead
The Future

01

지도학습과 비지도학습

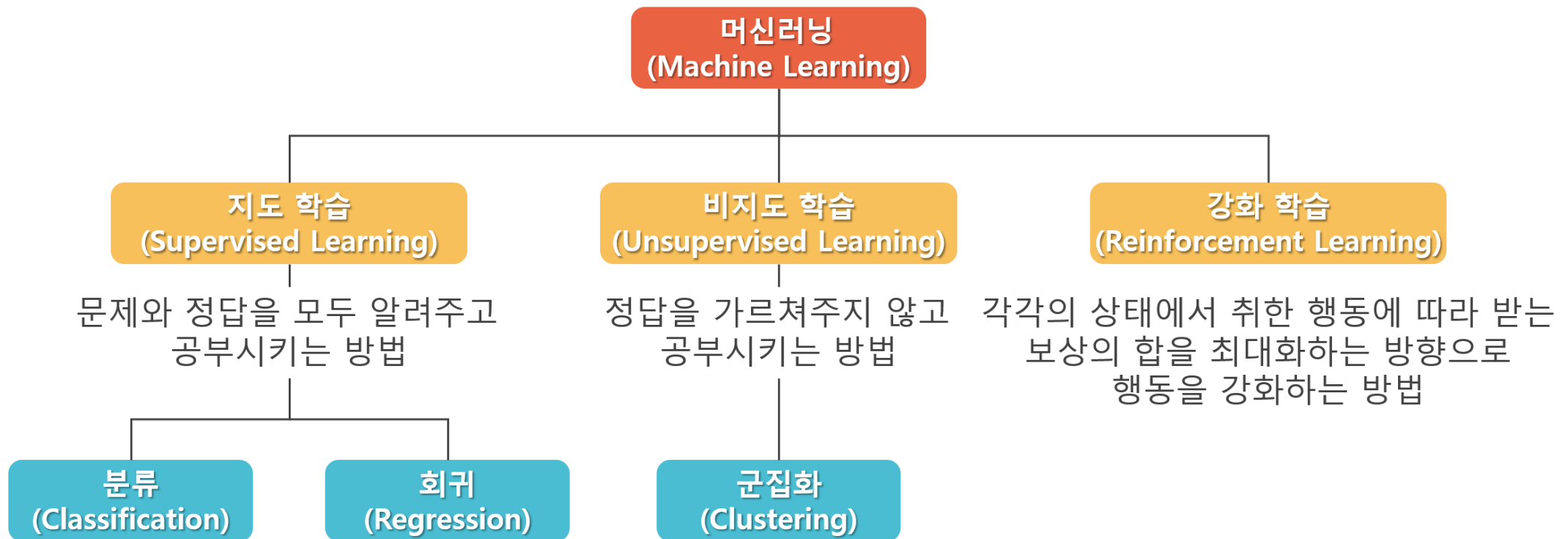
다음 자료를 기반으로 제작
난생처음 인공지능 입문 (출판사: 한빛아카데미)

머신러닝의 종류

인공지능 활용 Python language

- 머신러닝 모델을 학습시킴에 있어서, 입력값에 대한 결과값을 알고 있는지 여부에 따라 크게 "지도학습"과 "비지도 학습"으로 나누어짐

문제 ↔ 입력값
정답 ↔ 결과값, 레이블 (Label)



① 지도학습 (Supervised Learning) (1/4)

인공지능 활용 Python language

① 지도학습 (Supervised Learning) (1/4)

- 입력값에 대한 정답 (=결과값)을 알고 있는 학습 데이터를 활용하여 머신러닝 모델을 학습시키는 방식
- 입력값에 대한 정답 (=결과값)을 레이블 (Label)이라고 함



그림 2-8 구분된 학습 데이터를 활용하는 지도학습의 예

© Hanbit Academy Inc.

① 지도학습 (Supervised Learning) (2/4)

인공지능 활용 Python language

- ① 지도학습 (Supervised Learning) (2/4)
 - 지도학습에는 분류 (Classification)와 회귀 (Regression)라는 2가지 유형이 있음
 - ① 분류 (Classification)
 - 어떤 입력 데이터가 들어오더라도 학습에 사용한 레이블 (Label)중 하나로 결과값을 결정
 - 레이블 (Label)이 이산적인 (Discrete) 경우 (즉, [0, 1, 2, 3, ...]와 같이 유한한 경우)
 - Ex) 이미지가 주어졌을 때, 고양이 (Class = 0) 이미지인지 또는 개 (Class = 1) 이미지인지 분류
 - ② 회귀 (Regression)
 - 입력 데이터에 대한 결과값으로 학습에 사용한 레이블 이외의 값이 나올 수 있음
 - 레이블 (Label)이 실수인 경우
 - Ex) 키 (Height) 정보가 주어졌을 때, 몸무게를 예측

표 2-2 지도학습을 적용한 머신러닝 모델의 작업 © Hanbit Academy Inc.

작업	내용
분류	입력값에 대한 결과값이 정해진 레이블 중 하나로 결정되는 작업을 의미
회귀	입력값에 대한 결과값이 학습에 사용된 레이블 외의 값도 나올 수 있는 작업을 의미

① 지도학습 (Supervised Learning) (3/4)

인공지능 활용 Python language

① 지도학습 (Supervised Learning) (3/4)

- ㉠ 분류 (Classification) 작업

- 대표적인 예: 필기체 (손글씨) 인식
- 0부터 9까지의 손글씨 숫자 이미지와 레이블 (Label) 정보를 학습 데이터로 사용
- 어떤 이미지라도 (심지어는 숫자 이미지가 아니더라도) 0부터 9까지의 레이블 중 하나로 결과값을 결정함

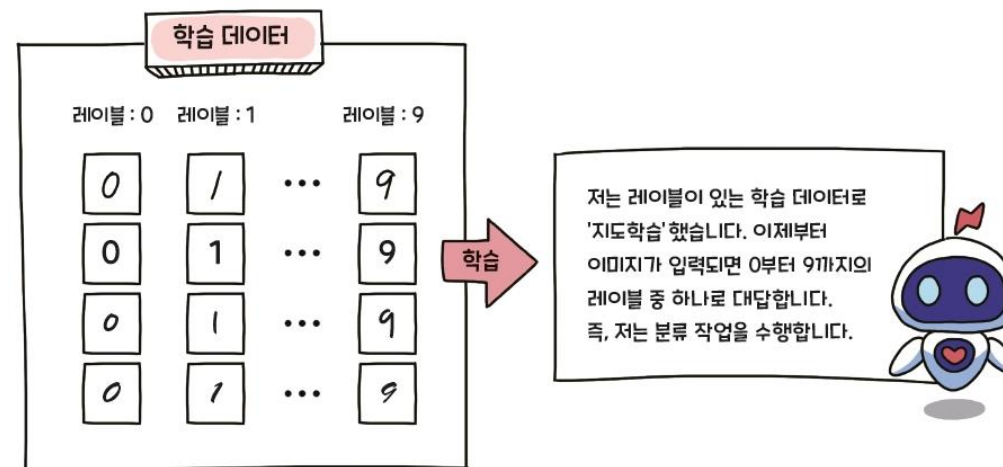


그림 2-9 머신러닝의 분류 작업 예: 숫자 필기체 인식하기

© Hanbit Academy Inc.

① 지도학습 (Supervised Learning) (4/4)

인공지능 활용 Python language

① 지도학습 (Supervised Learning) (4/4)

- ② 회귀 (Regression) 작업

- 대표적인 예: 몸무게 예측 작업
- 여러 명의 키와 그에 대응하는 몸무게를 학습 데이터로 사용하는데, 입력된 키에 대한 몸무게를 결과값으로 출력하기 때문에 몸무게가 레이블 (Label)이 됨
- 다양한 키에 대해서 레이블에 포함되지 않은 몸무게도 결과값으로 출력 가능

No.	이름	키 (cm)	몸무게 (kg)
1	김민성	100	30
2	박다인	120	40
3	윤이안	130	45
4	최서연	160	60
5	문진승	190	75

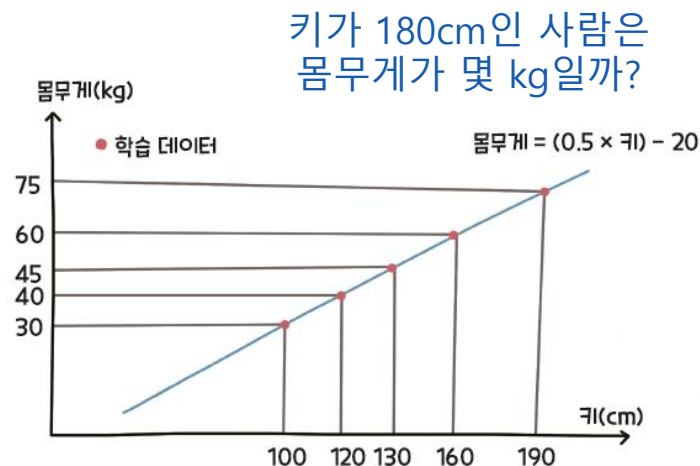


그림 2-7 몸무게와 키의 상관관계에 최적화된 직선의 방정식 © Hanbit Academy Inc.

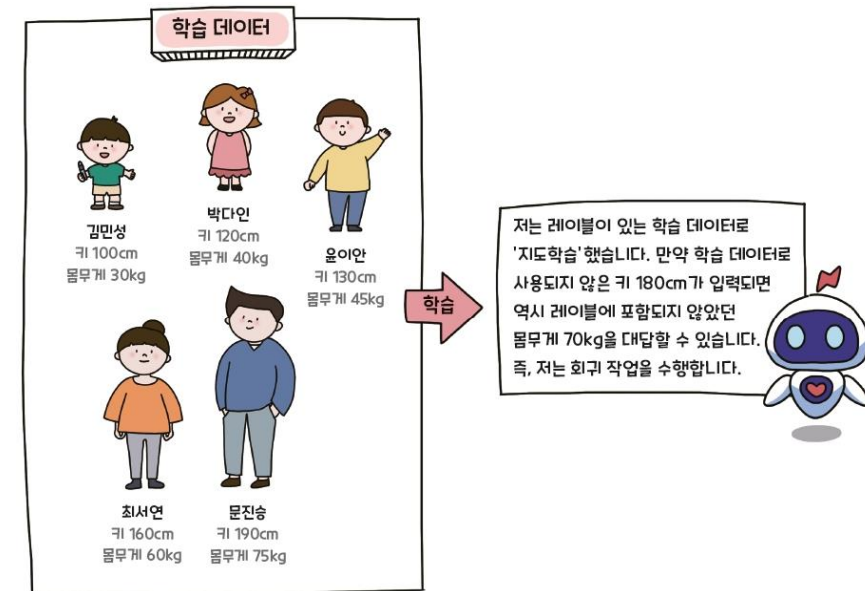


그림 2-10 머신러닝의 회귀 작업 예 : 키로 몸무게 예측하기 © Hanbit Academy Inc.

② 비지도학습 (Unsupervised Learning) (1/2)

인공지능 활용 Python language

② 비지도학습 (Unsupervised Learning) (1/2)

- 입력값에 대한 정답 (=결과값)이 없는, 즉 레이블 (Label)이 없는 데이터를 사용하여 머신러닝 모델을 학습시키는 방식
- 머신러닝 모델이 훈련 데이터 (Training Data)를 이용하여 데이터들 간의 규칙성을 찾음

훈련 데이터에
"고양이," "개"라는
Labels 정보가 없음

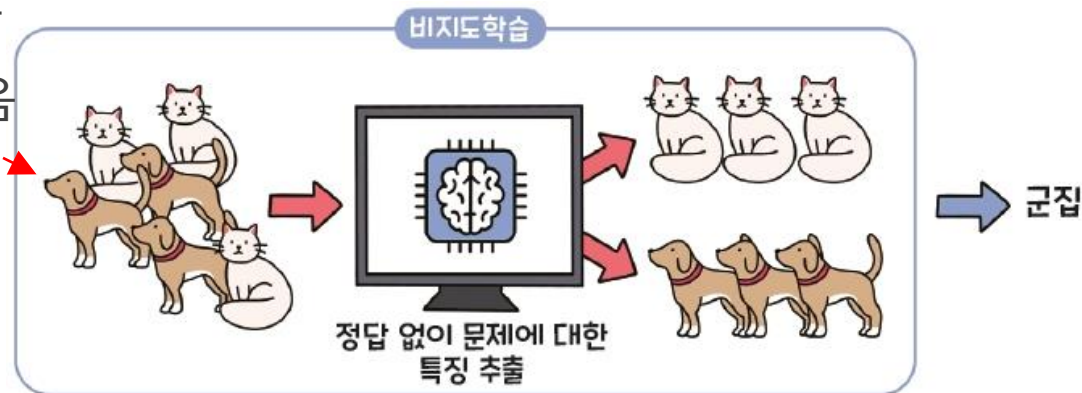


그림 8-12 비지도 학습 © Hanbit Academy Inc.

② 비지도학습 (Unsupervised Learning) (2/2)

인공지능 활용 Python language

• ② 비지도학습 (Unsupervised Learning) (2/2)

- x (=입력 데이터)와 y (= 지도학습에서 레이블)의 관계를 파악했던 지도학습과는 달리, 비지도학습은 y 없이 x 간의 관계를 스스로 파악함
- 즉, 지도학습과 다른 점은 y (= 레이블, Label)의 유무
- 비지도학습에서 사용하는 모델로는 군집화 (Clustering)가 있음

표 8-2 지도학습과 비지도학습 시 필요한 데이터 © Hanbit Academy Inc.

구분	지도학습	비지도학습
필요한 데이터 종류	x (학습 데이터), y (레이블)	x (학습 데이터)

AI Experts
Who Lead
The Future

02

군집화 기법: k-평균 군집화

다음 자료를 기반으로 제작
난생처음 인공지능 입문 (출판사: 한빛아카데미)

군집과 군집화

인공지능 활용 Python language

- **군집 (Cluster, 클러스터)**
 - 비슷한 특징을 가진 데이터들의 집단
- **군집화 (Clustering, 클러스터링) (1/2)**
 - 데이터가 주어졌을 때 그 데이터들을 유사한 정도에 따라 군집으로 분류하는 것
 - [그림 8-22]의 왼쪽 그래프를 보면 다양한 데이터들이 서로 섞여 있지만, 군집화 과정을 진행하면 오른쪽 그래프와 같이 비슷한 데이터끼리 군집으로 묶임

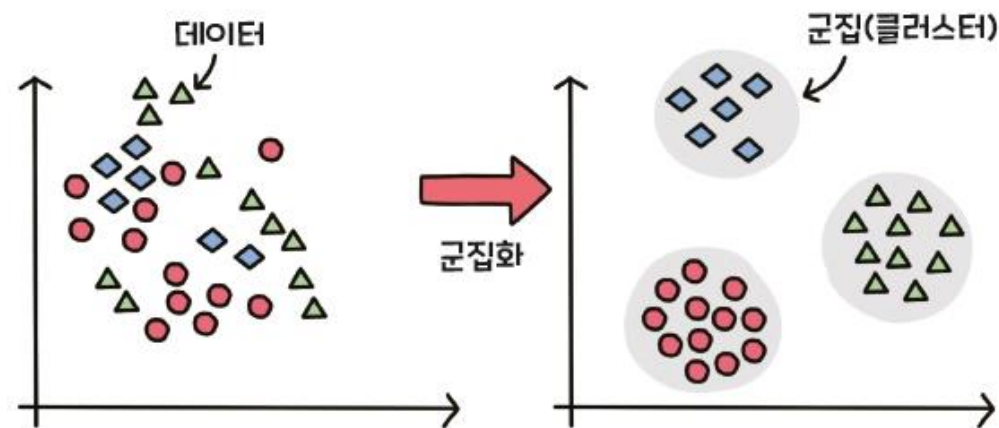


그림 8-22 군집화 © Hanbit Academy Inc.

군집화 (Clustering, 클러스터링)

인공지능 활용 Python language

- 레이블 (Label) 없이, 데이터 간에 존재하는 특성을 바탕으로 비슷한 데이터 구분하여 비슷한 집단으로 묶는 작업
- 군집 작업은 마케팅 분야에서 많이 응용
- 예를 들어,
 - 채팅 사이트를 운영하는 기업이 사람들의 대화 내용을 기반으로 군집 작업을 수행해 취미를 속성으로 묶었다면, 집단별로 인구통계학적 특성을 파악해 적절한 취미용품을 광고할 수 있음

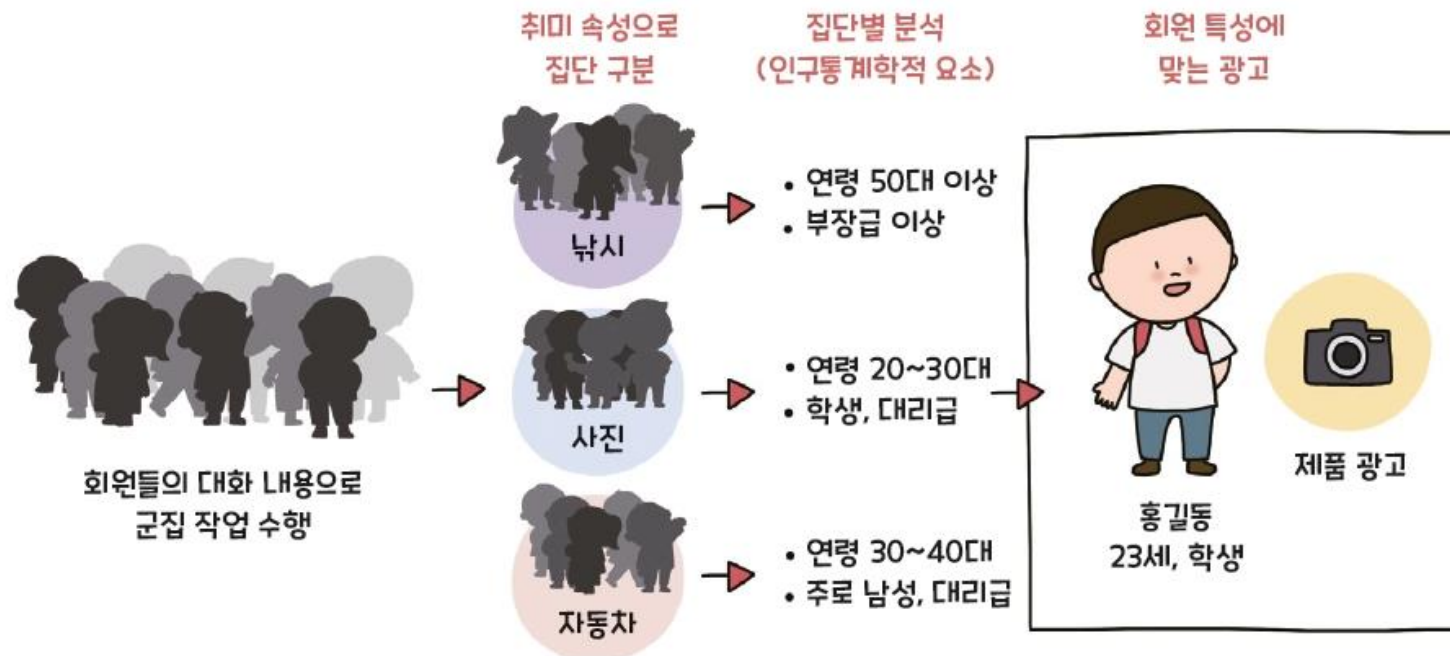


그림 2-12 마케팅 분야의 군집 작업 활용 예

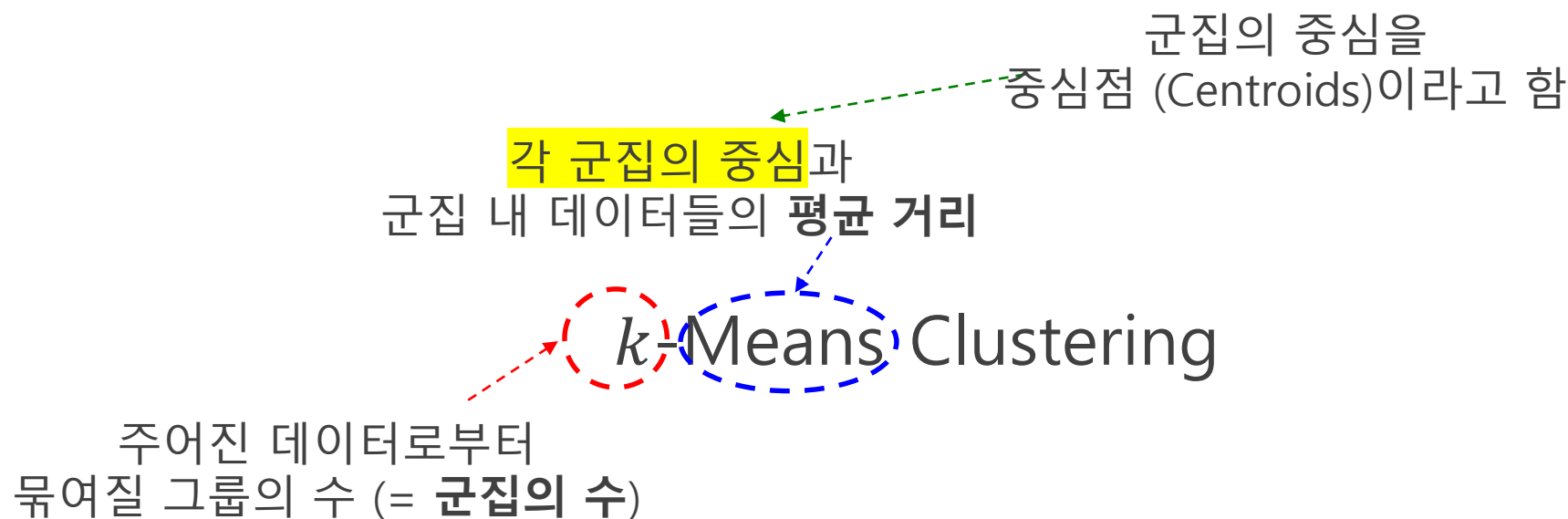
© Hanbit Academy Inc.

- 군집화 (Clustering) 문제를 해결하기 위한 기법
 - ① k -평균 군집화 (k -Means Clustering)
 - ② 밀도기반 클러스터링 (DBSCAN, Density Based Spatial Clustering of Applications with Noise)
 - ③ 계층적 군집화 (Hierarchical Clustering)

① k -평균 군집화 (k -Means Clustering) (1/9)

인공지능 활용 Python language

- ' k '는 주어진 데이터로부터 묶여질 그룹의 수 (= 군집의 수)
- 'Means'는 각 군집의 중심과 데이터들의 평균 거리를 의미
- 군집의 중심을 중심점 (Centroids)이라고 함



① k -평균 군집화 (k -Means Clustering) (2/9)

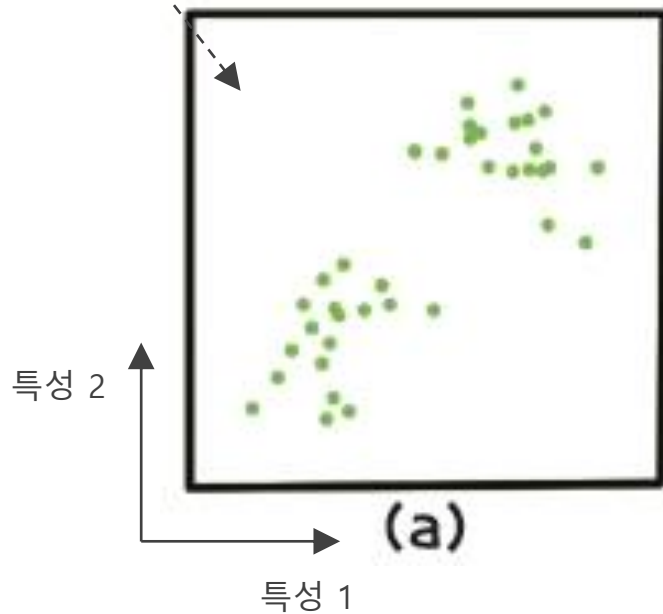
인공지능 활용 Python language

• k -Means Clustering 알고리즘의 군집화 과정을 살펴봅시다

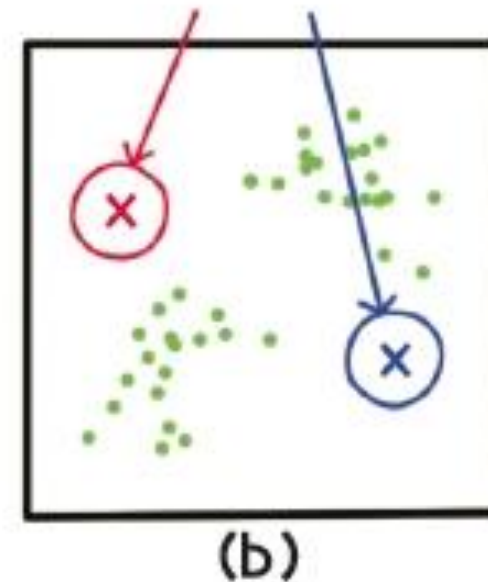
- Step 1)

주어진 학습 데이터셋 (Training Dataset)에서 k 개의 중심점 (Centroids)를 임의로 지정
본 예제에서는 $k = 2$ 로 가정

학습 데이터의 분포



2개의 중심점 선택($k=2$)

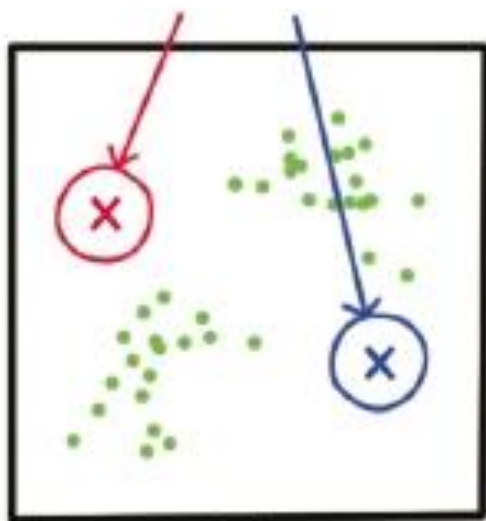


① k -평균 군집화 (k -Means Clustering) (3/9)

인공지능 활용 Python language

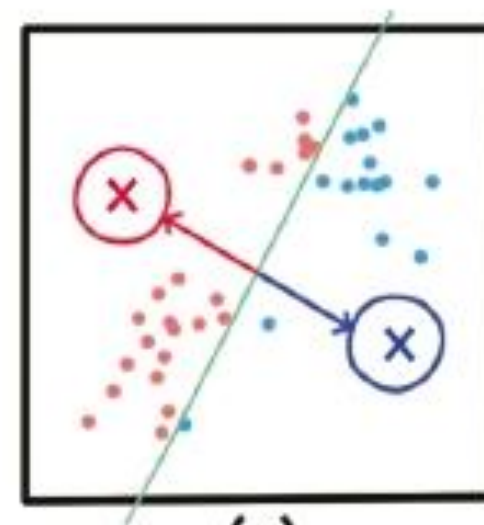
- k -Means Clustering 알고리즘의 군집화 과정을 살펴봅시다
 - Step 2) 데이터들을 가장 가까운 중심점 (Centroids)에 할당합니다

2개의 중심점 선택($k=2$)



(b)

가까운 K에 데이터 할당



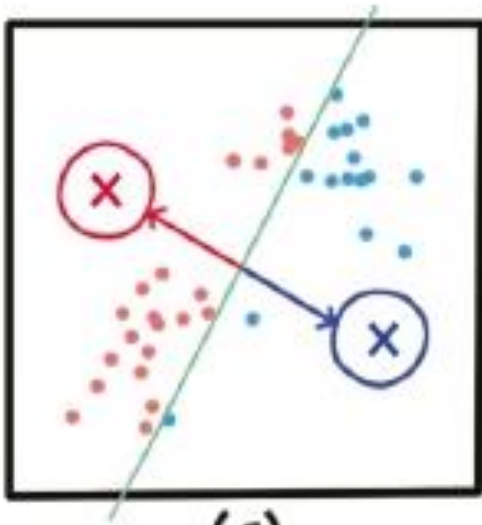
(c)

① k -평균 군집화 (k -Means Clustering) (4/9)

인공지능 활용 Python language

- k -Means Clustering 알고리즘의 군집화 과정을 살펴봅시다
 - Step 3) (c)에서 할당된 결과를 바탕으로 중심점을 새롭게 지정합니다

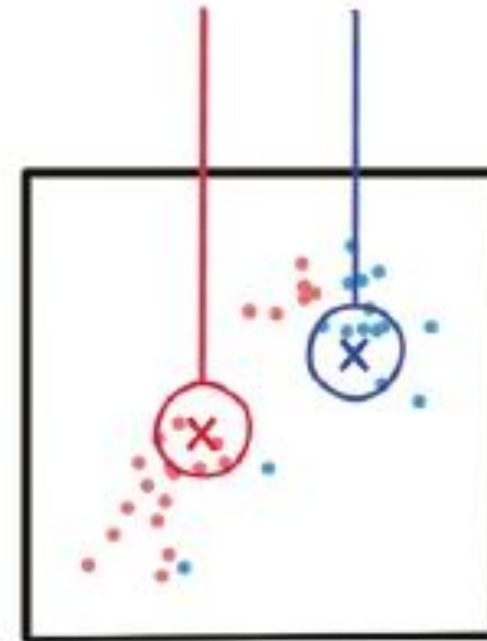
가까운 K에 데이터 할당



(c)



새로운 중심점 2개 선택



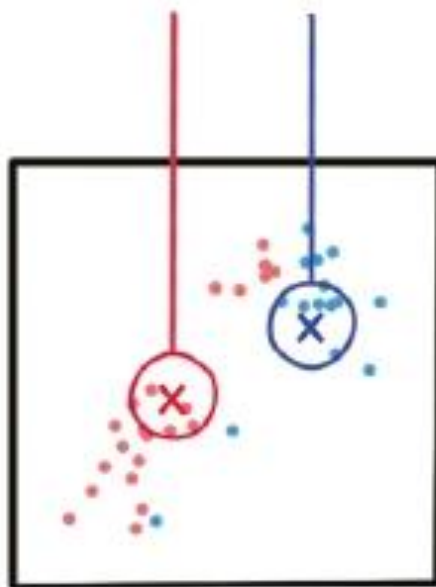
(d)

① k -평균 군집화 (k -Means Clustering) (5/9)

인공지능 활용 Python language

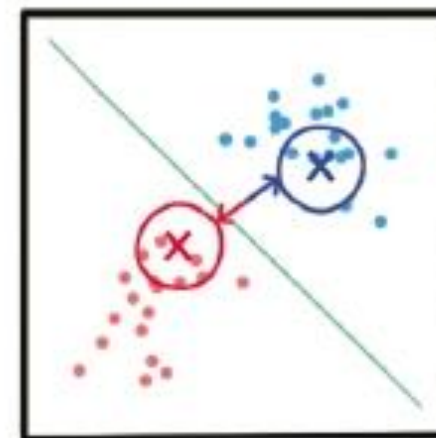
- k -Means Clustering 알고리즘의 군집화 과정을 살펴봅시다
 - Step 4) 중심점이 더 이상 변하지 않을 때까지 (c)부터 (d)까지의 과정을 반복 수행합니다

새로운 중심점 2개 선택



(d)

가까운 K에 데이터 할당



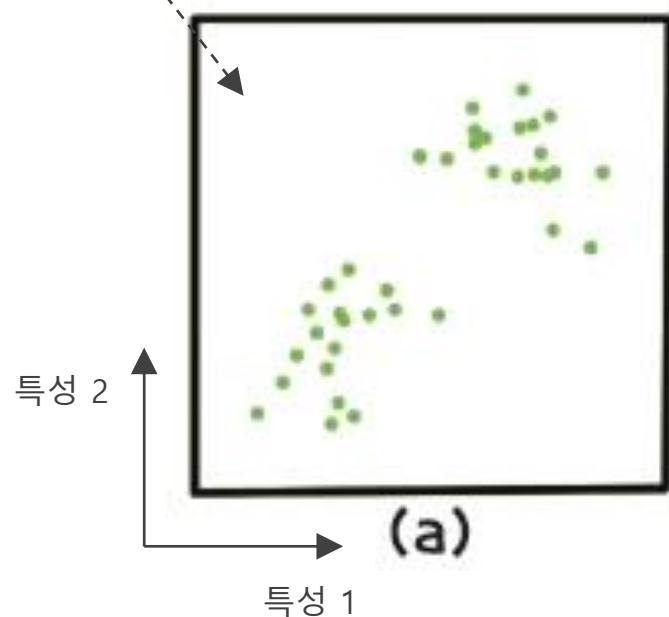
(e)

① k -평균 군집화 (k -Means Clustering) (6/9)

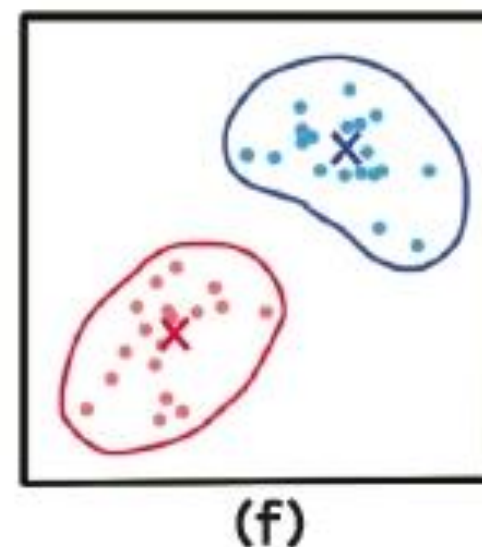
인공지능 활용 Python language

- k -Means Clustering 알고리즘의 군집화 과정을 살펴봅시다
 - Step 5) 최종적인 군집 (Clusters)이 생성됩니다

학습 데이터의 분포



알고리즘 수행 완료 후
최종 군집 형태



① k -평균 군집화 (k -Means Clustering) (7/9)

인공지능 활용 Python language

- k -Means Clustering 알고리즘의 군집화 과정을 살펴봅시다

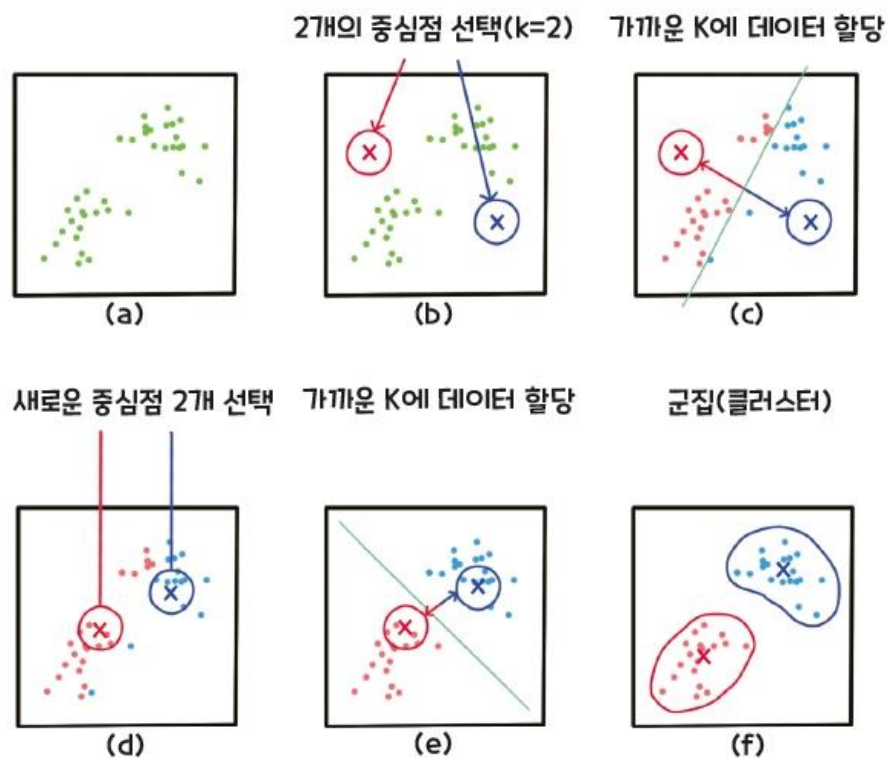


그림 8-23 K-평균 군집화 과정

© Hanbit Academy Inc.

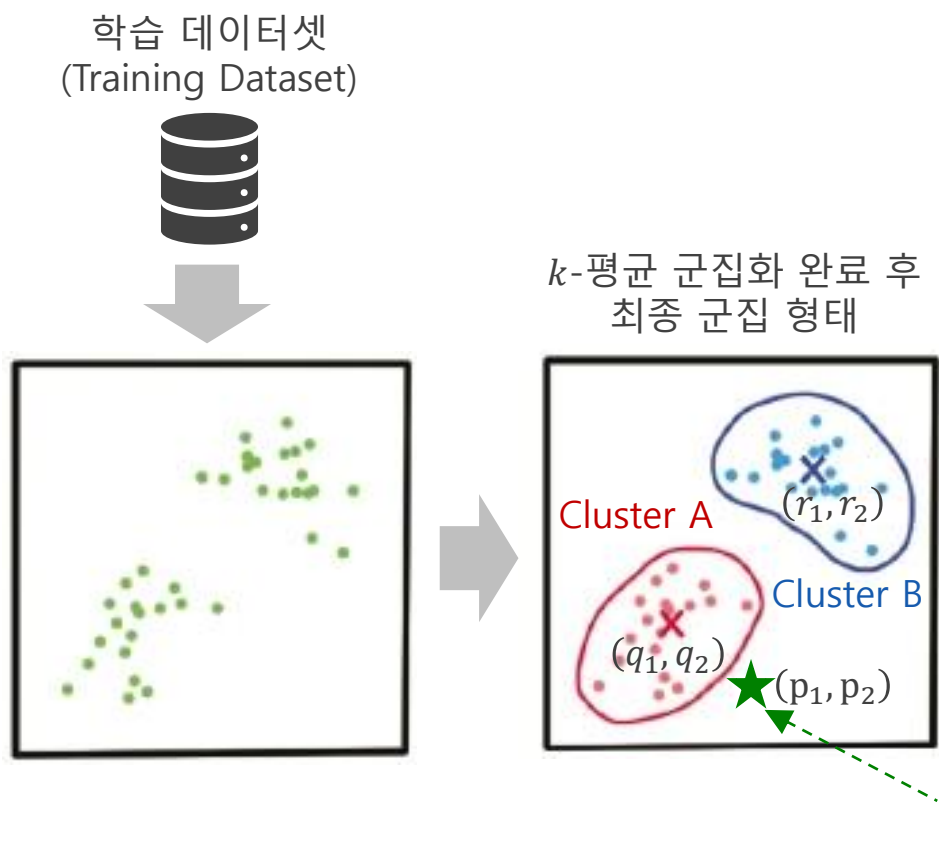
<군집화 과정 요약>

- 학습 데이터의 분포입니다
- 주어진 학습 데이터셋에서 k 개의 중심점을 임의로 지정합니다 (본 예제에서는 $k = 2$ 로 가정하였습니다)
- 데이터들을 가장 가까운 중심점에 할당합니다.
- (c)에서 할당된 결과를 바탕으로 중심점을 새롭게 지정합니다
- 중심점이 더 이상 변하지 않을 때까지 (c)~(d) 과정을 반복 수행합니다
- 최종적인 군집이 형성됩니다

① k -평균 군집화 (k -Means Clustering) (8/9)

인공지능 활용 Python language

- k -평균 군집화가 완료되고 나서, 시험 데이터 (Test Data)가 주어진 경우를 살펴봅시다



Step 1) 군집 A의 중심점과 ★ 사이의 거리를 계산

$$d_A = \sqrt{(p_1 - q_1)^2 + (p_2 - q_2)^2}$$

군집 B의 중심점과 ★ 사이의 거리를 계산

$$d_B = \sqrt{(p_1 - r_1)^2 + (p_2 - r_2)^2}$$

Step 2) $d_A > d_B$ 이면, ★을 군집 B로 분류

만약 $d_A < d_B$ 이면, ★을 군집 A로 분류

① k -평균 군집화 (k -Means Clustering) (9/9)

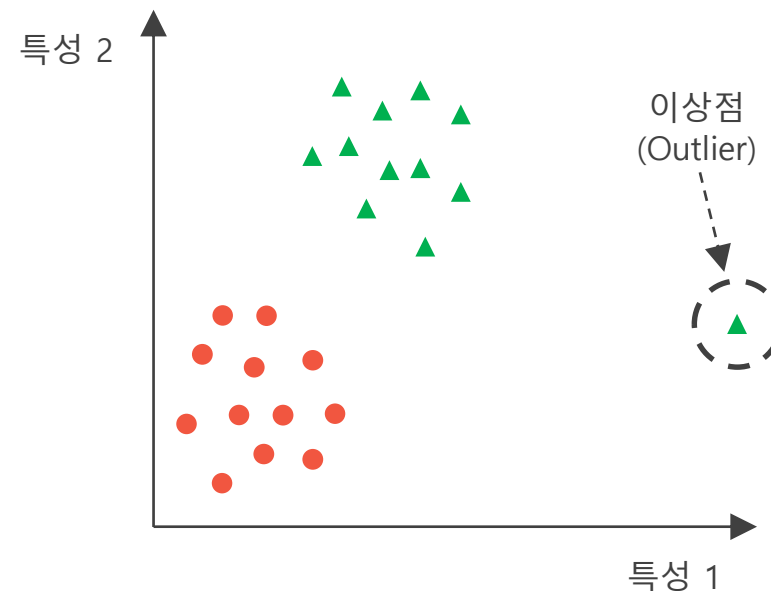
인공지능 활용 Python language

• 장점

- 알고리즘이 쉽고 간결함
- 여러 군집화 알고리즘 중에서 비교적 빠르게 수행됨

• 단점

- 최적의 k (= 군집의 수)를 모르기 때문에 찾아야 함
- 특성/특징 (Feature)의 개수가 많아지면 군집화 성능의 정확도가 떨어지는 경향이 있음
- 초기 중심점 (Initial Centroids)에 군집화 성능이 영향을 많이 받음
- 학습 데이터 내에 이상점 (Outlier)가 있을 경우, 좋은 성능이 안 나올 수 있음 (즉, 이상점에 대해 취약함)



AI Experts
Who Lead
The Future

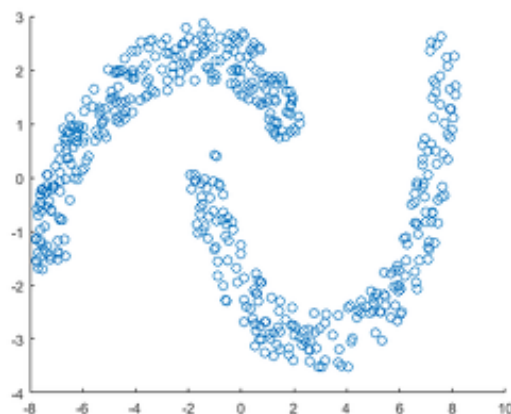
03

밀도 기반 클러스터링

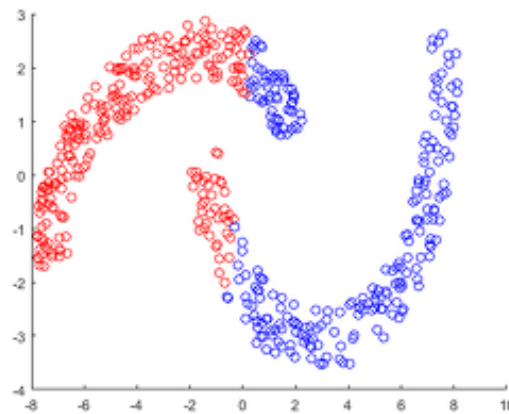
② 밀도기반 클러스터링 (DBSCAN) (1/12)

인공지능 활용 Python language

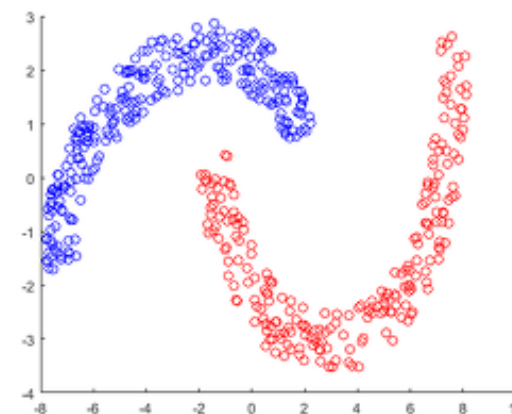
- DBSCAN(= **D**ensity **B**ased **S**patial **C**lustering of **A**pplications with **N**oise)
- 밀도 (Density)를 기반으로 군집화하는 군집 알고리즘



(a) 원본 데이터



(b) k-means clustering의 결과



(c) DBSCAN의 결과

[사진출처] <https://untitledblog.tistory.com/146>

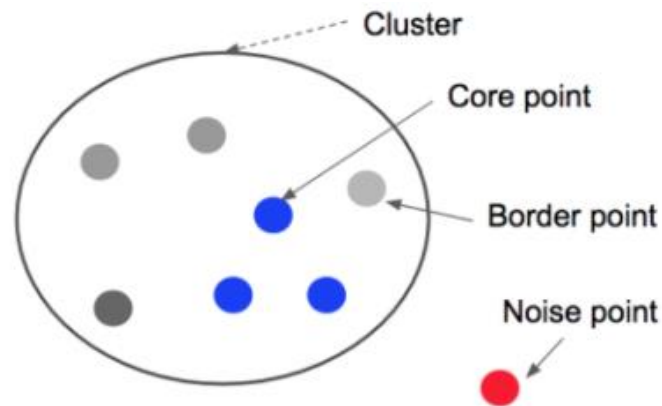
② 밀도기반 클러스터링 (DBSCAN) (2/12)

인공지능 활용 Python language

• 밀도기반 클러스터링의 주요 용어

- ϵ (Epsilon, 거리): 하나의 점으로부터의 반경
- minPts (**M**inimum **P**oints, 최소점): 군집을 이루기 위한 최소한의 데이터 수
 - 예를 들어 minPts = 4 라고 가정하면,
4개 이상의 데이터가 모여야 군집을 형성 할 수 있다
2개 또는 3개의 데이터들이 모인 것(데이터가 모두 어떤 군집에 속하지 않음)은 군집으로 인정하지 않는다

<명칭 요약>



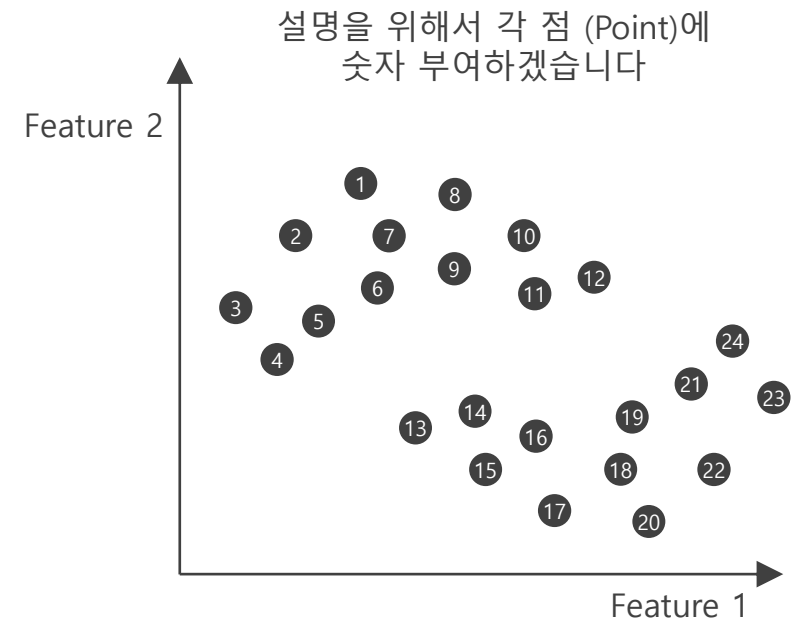
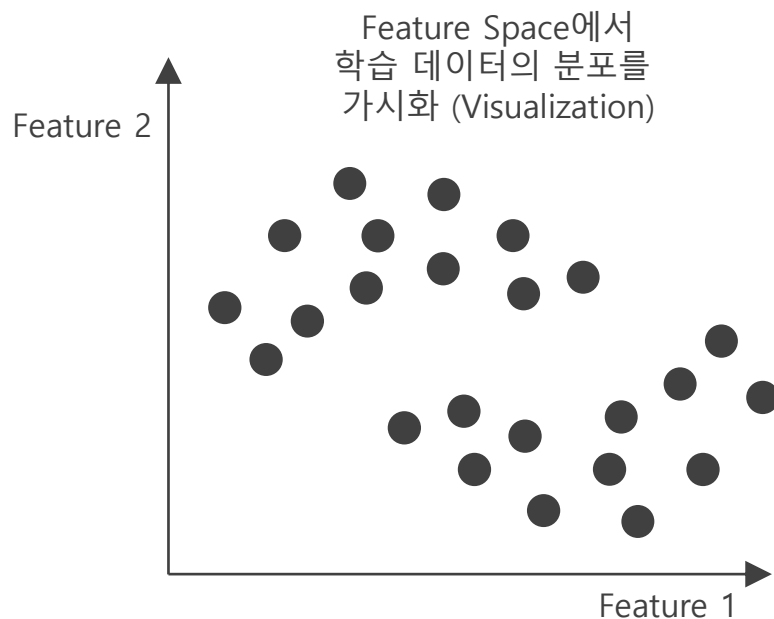
[사진출처] <https://bcho.tistory.com/1205>

[사진출처] <https://untitledblog.tistory.com/146>

② 밀도기반 클러스터링 (DBSCAN) (3/12)

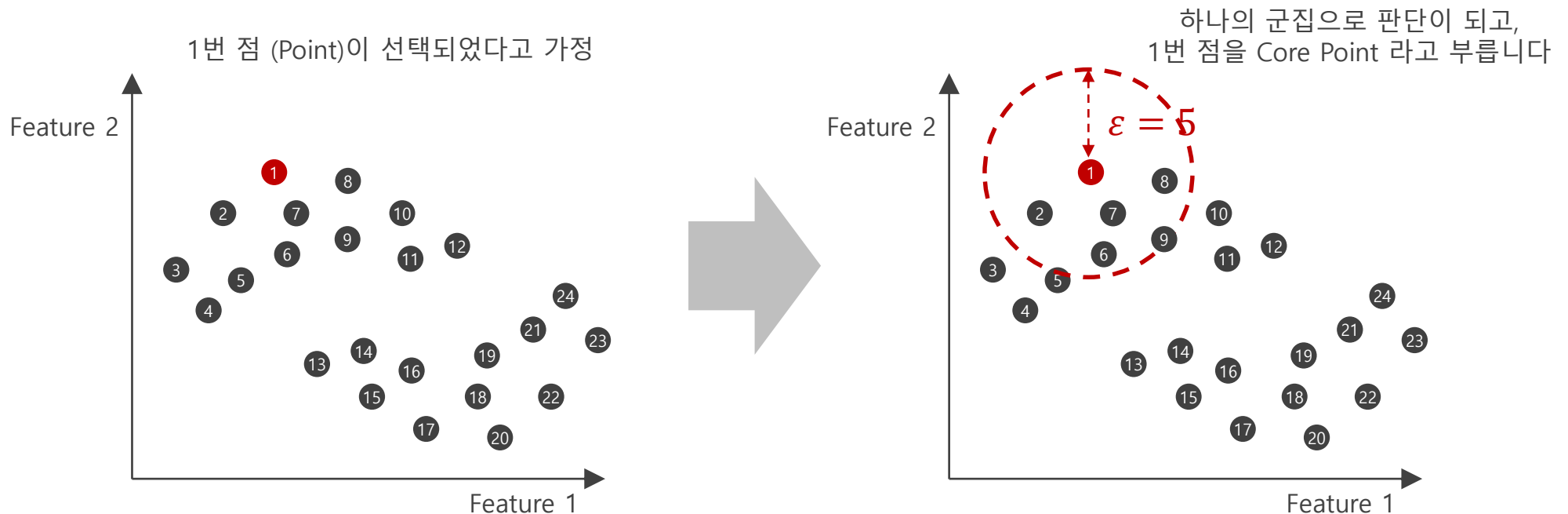
인공지능 활용 Python language

- 밀도기반 클러스터링의 진행 과정 ($\epsilon = 5$, minPts = 4 라고 가정)
 - Step 0) 학습 데이터 (Training Data)를 준비



② 밀도기반 클러스터링 (DBSCAN) (4/12)

- 밀도기반 클러스터링의 진행 과정 ($\epsilon = 5$, $\text{minPts} = 4$ 라고 가정)
 - Step 1) 임의로 한 점 (Point)을 선택 (1번 점이 선택되었다고 가정)
 - Step 2) 선택 된 점을 중심으로 반경 ϵ 거리에 minPts 이상의 데이터가 있는지 확인

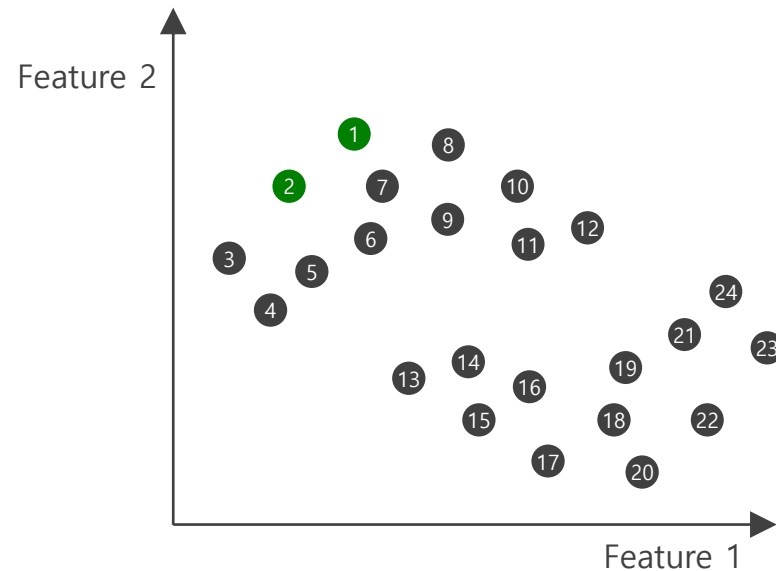
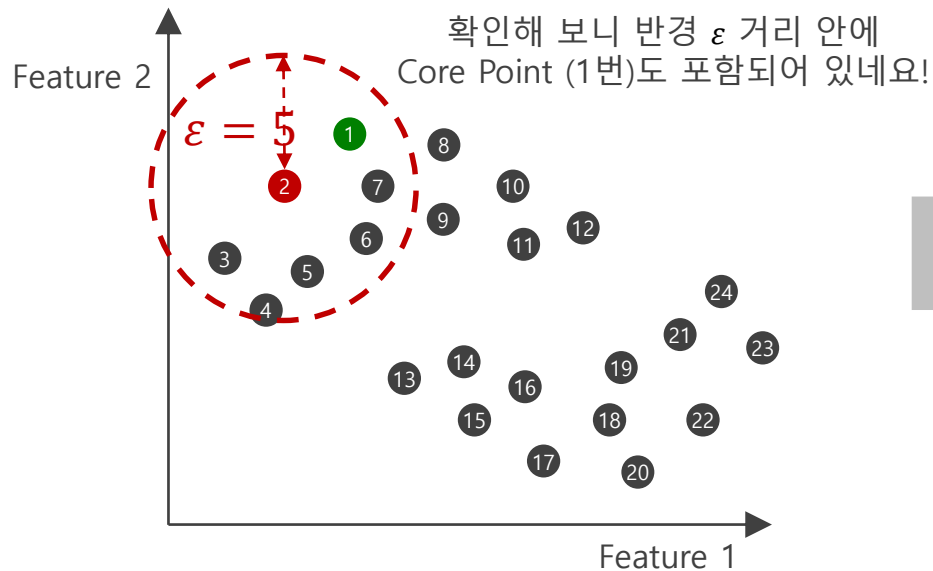


② 밀도기반 클러스터링 (DBSCAN) (5/12)

• 밀도기반 클러스터링의 진행 과정 ($\varepsilon = 5$, minPts = 4 라고 가정)

- Step 3)

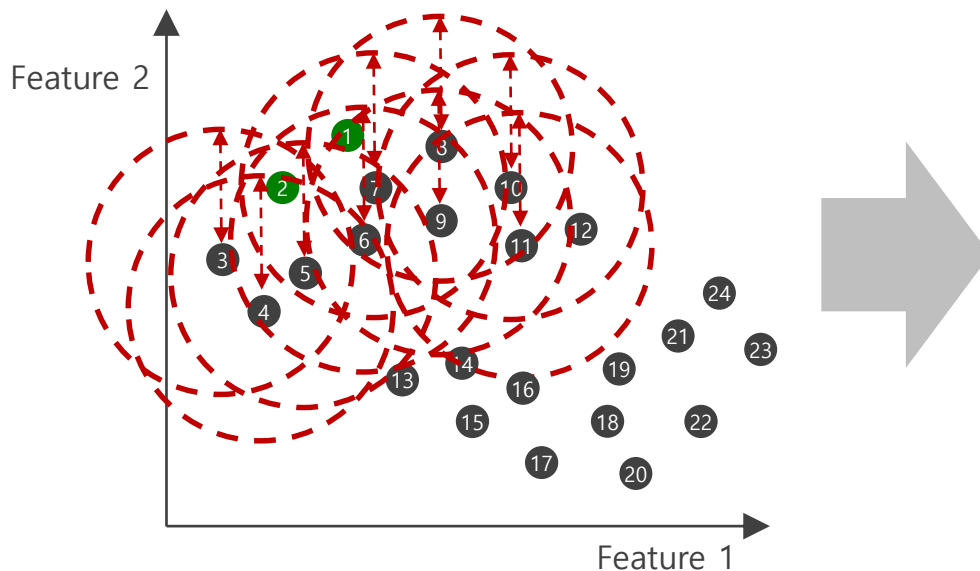
- 다음으로 선택된 점을 중심으로 반경 ε 거리에 minPts 이상의 데이터가 있는지 확인합니다
- 반경 ε 거리에 minPts 이상의 데이터가 존재하고 동시에 Core Point가 포함되어 있다면
- 같은 군집으로 포함시킴



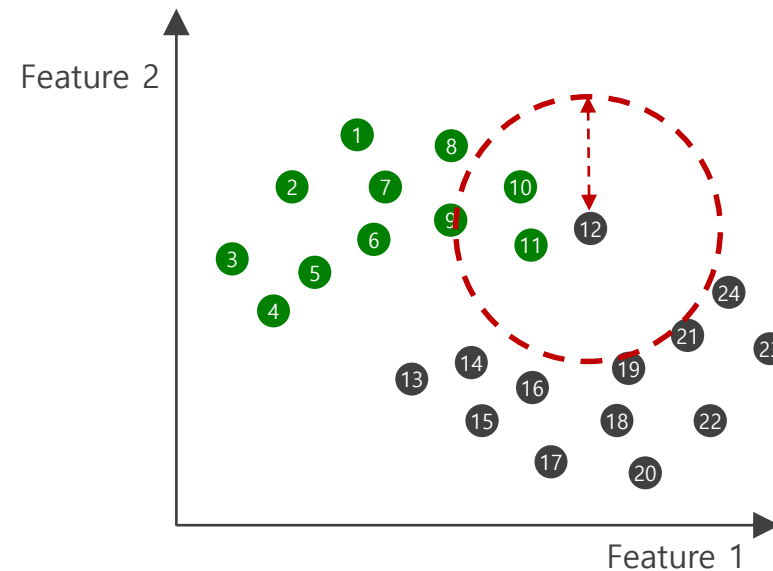
② 밀도기반 클러스터링 (DBSCAN) (6/12)

인공지능 활용 Python language

- 밀도기반 클러스터링의 진행 과정 ($\epsilon = 5$, $\text{minPts} = 4$ 라고 가정)
 - Step 4) 나머지 점들에 대해서 동일 작업을 반복 수행



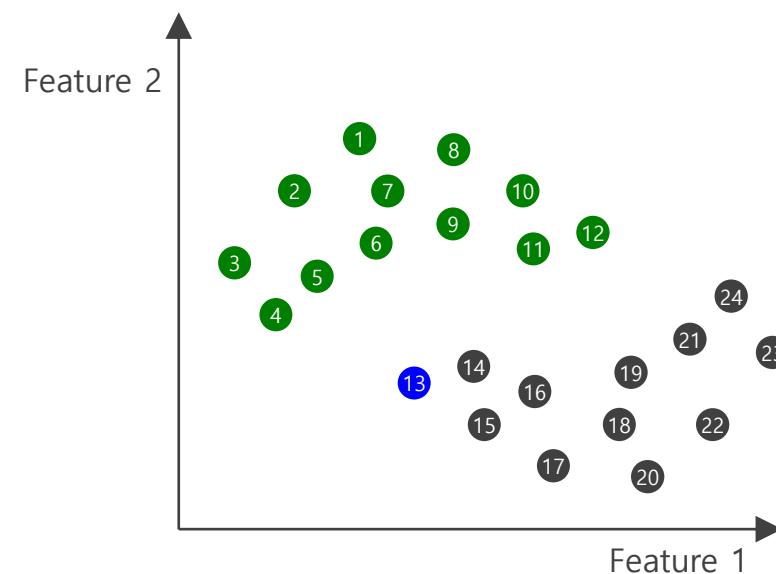
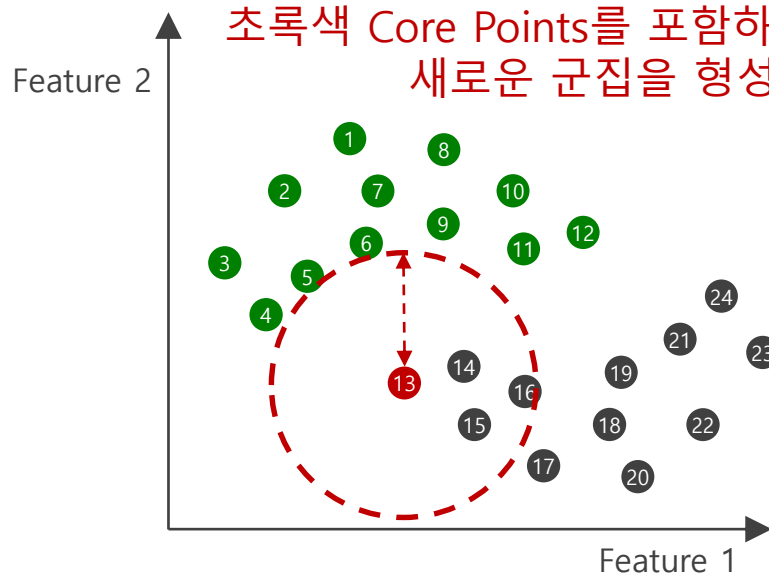
12번 점은 반경 내 점의 개수가 minPts보다 작기 때문에 Core Point가 될 수는 없지만 초록색 군집에는 속함.
이런 점 (Point)를 Border Point (경계점) 이라고 함



② 밀도기반 클러스터링 (DBSCAN) (7/12)

- 밀도기반 클러스터링의 진행 과정 ($\epsilon = 5$, $\text{minPts} = 4$ 라고 가정)
 - Step 4) 나머지 점들에 대해서 동일 작업을 반복 수행

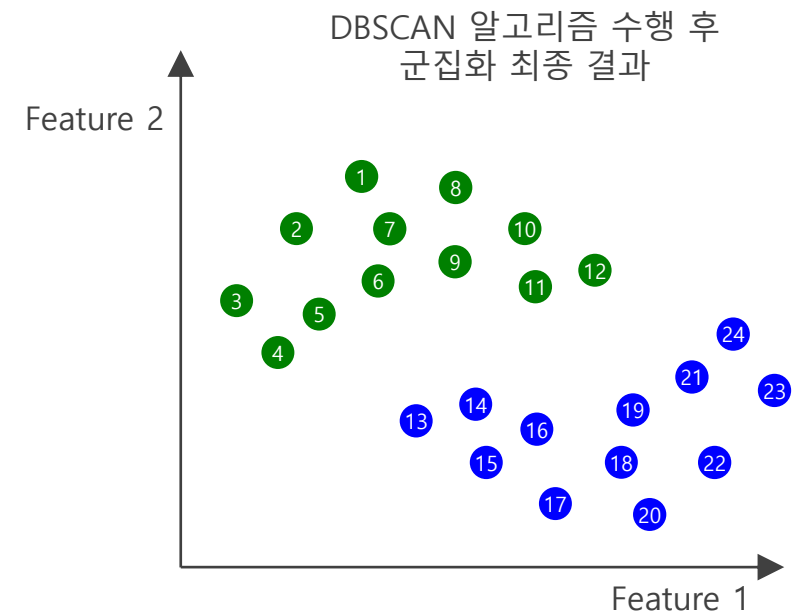
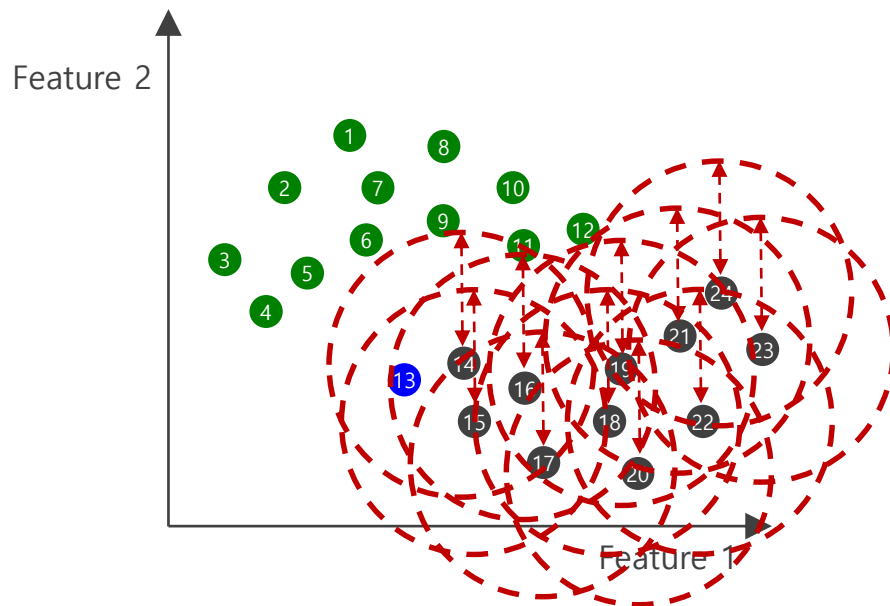
13번 점은 반경 내 4개의 데이터가 존재하니
Core Point가 되는데,
초록색 Core Points를 포함하지 않으므로
새로운 군집을 형성함



② 밀도기반 클러스터링 (DBSCAN) (8/12)

인공지능 활용 Python language

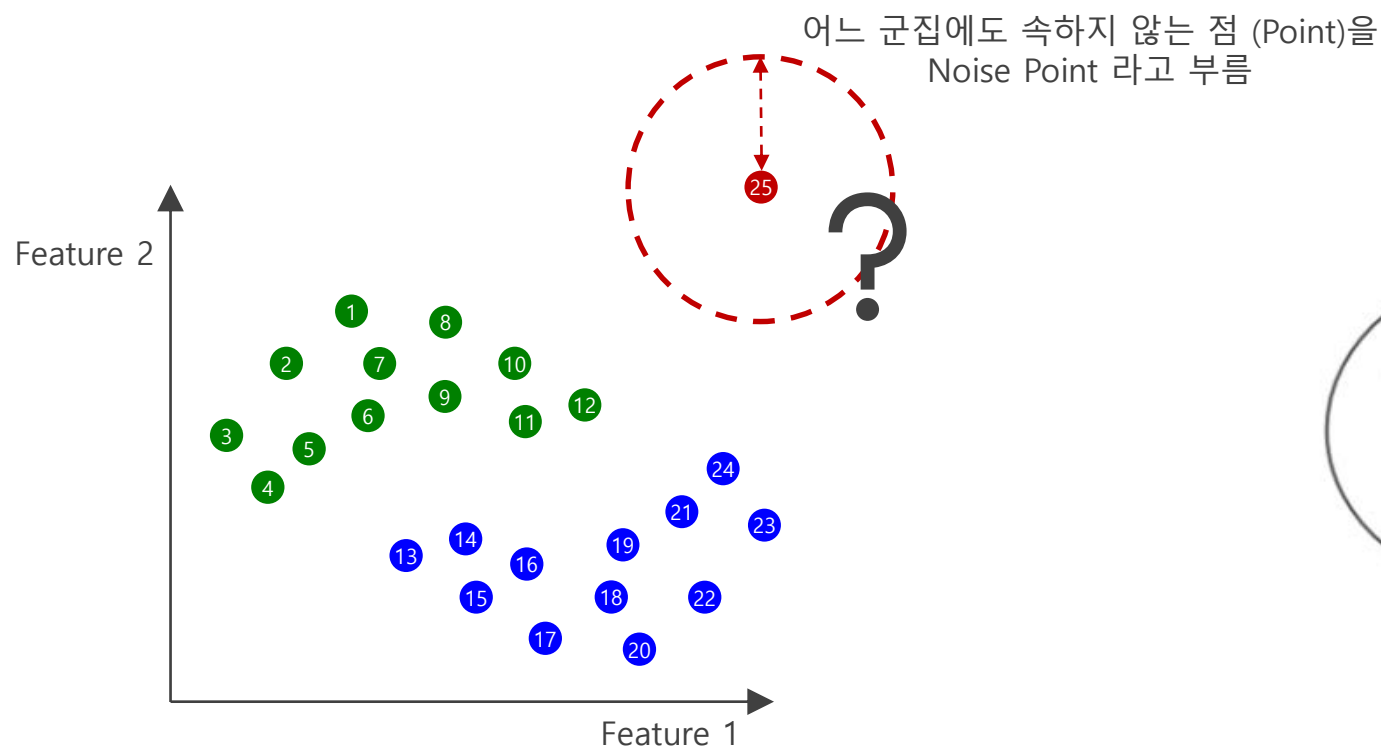
- 밀도기반 클러스터링의 진행 과정 ($\epsilon = 5$, $\text{minPts} = 4$ 라고 가정)
 - Step 4) 나머지 점들에 대해서 동일 작업을 반복 수행



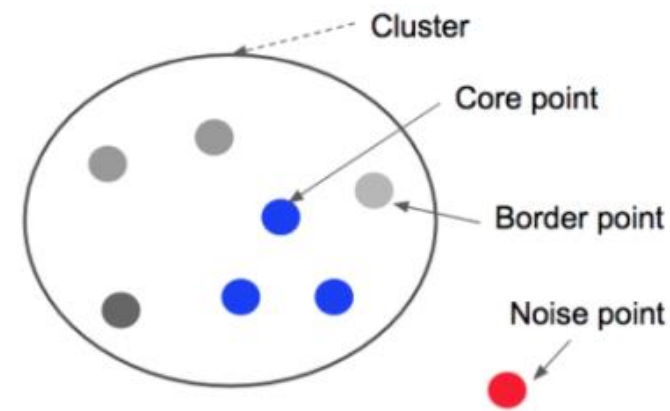
② 밀도기반 클러스터링 (DBSCAN) (9/12)

인공지능 활용 Python language

• Noise Point와 이상점 (Outlier) 제거



<명칭 요약>



[사진출처] <https://bcho.tistory.com/1205>

② 밀도기반 클러스터링 (DBSCAN) (10/12)

인공지능 활용 Python language

• 장점

- k -평균 군집화 (k -Means Clustering)와 달리 클러스터 수 ($= k$)를 지정할 필요가 없음
- k -평균 군집화가 표현할 수 없는 기하학적인 모양들을 가질 수 있다는 점임
 - 예를 들어, 밀도기반 클러스터링은 [그림 8-26]의 첫 번째 그림과 같이 다른 군집으로 둘러 싸인 상태에서 또 다른 군집을 가질 수 있음
- Noise Point를 통하여, 이상점 (Outlier) 검출이 가능 (Outlier를 제외하고 군집화를 진행)

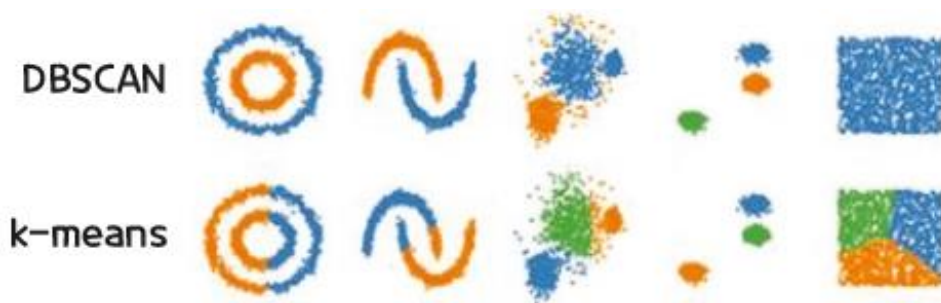
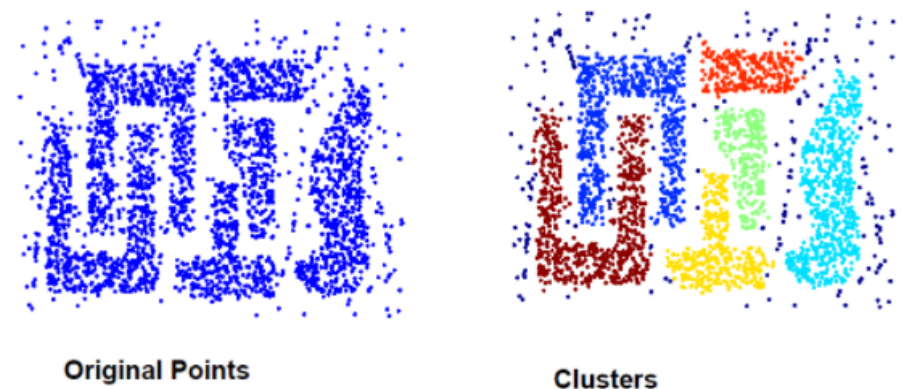


그림 8-26 밀도기반 클러스터링과 K-평균 군집화 비교

© Hanbit Academy Inc.



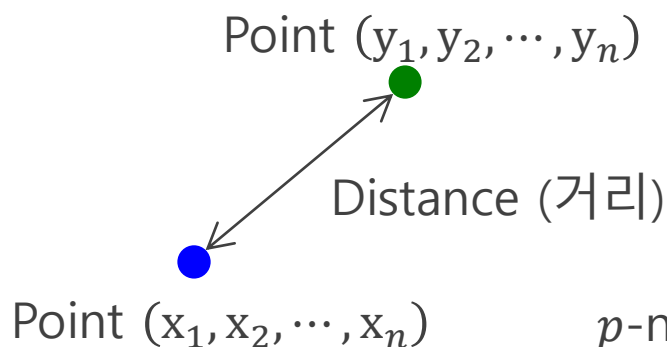
[사진출처] <https://lucy-the-marketer.kr/ko/growth/클러스터링과-dbscan/>

② 밀도기반 클러스터링 (DBSCAN) (11/12)

인공지능 활용 Python language

• 단점 (1/2)

- k -평균 군집화 (k -Means Clustering) 알고리즘보다 시간 복잡도 (Time Complexity)가 크다
 - 데이터 수가 많아지면 DBSCAN 알고리즘이 군집화 하는데 더 오래 걸림
- 군집화 성능을 올리기 위해서는 최적의 ϵ , minPts 값을 찾아야 함
- DBSCAN 알고리즘에서 사용되는 데이터 사이의 거리 측정 방법에 따라 군집화 결과가 변함
 - 유클리드 공간 (Euclidean Space)에서 거리 (Distance)



$$p\text{-norm distance} = \left(\sum_{i=1}^n |x_i - y_i|^p \right)^{1/p}$$

우리가 일상생활에서 흔히 사용하는 거리

$p = 2$ 인 경우, 2-norm distance (= Euclidean Distance, 유클리드 거리)

$$= \left(\sum_{i=1}^n |x_i - y_i|^2 \right)^{1/2} = \sqrt{(x_1 - y_1)^2 + \dots + (x_n - y_n)^2}$$

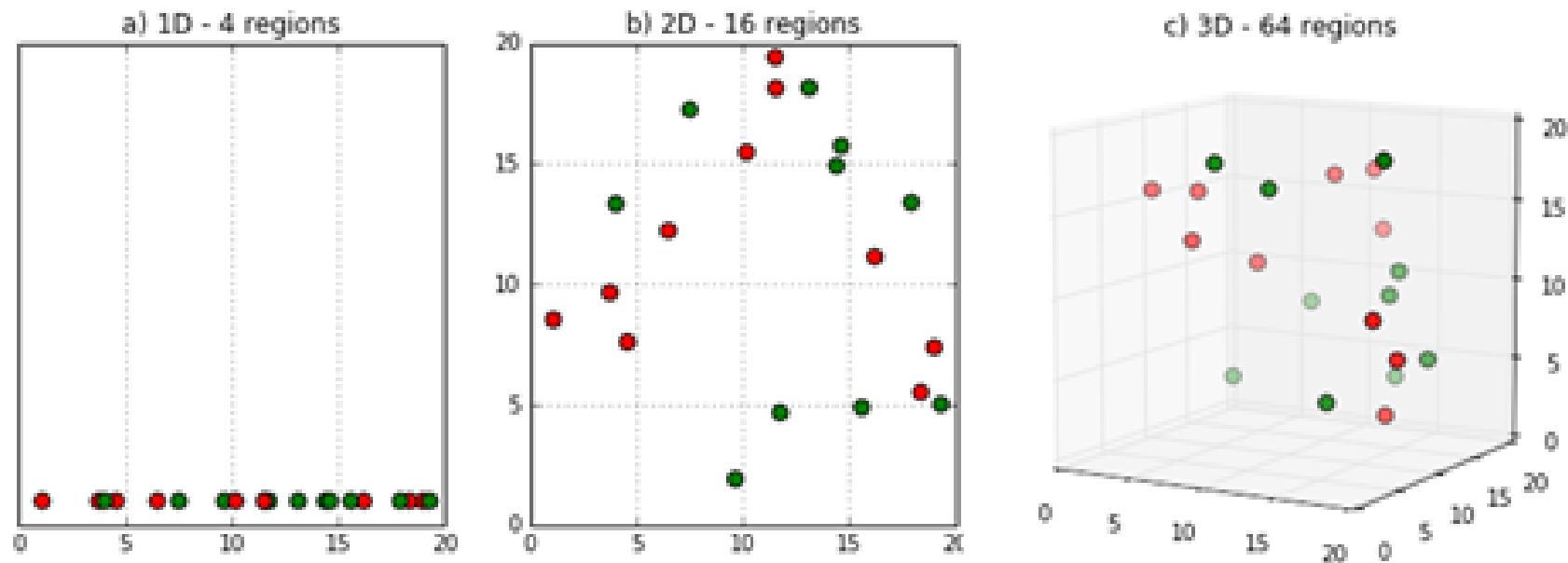
② 밀도기반 클러스터링 (DBSCAN) (12/12)

인공지능 활용 Python language

• 단점 (2/2)

- Feature Space의 차원 (Dimension)이 고차원으로 될수록 군집화 성능이 나빠짐
 - 차원의 저주 (= Curse of Dimensionality)

Feature Space의 차원이 높아지면 점 사이의 거리가 늘어나 서로 거리가 점점 멀어지고 군집화하기가 어려워짐



[사진출처] <https://deepai.org/machine-learning-glossary-and-terms/curse-of-dimensionality>

AI Experts
Who Lead
The Future

04

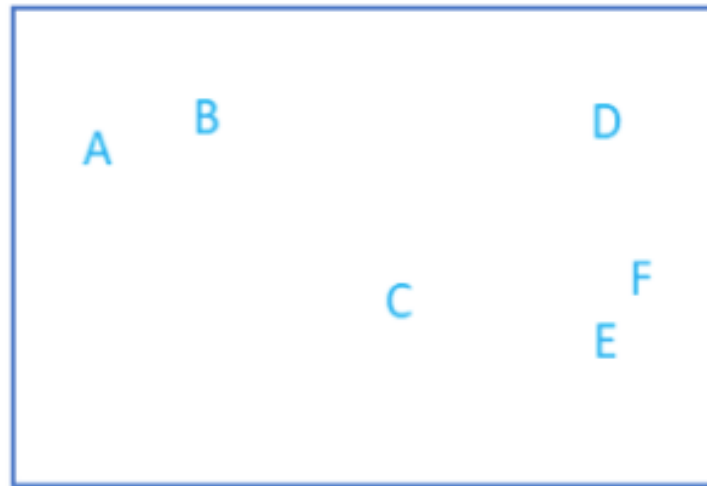
계층적 군집화

다음 자료를 기반으로 제작
난생처음 인공지능 입문 (출판사: 한빛아카데미)

③ 계층적 군집화 (Hierarchical Clustering) (1/11)

인공지능 활용 Python language

- 계층적 트리 (Tree) 모형 (= Dendrogram, 덴드로그램)을 이용하여 데이터들을 계층적으로 유사한 그룹으로 군집화를 수행하는 알고리즘
- 덴드로그램 (Dendrogram)이란?
 - 군집화(Clustering) 진행 과정과 결과를 시각화(Visualization)하기 위한 그래프 (Graph)의 일종



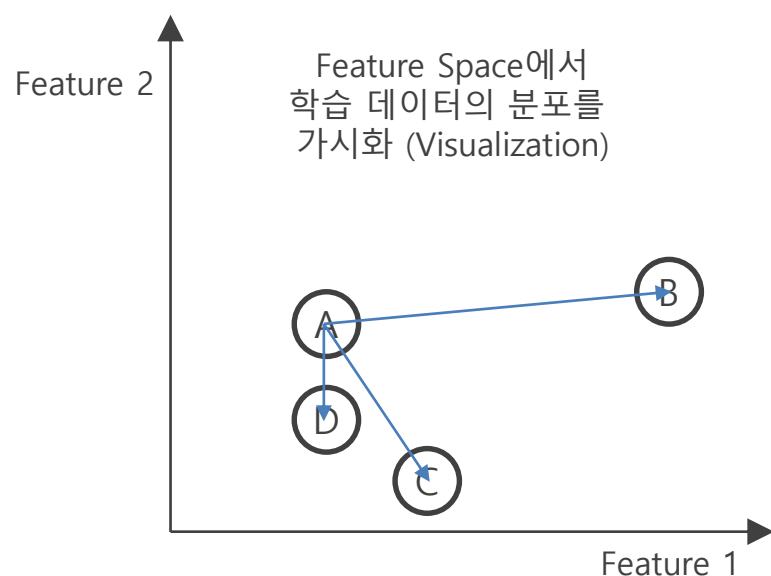
[사진출처] <https://www.displayr.com/what-is-dendrogram/>

③ 계층적 군집화 (Hierarchical Clustering) (2/11)

인공지능 활용 Python language

• 계층적 군집화 진행 과정

- **Step 1)** 준비한 학습 데이터셋 (Training Dataset)에서 각 데이터 사이의 거리 (Distance) 또는 유사도 (Similarity)를 계산



A B C D

거리 행렬 (Distance Matrix)

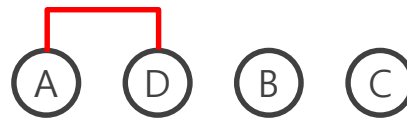
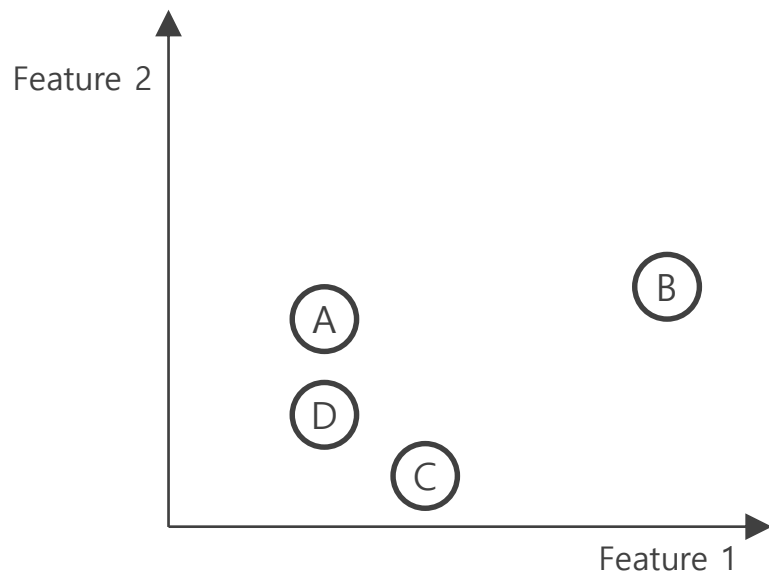
	A	B	C	D
A		20	7	2
B			10	25
C				3
D				

③ 계층적 군집화 (Hierarchical Clustering) (3/11)

인공지능 활용 Python language

• 계층적 군집화 진행 과정

- Step 2) 거리 행렬에서 가까운 거리를 찾고, 대응되는 데이터끼리 군집으로 묶음



거리 행렬 (Distance Matrix)

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

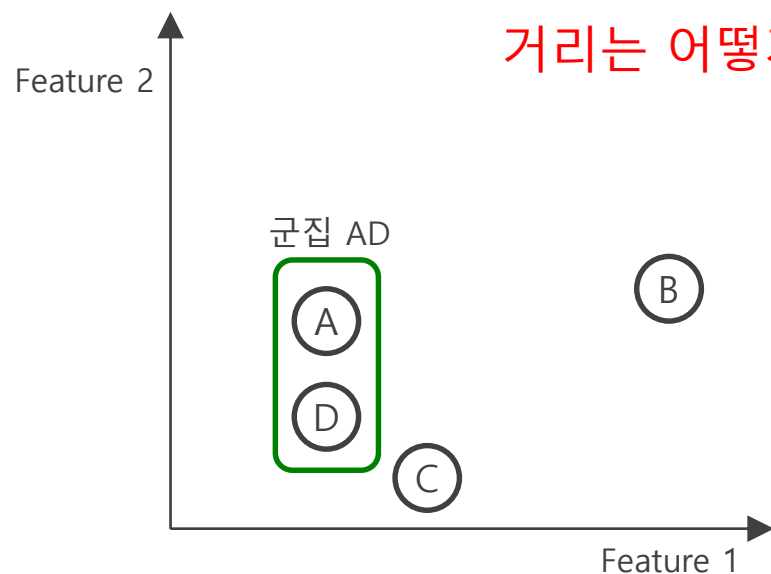
③ 계층적 군집화 (Hierarchical Clustering) (4/11)

인공지능 활용 Python language

• 계층적 군집화 진행 과정

- Step 3) A와 D가 군집으로 묶였으니, 거리 행렬을 업데이트 함

군집과 단일 개체사이의
거리는 어떻게 계산해야 하지?



거리 행렬 (Distance Matrix)

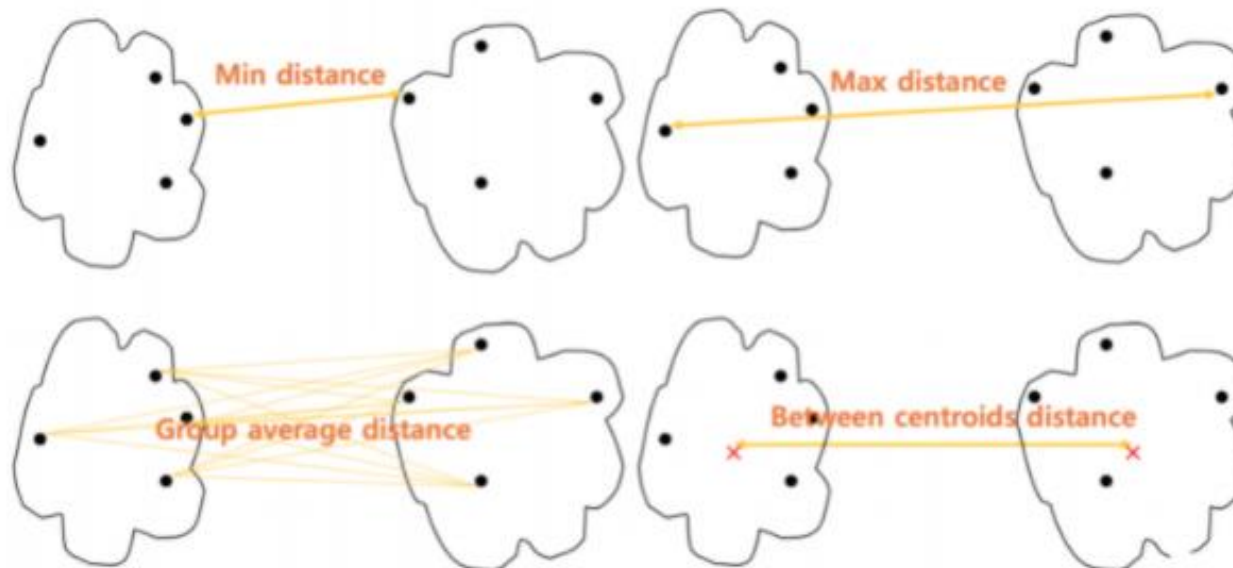
	AD	B	C	
AD				
B			10	
C				

③ 계층적 군집화 (Hierarchical Clustering) (5/11)

인공지능 활용 Python language

- 계층적 군집화 진행 과정
 - Step 3) A와 D가 군집으로 묶였으니, 거리 행렬을 업데이트 함

군집-개체 or 군집-군집 사이의 거리를 구하는 방법



[사진출처] <https://ratsgo.github.io/machine%20learning/2017/04/18/HC/>

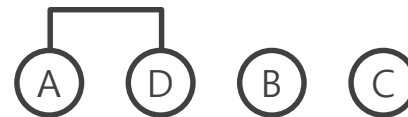
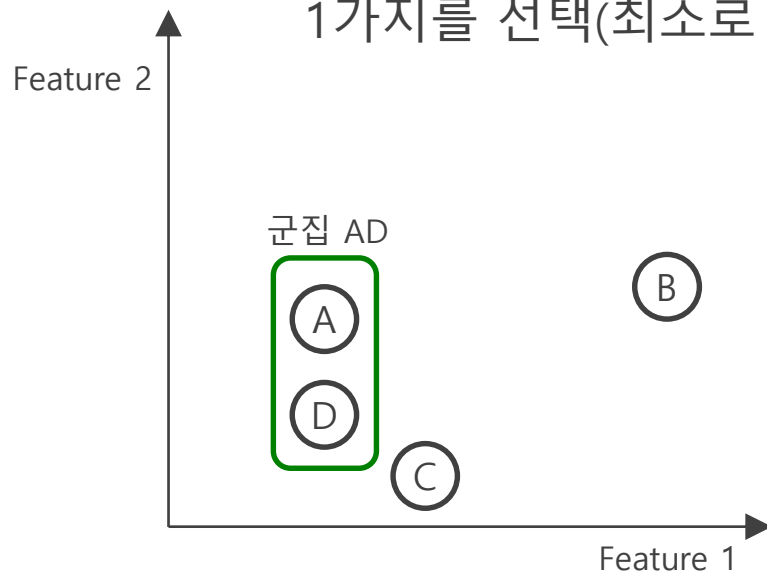
③ 계층적 군집화 (Hierarchical Clustering) (6/11)

인공지능 활용 Python language

- 계층적 군집화 진행 과정

- Step 3) A와 D가 군집으로 묶였으니, 거리 행렬을 업데이트 함

앞서 살펴본 4가지 거리 계산 방법 중
1가지를 선택(최소로 선택)하여 계산!



거리 행렬 (Distance Matrix)

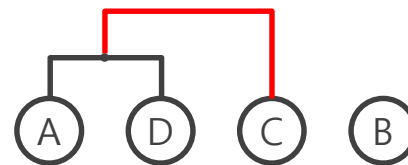
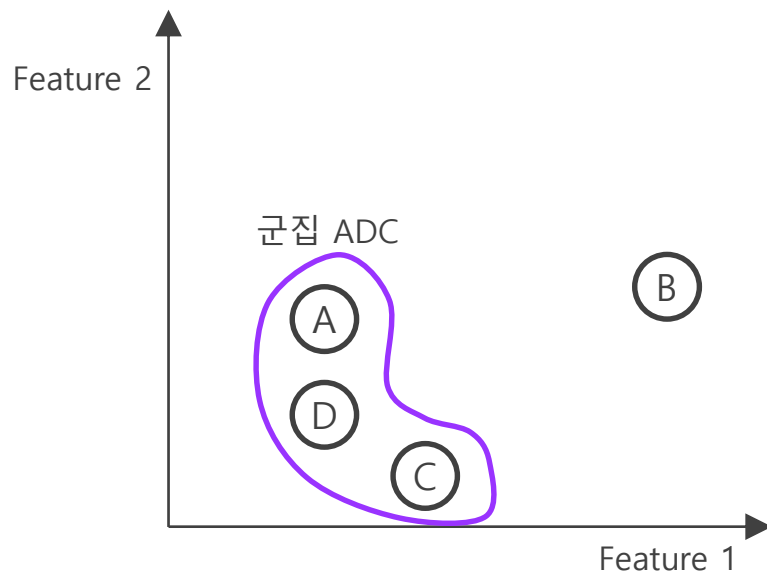
	AD	B	C	
AD		20	3	
B			10	
C				

③ 계층적 군집화 (Hierarchical Clustering) (7/11)

인공지능 활용 Python language

• 계층적 군집화 진행 과정

- Step 4) 거리 행렬에서 가까운 거리를 찾고, 대응되는 데이터끼리 군집으로 묶음



거리 행렬 (Distance Matrix)

	AD	B	C	
AD		20	3	
B			10	
C				

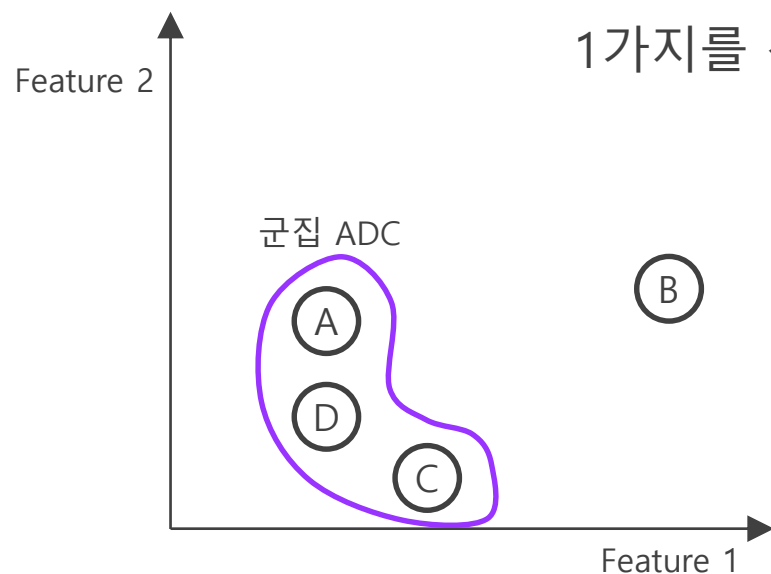
③ 계층적 군집화 (Hierarchical Clustering) (8/11)

인공지능 활용 Python language

• 계층적 군집화 진행 과정

- Step 5) AD와 C가 군집으로 묶였으니, 거리 행렬을 업데이트 함

앞서 살펴본 4가지 거리 계산 방법 중
1가지를 선택하여 계산!



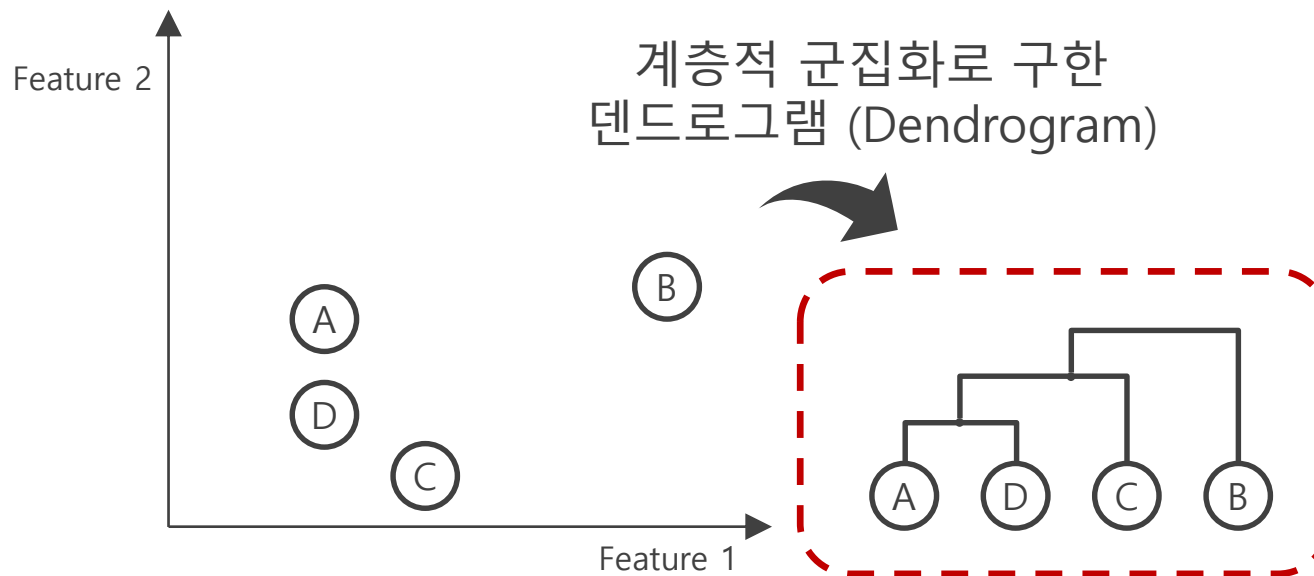
거리 행렬 (Distance Matrix)

	ADC	B		
ADC		10		
B				

③ 계층적 군집화 (Hierarchical Clustering) (9/11)

인공지능 활용 Python language

- 계층적 군집화 진행 과정
 - Step 6) 비교 대상이 없으므로, 군집화 학습 종료



거리 행렬 (Distance Matrix)

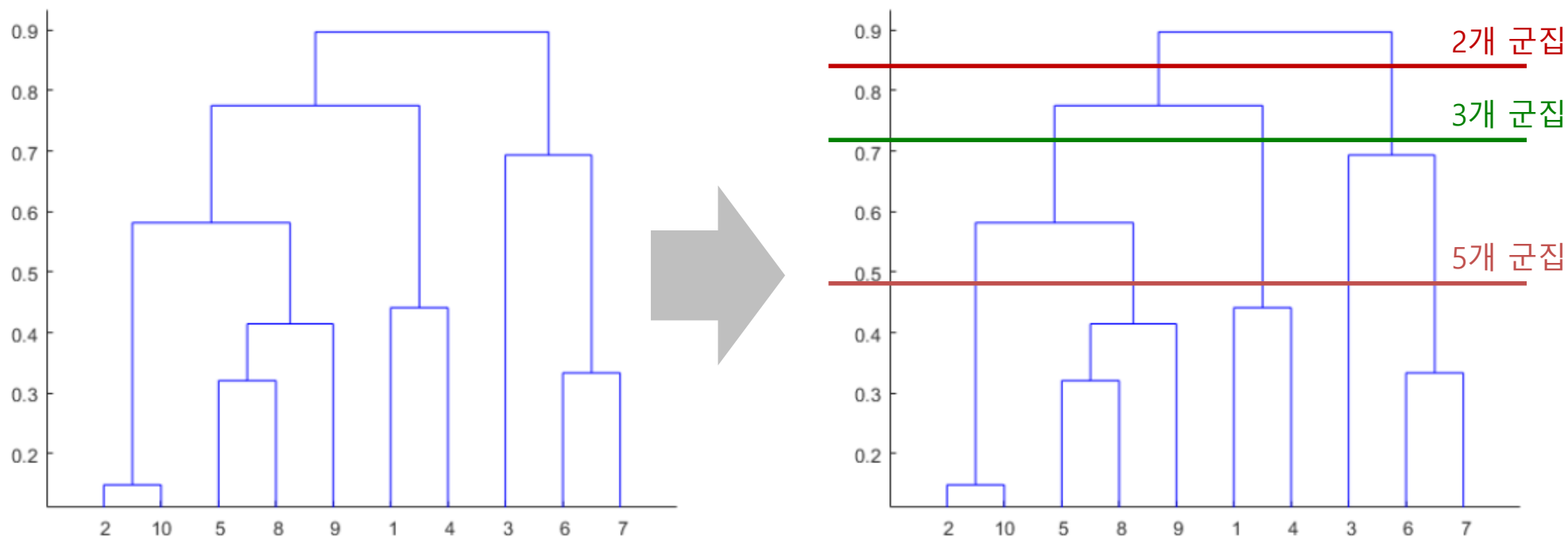
	A	D	C	B
A				
D				
C				
B				

③ 계층적 군집화 (Hierarchical Clustering) (10/11)

인공지능 활용 Python language

- 계층적 군집화 진행 과정

- Step 7) 덴드로그램 (Dendrogram)을 보고, 적절한 수준에서 트리를 잘라 최종 군집을 결정



③ 계층적 군집화 (Hierarchical Clustering) (11/11)

인공지능 활용 Python language

- **장점**

- 덴드로그램 (Dendrogram)을 통해 전체 군집들 사이의 구조적 관계를 쉽게 살펴볼 수 있음
- k -평균 군집화 (k -Means Clustering)와 달리 군집의 수를 사전에 결정할 필요가 없음

- **단점**

- 데이터셋이 매우 클 경우, 계산 속도가 느림 (Time Complexity가 높다)
- 이상치 (Outlier)에 의해서 전혀 엉뚱한 덴드로그램이 생성될 수 있음
- 일단 데이터가 특정 군집에 할당되면, 다른 군집으로 포함이 불가능 하기 때문에 때때로 잘못된 군집 결과를 초래할 수 있음

AI Experts
Who Lead
The Future

05

군집화 평가

다음 자료를 기반으로 제작
난생처음 인공지능 입문 (출판사: 한빛아카데미)

군집화 알고리즘의 적정성을 평가하는 방법 (1/6)

인공지능 활용 Python language

- 실루엣 분석 (Silhouette Analysis)
 - 각 군집 (Cluster) 간의 거리가 얼마나 잘 분리 되어 있는지 평가
- 실루엣 계수 (Silhouette Coefficients)
 - 평균 값이 1에 가까울 수록 군집화가 잘 되었다고 판정
 - 반대로, -1에 가까울 수록 군집화가 제대로 이루어지지 않았다고 판정

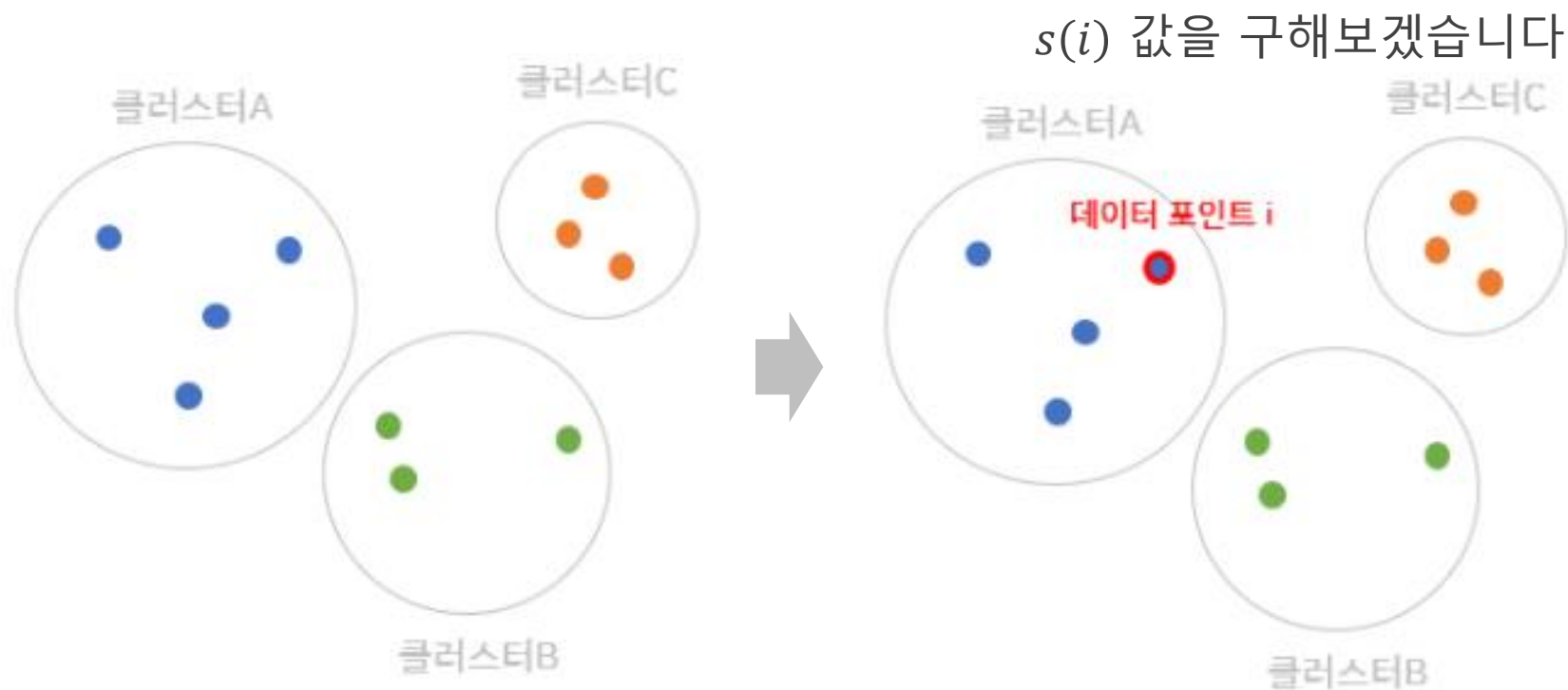


[사진출처] <https://studying-haeung.tistory.com/10>

군집화 알고리즘의 적정성을 평가하는 방법 (2/6)

인공지능 활용 Python language

- 실루엣 분석 (Silhouette Analysis) 절차를 살펴봅시다
 - $s(i)$: i 번째 데이터의 실루엣 계수(Silhouette Coefficients)

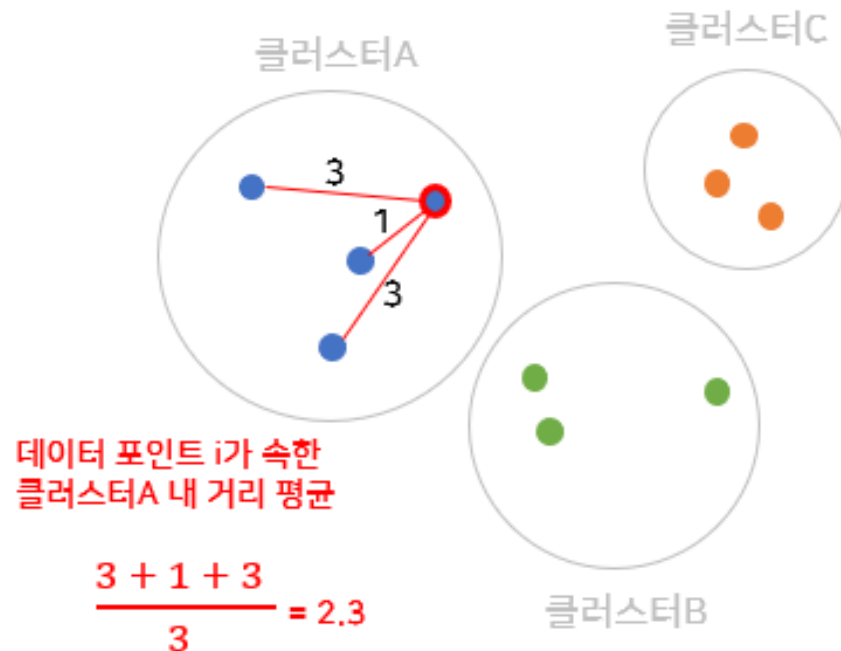


[사진출처] <https://studying-haeung.tistory.com/10>

군집화 알고리즘의 적정성을 평가하는 방법 (3/6)

인공지능 활용 Python language

- 실루엣 분석 (Silhouette Analysis) 절차를 살펴봅시다
 - $s(i)$: 데이터 포인트 i 의 실루엣 계수 (Silhouette Coefficients)



$a(i)$: 데이터 포인트 i 가 속한 클러스터 내 다른 데이터 포인트 사이의 평균 거리

$a(i) = 2.3$

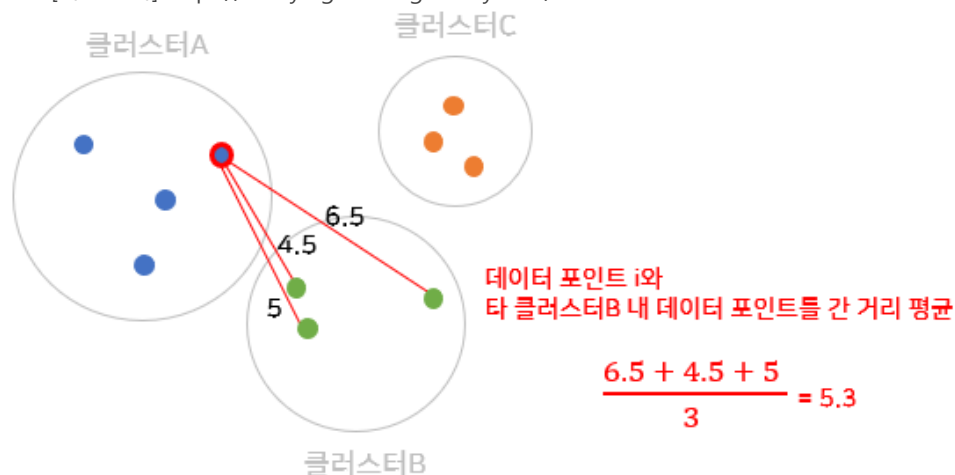
[사진출처] <https://studying-haeung.tistory.com/10>

군집화 알고리즘의 적정성을 평가하는 방법 (4/6)

인공지능 활용 Python language

- 실루엣 분석 (Silhouette Analysis) 절차를 살펴봅시다
 - $s(i)$: 데이터 포인트 i 의 실루엣 계수 (Silhouette Coefficients)

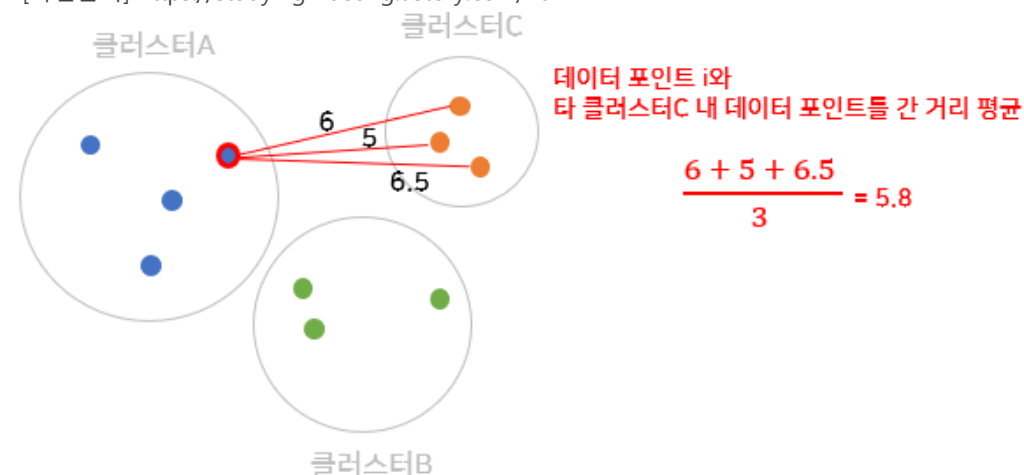
[사진출처] <https://studying-haeung.tistory.com/10>



$d(i, B)$: 데이터 포인트 i 와 클러스터B의 데이터 포인트 사이의 평균 거리

$$d(i, B) = 5.3$$

[사진출처] <https://studying-haeung.tistory.com/10>



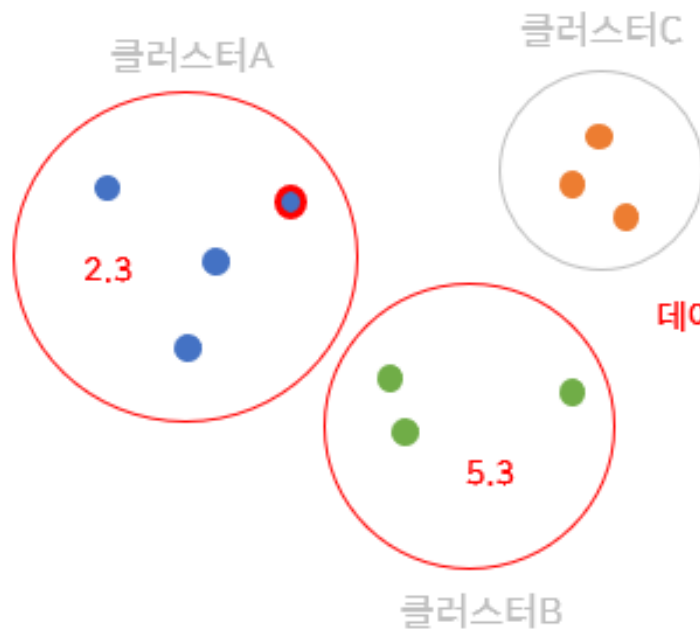
$d(i, C)$: 데이터 포인트 i 와 클러스터C의 데이터 포인트 사이의 평균 거리

$$d(i, C) = 5.8$$

군집화 알고리즘의 적정성을 평가하는 방법 (5/6)

인공지능 활용 Python language

- 실루엣 분석 (Silhouette Analysis) 절차를 살펴봅시다
 - $s(i)$: 데이터 포인트 i 의 실루엣 계수 (Silhouette Coefficients)



[사진출처] <https://studying-haeung.tistory.com/10>

$$\begin{aligned}
 b(i) &= \min\{d(i, B), d(i, C)\} \\
 &= \min\{5.3, 5.8\} \\
 &= 5.3
 \end{aligned}$$

$$\frac{5.3 - 2.3}{\max(5.3, 2.3)} = \frac{3}{5.3} = 0.57$$

$$\begin{aligned}
 s(i) &= \frac{b(i) - a(i)}{\max\{a(i), b(i)\}} \\
 &= 0.57
 \end{aligned}
 \quad -1 \leq s(i) \leq 1$$

군집화 알고리즘의 적정성을 평가하는 방법 (6/6)

인공지능 활용 Python language

- 실루엣 분석 (Silhouette Analysis) 절차를 살펴봅시다
 - $s(i)$: 데이터 포인트 i 의 실루엣 계수 (Silhouette Coefficients)



일련의 과정을 모든 데이터 포인트에 대해서 수행하여, 평균 $s(i)$ 값을 계산

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

\bar{s} 값이 1에 가까울 수록 군집화가 잘 되었다고 판정
-1에 가까울 수록 군집화가 제대로 이루어지지 않았다고 판정

[사진출처] <https://studying-haeung.tistory.com/10>

AI Experts
Who Lead
The Future

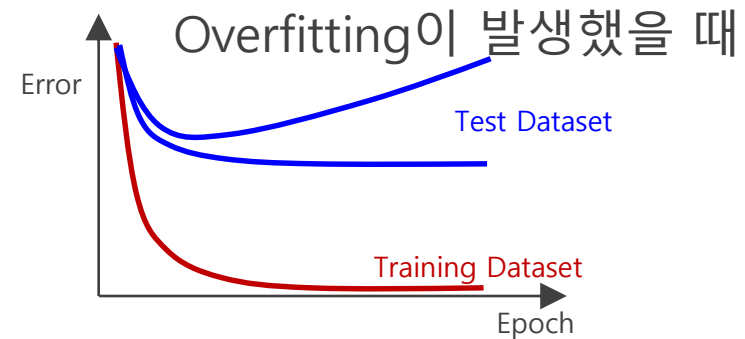
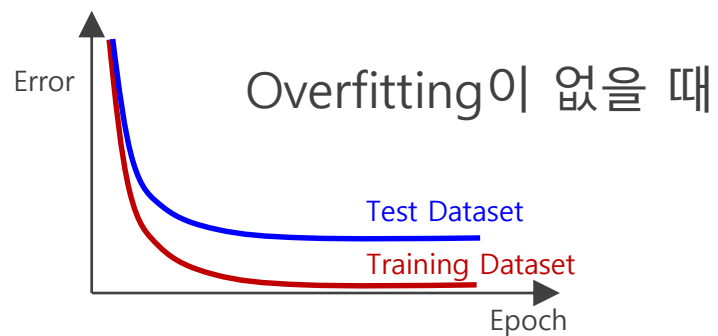
06

과적합

다음 자료를 기반으로 제작
난생처음 인공지능 입문 (출판사: 한빛아카데미)

• 과적합 (Overfitting)

- 머신러닝 모델이 학습 데이터 (Training Data)를 너무 과하게 학습하여, 시험 데이터 (Test Data)를 이용하여 모델을 평가할 때는 성능이 좋지 못한 것을 의미



• Overfitting이 발생하는 원인

- 1) 문제의 복잡도에 비해 학습 데이터 (Training Data)가 현저히 부족한 경우
 - 즉, 문제가 정의 된 전체 공간을 학습 데이터가 아우르지 못하고 일부 경우에만 집중했을 때 발생함
- 2) 데이터는 충분하지만, 머신러닝 모델이 너무 복잡한 경우

과적합 현상 2/4

인공지능 활용 Python language

- (a) 과소적합
 - 변수가 0인 쪽의 데이터 몇 개는 비교적 잘 근사하지만, 일정 시점 이후 데이터는 우하향하고 있음
- (b) 최적합
 - 데이터와 비슷하게 우상향하고 있어 제대로 반영하고 있다고 볼 수 있음
- (c) 과적합
 - 학습 데이터와 생성된 모델의 오차 (Error)를 구해보면 0에 가까울 것임
즉, 그래프가 모든 학습 데이터 (Training Data)를 지나고 있음
- 그렇다면 (b)보다 (c)가 더 좋은 그래프일까?
어떤 그래프가 좋은 그래프인지 알아보기 위해서는 시험 데이터 (Test Data)로 평가해 보면 됨

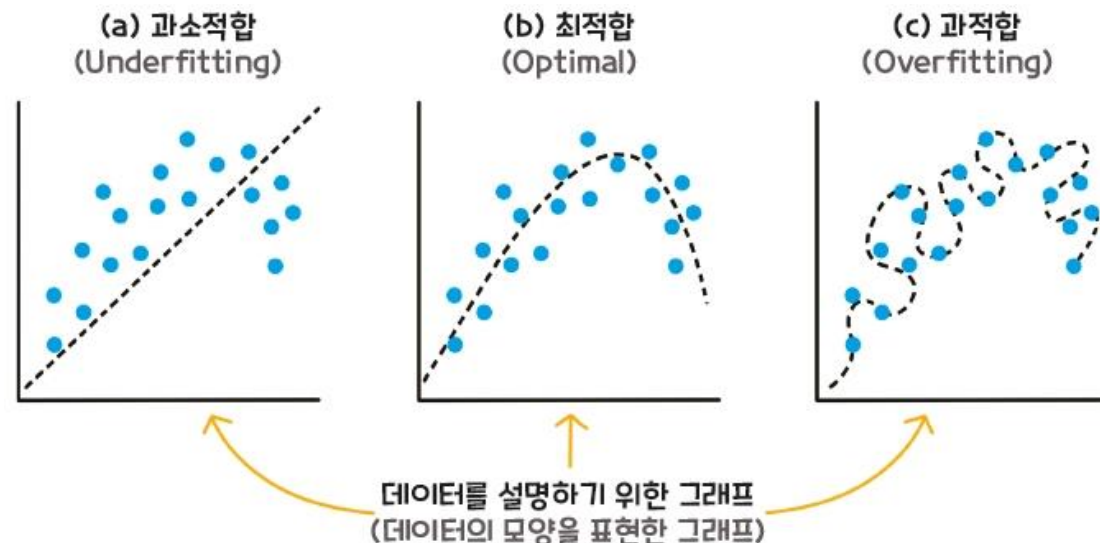


그림 8-33 과소적합, 최적합, 과적합 그래프

© Hanbit Academy Inc.

과적합 현상 3/4

- [그림 8-34]와 같이 하나의 시험 데이터 값을 불러왔을 때,
 - 시험 데이터 값과 머신러닝 모델이 내놓은 예측값 사이의 차이 (= 오차)가 가장 작은 그래프가 좋은 그래프라고 할 수 있음
 - 확인해 보면 (b)의 오차가 가장 작으므로 최적합(Optimal),
 - 즉 가장 좋은 그래프라고 할 수 있음

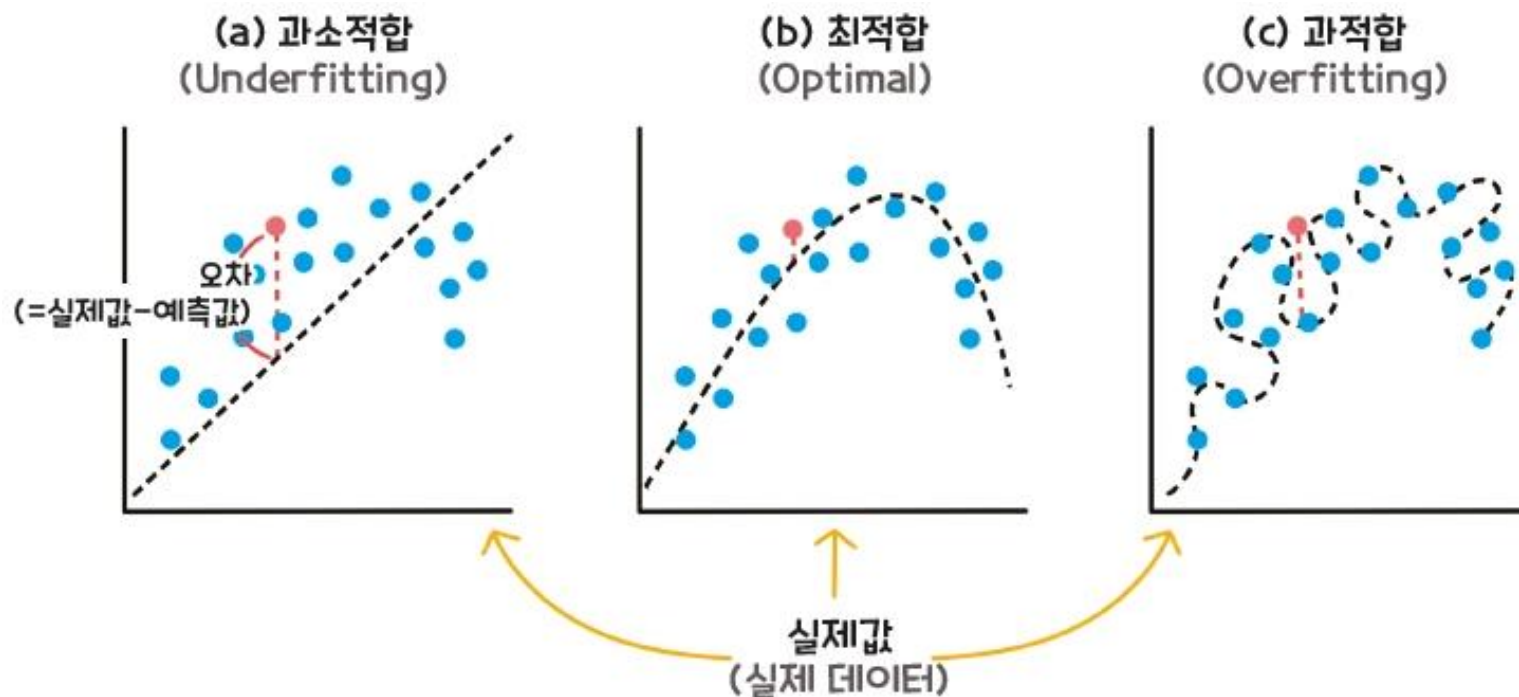


그림 8-34 과적합 판단 방법

© Hanbit Academy Inc.

• 과적합 현상이 발생하는 것을 막기 위한 방법

- 풀고자 하는 문제를 잘 대표할 수 있는 (양질의) 데이터들을 충분히 많이 확보해야 함
- 학습 데이터와 시험 데이터가 편향된 데이터들로 구성되지 않게 구성해야 함
- 너무 복잡한 모델을 사용하지 않아야 함



- 머신러닝은 유연성이 부족함
 - 학습에서 접하지 못했던 전혀 새로운 입력에 대해 오류 발생하거나 오차가 크게 나타남
 - 머신러닝은 데이터로 시작해서 데이터로 끝나는 기술
- 다른 사람이 만들어 놓은 모델은 재활용이 가능하더라도 데이터는 공유 어려움
- 실제로 공유된 데이터를 사용할 수도 있지만 분석하고자 하는 변수 중 일부가 누락된 경우가 많기 때문에 공유된 데이터를 이용한 분석은 그 목적에서 벗어난 경우가 많음
- 따라서 제대로 학습을 하려면 원하는 결과를 위한, 목적에 맞는 “나만의 데이터”가 필요함
- 결국 데이터가 없다면 머신러닝 알고리즘도, 딥러닝 알고리즘도 적용 어려움



다른 사람들이 만들고,
공유한 데이터
(= 공용 데이터셋, Public Dataset)



우리가 풀고자 하는 문제/목적에
바로 적용 가능한 공용 데이터셋은
거의 없음



결국 (쉽지 않지만)
“나만의 데이터”를
직접 만들어야 함