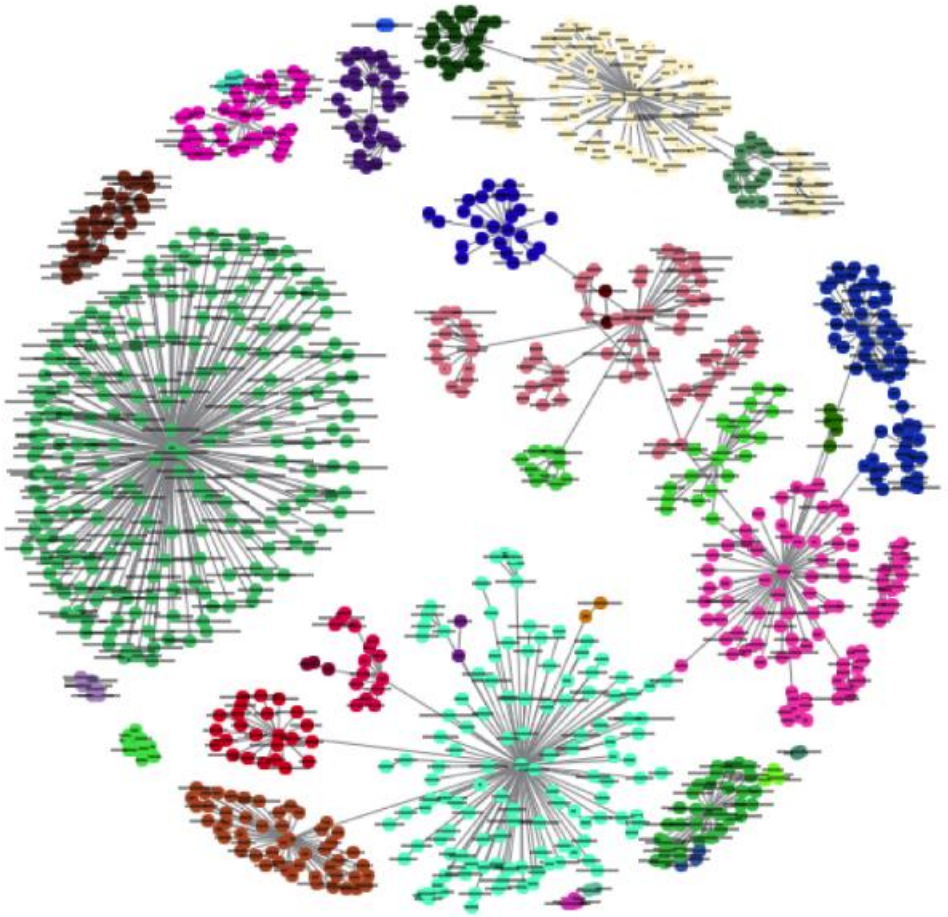
Clustering & Dimensionality Reduction

Data Scientist
안건이

목차

Why Clustering ?

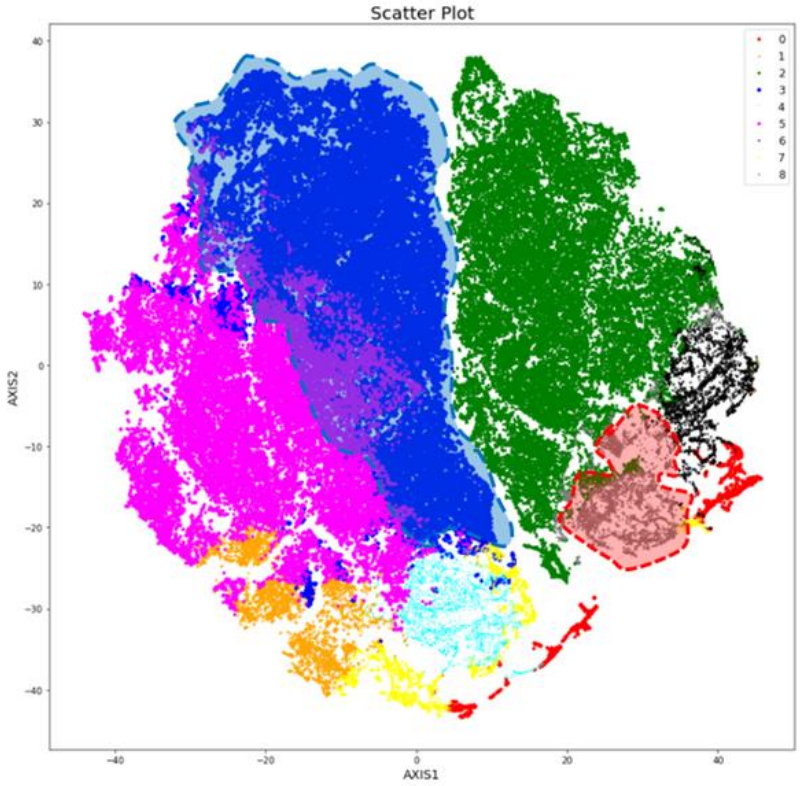
- Clustering (군집화) → Unsupervised Learning
  - Y가 존재하는데 왜 Clustering을 했을까?
  - 결국 우리는 X's 간의 특별한 조합으로 고효율 군이 도출될 거라는 생각
  - 복잡한 Supervised Learning Algorithm을 사용하지 말자 → 단순하게 생각하자 X's 간의 차이만 보자
  - Y에 방해를 받지 않고 X's만 가지고 군집을 만들어 봄
  - 그 후 군집들의 Y 평균 값을 통하여 고효율을 도출하는 군집을 찾아냄



Why Clustering ?

- Clustering (군집화) → Unsupervised Learning
  - Y가 존재하는데 왜 Clustering을 했을까?
  - 결국 우리는 X's 간의 특별한 조합으로 고효율 군이 도출될 거라는 생각
  - 복잡한 Supervised Learning Algorithm을 사용하지 말자 → 단순하게 생각하자 X's 간의 차이만 보자
  - Y에 방해를 받지 않고 X's만 가지고 군집을 만들어 봄
  - 그 후 군집들의 Y 평균 값을 통하여 고효율을 도출하는 군집을 찾아냄

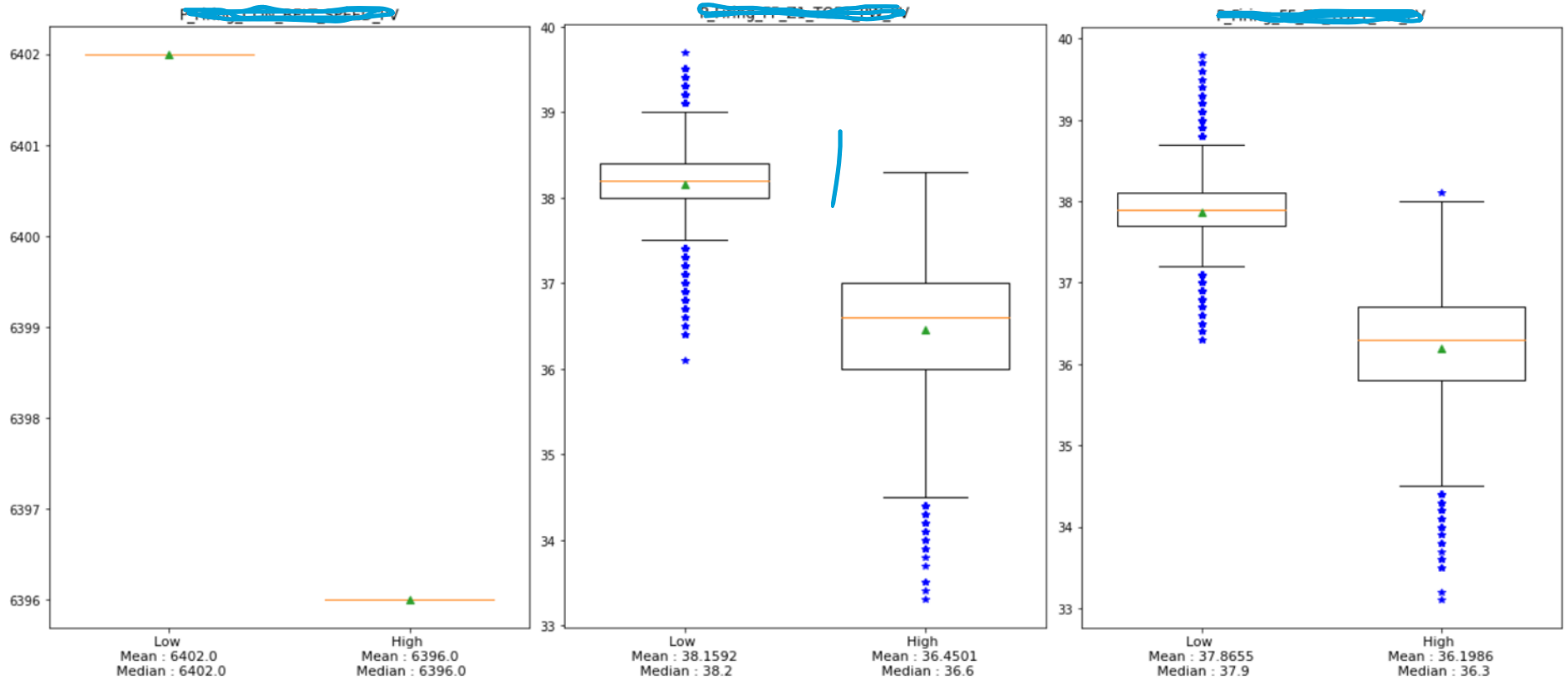
- 제어 가능한 [illegible] 변수(55개)만을 사용하여 Clustering 진행
- K-Means Clustering를 사용하였음 (Scaling 적용 X)
  - K=9 일때 군집간 Eff 차이가 가장 크게 발생함
- TSNE를 사용하여 군집 시각화를 진행하였음 (PCA는 설명력 부족함)



- Cluster내 데이터 개수가 풍부한 Cluster
  - High Eff Cluster = Label 8 (3765개, ●)
  - Low Eff Cluster = Label 3 (35848 개, ●)
  - GAP : [illegible]
- Cluster간 Feature Diff 검증 비교 – 안건이Y

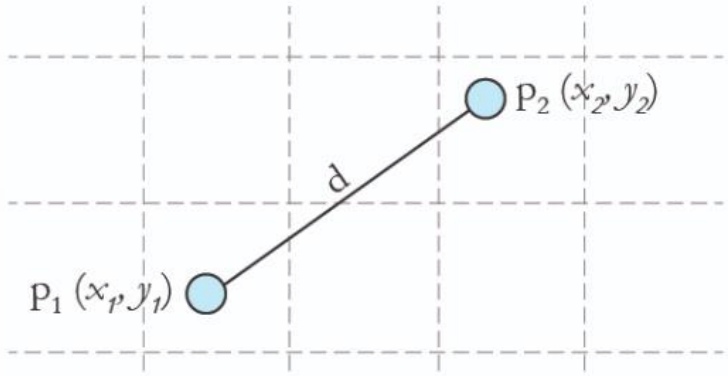
Why Clustering ?

- Clustering (군집화) → Unsupervised Learning
  - Y가 존재하는데 왜 Clustering을 했을까?
  - 결국 우리는 X's 간의 특별한 조합으로 고효율 군이 도출될 거라는 생각
  - 복잡한 Supervised Learning Algorithm을 사용하지 말자 → 단순하게 생각하자 X's 간의 차이만 보자
  - Y에 방해를 받지 않고 X's만 가지고 군집을 만들어 봄
  - 그 후 군집들의 Y 평균 값을 통하여 고효율을 도출하는 군집을 찾아냄
    - 고효율 Cluster vs 저효율 Cluster
      - 유의미한 변수 정리



Distance 종류

- Euclidian Distance
  - 중, 고등학교 시절 문, 이과 불문하고 수학을 공부했던 사람들에게 익숙한 Distance

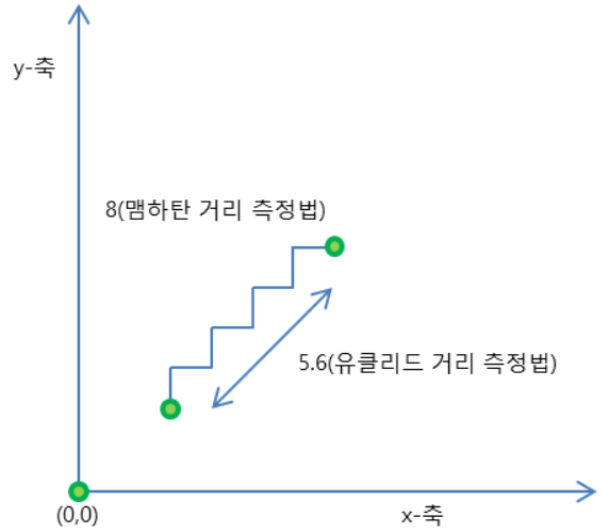


$$\mathrm{d}(\mathbf{p},\mathbf{q}) = \mathrm{d}(\mathbf{q},\mathbf{p}) = \sqrt{(q_1 - p_1)^2 + (q_2 - p_2)^2 + \cdots + (q_n - p_n)^2}$$

$$= \sqrt{\sum_{i=1}^{n} (q_i - p_i)^2}.$$

유클리디안 거리 공식

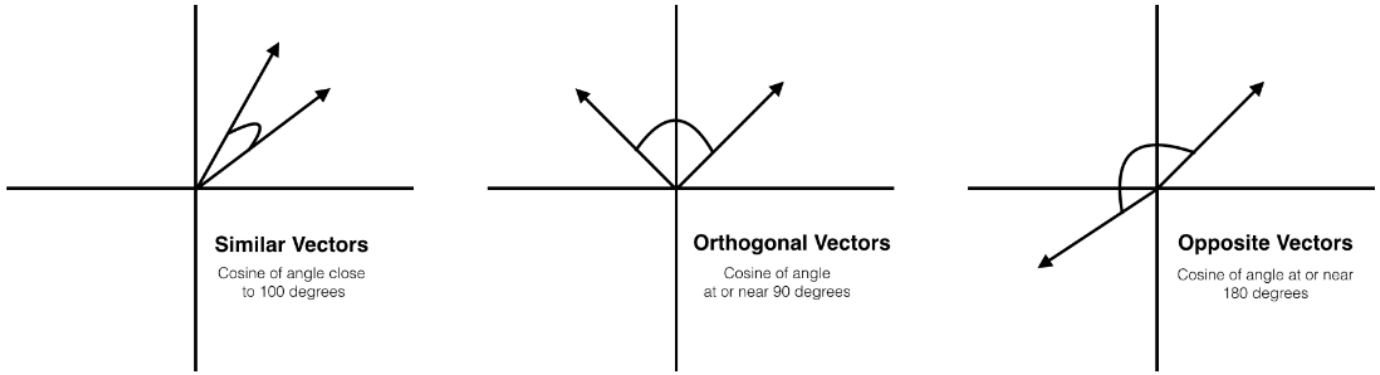
- Manhattan Distance
  - 두 점의 좌표 간의 절대값 차이를 합해서 구함
  - 블럭형으로 되어 있는 맨하탄 거리와 같다고 하여 Manhattan Distance라고 칭함



$$d = |a_1 - b_1| + |a_2 - b_2| + \cdots + |a_n - b_n|$$

Distance 종류

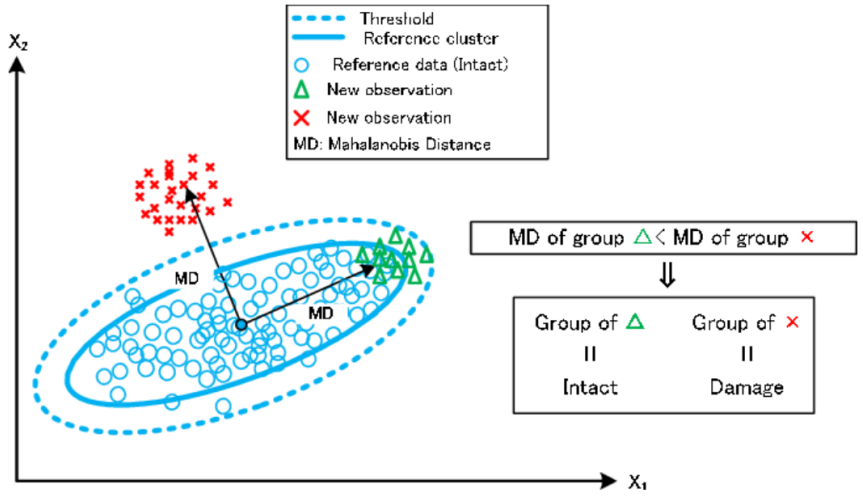
- Cosine Distance
  - 두 vector들 사이의 각도를 계산함
  - Vector의 크기는 무시하되 Vector의 방향의 차이만 계산함



$$similarity = \cos(\theta) = \frac{A \cdot B}{\|A\|\|B\|} = \frac{\sum_{i=1}^{n} A_i B_i}{\sqrt{\sum_{i=1}^{n} A_i^2}\sqrt{\sum_{i=1}^{n} B_i^2}}$$

Cosine Similarity의 공식

- Mahalanobis Distance
  - 다변량 데이터에서 분포의 형태를 고려하여 거리를 계산함
  - 변수들간 모두 독립적이고 Variance가 1로 정규화된다면 마할라노비스 거리는 유클리디안 거리와 같아진다.



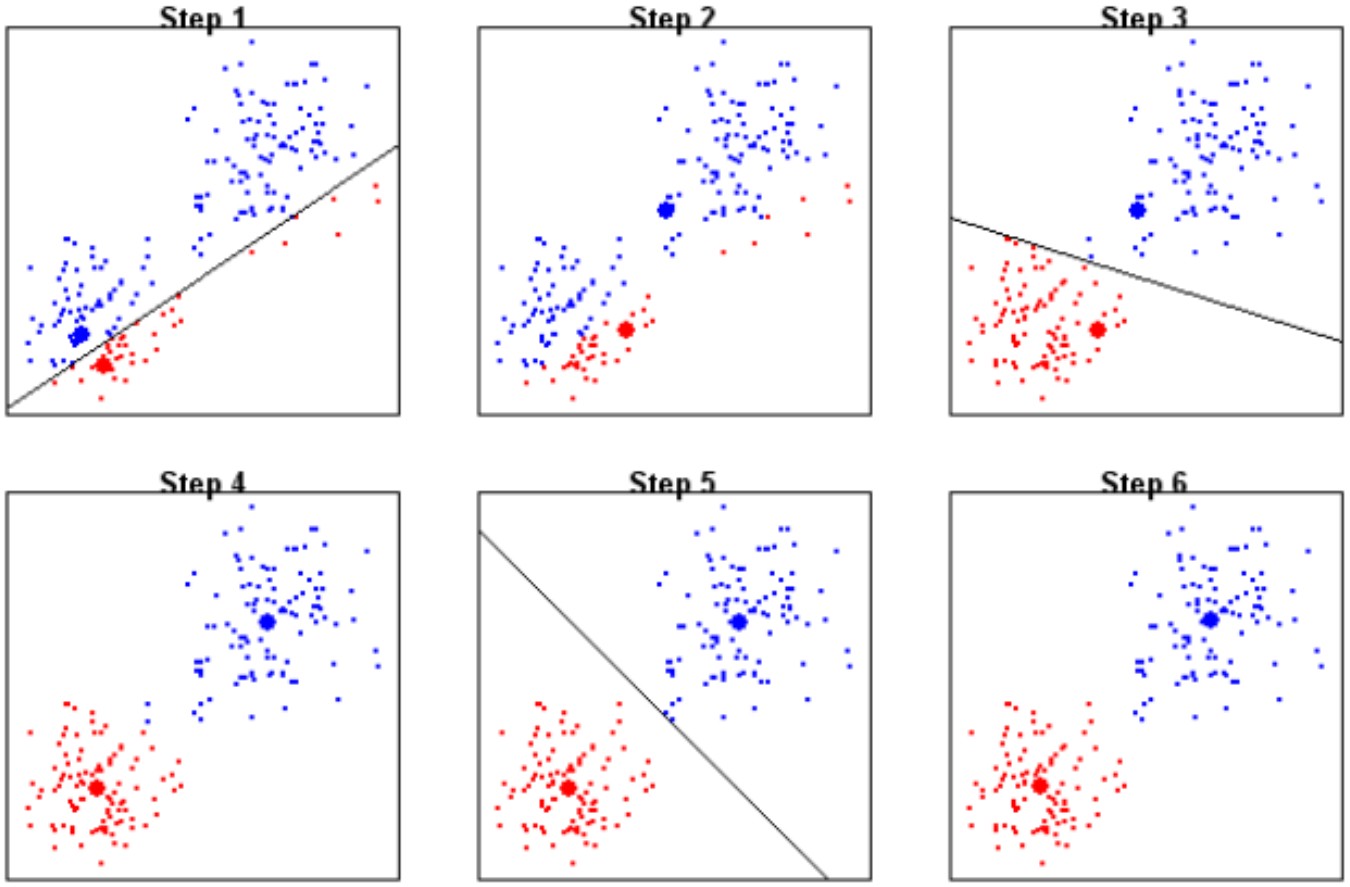
Concept of Mahalanobis distance (MD).

$$d(u,v) = \sqrt{(u-v)I^{-1}(u-v)^T} = \sqrt{(u-v)\cdot(u-v)} = \sqrt{(u_1-v_1)^2 + .. + (u_p-v_p)^2}$$

변수들 간에 독립적일 때는 유클리디안 거리공식으로 변하게 된다.

K-means Clustering

- K-means Clustering
  - 대표적인 분리형 군집화 알고리즘
  - 각 군집은 하나의 중심(centroid)를 가짐
  - 사용자가 사전에 군집 수(k)를 정해야 알고리즘을 실행할 수 있음 (k = Hyperparameter)
  - E(Expectation) M(Maximization) 알고리즘 기반으로 작동함
    - Cluster의 중심으로 이동하면서 Maximization

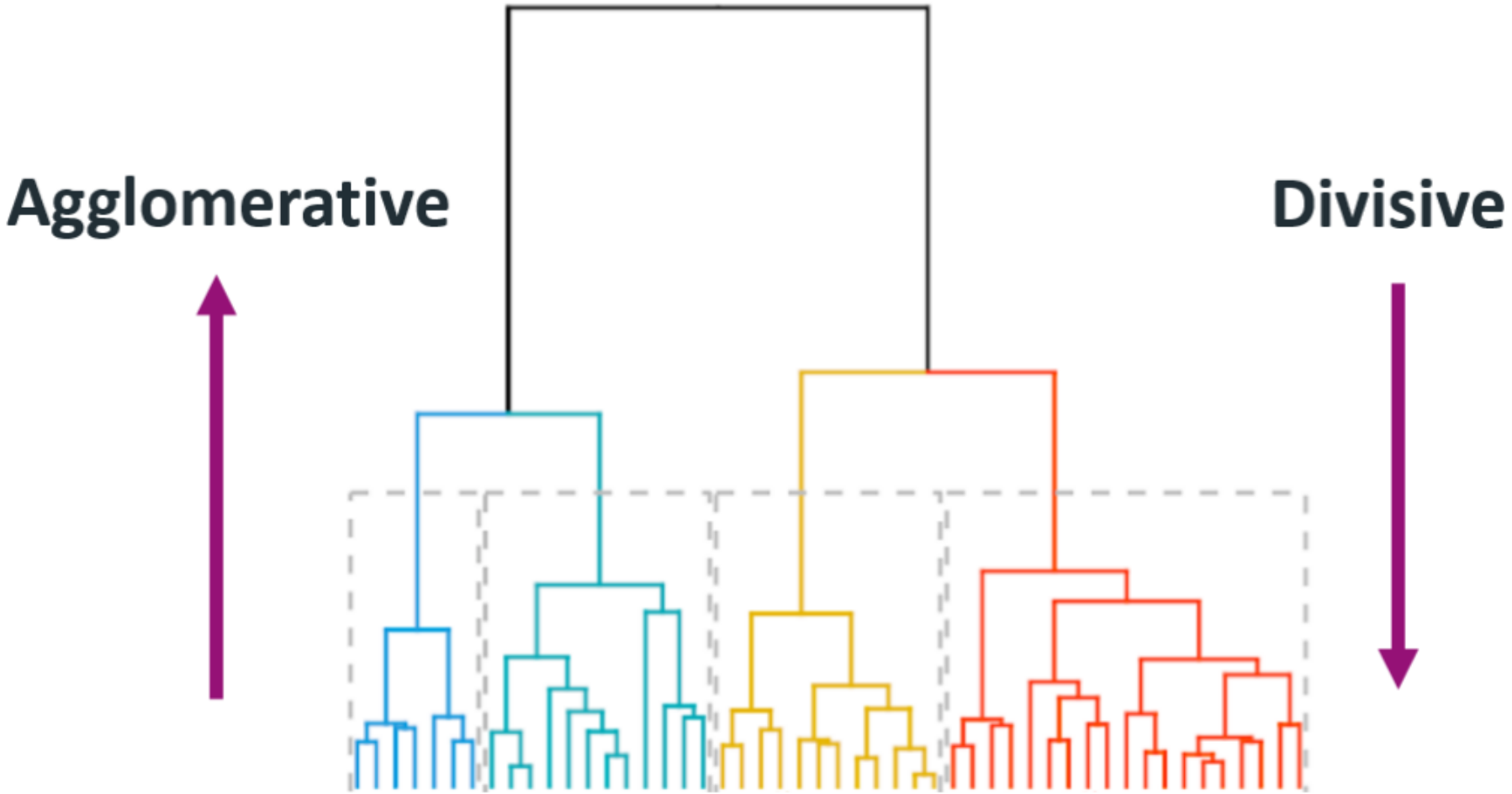


$$X = C_1 \cup C_2 \ldots \cup C_K, \quad C_i \cap C_j = \phi$$

$$argmin_C \sum_{i=1}^{K} \sum_{x_j \in C_i} \|x_j - c_i\|^2$$

Hierarchical Clustering

- Hierarchical Clustering
  - K-means와 달리 군집 수를 사전에 정하지 않아도 학습을 수행할 수 있음
    - 개체들이 결합되는 순서를 나타내는 트리 형태의 구조인 덴드로그램(Dendrogram) 때문임
  - 덴드로그램을 생성한 후 적절한 수준에서 트리를 자르면 전체 데이터를 몇 개 군집으로 나눌 수 있게 됨



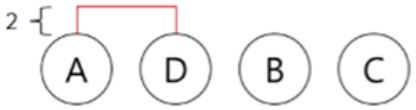
Hierarchical Clustering

- Hierarchical Clustering
  - K-means와 달리 군집 수를 사전에 정하지 않아도 학습을 수행할 수 있음
    - 개체들이 결합되는 순서를 나타내는 트리 형태의 구조인 덴드로그램(Dendrogram) 때문임
  - 덴드로그램을 생성한 후 적절한 수준에서 트리를 자르면 전체 데이터를 몇 개 군집으로 나눌 수 있게 됨

Step 1 : 각 데이터 마다의 Distance를 계산 함

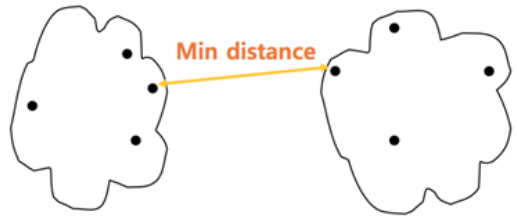
	A	B	C	D
A		20	7	2
B			10	25
C				3
D				

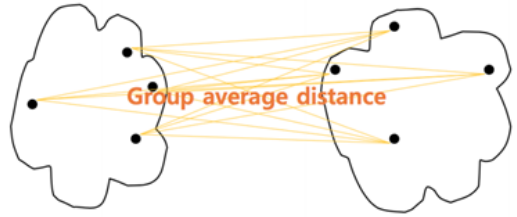
Step 2 : 가장 가까운 데이터를 서로 묶어줌

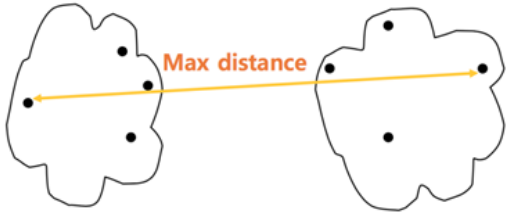


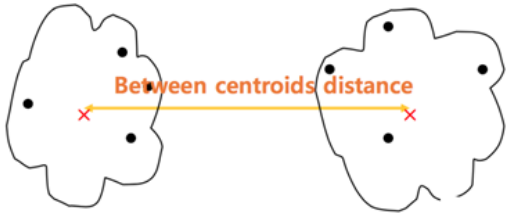
	A	B	C	D
A		20	7	2
B			10	25
C				3
D				

Step 3-1 : [A,D] 군집과 나머지 데이터들의 거리를 계산해 줌
계산 방식은 아래와 같이 크게 4가지가 있음





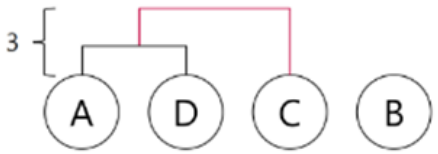




Hierarchical Clustering

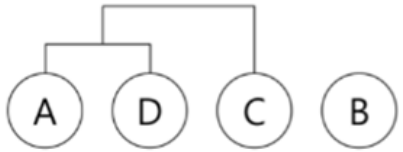
- Hierarchical Clustering
  - K-means와 달리 군집 수를 사전에 정하지 않아도 학습을 수행할 수 있음
    - 개체들이 결합되는 순서를 나타내는 트리 형태의 구조인 덴드로그램(Dendrogram) 때문임
  - 덴드로그램을 생성한 후 적절한 수준에서 트리를 자르면 전체 데이터를 몇 개 군집으로 나눌 수 있게 됨

Step 3-2 : 군집과 데이터(군집) 간 거리를 계산함



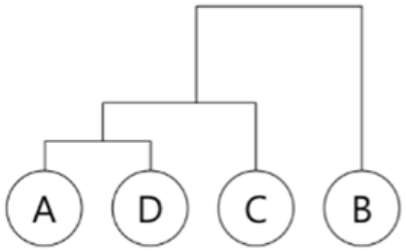
	AD	B	C	
AD		20	3	
B			10	
C				

Step 4 : Step 3을 반복함



	ADC	B		
ADC		10		
B				

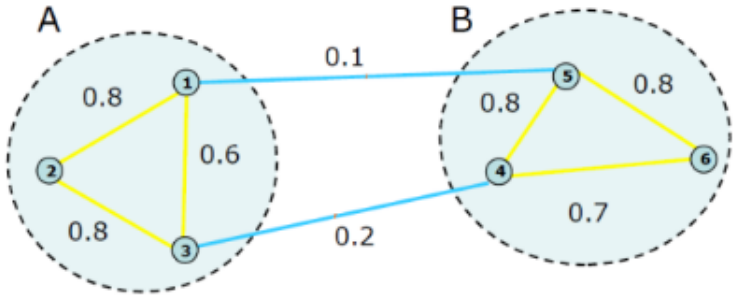
Step 5 : Step 3 반복



	AD CB			
AD CB				

Spectral Clustering

- Spectral Clustering
  - Graph 기반 군집화 기법
    - Raw 데이터를 그래프로 변환하기 위해 인접행렬(Adjacency Matrix)를 만듬
    - Adjacency Matrix를 만들때 Undirected Weighted Graph를 사용해서 만듬
  - Graph 구축
    - ε-neighborhood graph 구축 → k-nearest neighbor graph 구축
    - 혹은 minimum spanning tree를 사용하여 연결되지 않은 데이터까지 모두 연결시켜줌
  - Graph Cut
    - Minimum Cut method
      - 원 그래프를 A와 B라는 부 그래프로 나눌 때 끊어지는 엣지들의 가중치가 최소가 되도록 함
      - Degree Matrix(D), Laplacian Matrix(L = D - W) 필요



$$Cut(A,B) = \sum_{i \in A, j \in B} w_{ij} = 0.3$$

$$Cut(A,A) = \sum_{i \in A, j \in A} w_{ij} = 2.2$$

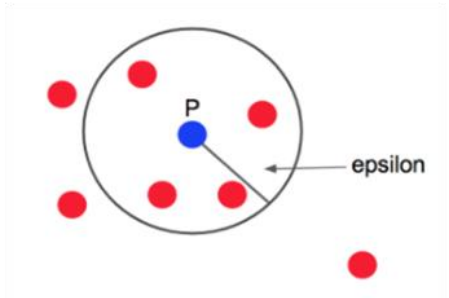
$$Cut(B,B) = \sum_{i \in A, j \in B} w_{ij} = 2.3$$

$$MinCut(A,B) = \min \frac{1}{4} q^T (D - W) q$$

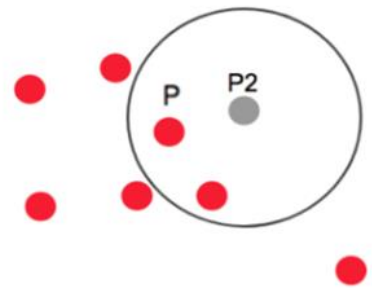
DBSCAN

- DBSCAN – Density Based Spatial Clustering of Applications with Noise)
  - K-means나 Hierarchical Clustering의 경우 군집간의 거리를 이용한 Clustering 기법
  - DBSCAN은 밀도기반의 기법이며 세밀하게 몰려 있어서 밀도가 높은 부분을 Clustering 하는 기법
    - 점 p가 있다고 할때, 점 p에서 부터 거리 e(epsilon)내에 점이 m(minPls)개 있으면 하나의 군집으로 인식함
    - 따라서 e와 m이 Hyperparameter임

Step 1 : e = 3, m = 4인 경우 → 충족 → Core Point

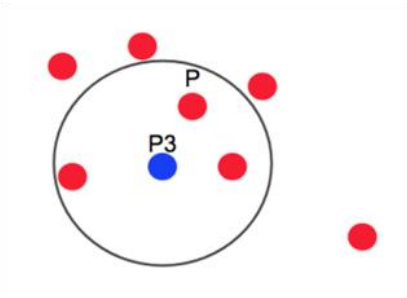


Step 2 : p 이동 → e = 3, m = 4인 경우 → 불충족 → Boders

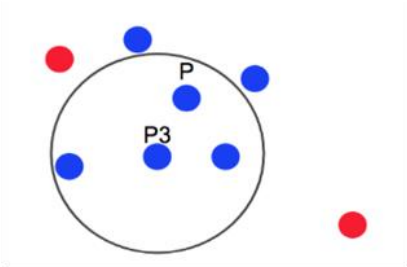


Epsilon 반경내의 점이 3개, m=4에 못미치기 때문에 core point는 되지 못하지만 앞의 점 P를 core point로 하는 군집에 속하기 때문에 boder point(경계점)이라고 함

Step 3 : e = 3, m = 4인 경우 → 충족 → Core Point



Step 4 : e = 3, m = 4인 경우 → 충족 → Core Point

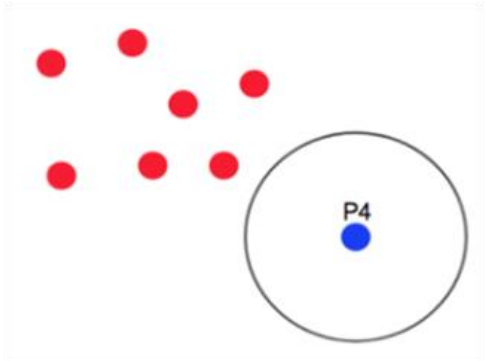


P3를 중심으로 하는 반경내에 다른 core point P가 포함되어 있는데, 이 경우 core point P와 P3는 연결되어 있다고 판단하고 하나의 군집으로 묶어줌

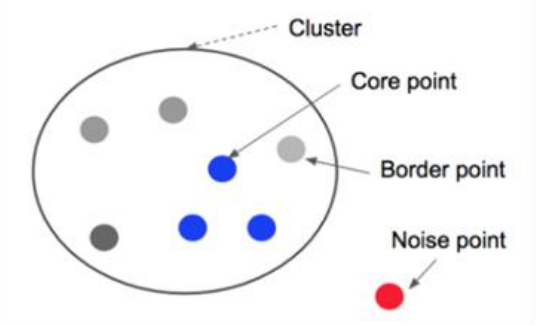
DBSCAN

- DBSCAN – Density Based Spatial Clustering of Applications with Noise)
  - K-means나 Hierarchical Clustering의 경우 군집간의 거리를 이용한 Clustering 기법
  - DBSCAN은 밀도기반의 기법이며 세밀하게 몰려 있어서 밀도가 높은 부분을 Clustering 하는 기법
    - 점 p가 있다고 할때, 점 p에서 부터 거리 e(epsilon)내에 점이 m(minPls)개 있으면 하나의 군집으로 인식함
    - 따라서 e와 m이 Hyperparameter임

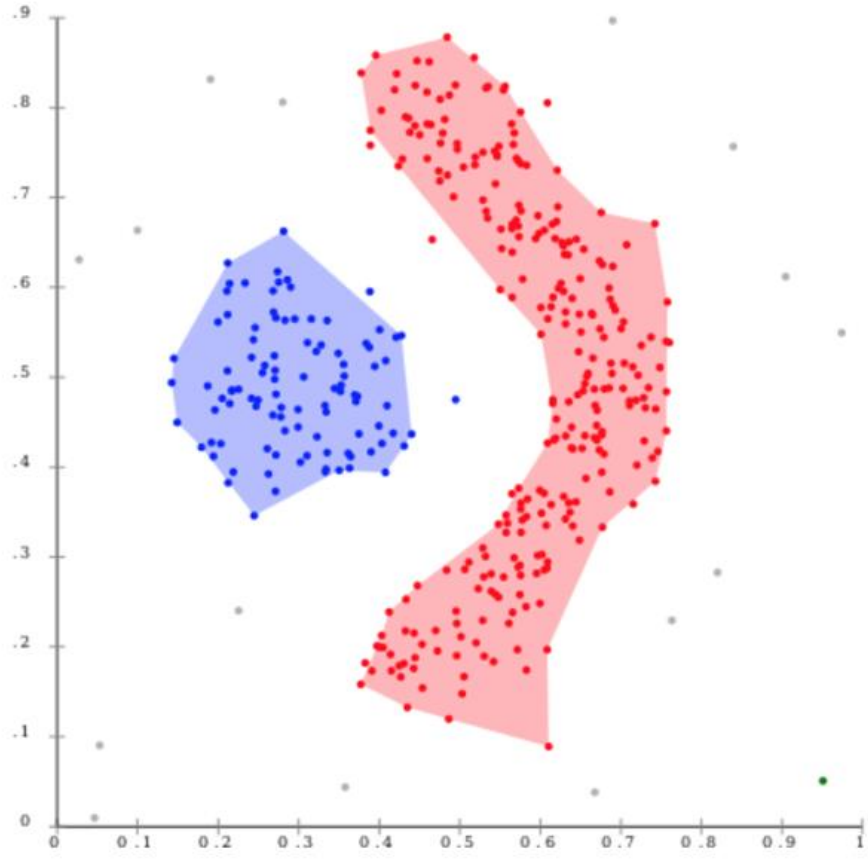
Step 5 : e = 3, m = 4인 경우 → 둘다 충족 → Noise



Final



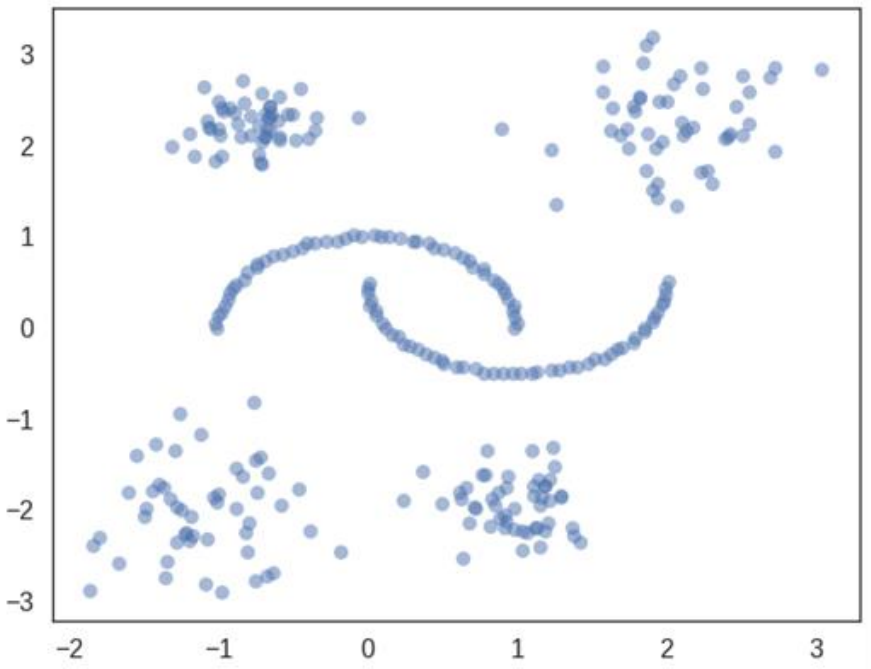
Example



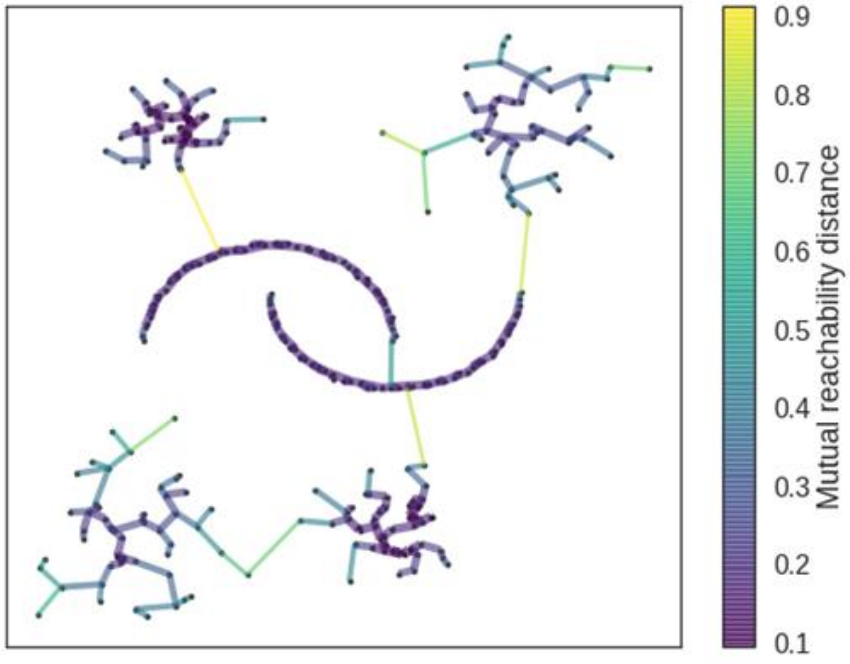
HDBSCAN

- HDBSCAN – Hierarchical Density Based Spatial Clustering of Applications with Noise)
  - HDBSCAN의 경우 더 이상 epsilon(e)이 필요하지 않으며 minPls(m)만 존재함
  - Point간의 mutual reachability distance 값을 기반으로 MST(minimum spanning tree)를 만듬
  - Tree를 바탕으로 Hierarchy를 구축함 → minPls 값을 기반으로 hierarchy를 축약함
  - 축약된 Cluster hierarchy에서 stable한 클러스터만 선택함

Step 1 : Raw Data



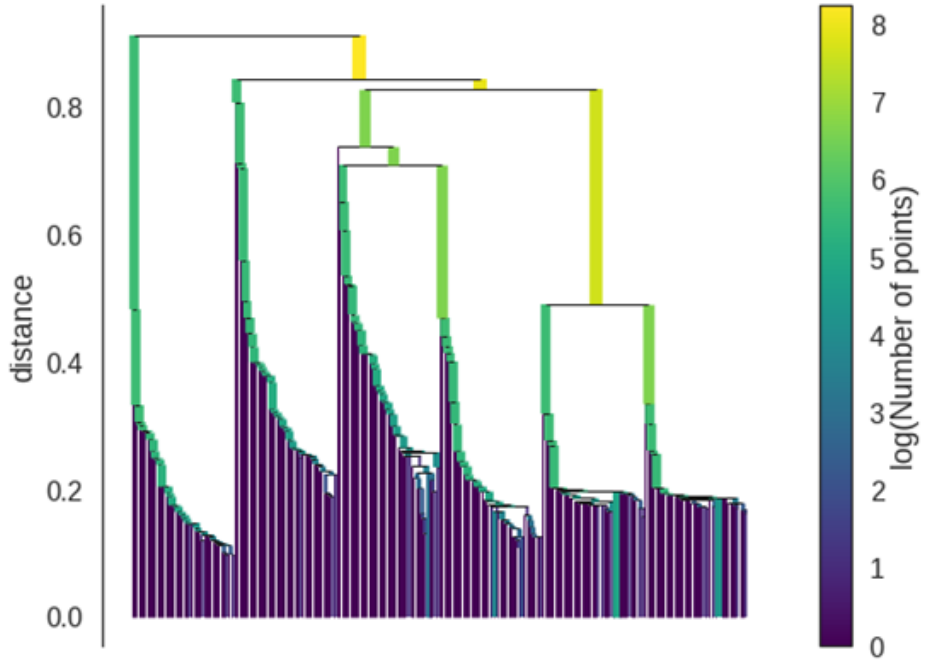
Step 2 : Minimum Spanning Tree 구축



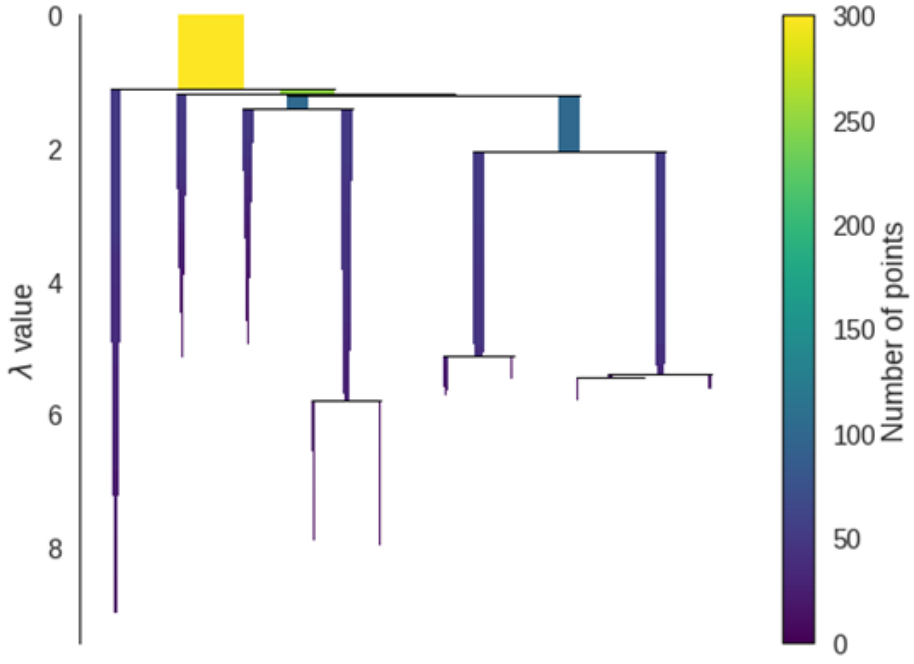
HDBSCAN

- HDBSCAN – Hierarchical Density Based Spatial Clustering of Applications with Noise)
  - HDBSCAN의 경우 더 이상 epsilon(e)이 필요하지 않으며 minPls(m)만 존재함
  - Point간의 mutual reachability distance 값을 기반으로 MST(minimum spanning tree)를 만듬
  - Tree를 바탕으로 Hierarchy를 구축함 → minPls 값을 기반으로 hierarchy를 축약함
  - 축약된 Cluster hierarchy에서 stable한 클러스터만 선택함

Step 3 : Hierarchy 구축



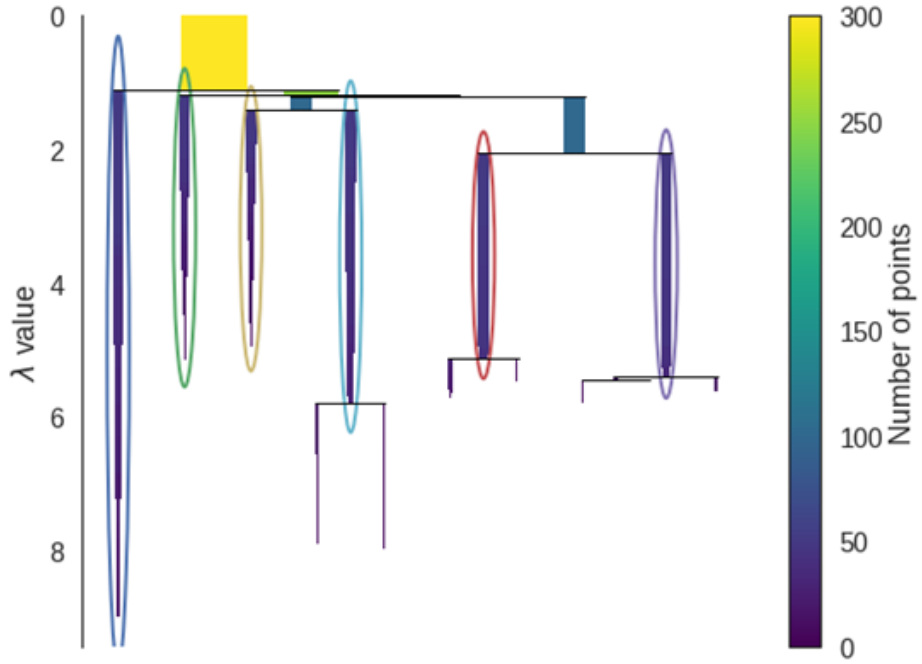
Step 4 : minPls(m) 기반 Cluster 축소



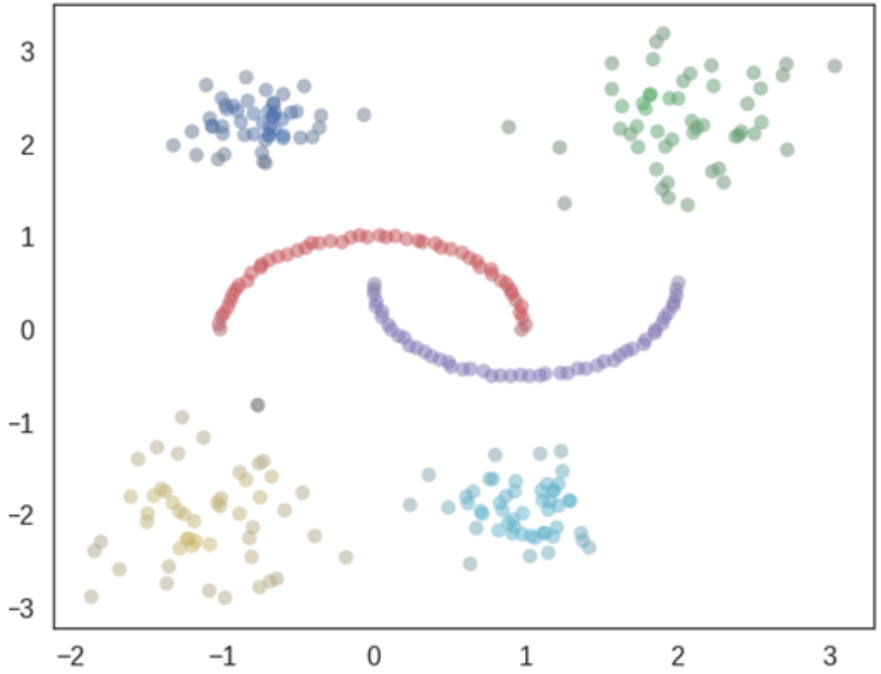
HDBSCAN

- HDBSCAN – Hierarchical Density Based Spatial Clustering of Applications with Noise)
  - HDBSCAN의 경우 더 이상 epsilon(e)이 필요하지 않으며 minPls(m)만 존재함
  - Point간의 mutual reachability distance 값을 기반으로 MST(minimum spanning tree)를 만듬
  - Tree를 바탕으로 Hierarchy를 구축함 → minPls 값을 기반으로 hierarchy를 축약함
  - 축약된 Cluster hierarchy에서 stable한 클러스터만 선택함

Step 5 : Stable한 클러스터만 선택



Final

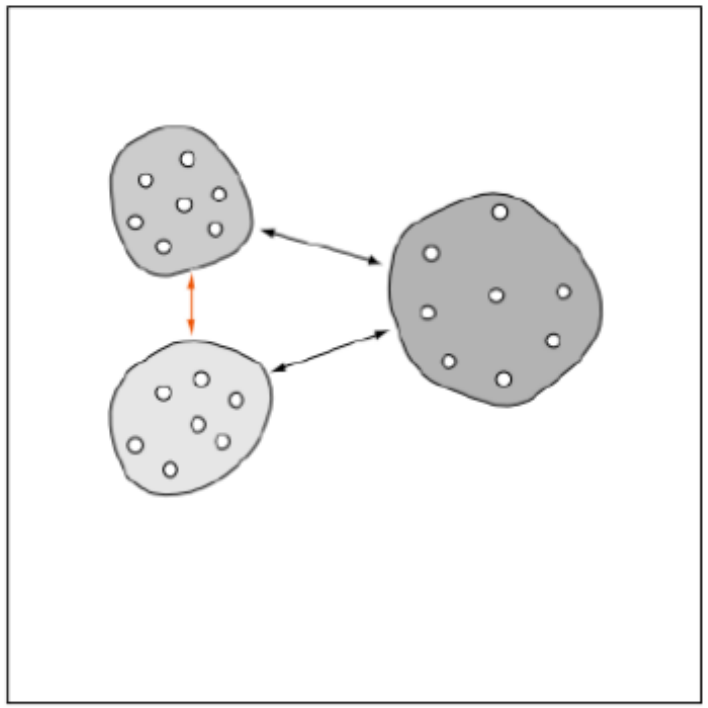


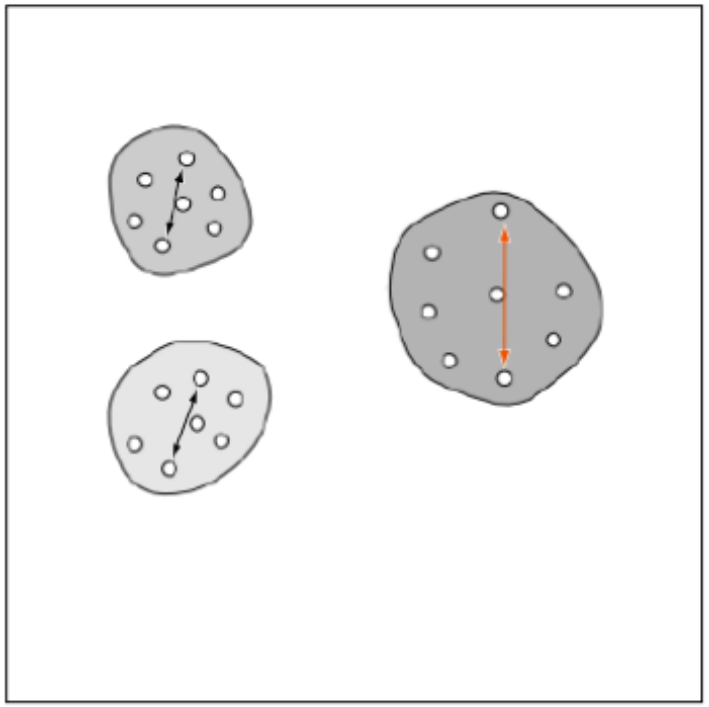
Real 추천 : https://www.kdnuggets.com/2020/02/understanding-density-based-clustering.html

Clustering 평가 지표

- Hyperparameter “K”의 타당성 지표
  - Dunn Index
    - 군집 감 거리의 최소값(좌측 하단)을 분자, 군자 내 요소 간 거리의 최대값(우측 하단)을 분모로 하는 지표
    - 군집 간 거리는 멀수록, 군집 내 분산은 작을 수록 좋은 군집화 결과라고 말할 수 있음

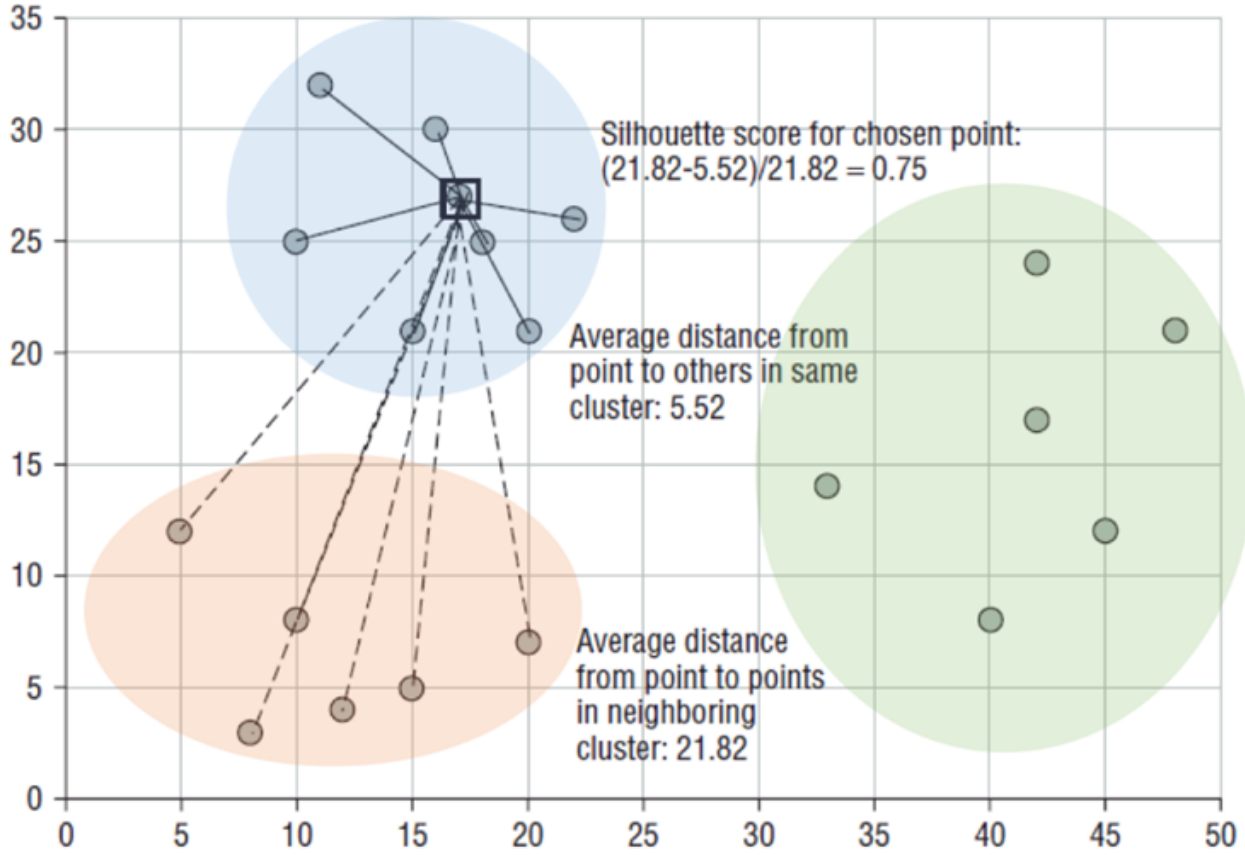
$$I(C) = \frac{\min_{i \neq j} \{d_c(C_i, C_j)\}}{\max_{1 \leq l \leq k} \{\triangle(C_l)\}}$$





Clustering 평가 지표

- Hyperparameter “K”의 타당성 지표
  - Silhouette
    - Dunn index의 경우 Clustering의 유효성을 검증하기 위한 하나의 값이 있음
    - Silhouette의 경우는 개체별로 그 적합성이 평가 됨
      - 즉, 모든 개체의 silhouette 값을 확인하고 Cluster별로 그 값의 분포에 문제가 없는지 확인하는 방식
    - 일반적으로 모든 개체의 silhouette 값이 0.5보다 크면, Clustering이 잘 된것이라고 판단함

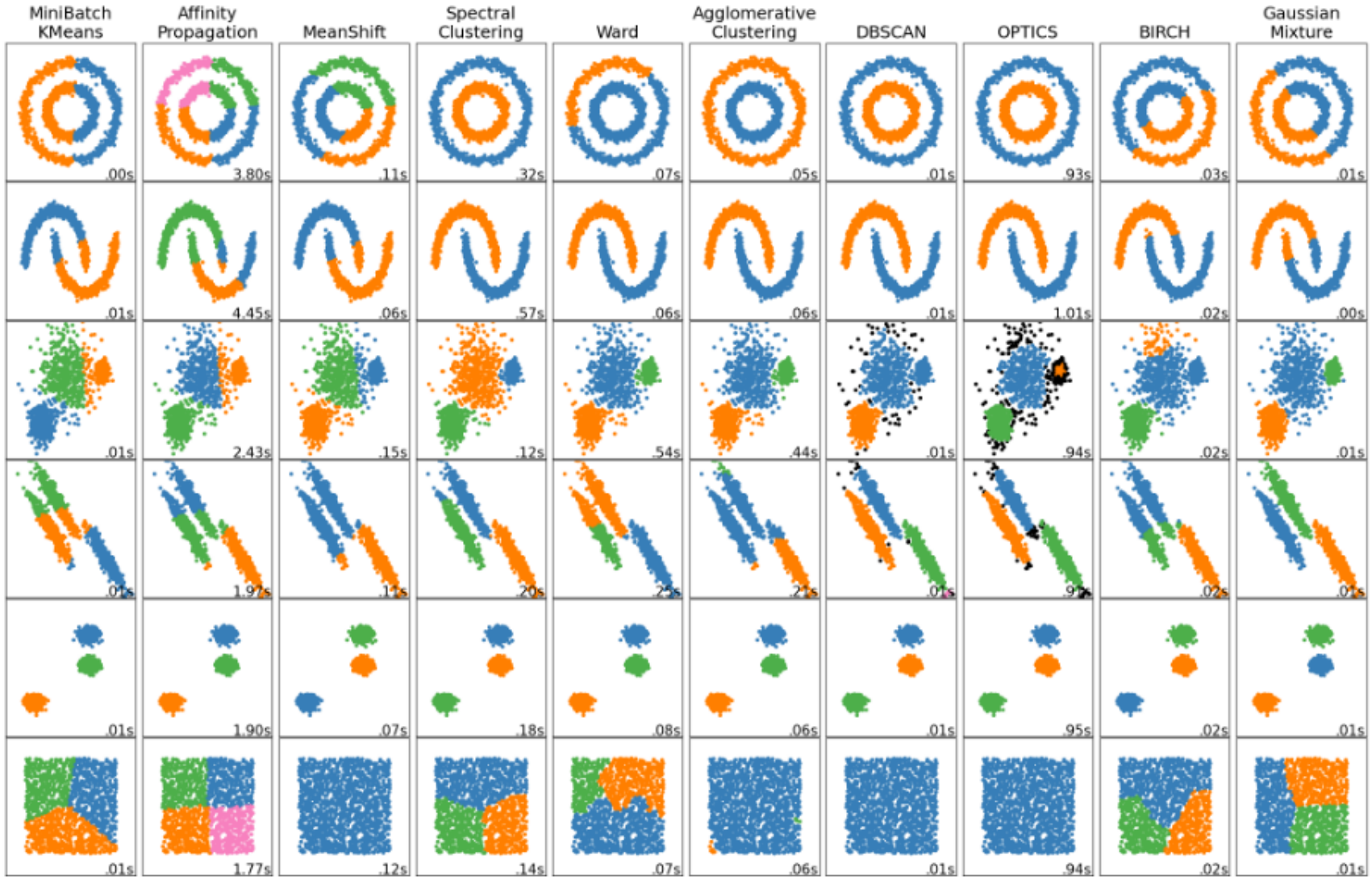


$$s(i) = \frac{b(i) - a(i)}{\max\{a(i), b(i)\}}$$

- 가장 이상적인 경우 (s = 1)
  - a(i) = 0
  - 한 군집의 모든 개체가 다 붙어 있음
- 최악의 경우 (s = -1)
  - b(i) = 0
  - 서로 다른 군집이 전혀 구분 되지 않음

Python Reference

- Overview of clustering methods



Real 추천 : https://scikit-learn.org/stable/modules/clustering.html

X's Scaling 해 ? 말아 ?

다양한 분야에 대한 데이터 과학자

GIGO

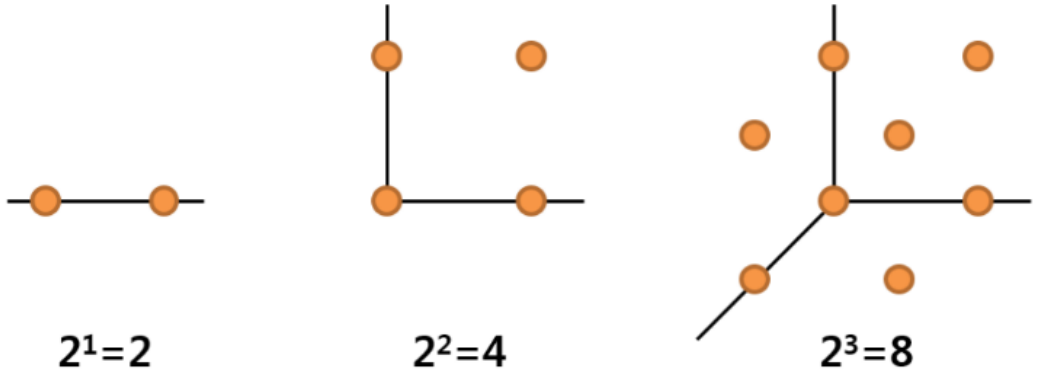
하나의 분야에 대한 데이터 과학자

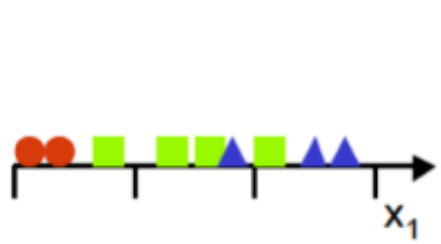
Domain에 맞게 !
내가 생각하는 분석에 맞게 !

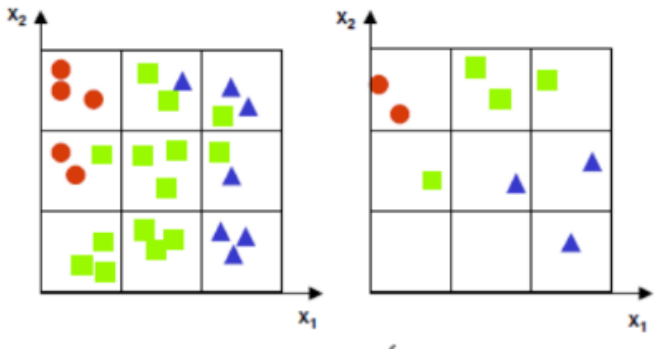
Dimensionality Reduction

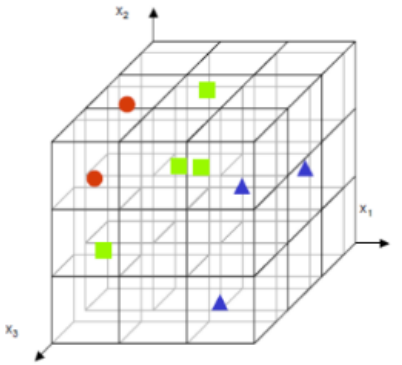
- Dimensionality Reduction – 차원 축소
  - 차원의 저주
  - Clustering 확인 지표2

> "If there are various logical ways to explain a certain phenomenon, the simplest is the best" - Occam's Razor





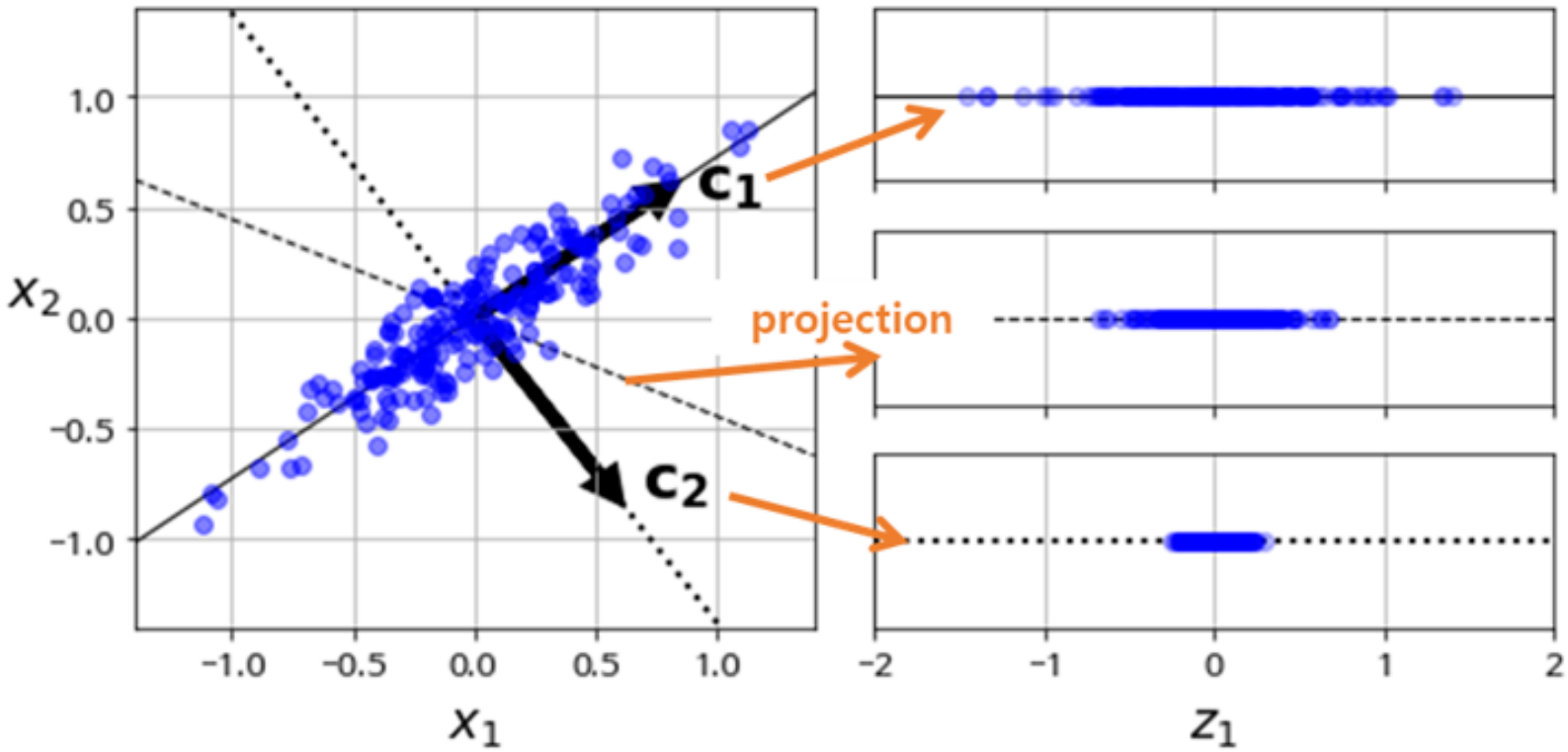




PCA

- Principal Component Analysis (PCA) – 주성분 분석
  - PCA는 가장 대표적인 차원 축소 알고리즘
  - PCA는 데이터에 가장 가까운 초평면(Hyperplane)을 구한 다음, 데이터를 이 초평면에 투영 시킴
  - 분산 보존
    - 저차원의 초평면에 데이터를 투영하기 전에 먼저 적절한 초평면을 선택해야함
    - PCA는 데이터의 분산이 최대가 되는 축을 찾음

- Key words : #Eigenvalue-Decomposition, #Singular Value Decomposition



T-SNE

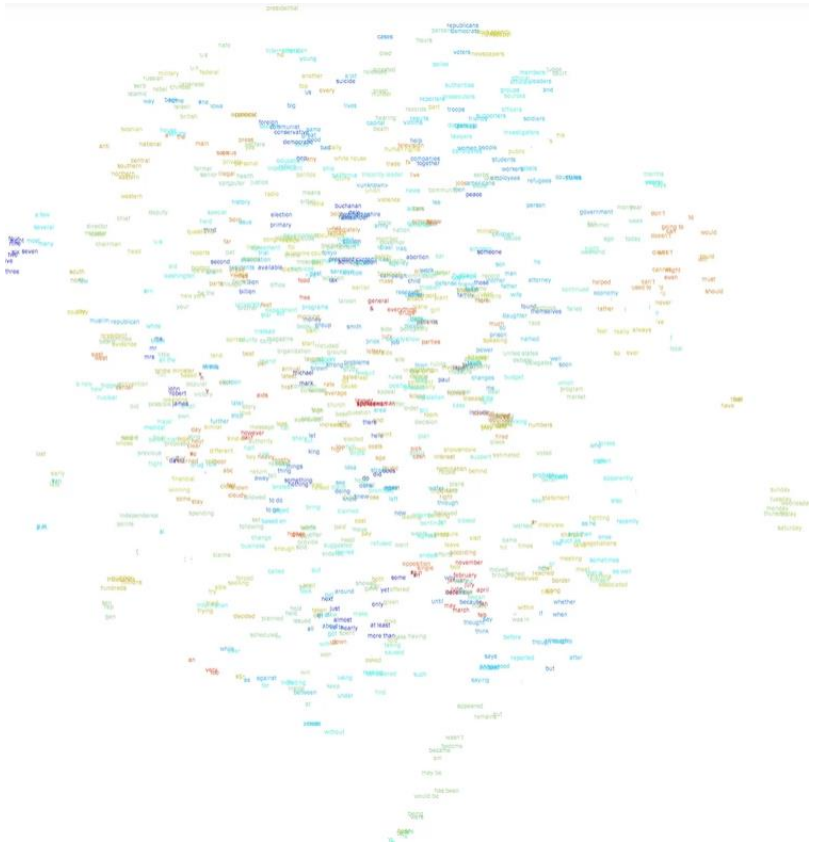
- T-SNE (Stochastic Neighbor Embedding)
  - 고차원 데이터를 시각화하는 데 가장 있기 있는 알고리즘
  - Text, Image를 차원 축소하여 시각화할 때 가장 많이 쓰임
  - 개념
    - 첫번째 식의 p는 고차원 원공간에 존재하는 i번째 개체 xi가 주어졌을 때 j번째 이웃인 xj가 선택될 확률을 의미
    - 두번째 식의 q는 저차원에 임베딩된 i번째 개체 yi가 주어졌을 때 j번째 이웃인 yj가 선택될 확률
    - SNE 목적 : p와 q의 분포 차이가 최대한 작게끔 하고자 하는 것
    - 두 확률 분포가 얼마나 비슷한지 측정하는 지표는 KL-divergence를 사용함

$$p_{j|i} = \frac{e^{-\frac{|x_i - x_j|^2}{2\sigma_i^2}}}{\sum_k e^{-\frac{|x_i - x_k|^2}{2\sigma_i^2}}}$$

$$q_{j|i} = \frac{e^{-|y_i - y_j|^2}}{\sum_k e^{-|y_i - y_k|^2}}$$

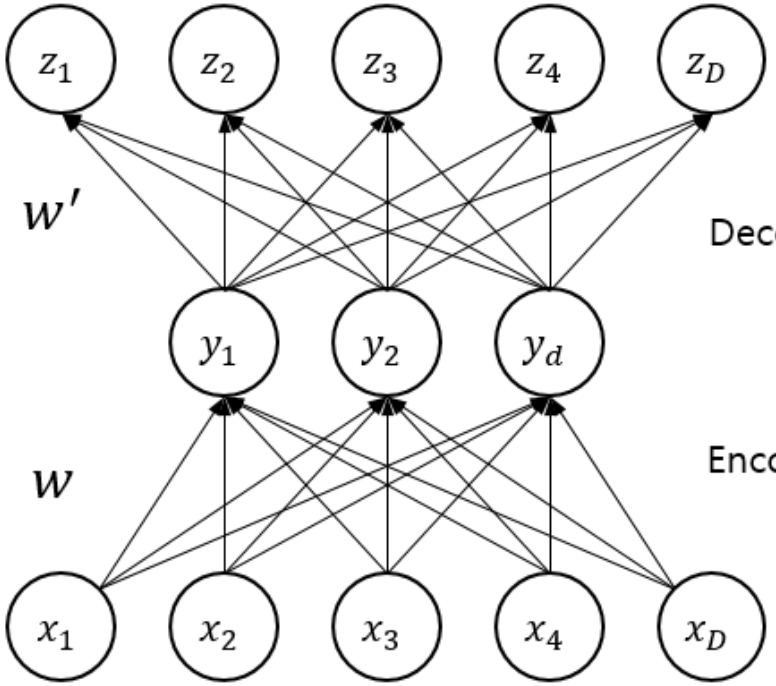
$$Cost = \sum_i KL(P_i||Q_i)$$

$$= \sum_i \sum_j p_{j|i} \log \frac{p_{j|i}}{q_{j|i}}$$



AutoEncoder

- AutoEncoder
  - Deep Learning 계열에서 가장 많이 쓰이는 차원 축소 방법
  - 자기 자신을 복제 함으로써 자기 자신을 가장 잘 표현하는 Hidden layer를 찾음



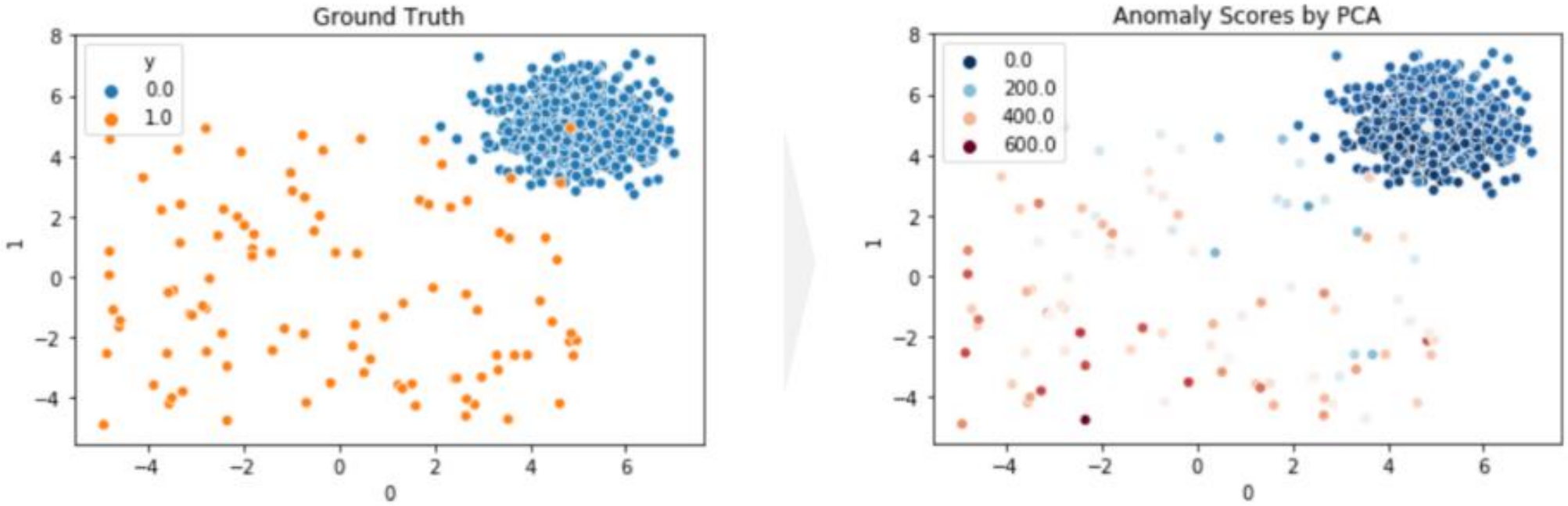
Decoder : $z = g_{\theta'}(y) = s(W'y + b')$

Encoder : $y = f_\theta(x) = s(Wx + b)$

$$\theta^*, \theta'^* = \arg\min \frac{1}{N} \sum_{i=1}^{n} L(x^{(i)}, z^{(i)})$$

차원 축소 기법을 활용한 이상치 탐지

- Anomaly Detection using DR
  - AD : 비정상 → 비정상은 1% 내외
  - PCA
    - 분산을 최대로 하는 새로운 축을 건설함
    - 새로운 축을 건설할 때 **데이터의 분포를 잘 대변하는 방향**으로 만들어짐
      - 이상치는 반영이 안될 것
    - 새로운 축으로 사영했을 때의 기존 고차원에서 얼마만큼 이동을 했는지 → Anomaly Score
  - Autoencoder
    - Reconstruction Error
      - 자기 자신을 잘 복제하는 Model을 Training 시킴
      - 정상만을 가지고 잘 학습했을 때 **새로운 분포인 비정상이 들어오면 복제를 잘 못함** → Recon Error



Q & A