

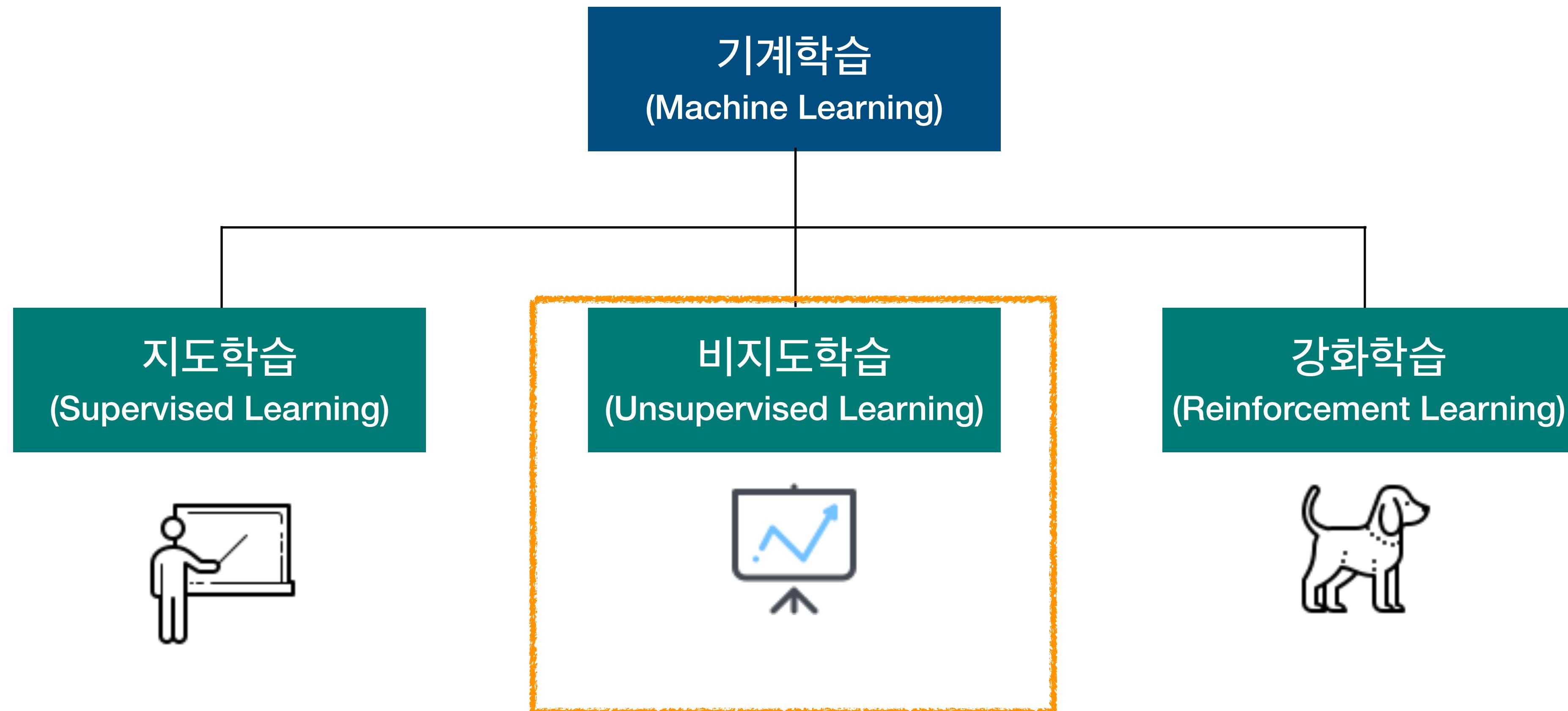
# 빅데이터 분석 프로그래밍

## Application

### Week-01. Unsupervised Learning

Jungwon Seo, 2022-Fall

# 기계학습의 종류



# 비지도 학습의 예

## 군집화와 차원축소

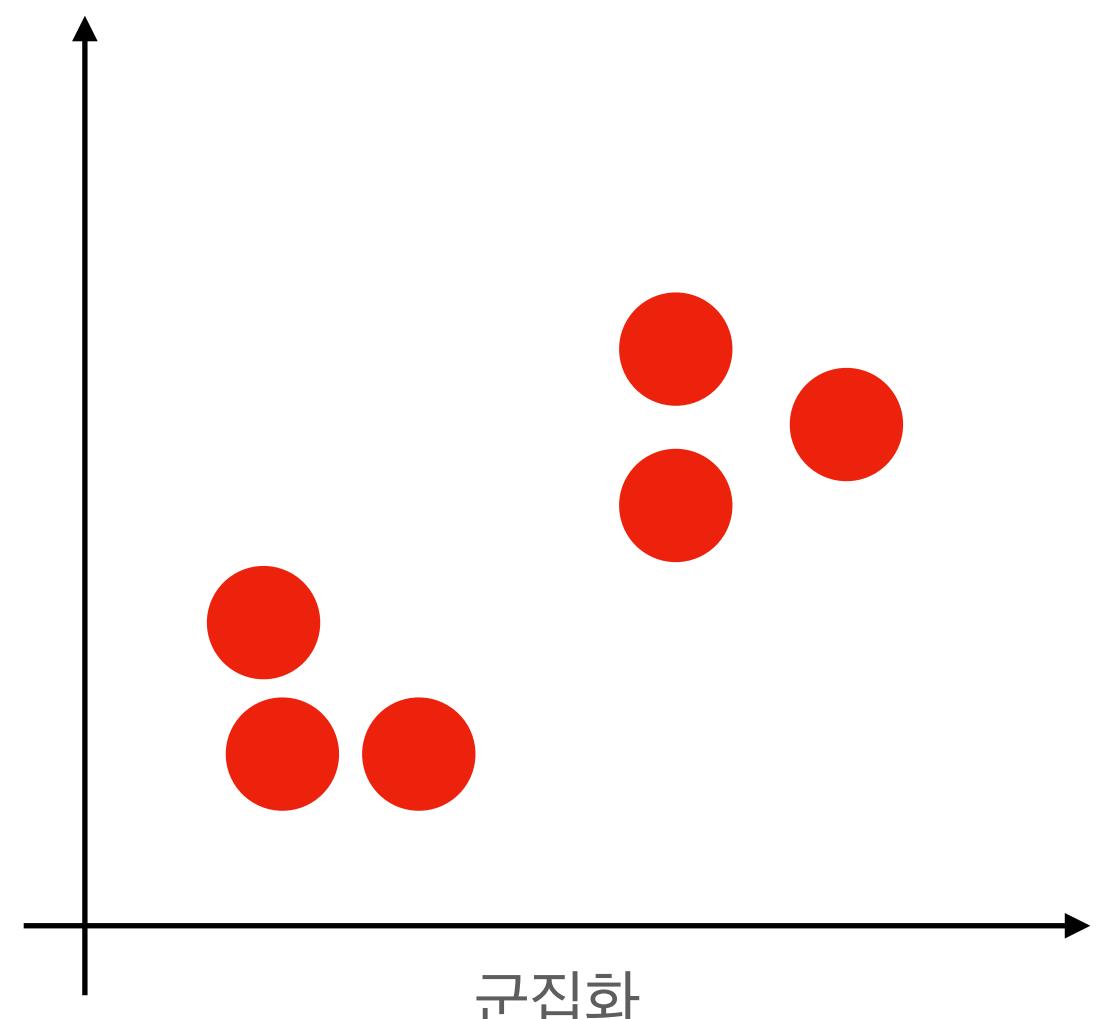
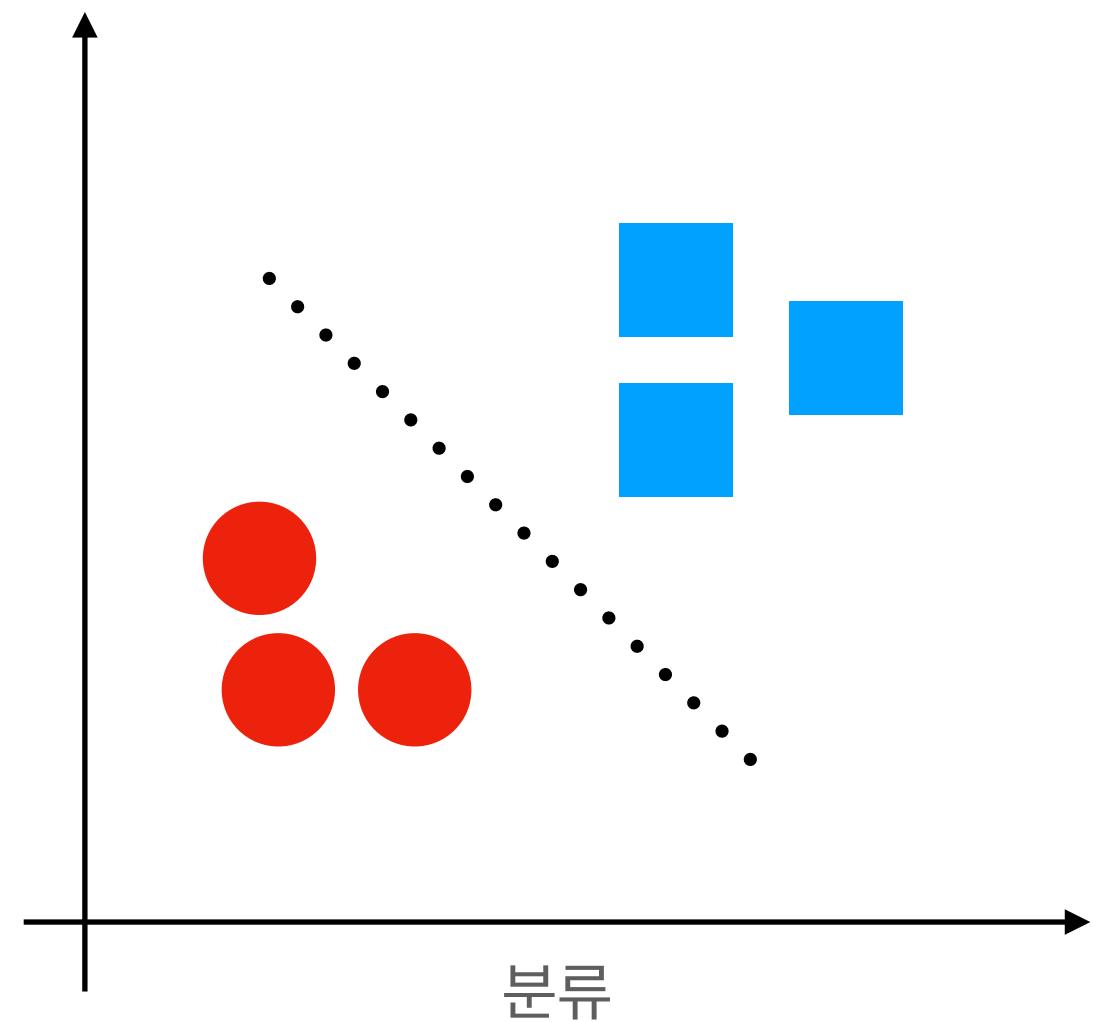
- 군집화 (Clustering)
  - 주어진 데이터를 기반으로 비슷한 부류의 데이터끼리 묶는 과정
  - 예:
    - Targeted 마케팅
    - 유저/소비자 Segmentation
    - 추천 시스템
- 차원축소 (Dimensionality reduction)
  - 고차원의 데이터를 저차원으로 축소하는 과정
  - 이상적인 차원 축소는, 적은 차원으로 원본의 데이터의 특징을 손실 없이 표현 할 수 있는 경우
  - 예:
    - 데이터 압축
    - 빅데이터 시각화

# Clustering

# 군집화 (Clustering)

Label이 없지만, 분류를 해야한다

- 지도학습의 분류문제와의 차이점
  - 분류문제와 다르게, 클러스터링은 사전에 샘플이 어느 클래스에 속하는지 알 수 없는 상황에서 진행
  - 데이터 사이에 내재된 관계를 통해 같은 클래스에 속한 샘플들의 유사도를 높게 만드는 몇 개의 클래스로 분류
- 군집화의 접근법
  - **Partitional clustering**
  - Hierarchical clustering



# K-Means Clustering

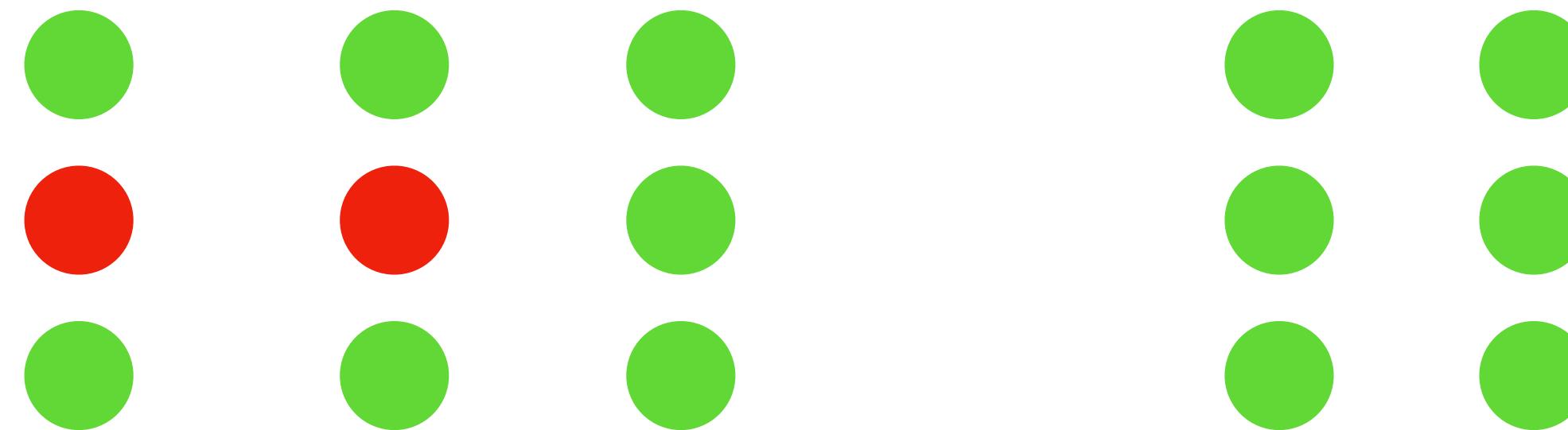
## K 평균 군집

- 가장 기초적이고 자주 사용되는 클러스터링 알고리즘
- 알고리즘
  1. 데이터 전처리(정규화 또는 Outlier 제거)
  2. 랜덤으로 K개의 군집 중심 선택 ( $\mu_1^{(0)}, \mu_2^{(0)}, \dots, \mu_k^{(0)}$ )
  3. Cost function 정의  $J(c, \mu) = \min_{\mu} \min_c \sum_{i=1}^M \|x_i - \mu_{c_i}\|^2$ 
    1.  $x$ : 데이터 포인트,  $\mu$ : 중심(centroid),  $M$ : 전체 데이터 수,  $c$ : 군집
    4. 반복 횟수  $t = 0, 1, 2, \dots$ 를 지정. 그리고 다음 과정이  $J$ 에 수렴할 때까지 반복
      1. 각 샘플  $x_i$ 에 대해 거리가 가장 가까운 군집에 배정
      2. 각 군집  $k$ 에 대해 해당 군집의 중심(centroid)을 다시 계산

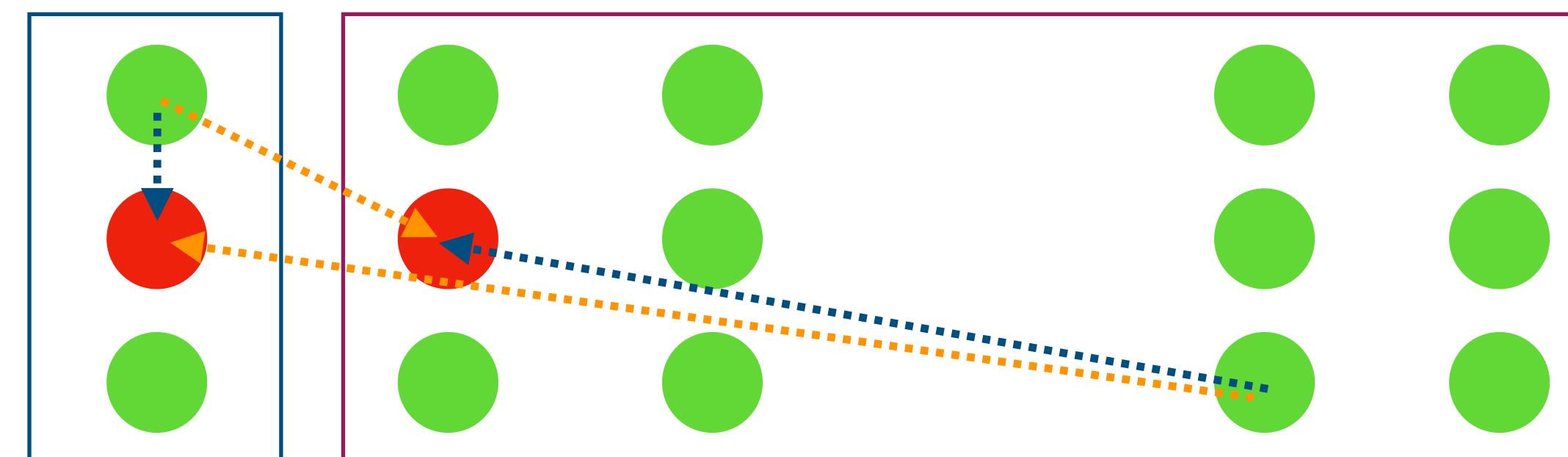
# K-Means Clustering

## Step by step

- 랜덤으로 K(예: 2) 개의 군집의 중심 선택



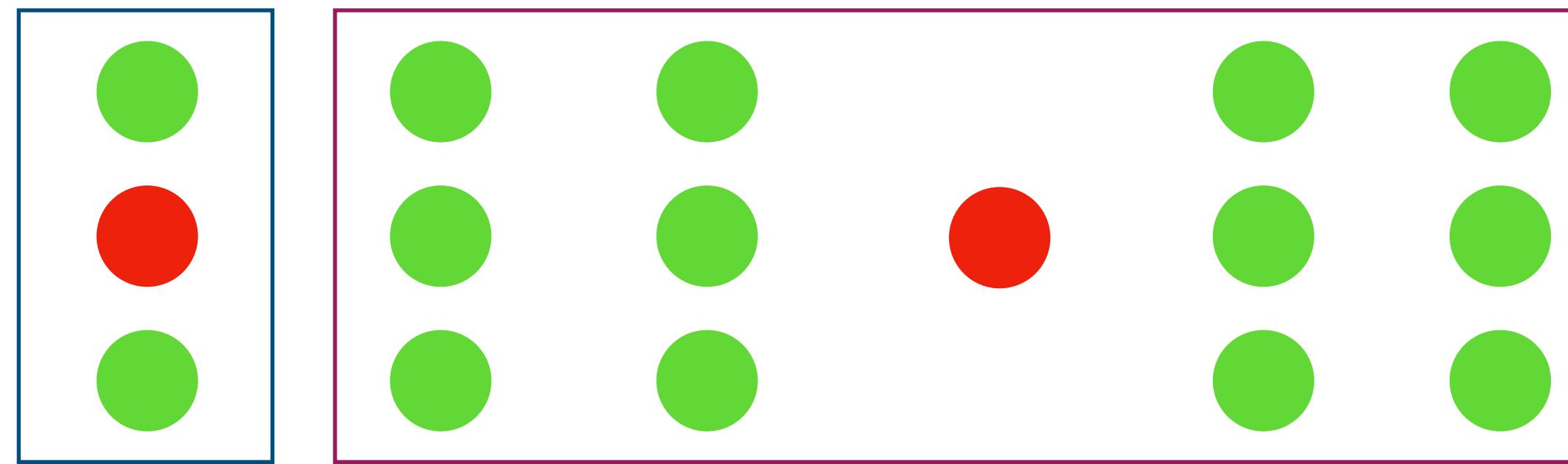
- 각각의 데이터 포인트와 군집의 중심과의 거리 계산후, 가까운 군집에 편성



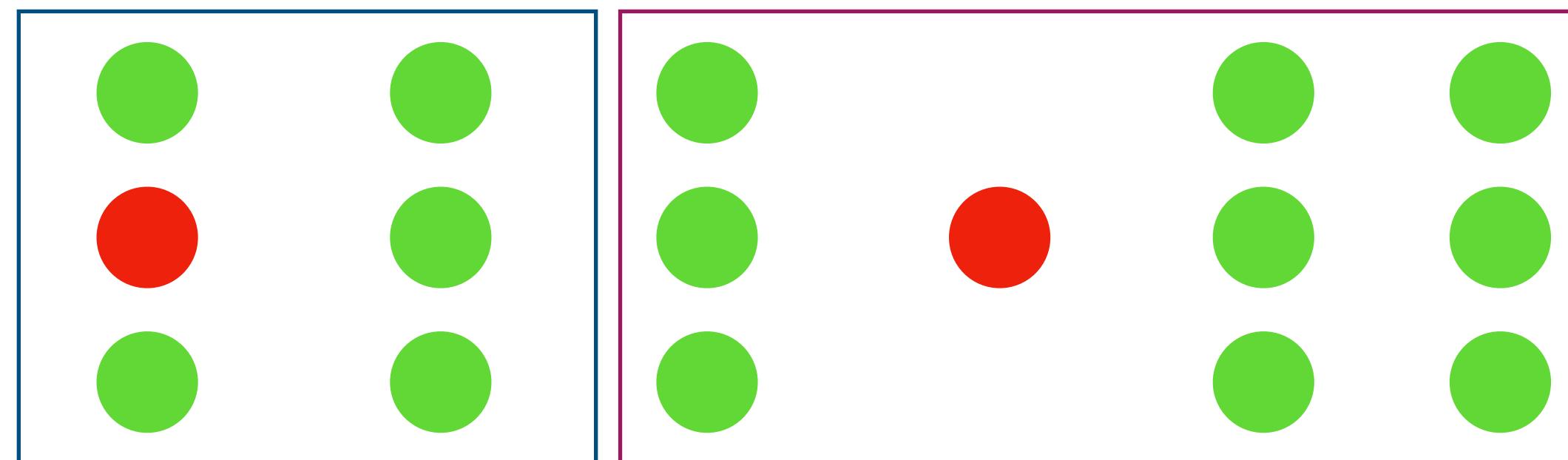
# K-Means Clustering

## Step by step

- 군집의 **중심점** 다시 계산



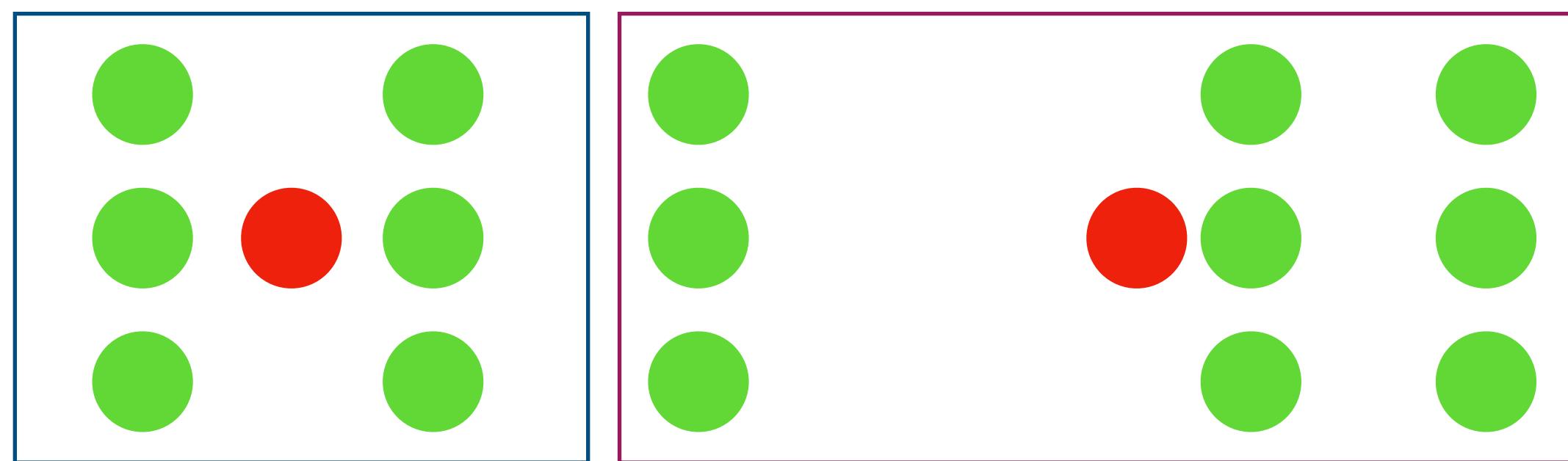
- 마찬가지로 거리 계산후 군집 재편성



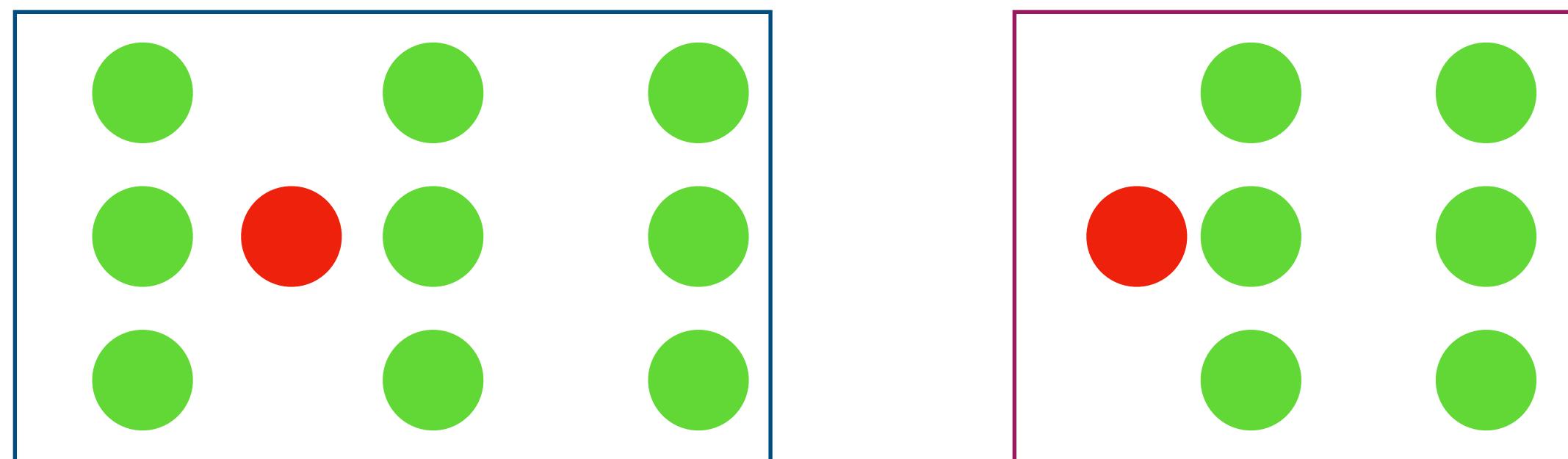
# K-Means Clustering

## Step by step

- 군집의 **중심점** 다시 계산



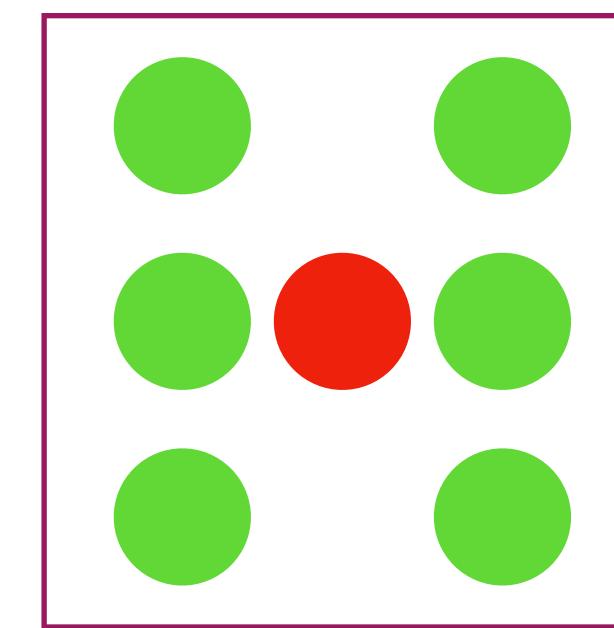
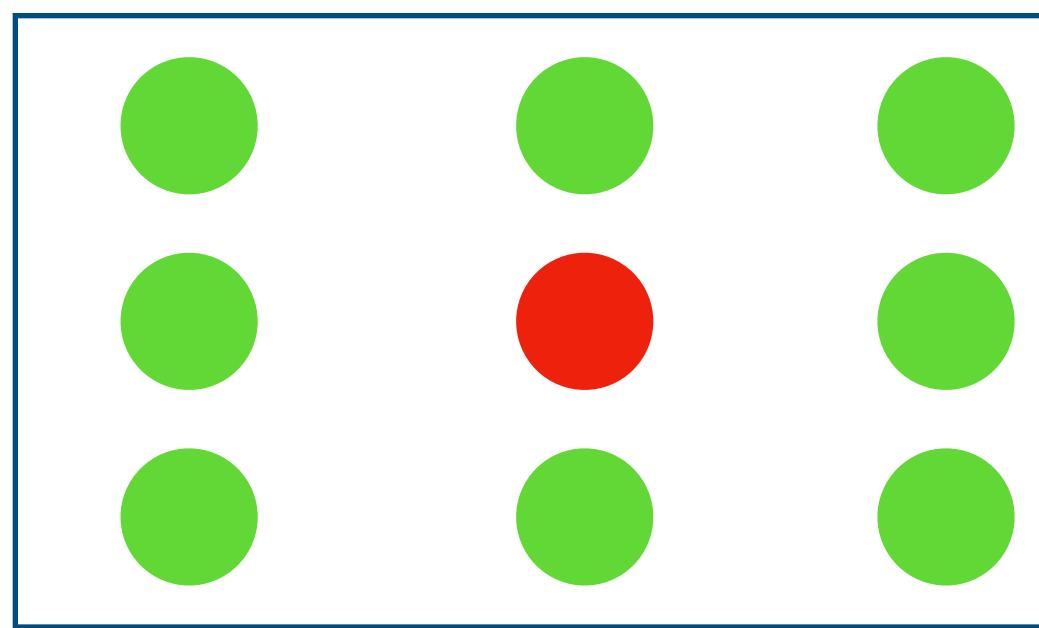
- 마찬가지로 거리 계산후 군집 재편성



# K-Means Clustering

## Step by step

- 군집의 **중심점** 다시 계산



- 결과적으로 Cost Function J가 가장 적게되는 방향으로 군집이 수렴

$$J(c, \mu) = \min_{\mu} \min_c \sum_{i=1}^M \| x_i - \mu_{c_i} \|^2$$

# K-Means Clustering

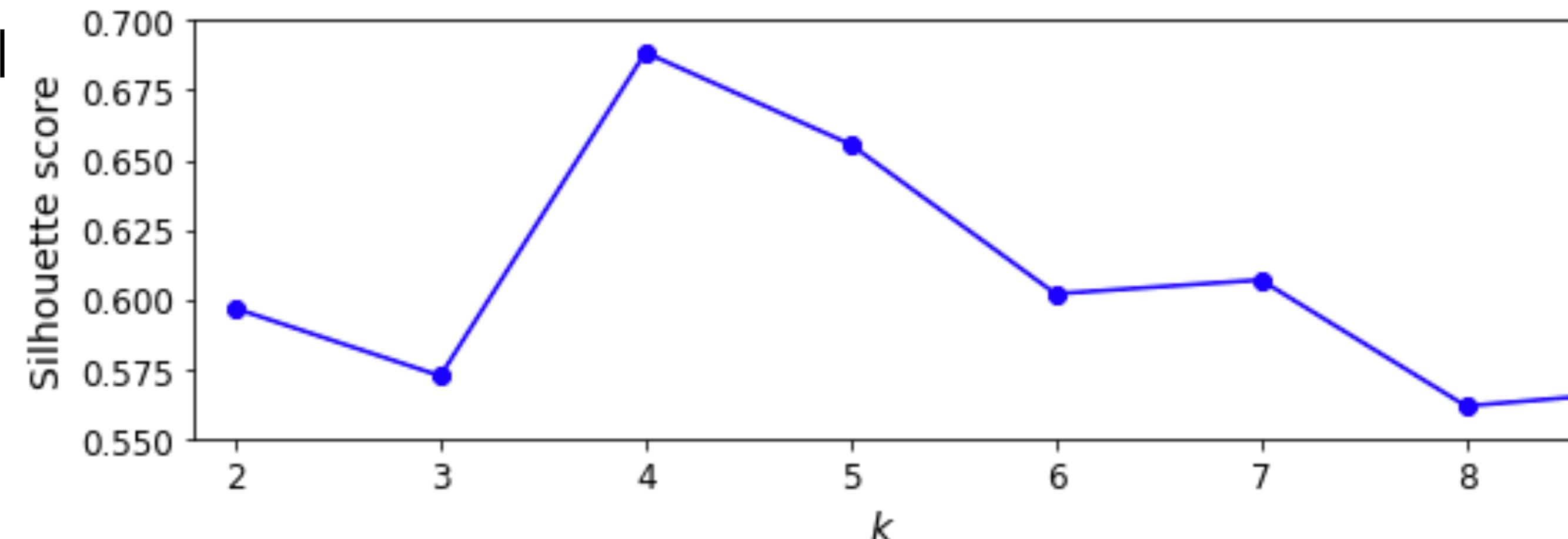
## Pros and Cons

- 장점
  - 빠른 속도와 용이한 확장성
  - 계산 복잡도  $O(NKt)$ , N: 데이터수, K: 군집수, t: 반복횟수
- 단점
  - 국소(Local) 최적을 피하려면 알고리즘을 여러번 돌려봐야 함
  - K를 사람이 직접 지정해줘야함
  - 클러스터의 크기와 밀집도가 다르거나 원형인 경우 잘 동작하지 않음
- 보완책
  - K-means++ : 초기 센트로이드에 사용, 다른 센트로이드와 거리가 먼 센트로이드를 선택하는 초기화 단계를 포함
  - ISODATA : K값을 정하기가 어려울 때 사용, 해당 군집의 샘플 수가 작아지면, 해당 군집 삭제, 많아지면 분리

# K-Means Clustering

## Evaluation

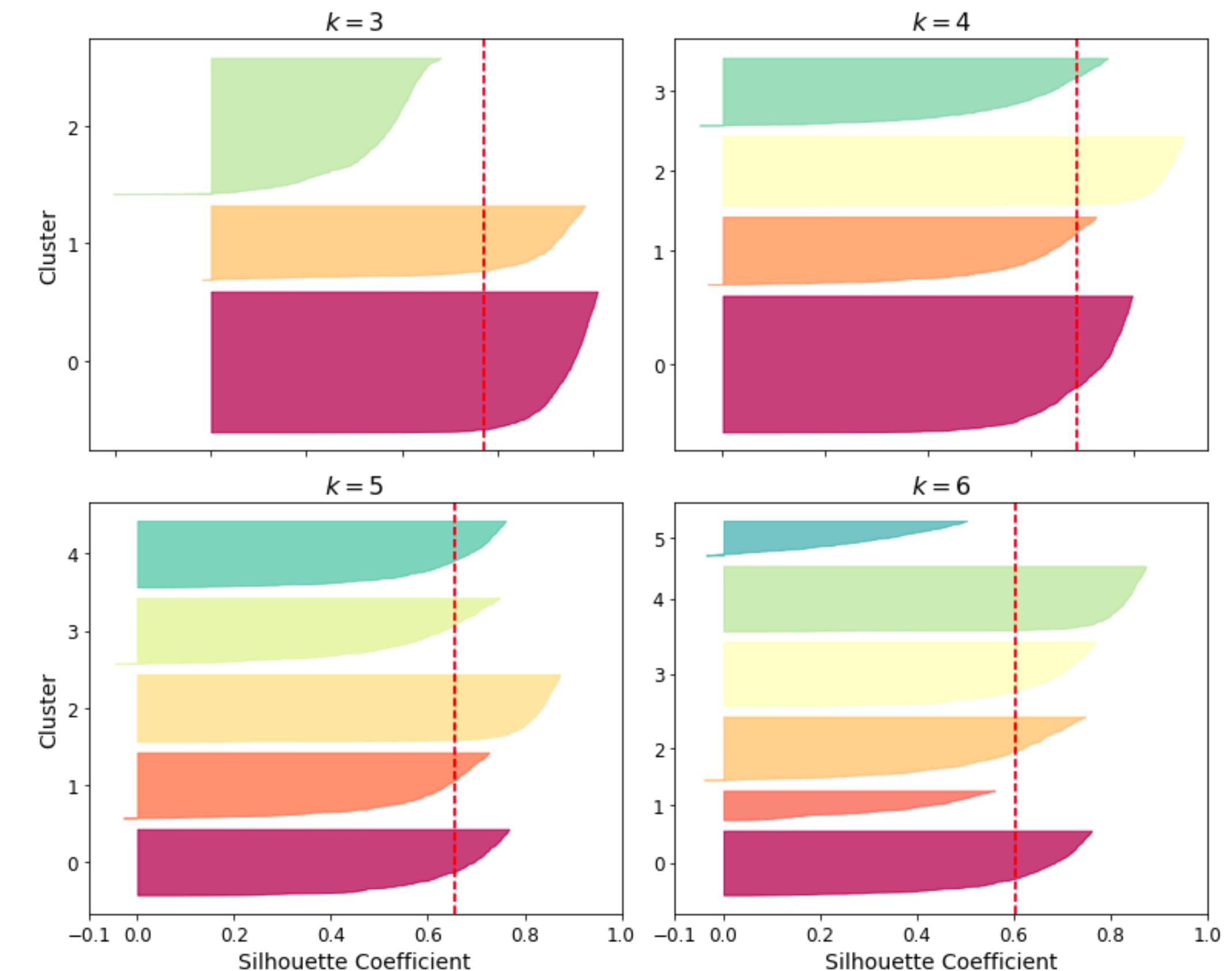
- Clustering이 제대로 되었다는 것을 어떻게 평가 할 수 있을까?
  - 비지도 학습이니 이것도 그냥 모르는 건가?
- Silhouette Score (실루엣 계수/점수)
  - 기본 컨셉은, 잘 분리된 군집의 경우 "내 군집과의 거리는 가깝지만, 남의 군집과의 거리는 멀다"
  - $\text{Silhouette Score} = (b - a) / \max(a, b)$ ,
    - a: 동일한 클러스터 안에있는 다른 샘플과의 거리
    - b: 가장가까운 클러스터까지의 거리
    - 둘다 평균



# K-Means Clustering

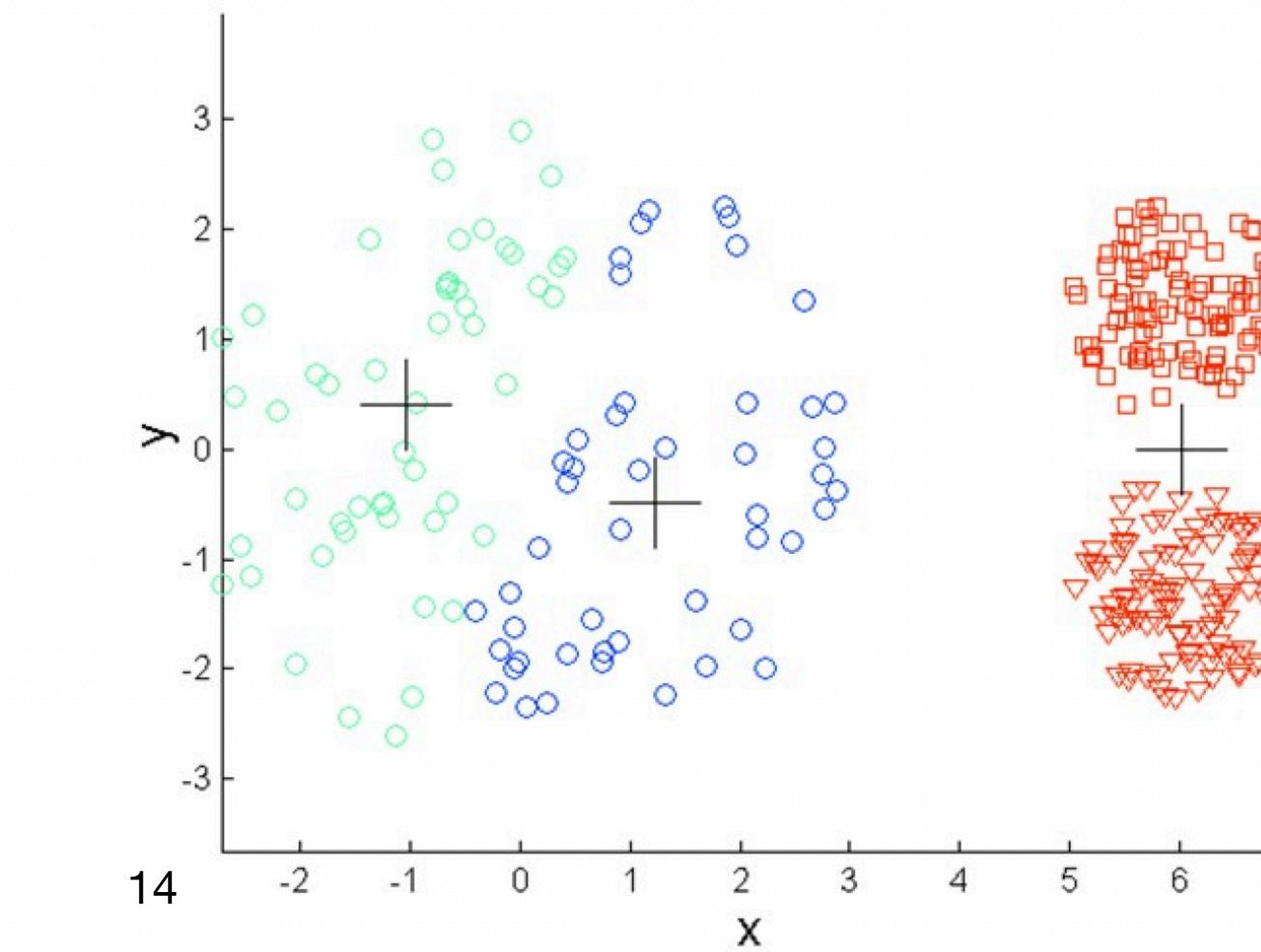
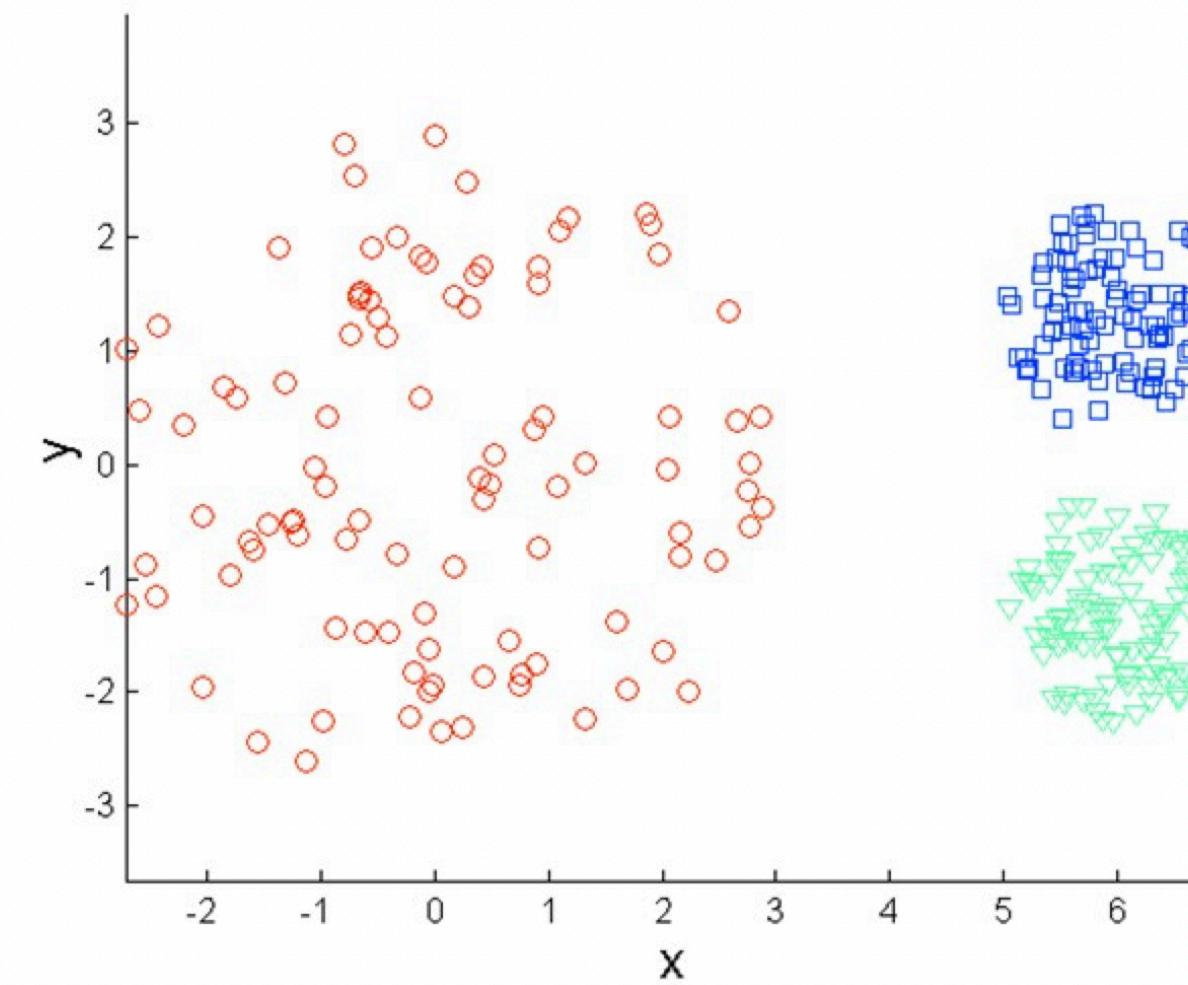
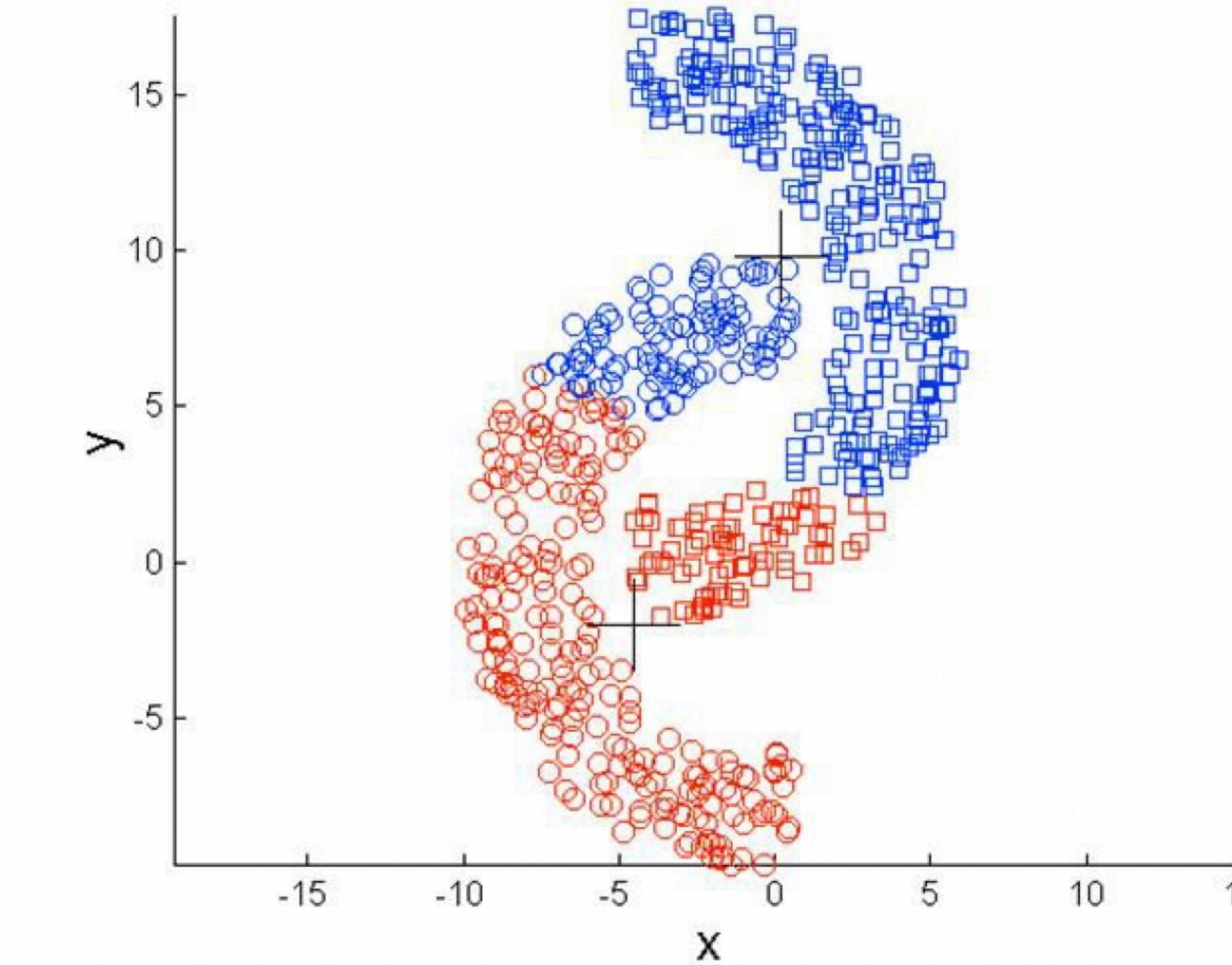
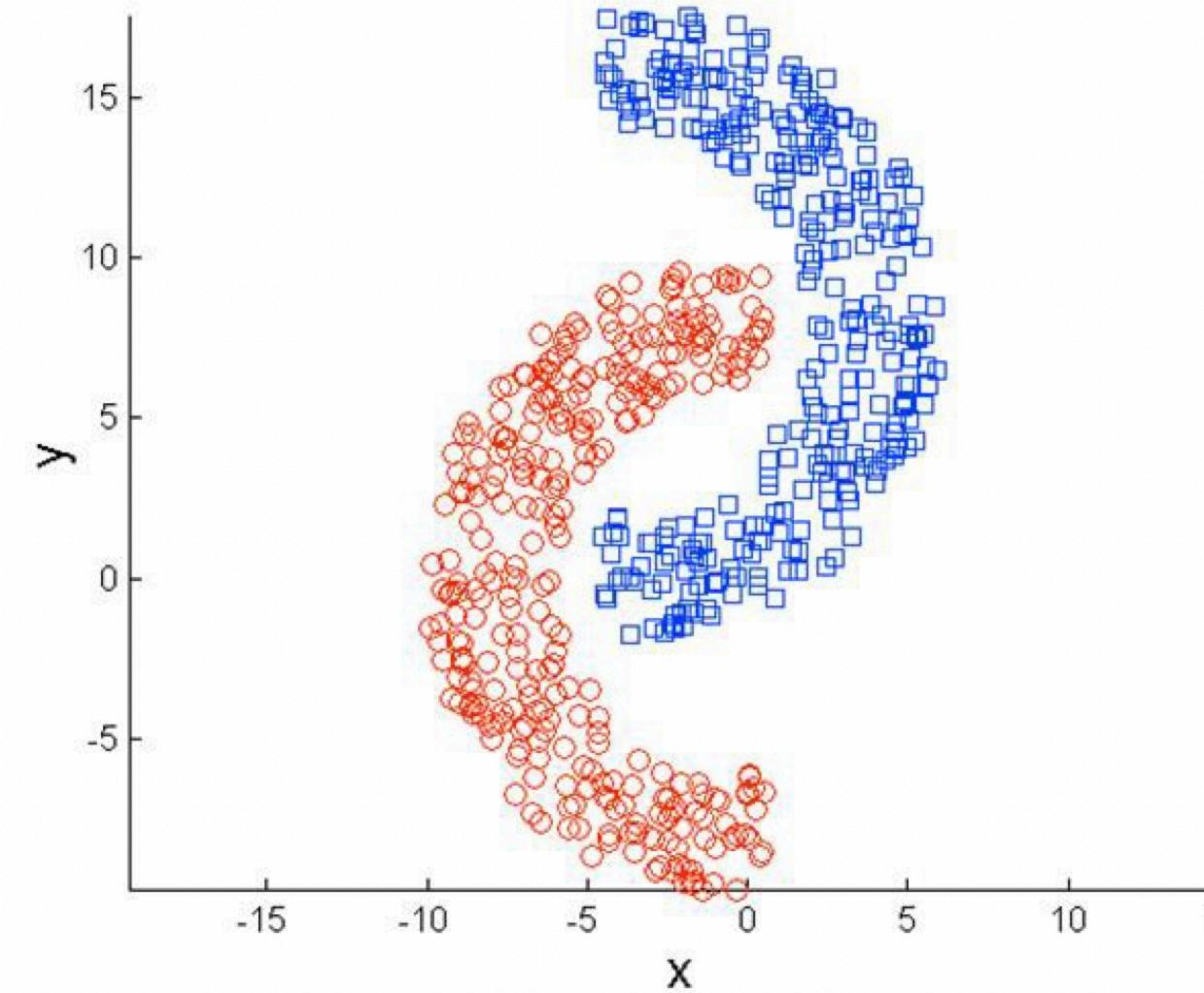
## Evaluation

- Silhouette Diagram
  - 높이: 클러스터가 포함하고 있는 샘플의 개수
  - 너비: 클러스터에 포함된 샘플의 정렬된 실루엣 계수
    - 그렇기 때문에 칼모양을 보임
  - 점선: 실루엣 점수 (전체 실루엣 계수의 평균)
- 해석
  - 잘 군집화가 되어진 경우에 오른쪽의 다이어그램에서 각각의 군집이 점선을 골고루 넘어가 있다. ( $K=5$ )



# K-Means Clustering

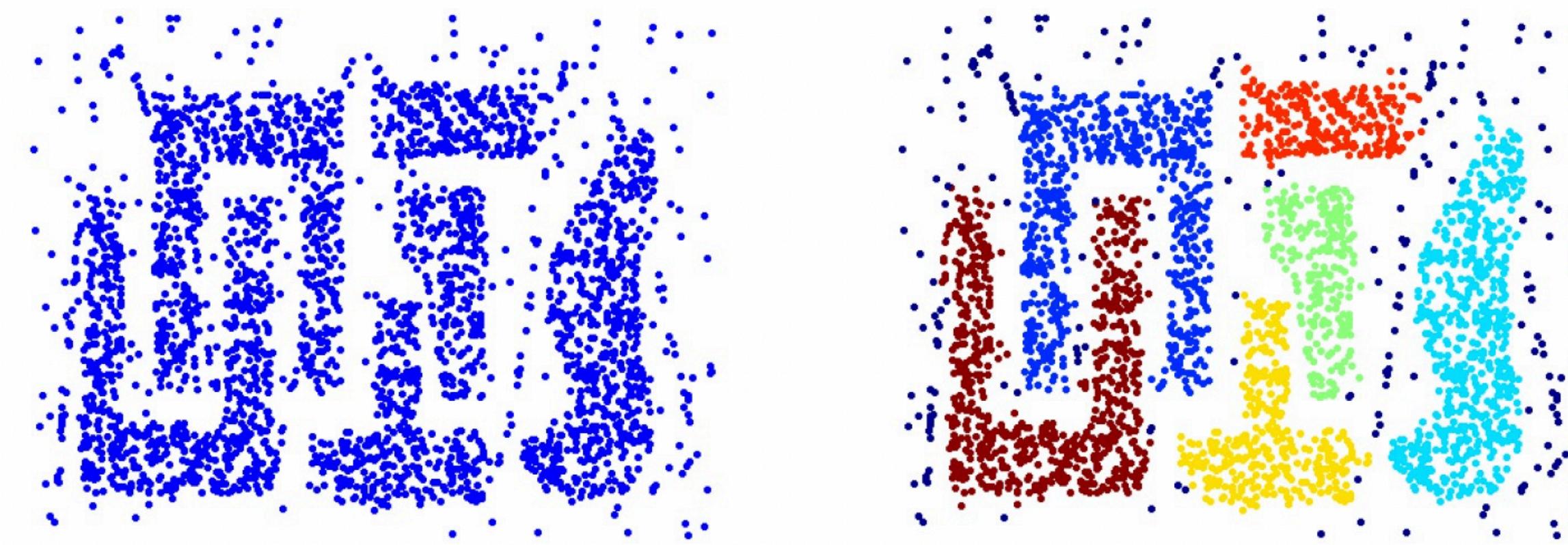
잘못 군집화 되는 경우



# DBSCAN

## Density-based spatial clustering of applications with noise

- 밀도기반 클러스터링의 알고리즘
  - 데이터의 분포와 밀도 (density)를 고려하여 클러스터를 구성하는 방법
  - 구형이 아닌 임의의 모양으로 생긴 클러스터에도 잘 동작
  - 클러스터링 과정에서 노이즈를 제거하는 것이 가능함



Original Points

Clusters

# DBSCAN

## 파라미터와 알고리즘

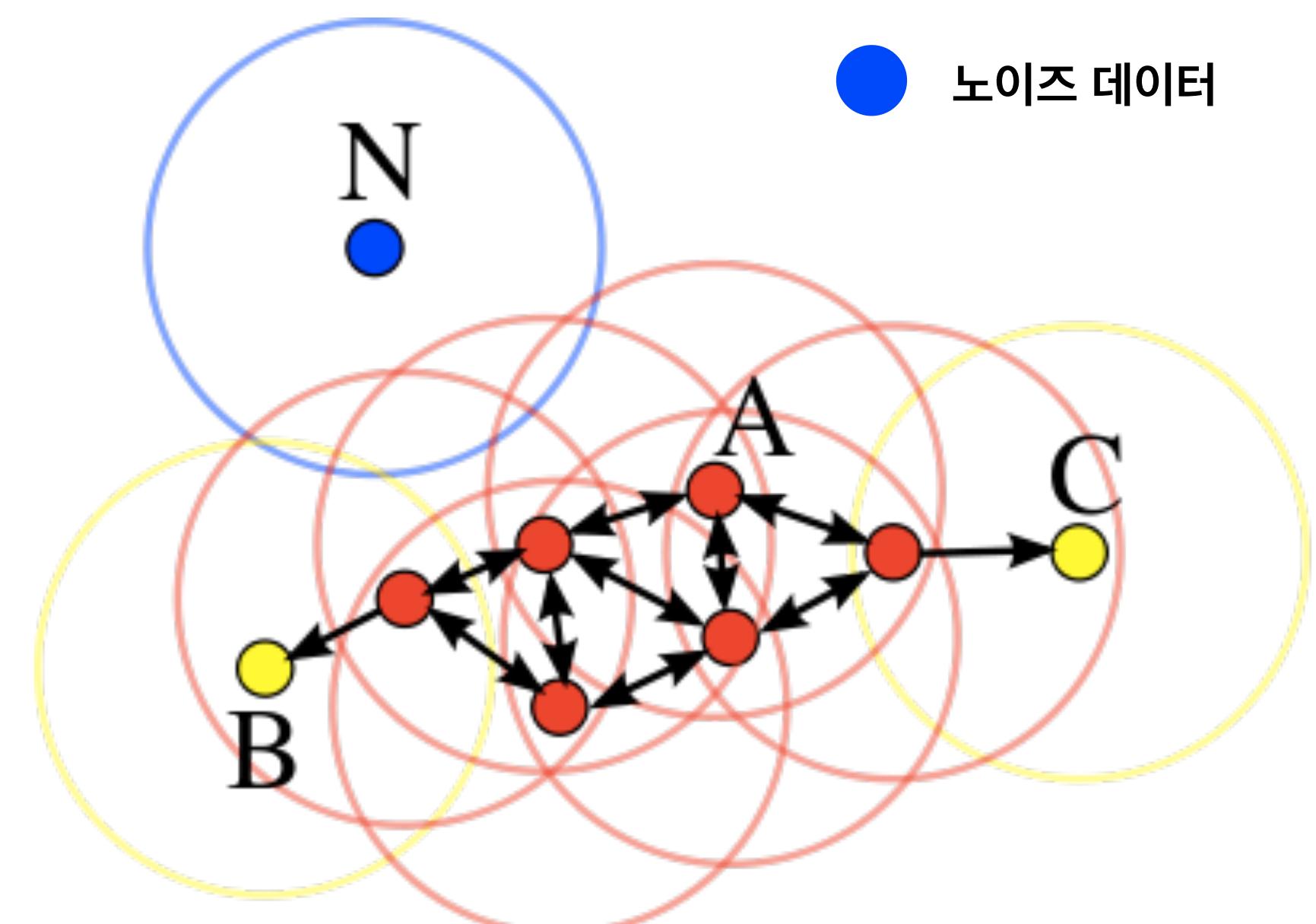
- 데이터 포인트의 종류

- 핵심 데이터 : 임의의 점  $p$ 에서부터 거리  $\epsilon$  이하의 점이  $m(=3)$ 개 이상
- 이웃 데이터 : 임의의 점  $p$ 와의 거리가  $\epsilon$  이하인 데이터를  $p$ 의 이웃 데이터라고 한다
- 외각 데이터 : 핵심 데이터는 아니지만, 임의의 핵심 데이터의 이웃 데이터인 점
- 노이즈 데이터 : 핵심 데이터도 아니고 외각데이터도 아닌 점

- 핵심 데이터
- 외각 데이터
- 노이즈 데이터

- 알고리즘

- 임의의 점에서 시작
- 해당 점에서, 주변에 거리  $\epsilon$ 이내에 몇개의 샘플이 있는지 확인
- 적어도  $m$ 개의 샘플이 있다면 핵심샘플로 간주
- 핵심샘플의 범위안에 있는 샘플도 클러스터에 포함, 이웃에 이웃으로 전파
- 그 외의 포인트는 이상치로 판단



# Dimensionality Reduction

# 차원 축소

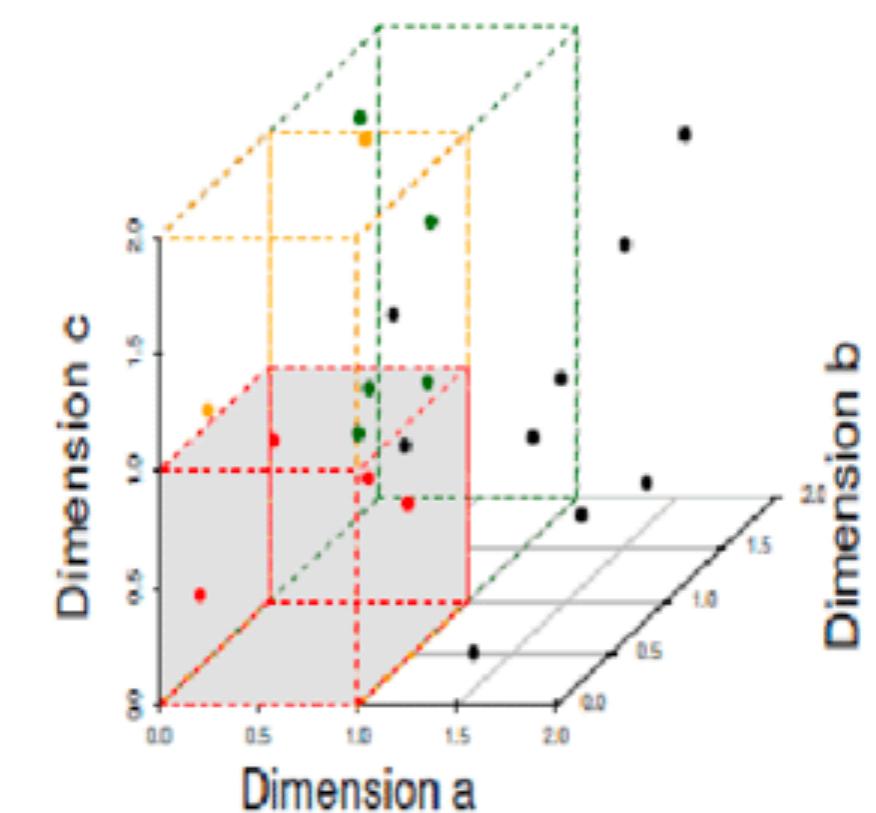
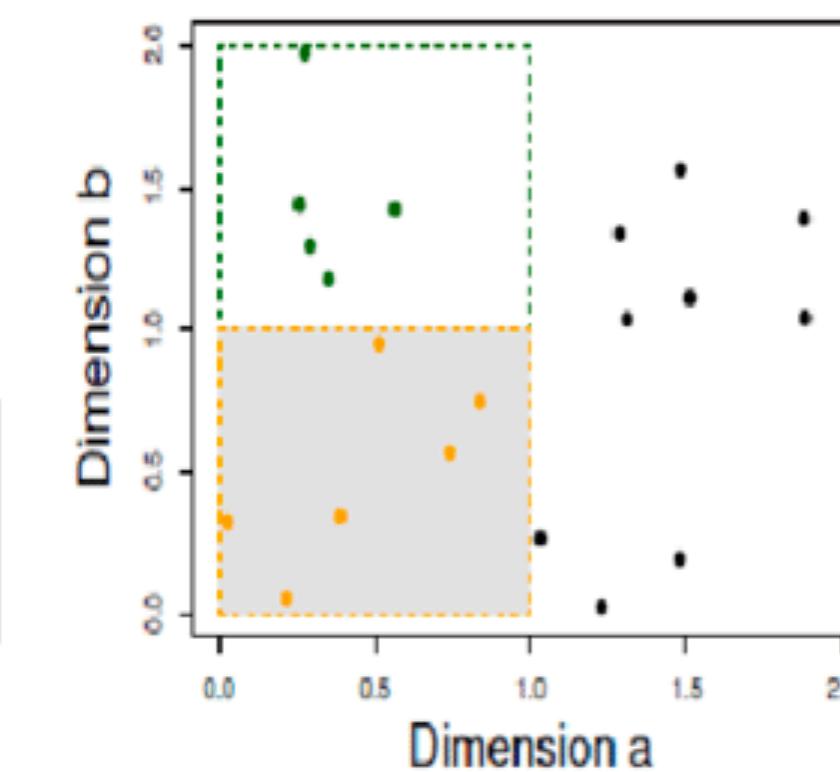
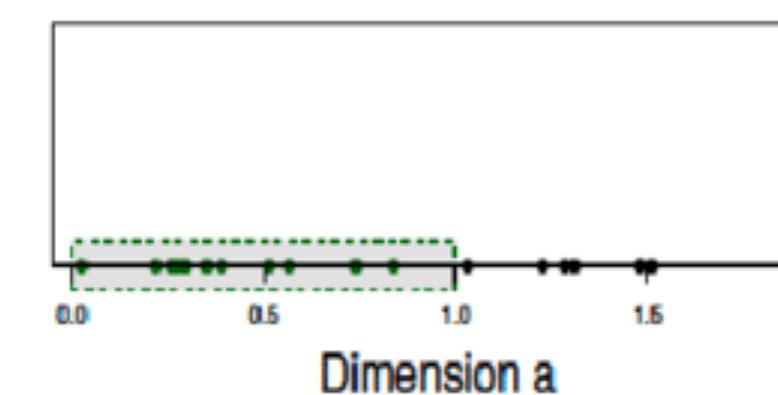
## Motivation

- 데이터 경량화
  - 더 빠른 훈련속도를 위해
  - 차원의 저주를 피하기 위해
- 시각화
  - 결국에 사람이 볼 수 있는 것은 최대 3차원
  - 의사 결정권자에게 보고하기 위해서는 물론
  - 본인 또한 시각화를 통해 추가적인 인사이트를 얻을 수 있음 (EDA)

# 차원의 저주

## 고차원 데이터의 문제

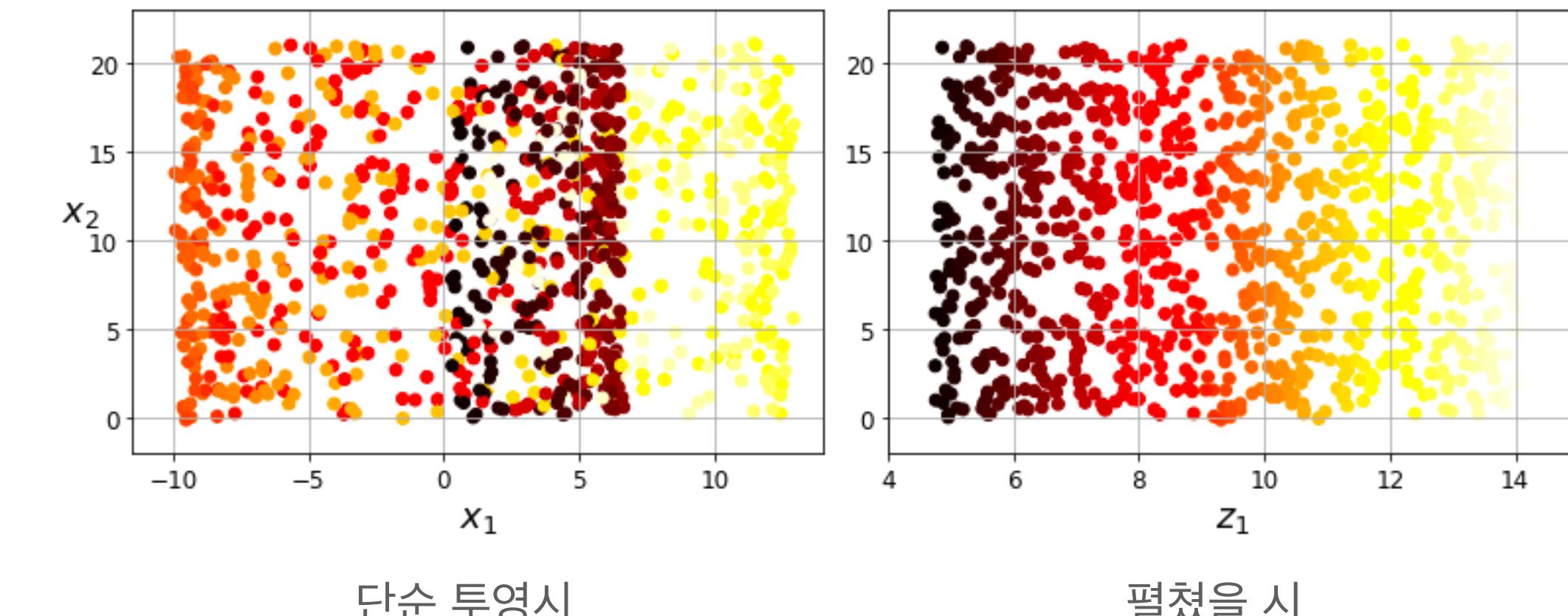
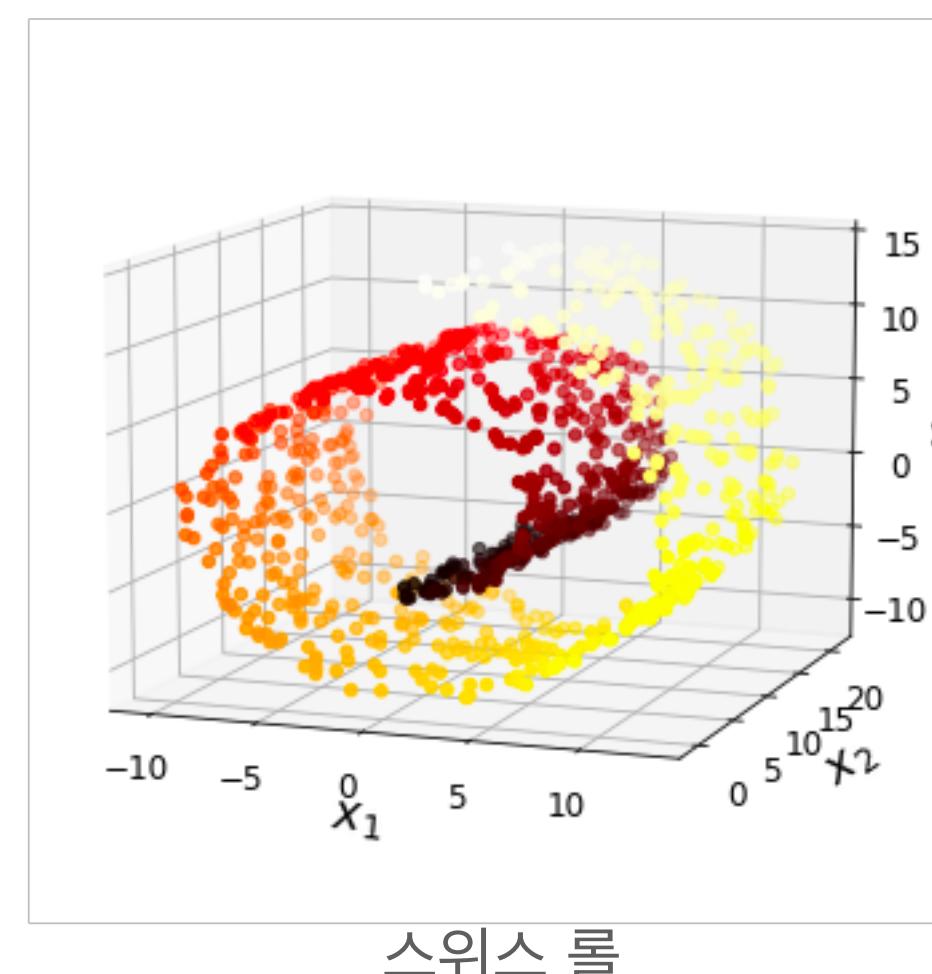
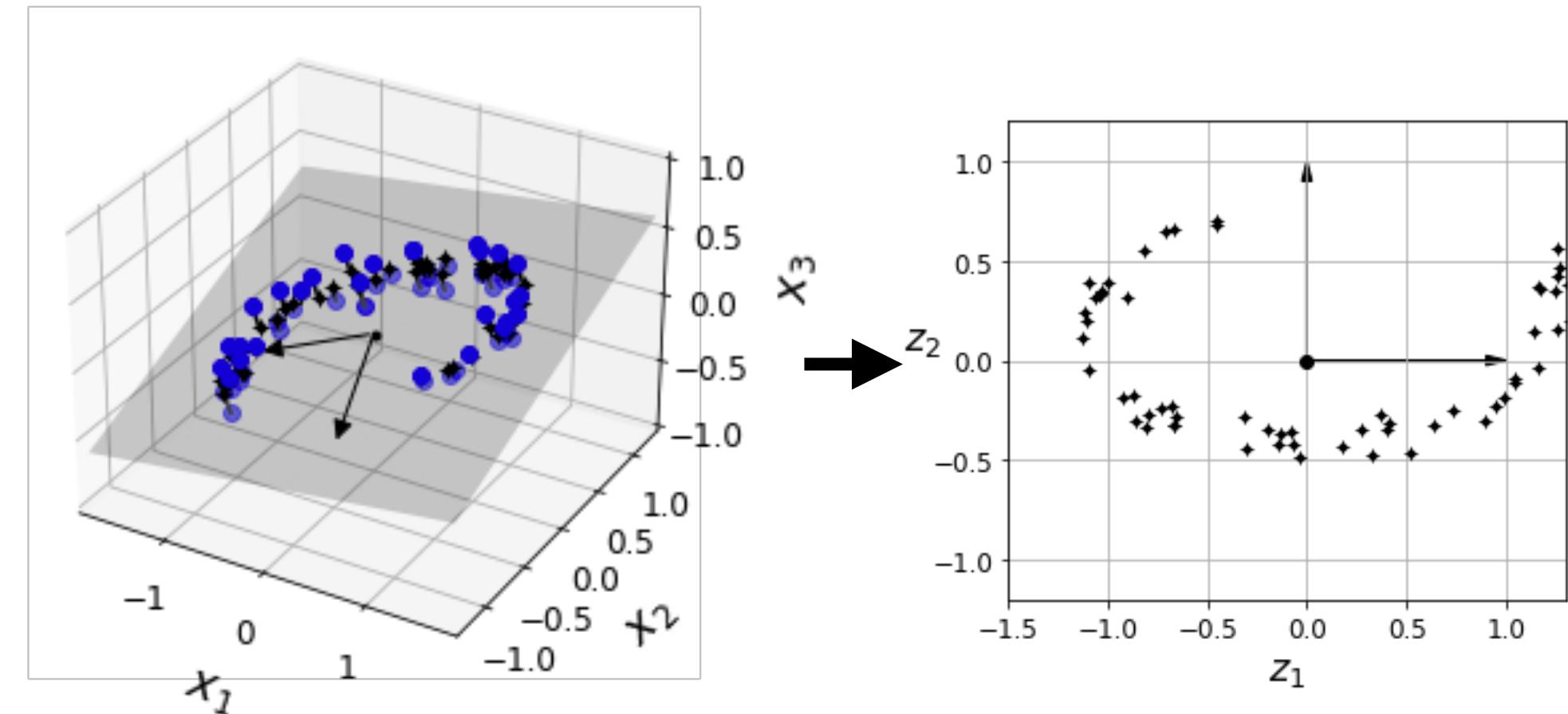
- Data Science에서 차원이라함은 Feature의 개수를 의미함
  - Feature가 많으면 훈련에 용이한 것이 아닌가?
  - 한 차원씩 증가 할 수록, 특정한 패턴을 보여주는 단위 영역에 필요로 되어지는 데이터의 양은 기하급수적으로 증가
    - 대부분의 훈련 데이터가 서로 떨어져있게 됨
    - 많은 알고리즘이 데이터 포인트간의 거리를 기반으로 모델을 생성
  - 고차원 데이터셋의 경우 더 많은 데이터를 필요
  - 더 많은 데이터는 더 많은 훈련시간을 필요



# 차원 축소를 위한 접근법

## Projection과 Manifold

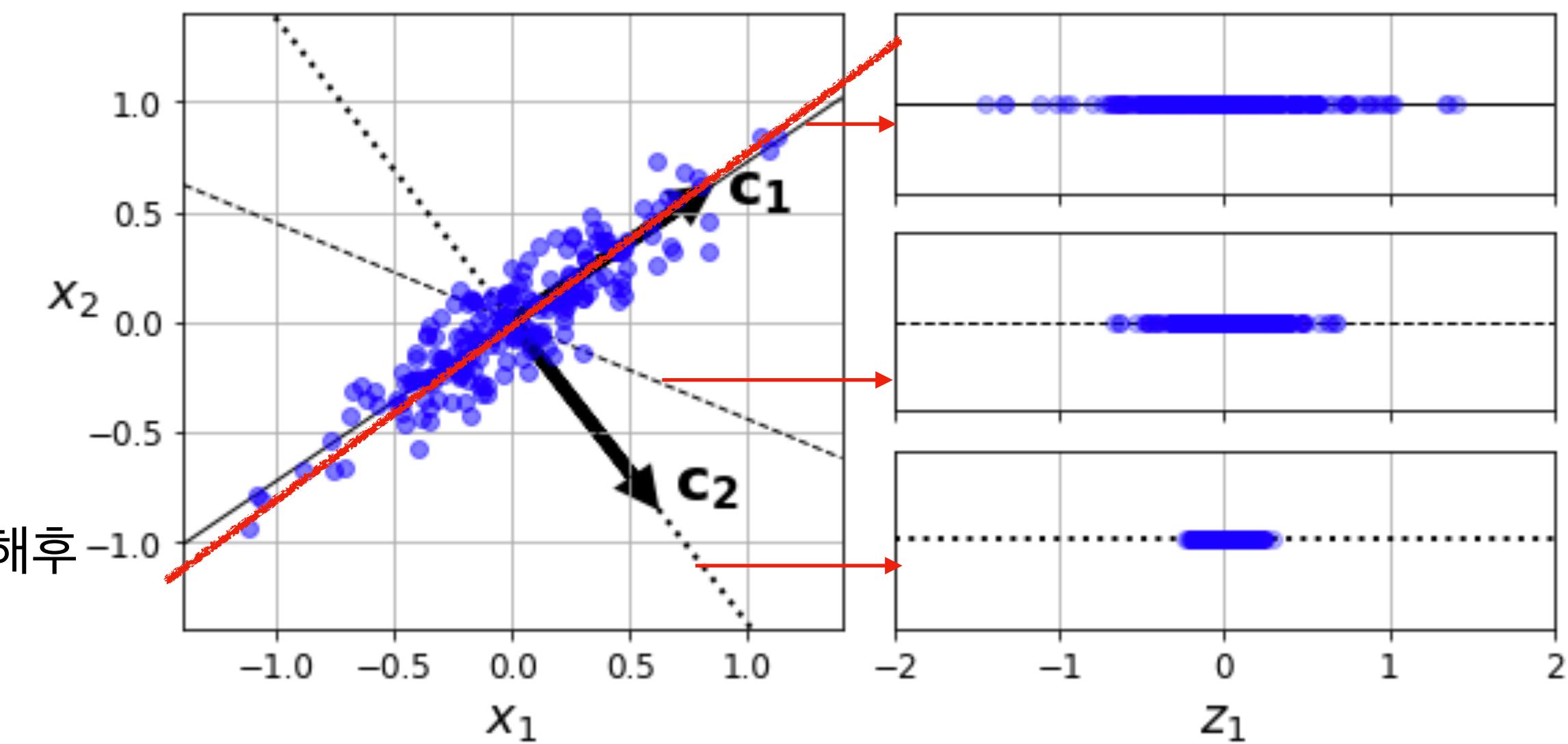
- 투영 (Projection)
  - 고차원 공간에 있는 데이터들을 저차원 공간으로 투영 시키는 방법
- Manifold
  - 고차원 공간에 내재한 저차원 공간
  - 단순 Projection을 통해서 축소가 되지 않는 경우가 존재 : non-linear
  - Manifold Learning
    - 고차원의 공간을 설명할 수 있는 저차원의 공간을 찾는 학습



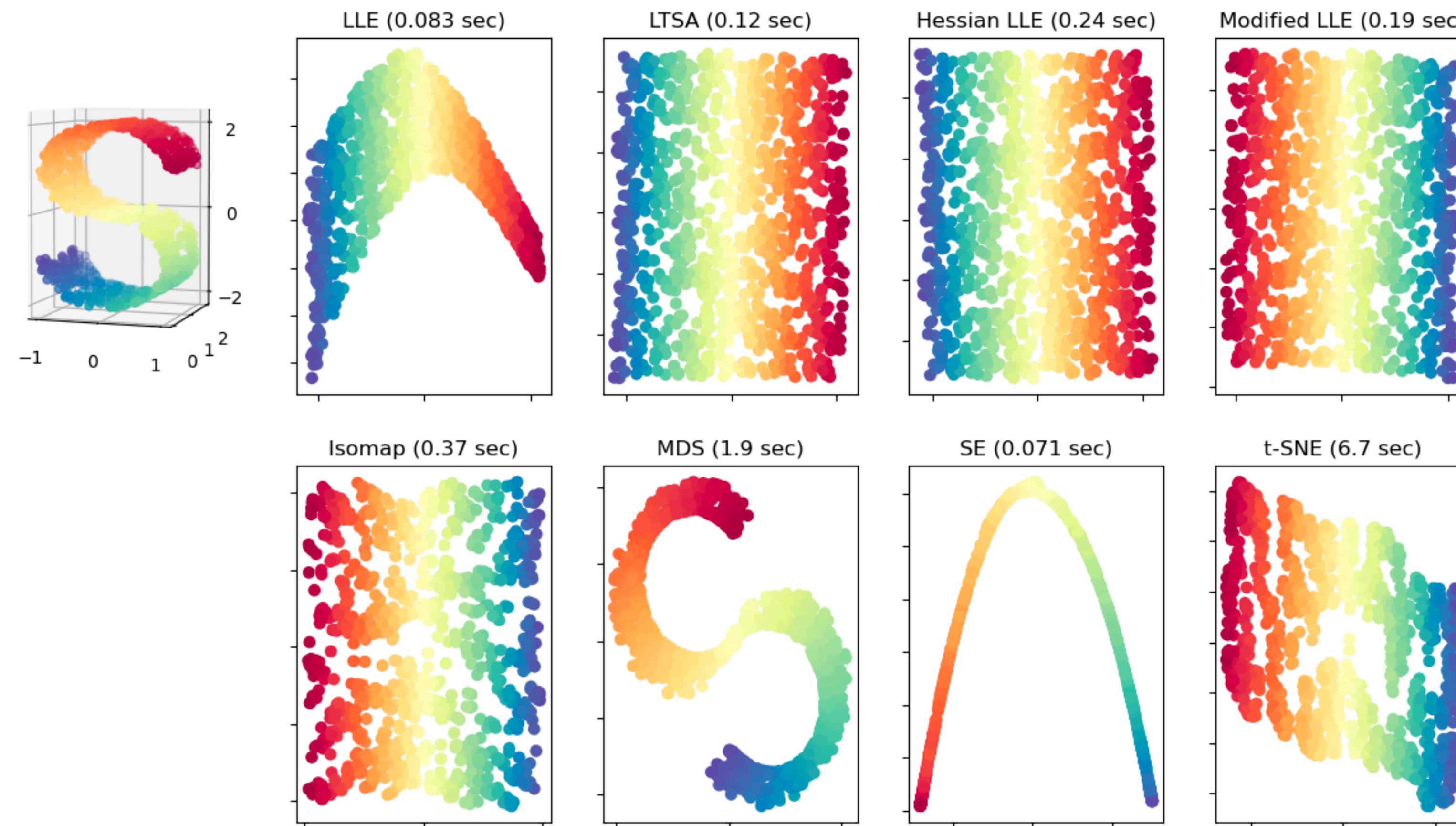
# PCA

## 주성분 분석

- Principal Component Analysis
  - 데이터에 가장 가까운 초평면(Hyperplane)을 정의한 다음, 데이터를 이 평면에 투영(projection)
  - 가장 가까운 초평면?
    - 분산(Variance)을 최대로 보존하는 평면
- 간단한 설명
  - 첫번째로 분산이 최대인 축을 찾는다
  - 두번째로 첫번째 축과 직교인 축을 찾아 높친 분산을 보존한다
- 덜 간단한 설명
  - 특이값 분해(SVD)를 이용해 데이터셋을  $U \sum V^T$ 의 행렬로 분해후
  - 주성분의 단위벡터들을 V에서 추출 ( $c_1, c_2, \dots$ )
  - 추출한 단위 벡터를 이용하여 원 데이터를 투영



Manifold Learning with 1000 points, 10 neighbors



# References

- Géron, Aurélien. Hands-on machine learning with Scikit-Learn, Keras, and TensorFlow: Concepts, tools, and techniques to build intelligent systems. O'Reilly Media, 2019.

**E.O.D**