Distance Measure and K-means

I. 군집분석과 비지도학습에 대해,

머신 러닝은 크게는 두 가지, 지도학습과 비지도학습으로 나뉜다. 지도학습은 이미 결과를 알고 있는, label이 있는 데이터를 예측하거나 분류하는 것이다. 반면 비지도학습은 label이 없는 데이터에서 패턴을 발견하고, 숨겨진 구조를 찾아내는 것이다.

<지도학습>

"데이터에 대한 Label 이 주어진 상태에서 컴퓨터를 학습시키는 방법"

Feature : "개별적이고 측정 가능한 Heuristic한 속성"

NO.	SIZE	COLOR	SHAPE	FRUIT NAME
1	Big	Red	Rounded shape with a depression at the top	Apple
2	Small	Red	Heart-shaped to nearly globular	Cherry
3	Big	Green	Long curving cylinder	Banana
4	Small	Green	Round to oval,Bunch shape Cylindrical	Grape

(Training) Data

Label : "명시적인 정답"

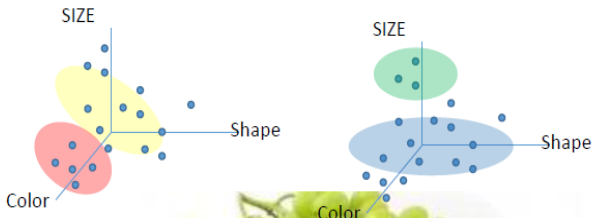
- ✓ 수박? -> (Big, Green, Round shape)
- ✓ Feature가 다양할수록 복잡한 문제를 풀 수 있다

<비지도학습>

"데이터에 대한 Label 이 주어지지 않은 상태에서 컴퓨터를 학습시키는 방법"

NO.	SIZE	COLOR	SHAPE
1	Big	Red	Rounded shape with a depression at the top
2	Small	Red	Heart-shaped to nearly globular
3	Big	Green	Long curving cylinder
4	Small	Green	Round to oval,Bunch shape Cylindrical

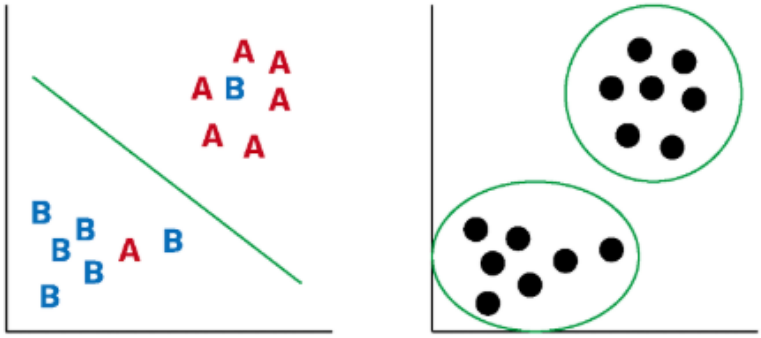
(Training) Data



- Then you will arrange them on considering base condition as **color.**
- Then the groups will be some thing like this.
- RED COLOR GROUP: apples & cherry fruits.
- GREEN COLOR GROUP: bananas & grapes.

- so now you will take another physical character such as **size**
- RED COLOR AND BIG SIZE: apple.
- RED COLOR AND SMALL SIZE: cherry fruits.
- GREEN COLOR AND BIG SIZE: bananas.
- GREEN COLOR AND SMALL SIZE: grapes.

대표적인 비지도학습으로 군집분석이 있다. 조금 더 쉬운 이해를 위해 그림으로 표현해보겠다.

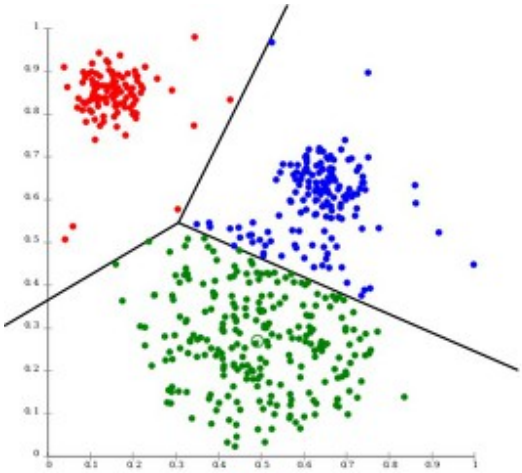


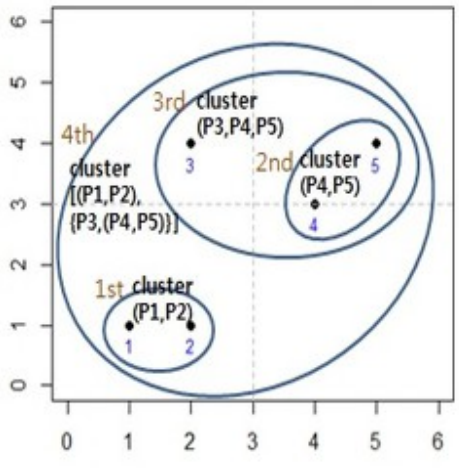
왼쪽의 점들은 A, B라는 label, 즉 그들의 정체가 밝혀져 있다. 오른쪽 점들은 그것들의 정체를 가려 놓은 것이다.

1) 왼쪽 데이터의 경우, 직선 하나를 그어 놓고 점들을 분류한다고 생각해보자. 우리는 그들의 정체를 알고 있기 때문에, 분류에 대한 평가까지 할 수 있다. 만약 직선 위의 데이터라면 A. 직선 아래의 데이터라면 B이다. A, B 라는 기준에 따라 이리저리 선을 긋다 보면 위와 같이 적당한 분류를 하는 선을 그을 수 있다. 또한, 총 14개의 데이터중 2개가 다른 위치에 있기 때문에 2/14 라는 오답율까지 구할 수 있으며, 새로운 데이터가 들어왔을 때 A 혹은 B 라고 정해줄 수도 있다.

2) 오른쪽 데이터의 경우, 우리는 저 까만 점들의 정체를 알 수 없다. 그렇기 때문에 직선을 그어 그들을 평가할 만한 기준이 없는 것이다. 하지만 데이터들의 패턴을 발견할 수는 있다. 위 데이터는 두 개의 지점을 기준으로 군집을 형성하고 있다는 것이다. 따라서, 그림과 같이 두 개의 군집으로 우리는 저 데이터를 나눌 수 있다. 이것이 군집분석과 비지도학습의 매우 간단한 개요이다.

II. 계층적 군집분석과 비계층적 군집분석



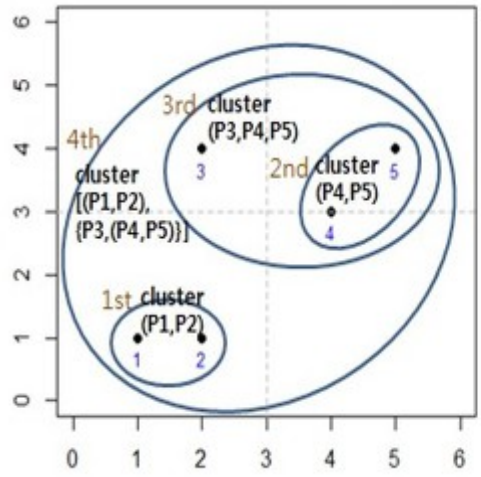


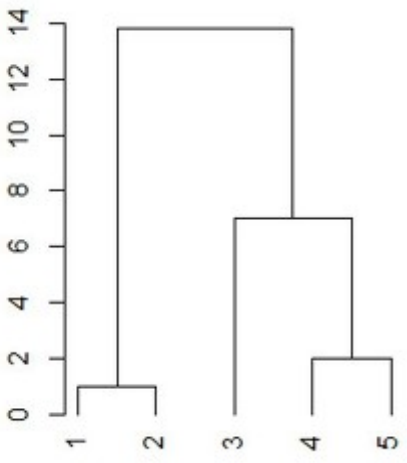
군집분석은 입력된 데이터들의 값에 따라 어떤 데이터들이 좀 더 비슷한 성질을 가지고 있는지 파악하여 비슷한 것들끼리 군집으로 묶어주는 분석방법이다. 군집분석의 목적을 간략하게 표현한다면 다음과 같다고 할 수 있다.

- 유사한 성향을 가진 개체를 모아 군집을 형성할 수 있다.
- 시각적 표현을 통하여 군집 간의 특성을 관찰할 수 있다.
- 개체를 분류하기 위한 명확한 분류 기준이 존재하지 않거나 기준이 밝혀지지 않은 상태에서 유용하게 이용할 수 있다.

군집분석 알고리즘에는 계층적 군집분석과 비계층적 군집분석이 있는데 용도에 따라 다르게 사용된다.

1) 계층적 군집분석

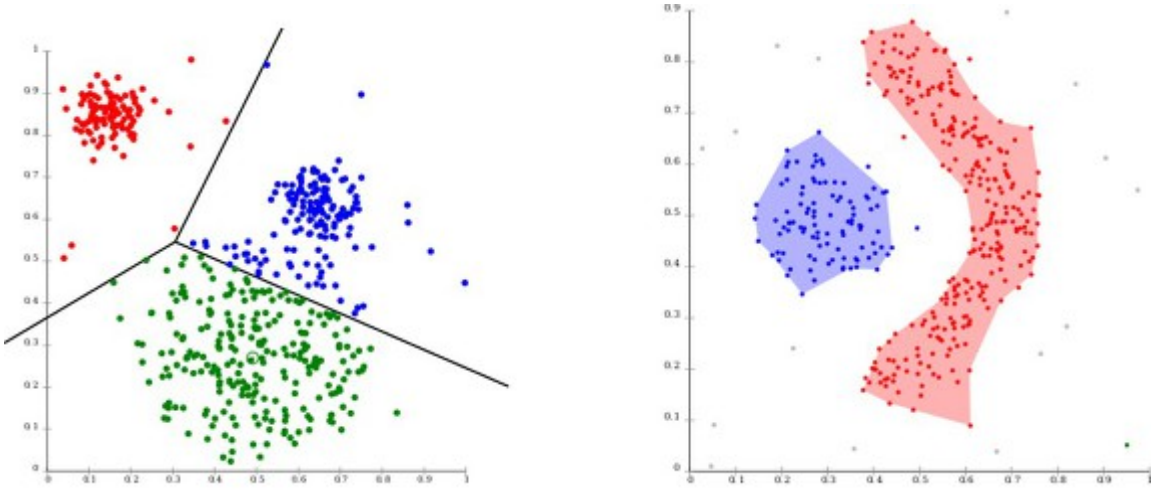




먼저 계층적 군집분석은 한 군집이 다른 군집을 포함할 수 있는 구조로 군집을 만드는 기법이다.

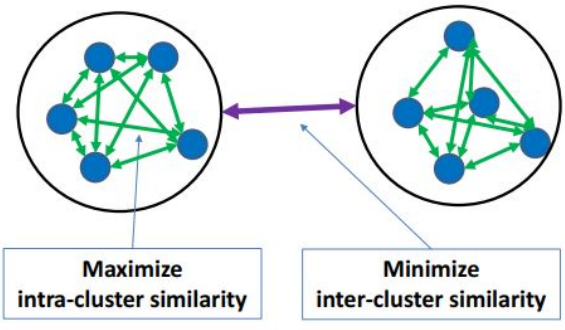
음악 장르를 구분하는 것과 비슷하게 음악이라는 것을 댄스, 발라드, 힙합, 락으로 구분하고 락에는 펑크락과 하드락이 있고 그 아래에 또 세분화되고 이런 식으로 계층화되는 형태로 데이터를 군집화한다. 그래서 계층적 군집분석은 오른쪽 위의 그림과 같이 가지형태로 파생되는 형태로 결과가 얻어지는 데 이것을 덴드로그램(Dendrogram)이라고 한다.

2) 비계층적 군집분석



비계층적 군집분석은 군집끼리 포함관계를 이루지 않고 서로 독립적인 한 군집으로 만드는 기법이다. 위와 같이 분포한 데이터를 빨강그룹, 파랑그룹, 초록그룹으로 구분하여 그룹 짓는 것이 바로 비계층적 군집분석이라고 할 수 있다. 비계층적 군집분석은 거리를 기반으로 군집화하는 방법(K-means)(왼쪽 위 그림)과 밀도를 기반으로 군집화 하는 방법(DB-SCAN)(오른쪽 위 그림)이 있는데 데이터가 분포한 특성에 따라 좀 더 말을 잘 듣는 방법을 선택해서 사용할 수 있다.

III. Distance Measure



동일한 군집에 속한 데이터는 서로 유사할수록 좋고, 다른 군집에 속한 데이터들은 서로 다를수록 좋다. (High intra-class similarity & Low inter-class similarity)

여기서 유사하다(Similarity) or 유사하지 않다(Dis-similarity)를 어떻게 측정할까?

1) 유클리디안 거리(Euclidean distance)

계산 값이 0에 가까울수록 유사함.

$$dist = \sqrt{\sum_{k=1}^{n}(p_k - q_k)^2}$$

where n is **the number of dimensions (attributes)** and
p_k and q_k are **the value of k^{th} attribute** of data objects p and q.

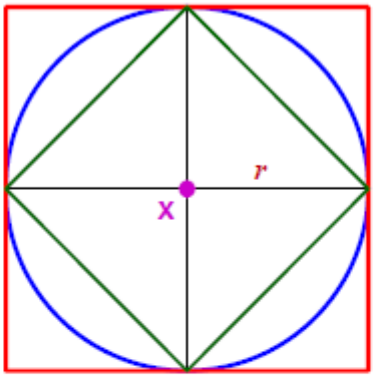
2) 민코스키 거리(Minkowski Distance)

유클리디안 거리를 일반화 한 것이다. (2→r)
(r=1이면 맨하탄 거리, r=2일 때는 유클리디안 거리와 동일

$$dist = (\sum_{k=1}^{n}|p_k - q_k|^r)^{\frac{1}{r}}$$

where r is a **parameter (r = 2 → Euclidean Distance)**,
n is **the number of dimensions (attributes)** and
p_k and q_k are **the value of k^{th} attribute** of data objects p and q.

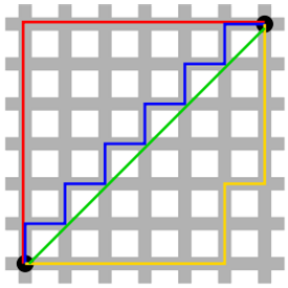
- *r=1이면 Manhattan Dist.(L1), r=2면 Euclidean Dist.(L2), r→∞면 Supremum Dist. (Lmax)*



- Green: All points y at distance $L_1(x, y) = r$ from point x
- Blue: All points y at distance $L_2(x, y) = r$ from point x
- Red: All points y at distance $L_\infty(x, y) = r$ from point x

3) 맨하탄 거리(Manhattan distance)

맨하탄 거리는, 마치 사각형 격자로 이뤄진 지도에서 사각형을 가로지르지 않고 갈 수 있는 최단 거리를 구하는 공식이다. → 변수 간 상관성이 없고, 데이터 변수가 많을 때(차원이 클 때) 맨하탄 거리를 이용하는 것이 좋다.



$$d_M(x,y)=\sum_{j=1}^{m}|x_j-y_j|$$

초록색 경로는 유클리디안 거리, 나머지는 맨하탄 거리

4) 마할라노비스 거리(Mahalanobis distance)

유클리디언 거리에서 데이터의 속성들의 공분산(covariance)을 반영하여 거리를 계산하는 방법.

계산값이 0에 가까울수록 유사함. → 변수 간의 상관 관계가 존재할 때 사용

$$D_{Mahal}(p,q)=(p-q)\Sigma^{-1}(p-q)^T$$

- Σ is the **covariance matrix** of the input data $X=\begin{bmatrix}X_1\\ \vdots \\ X_n\end{bmatrix}$ (*n* is **the dimension of data**)
- $\Sigma_{ij}=\mathrm{cov}(X_i,X_j)=\mathrm{E}[(X_i-\mu_i)(X_j-\mu_j)]$

$$\Sigma_{ij}=\frac{1}{m-1}\sum_{k=1}^{m}(X_{ik}-\overline{X}_i)(X_{jk}-\overline{X}_j) \sim \textit{sample covariance (m is the number of data)}$$

- $\Sigma=\begin{bmatrix}\mathrm{E}[(X_1-\mu_1)(X_1-\mu_1)] & \mathrm{E}[(X_1-\mu_1)(X_2-\mu_2)] & \cdots & \mathrm{E}[(X_1-\mu_1)(X_n-\mu_n)]\\ \mathrm{E}[(X_2-\mu_2)(X_1-\mu_1)] & \mathrm{E}[(X_2-\mu_2)(X_2-\mu_2)] & \cdots & \mathrm{E}[(X_2-\mu_2)(X_n-\mu_n)]\\ \vdots & \vdots & \ddots & \vdots \\ \mathrm{E}[(X_n-\mu_n)(X_1-\mu_1)] & \mathrm{E}[(X_n-\mu_n)(X_2-\mu_2)] & \cdots & \mathrm{E}[(X_n-\mu_n)(X_n-\mu_n)]\end{bmatrix}$.
- Σ^{-1} is called **the inverse covariance matrix** (the inverse of the covariance matrix)

X, Y 변수 간에 양의 상관관계가 존재할 때, 이를 고려 (단순히 유클리디안 거리로 구하게 된다면 A는 C보다 B와 가깝다고 측정될 것- 데이터 분포를 생각하지 않고)

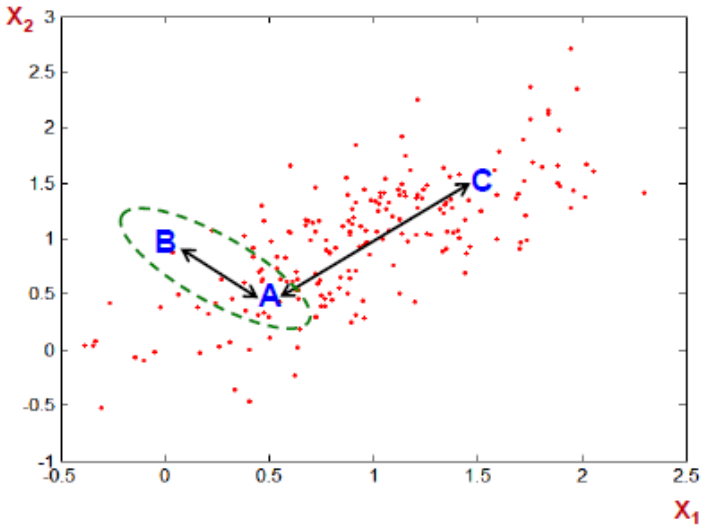
- **Data Points**
  - A: (0.5, 0.5)
  - B: (0, 1)
  - C: (1.5, 1.5)
- **Covariance Matrix:**

$$\Sigma = \begin{bmatrix} 0.3 & 0.2 \\ 0.2 & 0.3 \end{bmatrix}$$

- **Mahalanobis Distance**

$D_{Mahal}(A, B)$ =

$D_{Mahal}(A, C)$ =



계산 시, 위의 값은 5, 아래는 4로 A-C가 마할로비스 거리는 더 가까운 것을 볼 수 있다.

5) 코사인 유사도(Cosine Similarity)

위의 거리 기반의 유사도와 달리 코사인 유사도는 각(radian) 기반의 계산법. 벡터의 크기에 영향을 받지 않는다는 특징이 있다. 값의 범위는 -1~1이며, 1에 가까울수록 유사하고 -1에 가까울수록 다르다고 볼 수 있다. 문서 간 거리를 구할 때 많이 쓰인다. (두 벡터 사이의 거리를 구하는 유클리드 거리와 달리 코사인 유사도는 두 벡터 사이의 각도를 이용하여 구한다)

→데이터의 절대적 크기가 중요하지 않고, 변수 간의 상대적 비율만 고려해야 할 경우에 쓰임.

If d_1 and d_2 are two **document vectors**, then

- $\cos(d_1, d_2) = (d_1 \bullet d_2) / \|d_1\| \|d_2\|$

  where • indicates **vector dot product** and $\|d\|$ is the **length of vector** d.

Example:

d_1 = 3 2 0 5 0 0 0 2 0 0

d_2 = 1 0 0 0 0 0 0 1 0 2

- $d_1 \bullet d_2$
- $\|d_1\|$ =
- $\|d_2\|$ =
- $\cos(d_1, d_2)$ =

정답) 5, 6.48, 2.45, 0.315

6) Similarity between Binary Vectors (SMC/Jaccard)

주로 범주에서 쓰이는 기법으로 p와 q라는 두가지 object가 있을 때 쓰이는 유사도 측정법이다.

SMC(Simple Matching)와 Jaccard가 있는 데 수식은 다음과 같다.

Simple Matching and Jaccard Similarity Coefficients

- **SMC = number of matches / number of attributes**
  = $(M_{11} + M_{00}) / (M_{01} + M_{10} + M_{11} + M_{00})$
- **J = # of 11 matches / # of not-both-zero attributes values**
  = $(M_{11}) / (M_{01} + M_{10} + M_{11})$

M값에 대한 설명은 다음과 같다.

Compute **similarities** using the following **quantities**

- M_{01} = the number of attributes where **p was 0** and **q was 1**
- M_{10} = the number of attributes where **p was 1** and **q was 0**
- M_{00} = the number of attributes where **p was 0** and **q was 0**
- M_{11} = the number of attributes where **p was 1** and **q was 1**

EXAMPLE:

***p* = 1 0 0 0 0 0 0 0 0 0**

***q* = 0 0 0 0 0 0 1 0 0 1**

M_{01} = 2 (the number of attributes where p was 0 and q was 1)

M_{10} = 1 (the number of attributes where p was 1 and q was 0)

M_{00} = 7 (the number of attributes where p was 0 and q was 0)

M_{11} = 0 (the number of attributes where p was 1 and q was 1)

SMC = $(M_{11} + M_{00}) / (M_{01} + M_{10} + M_{11} + M_{00})$ =

J = $(M_{11}) / (M_{01} + M_{10} + M_{11})$ =

정답) 0.7, 0

IV. K-means Clustering

1) 알고리즘 개요

K-means 알고리즘은 비계층적 군집방법 중 가장 널리 사용되는 것으로 K개의 군집 중심좌표를 고려하여 각 개체를 가까운 군집에 배정하는 반복적 알고리즘이다. 구체적 절차는 다음과 같다.

1: Select K points as the initial centroids.
2: **repeat**
3: Form K clusters by assigning all points to the closest centroid.
4: Recompute the centroid of each cluster.
5: **until** The centroids don't change (stopping condition)

단계0. 초기 객체 선정
어떤 규칙에 의하여 K개의 객체 좌표를 초기 군집의 중심좌표(Centroid)로 선정한다.

단계1. 객체의 군집 배정
각 개체에 대하여 K개의 군집 중심좌표(Centroid)와의 거리(주로 유클리디안 사용)를 산출한 후 가장 가까운 군집에 그 객체를 배정한다.

단계2. 군집 중심좌표의 산출
새로운 군집에 대한 중심좌표를 산출한다.

단계3. 수렴조건 점검
새로 산출된 중심 좌표값과 이전 좌표값을 비교하여 수렵 조건 내에 들면 마치며, 그렇지 않으면 단계1로 돌아가 반복한다.

위의 초기 K개의 객체를 선정하는 방법으로 다음과 같은 규칙이 사용된다.

무작위방법: 대상 객체 중 무작위로 K개를 선정한다

외각 객체선정: 전체 객체의 중심좌표에서 가장 멀리 위치하는 K개의 객체를 선정한다.

위에서 군집의 중심좌표란, 해당 군집에 속한 객체들의 평균치로 이루어진 값을 의미한다. 즉, j번째 군집의 중심좌표 c_j는 다음과 같다.

$$c_j = \left(\overline{X_1}^{(j)}, \overline{X_2}^{(j)}, \dots, \overline{X_p}^{(j)}\right)^T, \qquad j = 1,2,\dots,K$$

이상과 같이 K-means알고리즘에서는 객체들의 군집이 결정되면 중심좌표가 산출되며 중심좌표가 정해지면 이와 관련된 새로운 군집이 형성된다. 결국, n개의 객체를 K개의 군집으로 나눌 때 각

객체를 어떤 군집에 배정하는 것이 전체 거리를 최소로 하는가 하는 최적화 문제로 해석이 가능하다.

Let,

$$a_{ij} = \begin{cases} 1, & 객체i가\ 군집j에\ 배정될때 \\ 0, & otherwise \end{cases}$$

K-means알고리즘은 다음과 같은 최적화 문제로 표현이 가능하다.

MIN $$Z = \sum_{i=1}^{n}\sum_{j=1}^{k} d(x_i, c_j) * a_{ij} \quad (a)$$

Subject to

$c_j = \sum_{i=1}^{n}(x_i * a_{ij}) / \sum_{i=1}^{n} a_{ij}\ , j = 1, \dots, K$ (b)

$\sum_{j=1}^{K} a_{ij} = 1\ , i = 1, \dots, n$ (c)

$\sum_{j=1}^{n} a_{ij} \geq 1\ , j = 1, \dots, n$ (d)

$a_{ij} = 1\ or\ 0, for\ i = 1, \dots, n; j = 1, \dots, K$ (e)

식(a)의 목적함수는 객체와 해당 군집의 중심좌표와의 거리합을 나타낸 것이며, 식(b)는 중심좌표를 나타낸 식이며, 식 (c)는 각 객체가 K개의 군집 중 하나에 속하여야 함을 의미한다. 식(d)는 한 군집에 하나 이상의 객체를 포함하여야 함을 의미한다.