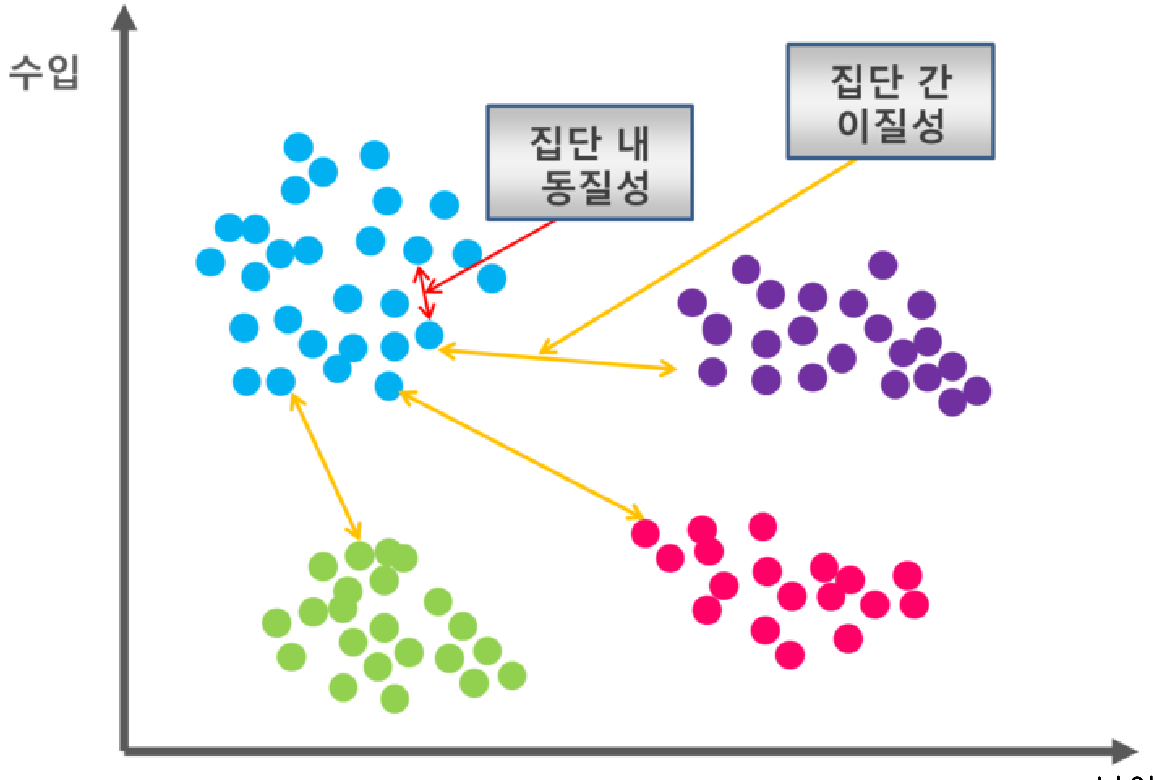
군집 분석

- 군집분석 개념
- 군집 분석에서의 유사성
- 분할기반 군집분석 알고리즘 : K-means
- K-means 클러스터링 실습
- 군집분석 결과 평가
- 군집분석 결과 비교
- SSE 비교를 통한 k 탐색

나이

군집 분석

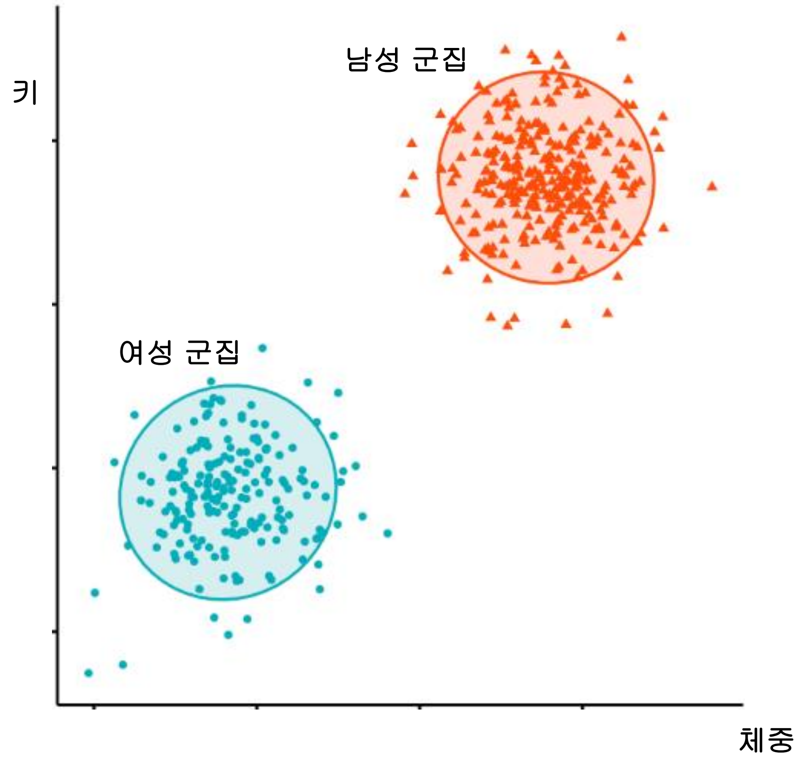
- 각 객체의 유사성을 측정하여 비슷한 특성을 가진 그룹을 찾는데 사용되는 분석 방법
- 예를 들어, 사람들의 나이와 경제적 수입이 유사한 그룹을 찾아내어 해당 그룹의 공통적인 특성, 즉 직업이나 거주지역, 근무환경 등을 분류할 수 있음



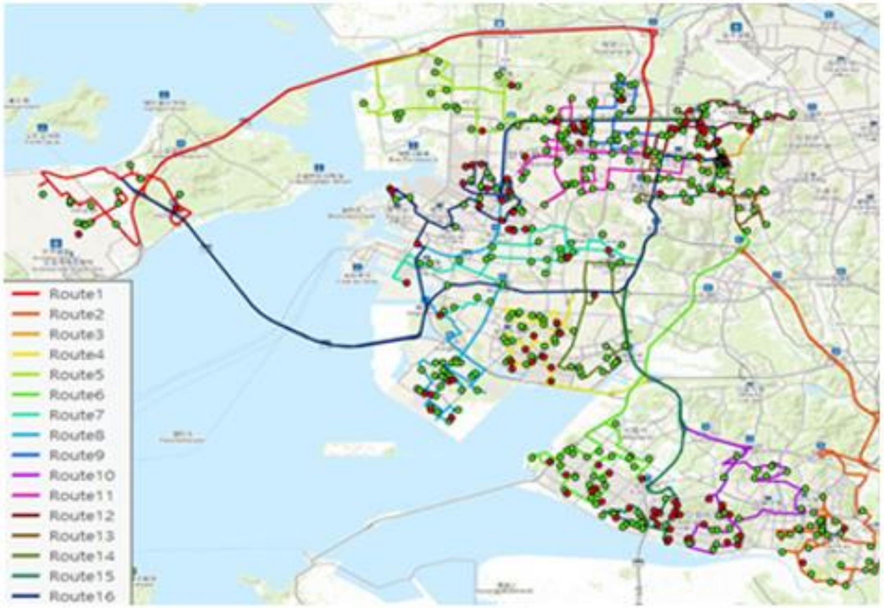
어느 그룹의 나이와 경제수입의 군집분석

군집 분석

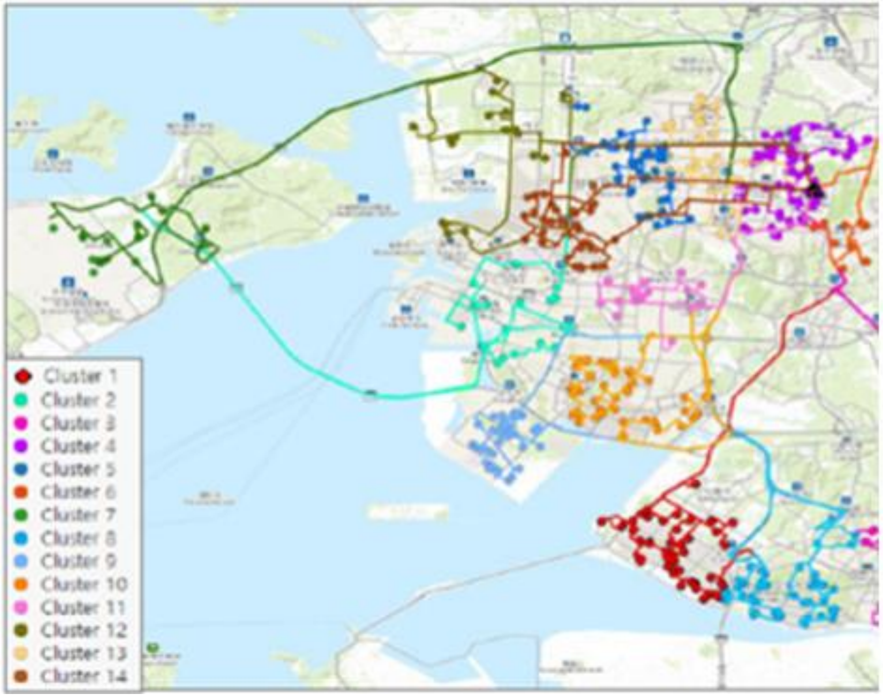
- 활용 - 사람들의 키와 체중을 대상으로 군집 분석 수행
- 키와 체중의 값에 따라 남성과 여성의 군집 분석 결과



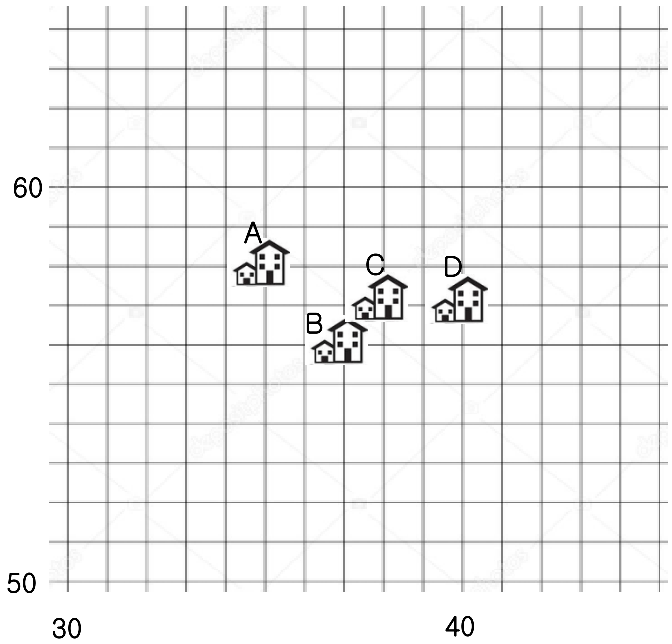
군집분석 - 화물 배송 경로 개선



- 고객 위치를 기준으로 유사한 지역 군집
- 투입 차량 16대 -> 14대로 감소
- 차량 이동거리 평균 66.08km에서 56.61km로 감소



객체 간의 유사성 - 거리 기반



위도 경도	x	y
상점A	35	58
상점B	37	56
상점C	38	57
상점D	40	57

	상점A	상점B	상점C	상점D
상점A	0	2.828	3.162	5.099
상점B	2.828	0	1.414	3.162
상점C	3.162	1.414	0	2
상점D	5.099	3.162	2	0

Distance Matrix

- 거리함수 *d*를 Euclidean distance 로 사용
- 상점의 좌표 (x, y)

분할 기반 알고리즘 : 기초

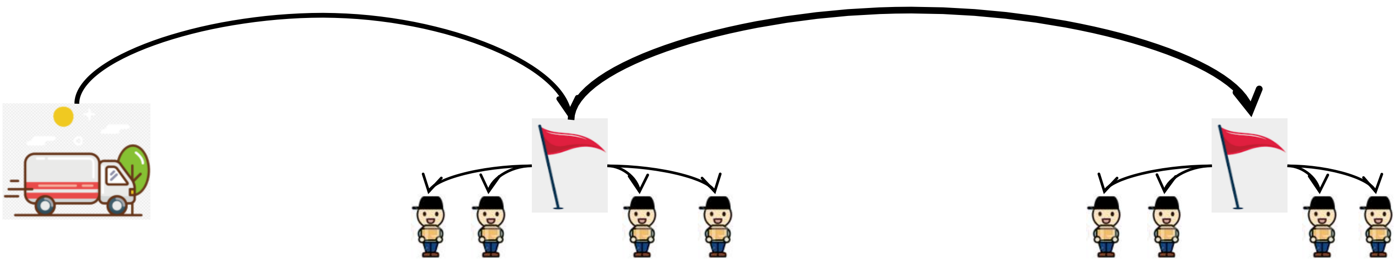
- 객체들을 대상으로 **k**개의 군집으로 분할을 수행
- 주어진 **k**에 선택한 분할 기준을 최적화할 수 있는 **k**개의 군집을 찾음
  - 반복적으로 수많은 분할을 수행 검토함
  - 각 군집은 중심점에 의해 표현됨
  - 객체들이 어느 중심점에 가까운가에 따라 군집을 결정함

K-Means 군집 분석 알고리즘

- 주어진 *k로*, ***k-means*** 알고리즘은 **4**단계로 수행함
  - 모든 객체들을 *k*개의 그룹으로 분할함
  - 분할한 클러스터 내의 객체들로부터 새로 **seed** 객체를 탐색, 클러스터의 중심(또는 평균) 값을 **centroid**로 정함
  - 각 객체들을 인접한 **seed** 객체로 할당함
  - 클러스터가 변화하지 않을때까지 **2**번째 단계부터 반복함

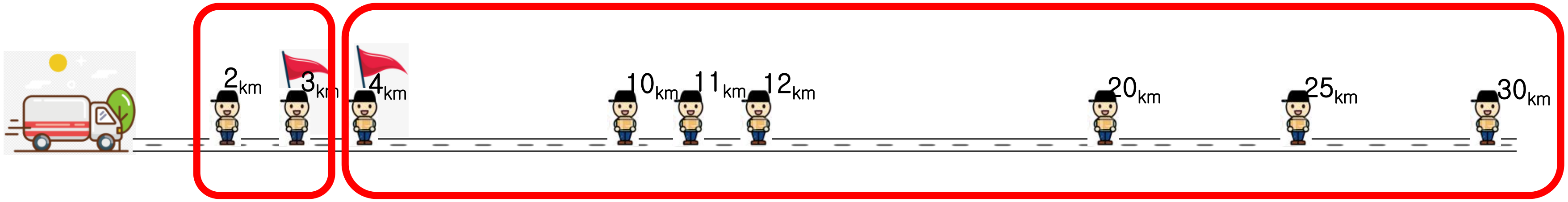
K-Means 예시

- 물류업체는 택배배송을 위해 각 택배 도착지까지 배송거리를 최소화
- 각 도착지를 모두 오가는 비용을 줄이고자, 중간지점을 찾아 인접 도착지들로 배송
- 도착지들을 군집분석을 수행, 인접한 도착지 군집을 분석
- 거리를 최소로 하는 군집을 찾기 위해 **K-means** 수행



K-means 1회차

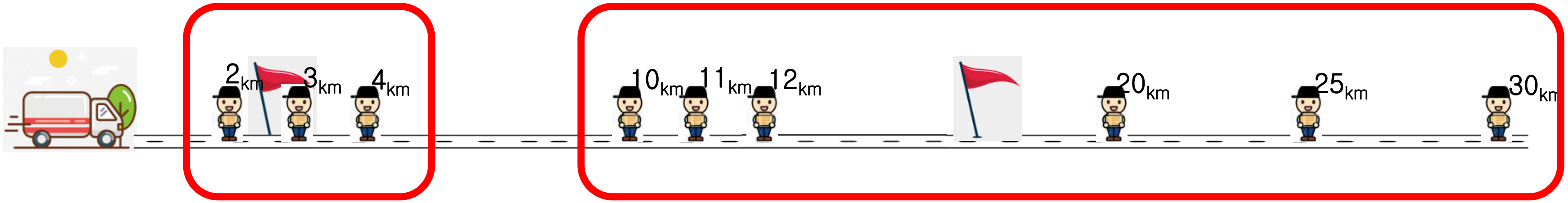
- 출발지로부터 도착지들의 거리를 1-d 클러스터링 수행
- {2_{km}, 4_{km}, 10_{km}, 12_{km}, 3_{km}, 20_{km}, 30_{km}, 11_{km}, 25_{km}}
- 임의의 군집 중심: $m_1=3, m_2=4$



- 각 중심에 가까운 군집#1, 군집#2 생성
- 각 군집#1, 군집#2의 새 중심 계산
  - 중심#1= (2+3)/2 = 2.5
  - 중심#2=(4+10+11+12+20+25+30)/7 = 16

K-means 2회차

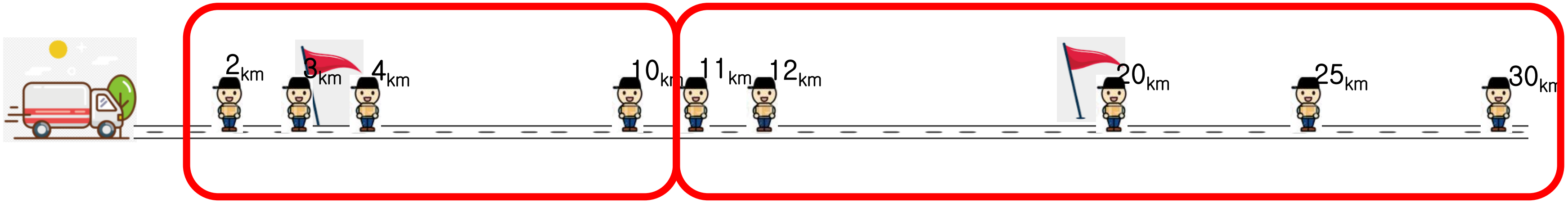
- 중심#1=2.5, 중심#2=16
- 각 중심에 가까운 군집#1, 군집#2 생성



- 각 군집#1, 군집#2의 새 중심 계산
  - 중심#1= (2+3+4) /3=3
  - 중심#2= (10+11+12+20+25+30)/6=18

K-means 3회차

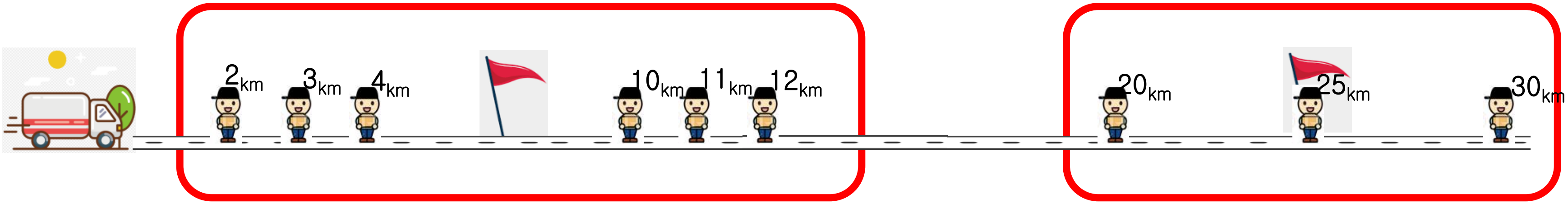
- 중심#1=3, 중심#2=18
- 각 중심에 가까운 군집#1, 군집#2 생성



- 각 군집#1, 군집#2의 새 중심 계산
  - 중심#1= 4.75
  - 중심#2= 19.6

K-means 4회차

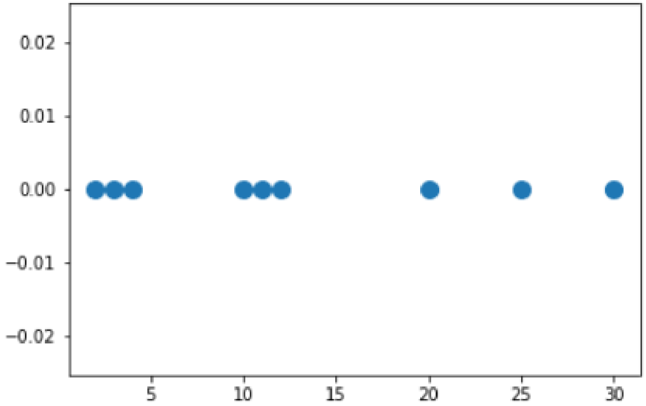
- 중심#1=4.75, 중심#2=19.6
- 각 중심에 가까운 군집#1, 군집#2 생성



- 각 군집#1, 군집#2의 새 중심 계산
  - 중심#1= 7
  - 중심#2= 25
- 중심#1, #2에 대해 더 이상 군집의 변화가 없으므로 군집분석 종료

PYTHON 실습1. 1차원 클러스터링 예제

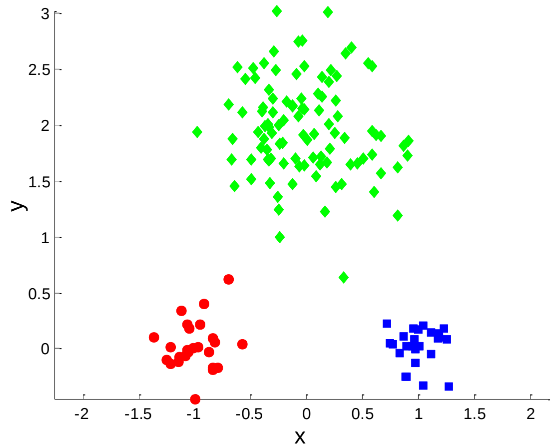
In [76]:
```
from __future__ import print_function
from sklearn.datasets import make_blobs
from sklearn.cluster import KMeans
from sklearn.metrics import silhouette_samples, silhouette_score
from pandas import DataFrame
import matplotlib.pyplot as plt
import matplotlib.cm as cm
import numpy as np
import pandas as pd
import math

X = np.array([[2, 0], [3, 0], [4, 0], [10, 0], [11, 0], [12, 0],
              [20, 0], [25, 0], [30, 0]])
plt.scatter(X[:, 0], X[:, 1], s=100)
plt.show()
```



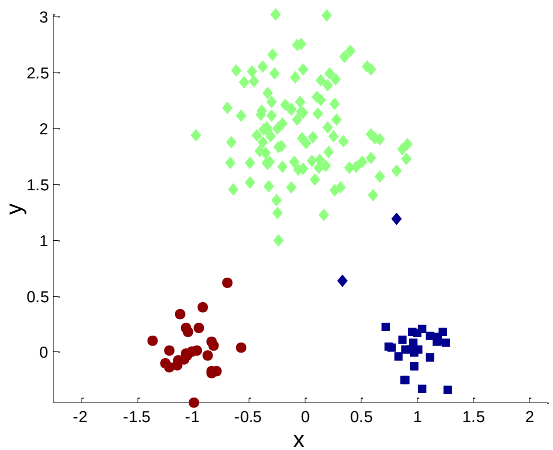
In [70]:
```
# 1회차
model1 = KMeans(n_clusters=2, init=np.array([[3,0],[4,0]]), n_init=1,
                max_iter=1, random_state=1).fit(X)
c0, c1 = model1.cluster_centers_
c0, c1
```

Out[70]:
```
(array([2.5, 0. ]), array([16.,  0.]))
```

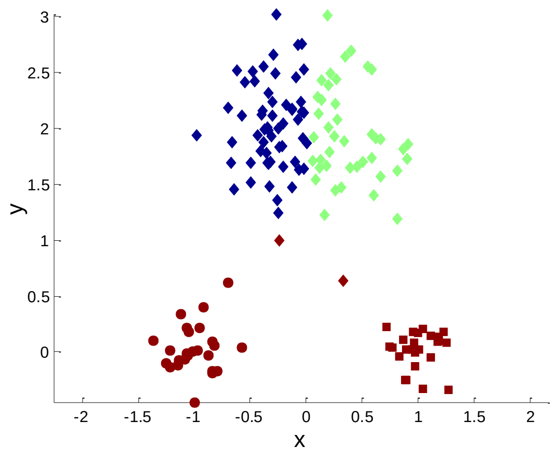
K-means 분석 결과 비교



Original Points

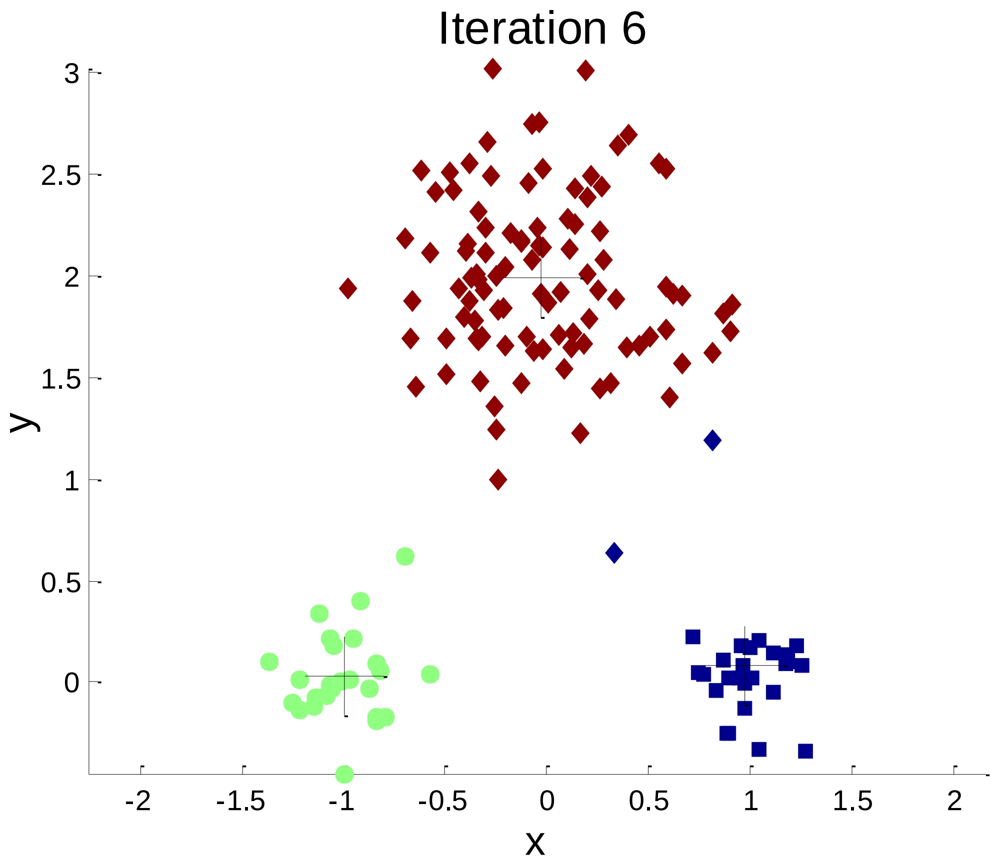


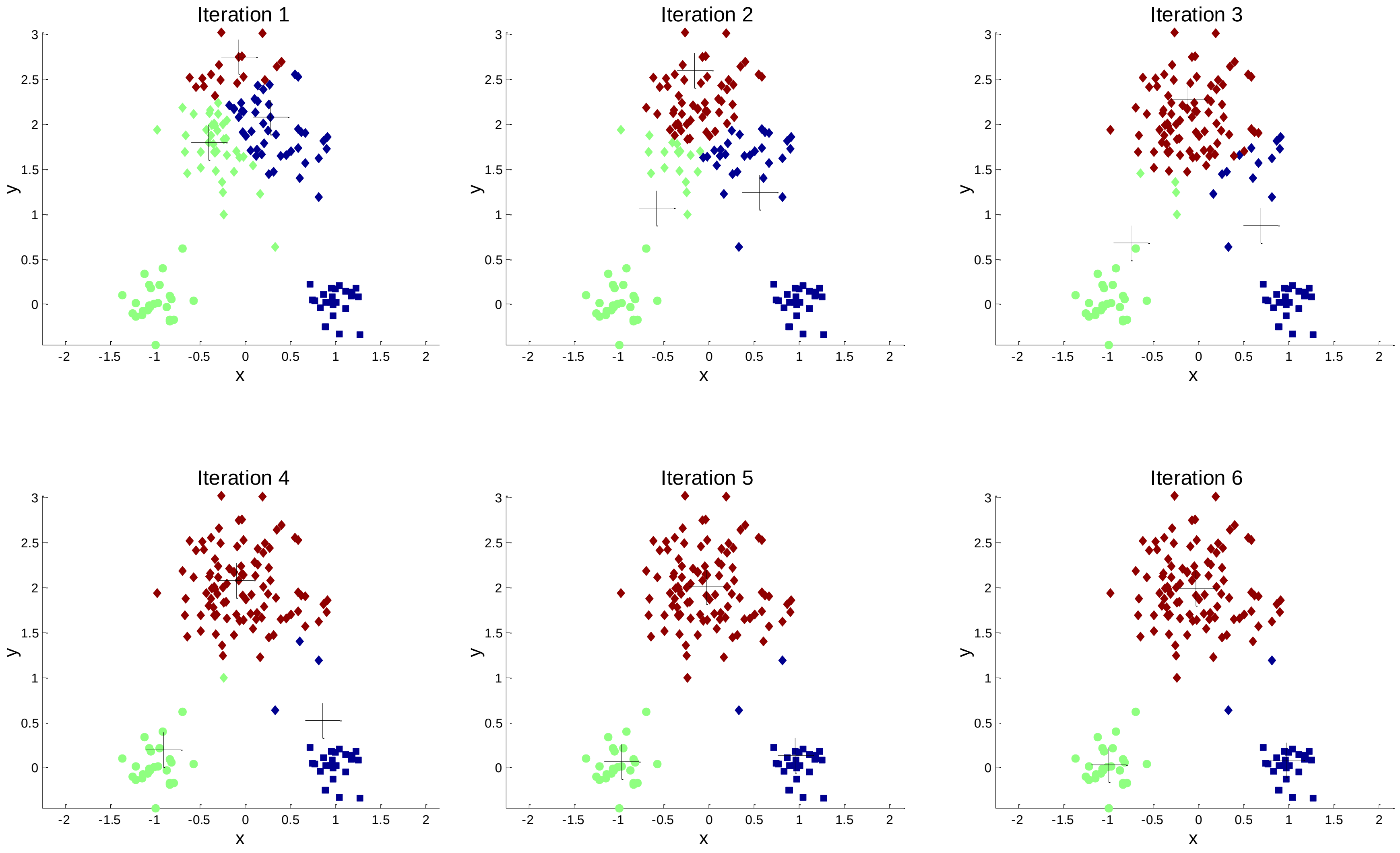
Optimal Clustering



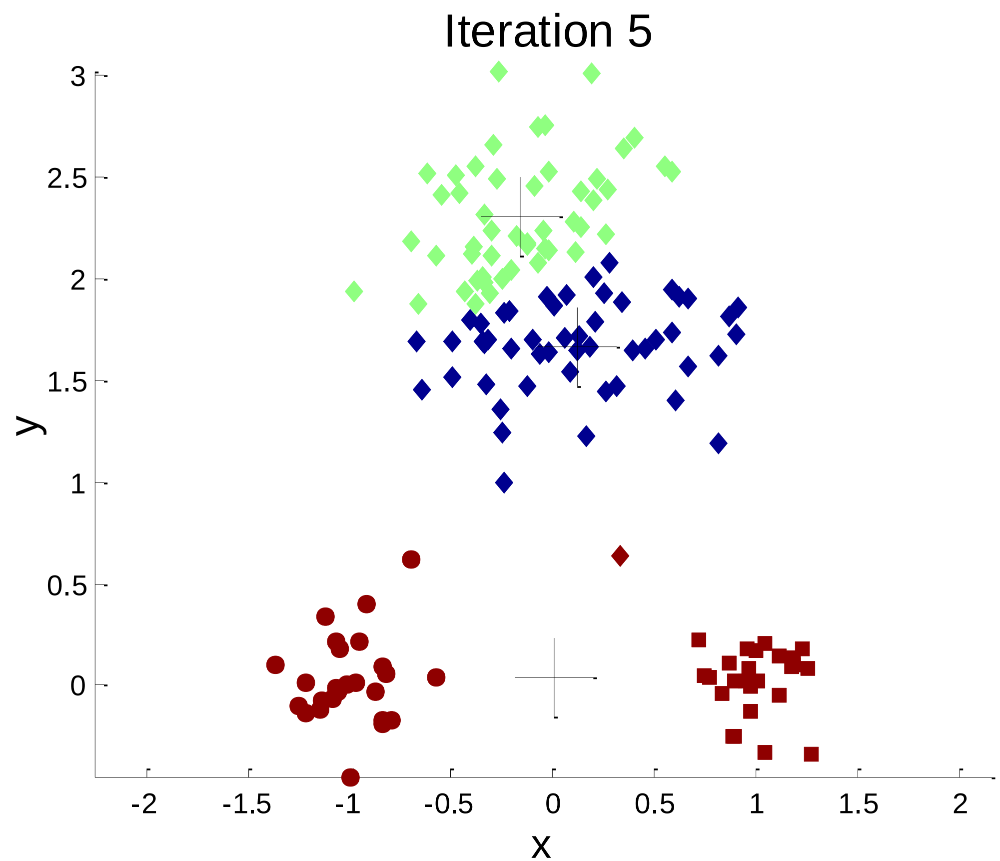
Sub-optimal Clustering

초기 중심값 선택

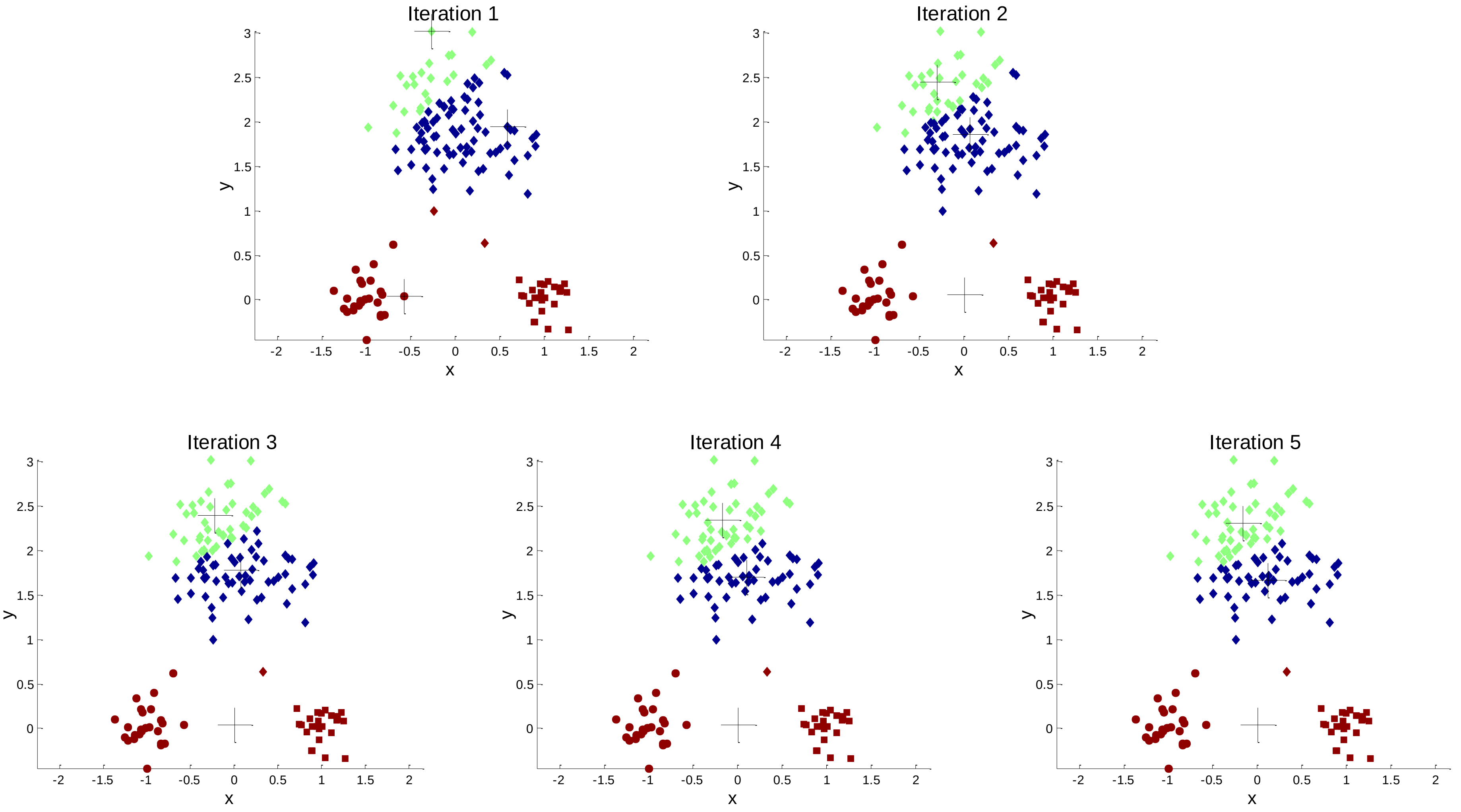



Iteration 1
Iteration 2
Iteration 3
Iteration 4
Iteration 5
Iteration 6
x
y
-2 -1.5 -1 -0.5 0 0.5 1 1.5 2
0 0.5 1 1.5 2 2.5 3

초기 중심값 선택

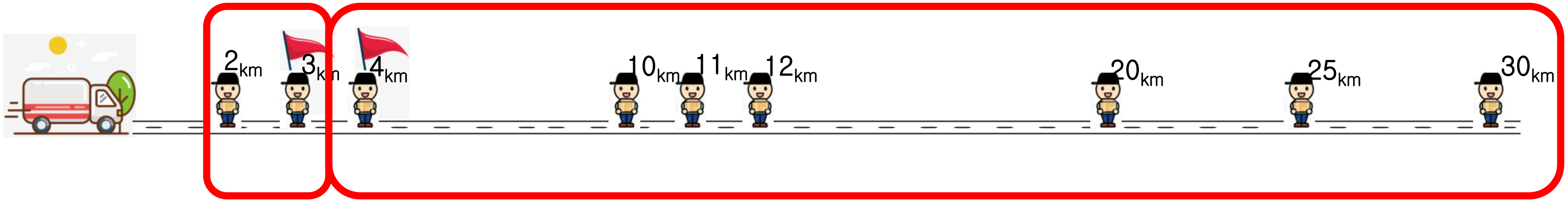


Importance of Choosing Initial Centroids



K-means 군집 평가

- 가장 보편적으로 **Sum of Squared Error (SSE)** 사용
  - 각 객체마다 인접한 클러스터와의 거리로 구함



  - **k_1={2,3}, k_2={4,10,11,12,20,25,30} m_1=3, m_2=4**
  - **$SSE = \sum |k_i - m_i|$ = 1+0+0+6+7+8+16+21+26=85**

K-means 군집 평가

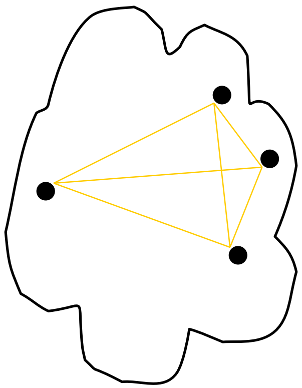
- 가장 보편적으로 **Sum of Squared Error (SSE)** 사용
  - **SSE**는 거리 제곱의 합으로 정의됨

$$SSE = \sum_{i=1}^{K} \sum_{x \in C_i} dist^2(m_i, x)$$

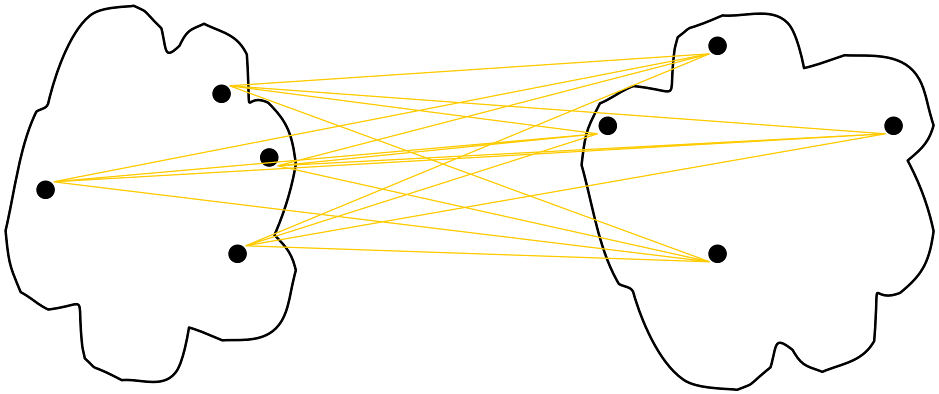
  - ***x***: 군집 ***C*****i** 내의 데이터 객체, ***m*****i** 해당 군집에서의 중심(대표)객체
    - ***m*****i** 이 군집의 평균 중심에 가까운 것을 산출
  - 두 클러스터 결과로부터 최소 에러의 결과를 선택할 수 있음
  - **SSE**를 줄이는 방법으로 클러스터 군집의 수 ***k*** 를 증가시킬 수 있음
    - 작은 ***k***에서의 좋은 분석 결과는 높은 ***k***에서의 나쁜 분석 결과보다 낮은 **SSE**를 가짐

군집 평가 : 응집도(Cohesion)와 분리도(Separation)

- 응집도와 분리도는 그래프 기반으로 해석가능 함
  - 클러스터 응집도는 클래스터 내의 모든 거리 가중치의 합으로 표현
  - 클러스터 분리도는 클러스터 외부 객체 간의 모든 거리 가중치의 합으로 표현됨



클러스터 응집도

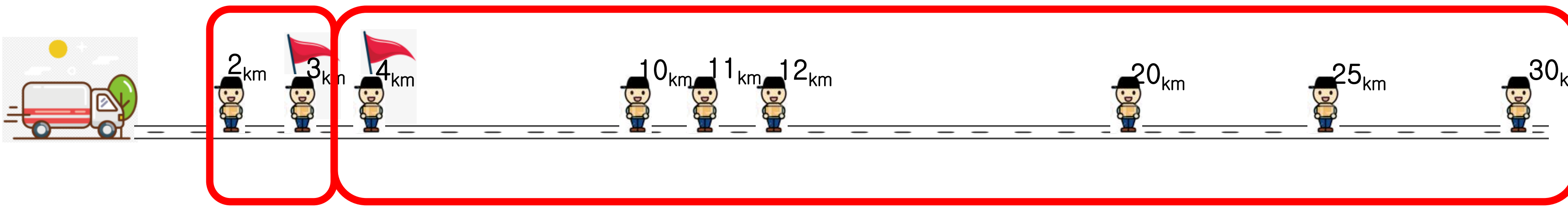


클러스터 분리도

군집 평가 : 응집도(Cohesion)와 분리도(Separation)

- 클러스터 응집도 : 한 클러스터 내의 객체들이 밀집해있는 정도를 측정함 (SSE)
- 클러스터 분리도: 한 클러스터가 다른 클러스터와 잘 분리되어 있는 정도를 측정함

- 평가 #1



- $k_1=\{2,3\}$, $k_2=\{4,10,11,12,20,25,30\}$ $m_1=3$, $m_2=4$, $m=58.5$
- 응집도(*SSE*)= $\sum |k_i - m_i| = 1+0+0+6+7+8+16+21+26=$**85**
- **분리도 = $2*(13-3)^2+7*(13-4)^2=200+81*7=200+567=$767**

군집 평가 : 응집도(Cohesion)와 분리도(Separation)

- 클러스터 응집도 : 한 클러스터 내의 객체들이 밀집해있는 정도를 측정함 (SSE)
  - 응집도는 클러스터 내의 "sum of squares (SSE)" 값으로 측정

$$WSS = \sum_{i} \sum_{x \in C_i} (x - m_i)^2$$

- 클러스터 분리도: 한 클러스터가 다른 클러스터와 잘 분리되어 있는 정도를 측정함 (Squared Error 값 사용)
  - 분리도는 클러스터 외부 객체와의 "sum of squares"으로 측정함

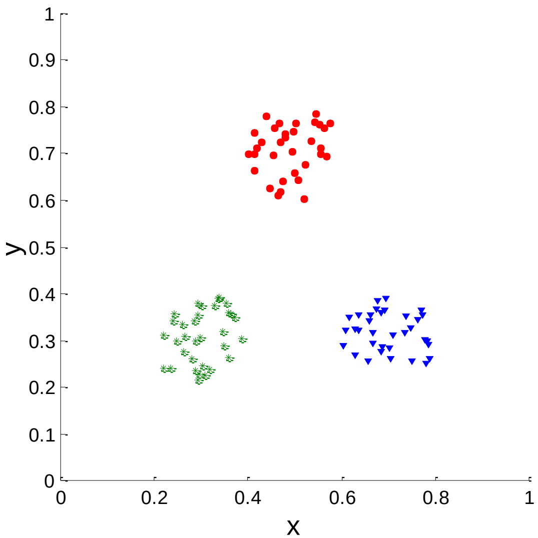
$$BSS = \sum_{i} |C_i| (m - m_i)^2$$

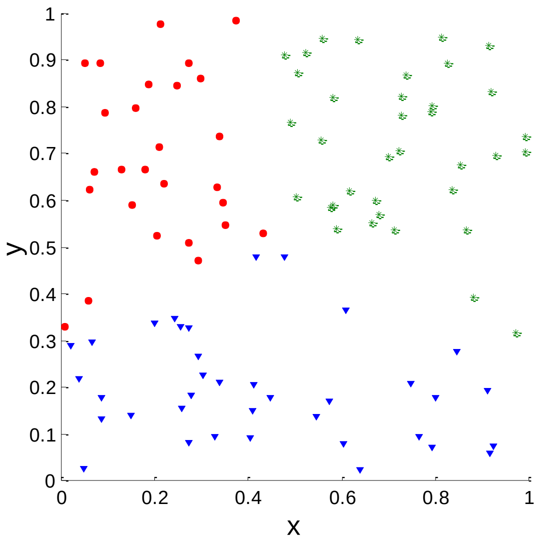
  - Where $|C_i|$ is the size of cluster i

군집 평가 - 유사도 매트릭스

- 유사도 매트릭스
  - x, y 축의 한 값이 각각의 데이터 객체로 표현됨
  - x, y 쌍의 객체가 동일한 클러스터에 위치한다면 **'1'**의 값
  - x, y 쌍의 객체가 서로 다른 클러스터에 위치한다면 **'0'**의 값

- 일부 밀도기반의 군집이나 연속성 기반의 군집에는 적절하지 않을 수 있음





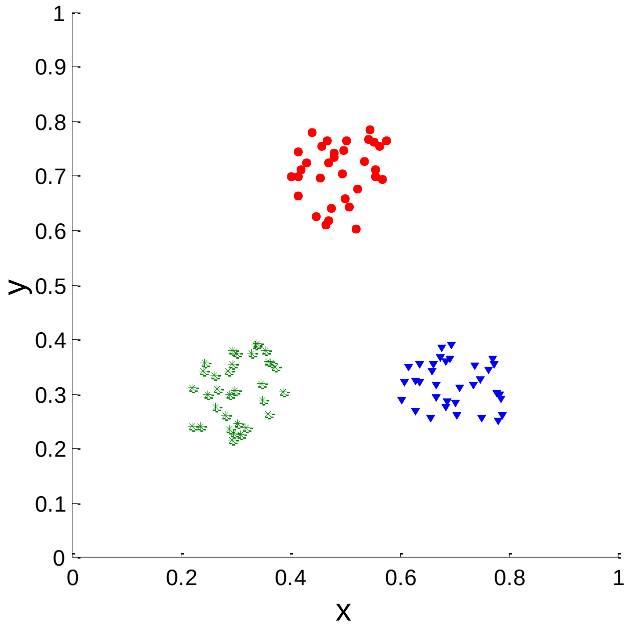
- 군집 평가 - 유사도 매트릭스
  - #1

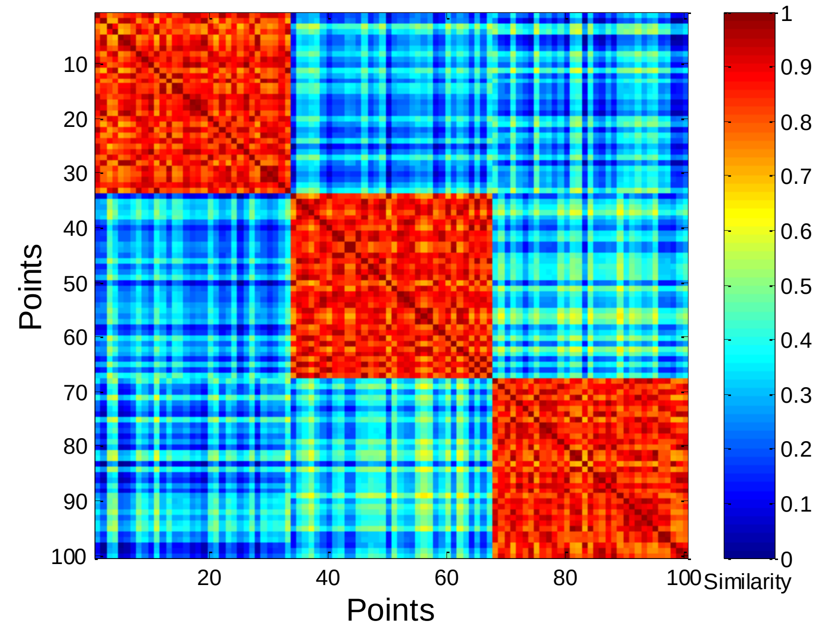
#1 유사도 매트릭스

	2km	3km	4km	10km	11km	12km	20km	25km	30km
2km	1	1	1	0	0	0	0	0	0
3km	1	1	1	0	0	0	0	0	0
4km	1	1	1	0	0	0	0	0	0
10km	0	0	0	1	1	1	0	0	0
11km	0	0	0	1	1	1	0	0	0
12km	0	0	0	1	1	1	0	0	0
20km	0	0	0	0	0	0	1	1	1
25km	0	0	0	0	0	0	1	1	1
30km	0	0	0	0	0	0	1	1	1

유사성 매트릭스 기반의 군집 평가

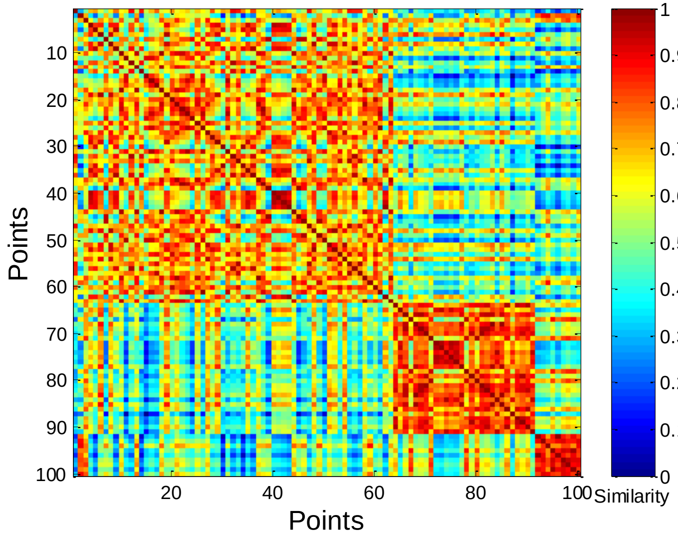
- 군집분석 결과 포함된 군집의 번호 순서대로 객체들을 정렬함

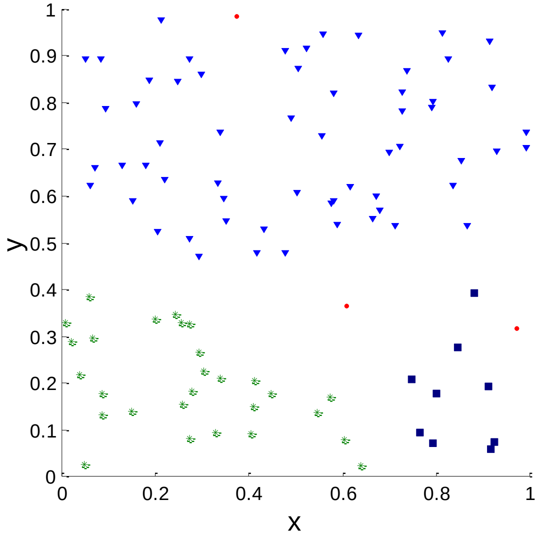




유사성 매트릭스 기반의 군집 평가

- 랜덤 객체를 대상으로 유사성 매트릭스는 좋은 값이 나오지 않음

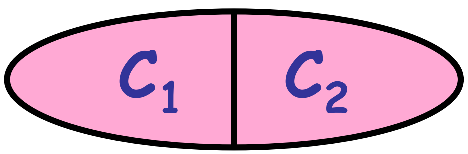




엔트로피 기반의 군집 평가

- 각 군집의 엔트로피를 측정하여 군집 평가에 사용함

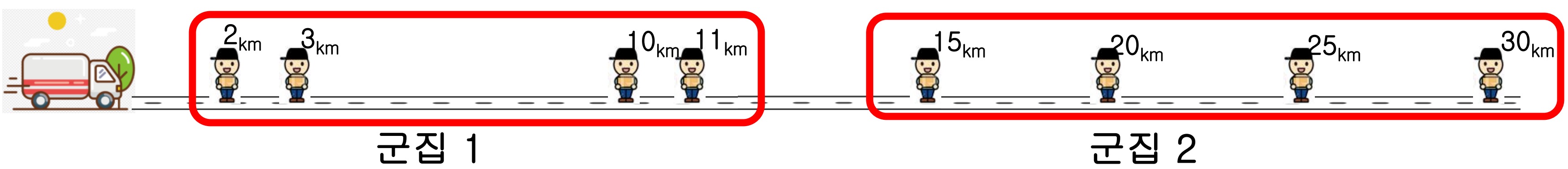
  (엔트로피 값이 낮을수록 좋은 군집으로 평가함)



$$P_1 = \frac{1}{2}$$

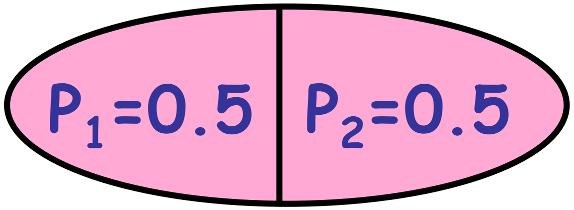
$$P_2 = \frac{1}{2}$$

엔트로피 예제 #1)



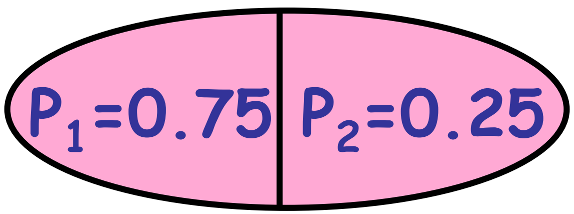
Entropy = $-\sum 0.5 \log_2 0.5 = 1.0$

$$\text{Entropy} = -\sum_{i=1}^{2} P_i \log_2 P_i$$

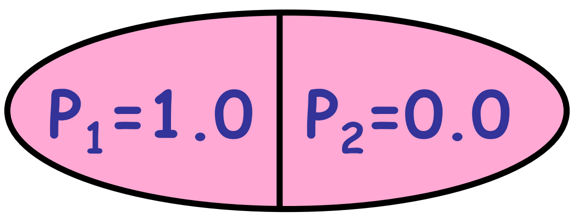


Entropy = 1.0 ;

엔트로피 값이 높을수록 유동적이고
변화가 많음



Entropy = 0.85



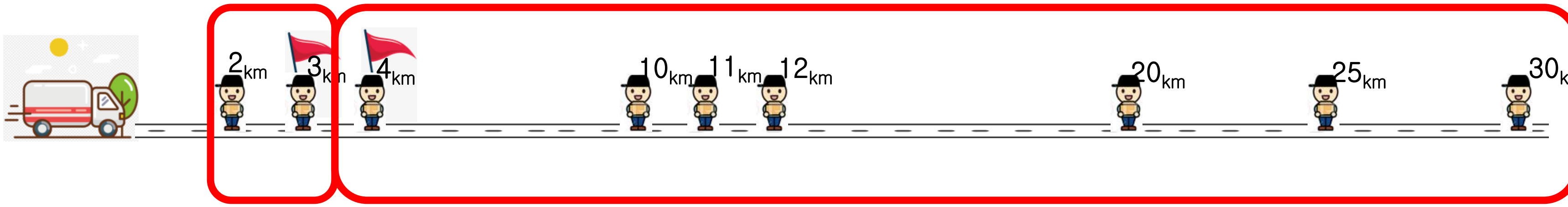
Entropy = 0.0 ;

엔트로피 값이 낮을수록 안정적이고
변화가 없음

PYTHON 실습2. 군집평가

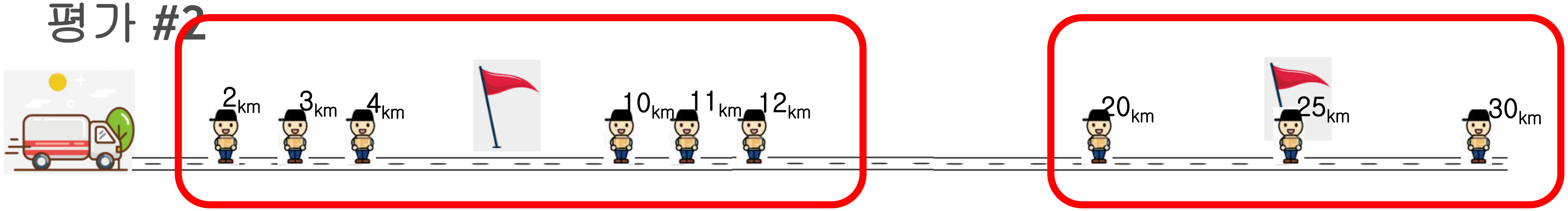
Cohesion and Separation 군집 비교

- 평가 #1



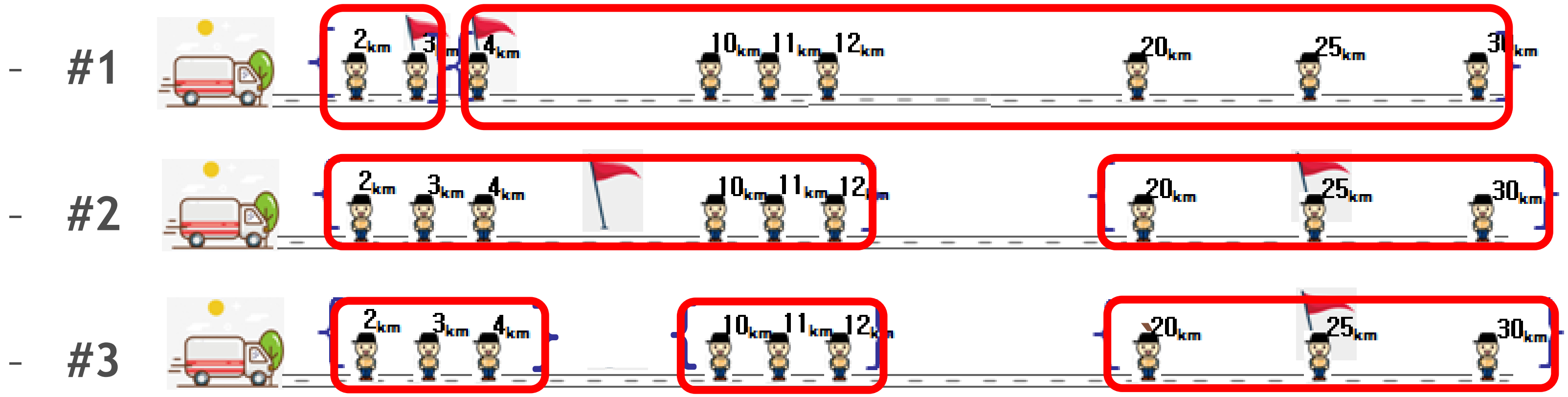
- $k_1=\{2,3\}$, $k_2=\{4,10,11,12,20,25,30\}$ $m_1=3$, $m_2=4$, $m=58.5$
- *Cohesion(SSE)*$= \sum |k_i - m_i| = 1+0+0+6+7+8+16+21+26=$**85**
- **Separation** $= 2*(13-3)^2+7*(13-4)^2=200+81*7=200+567=$**767**

- 평가 #2



- $k_1=\{2,3,4,10,11,12\}$, $k_2=\{20,25,30\}$ $m_1=7$, $m_2=25$, $m=13$
- *Cohesion(SSE)*$= \sum |k_i - m_i| = 5+4+3+3+4+5+5+0+5=$**34**
- **Separation** $= 6*(13-7)^2+3*(25-13)^2=6*36+3*144=216+432=$**648**

군집 비교 - 유사도 매트릭스



#1 유사도 매트릭스

1	1	0	0	0	0	0	0	0
1	1	0	0	0	0	0	0	0
0	0	1	1	1	1	1	1	1
0	0	1	1	1	1	1	1	1
0	0	1	1	1	1	1	1	1
0	0	1	1	1	1	1	1	1
0	0	1	1	1	1	1	1	1
0	0	1	1	1	1	1	1	1
0	0	1	1	1	1	1	1	1

#2 유사도 매트릭스

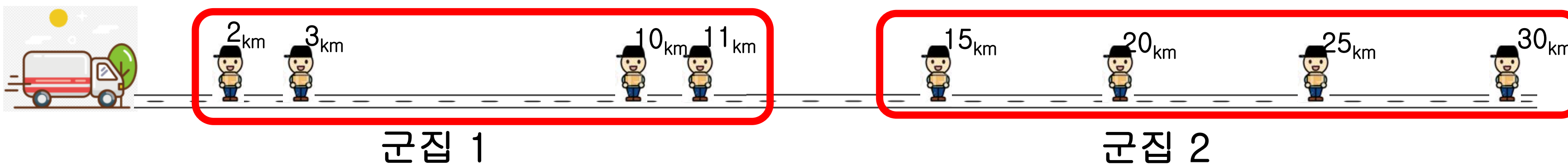
1	1	1	1	1	1	0	0	0
1	1	1	1	1	1	0	0	0
1	1	1	1	1	1	0	0	0
1	1	1	1	1	1	0	0	0
1	1	1	1	1	1	0	0	0
1	1	1	1	1	1	0	0	0
0	0	0	0	0	0	1	1	1
0	0	0	0	0	0	1	1	1
0	0	0	0	0	0	1	1	1

#3 유사도 매트릭스

1	1	1	0	0	0	0	0	0
1	1	1	0	0	0	0	0	0
1	1	1	0	0	0	0	0	0
0	0	0	1	1	1	0	0	0
0	0	0	1	1	1	0	0	0
0	0	0	1	1	1	0	0	0
0	0	0	0	0	0	1	1	1
0	0	0	0	0	0	1	1	1
0	0	0	0	0	0	1	1	1

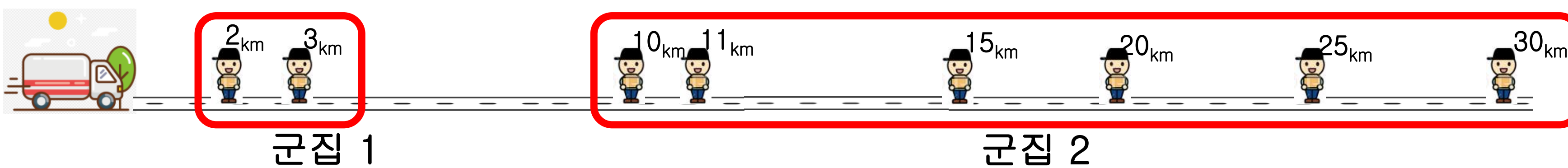
군집 비교 - 엔트로피

엔트로피 예제 #1)



$$\text{Entropy} = -\sum 0.5 \log_2 0.5 = 1.0$$

엔트로피 예제 #2)



$$\text{Entropy} = -\ 0.25 \log_2 0.25 - 0.75 \log_2 0.75 = 0.85$$

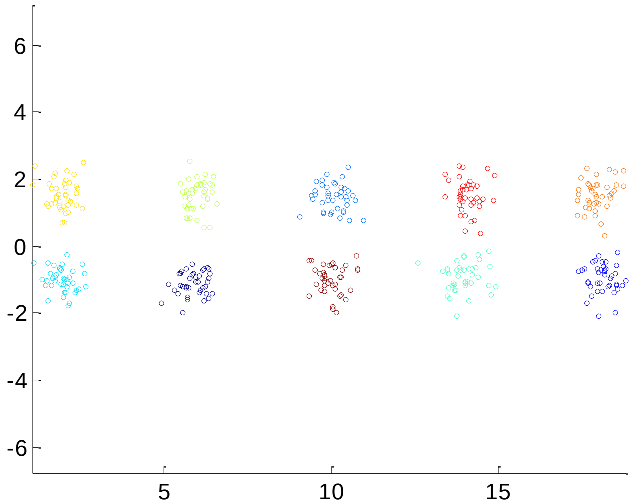
PYTHON 실습3. 군집평가 비교

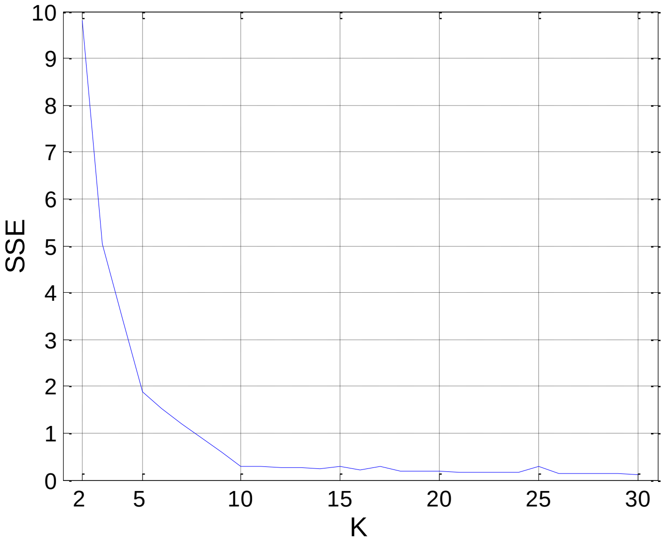
Internal Measures: SSE

- 클러스터링 방법이나 외부조건과 관계없이 분석 결과 자체 정보만으로 클러스터의 우수성을 평가할 수 있음

- 두 개 이상의 클러스터링 분석 결과를 비교하거나 다른 클러스터링 방법의 결과를 서로 비교하는데 좋은 기준이 됨

- 좋은 군집의 수 **k**를 찾는 방법으로 사용 가능함





PYTHON 실습4. SSE 기반의 k 탐색 실습

PYTHON 실습5. K-means 2차원 데이터

K-Means 기법의 장단점

- 장점
  - *상대적 효율성*: *O*(*tkn*), where *n* is # objects, *k* is # clusters, and *t* is # iterations. Normally, *k*, *t* << *n*.
  - 주로 *local optimum* 클러스터 군집을 찾음. The *global optimum* 군집의 경우, 다수의 초기 seed 선택, 진화적 seed 선택 등의 개선된 기법들이 필요함

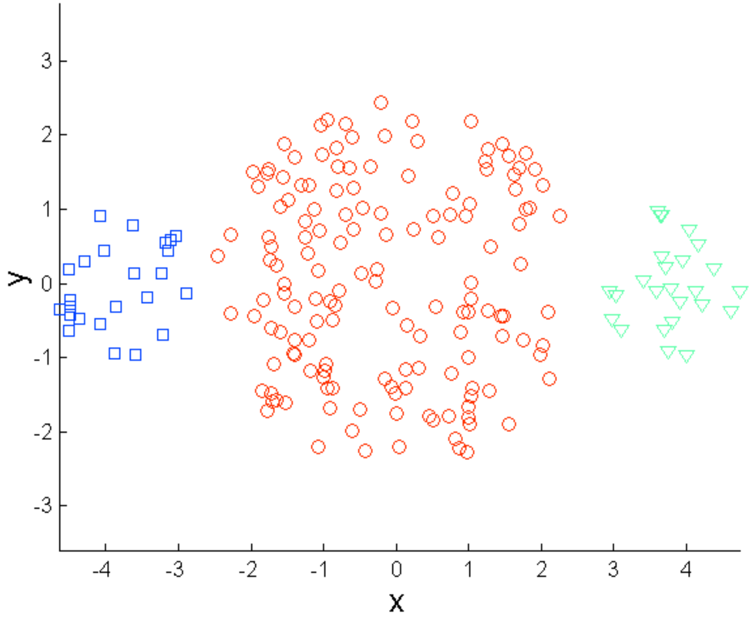
- 단점
  - 중심값을 구할 수 있는 경우에 적용가능함. 아이템 형식의 데이터를 대상으로 적용하기 어려움
  - *k*, the *number* of clusters가 적절히 정의되어야 함
  - 잡음 데이터나 이상치 데이터에 영향을 받음
  - 임의 모양의 클러스터 군집을 찾는데 적합하지 않음

K-means의 한계점

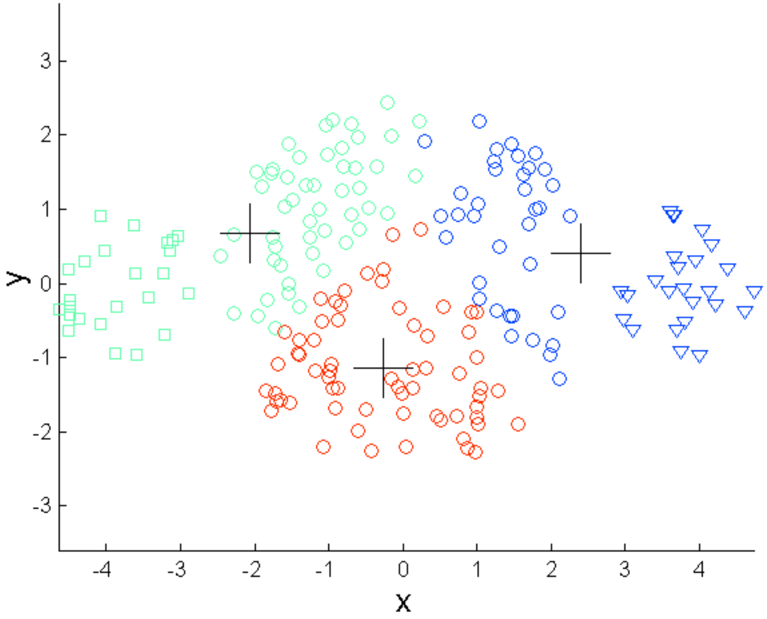
- **K-means** 는 군집들이 다양성을 가질때 효율적이지 못함
  - 군집의 크기
  - 군집의 밀도
  - 비원형 형태의 군집

- **K-means** 이상치를 포함한 데이터 처리에 부적합함

Limitations of K-means: Differing Sizes

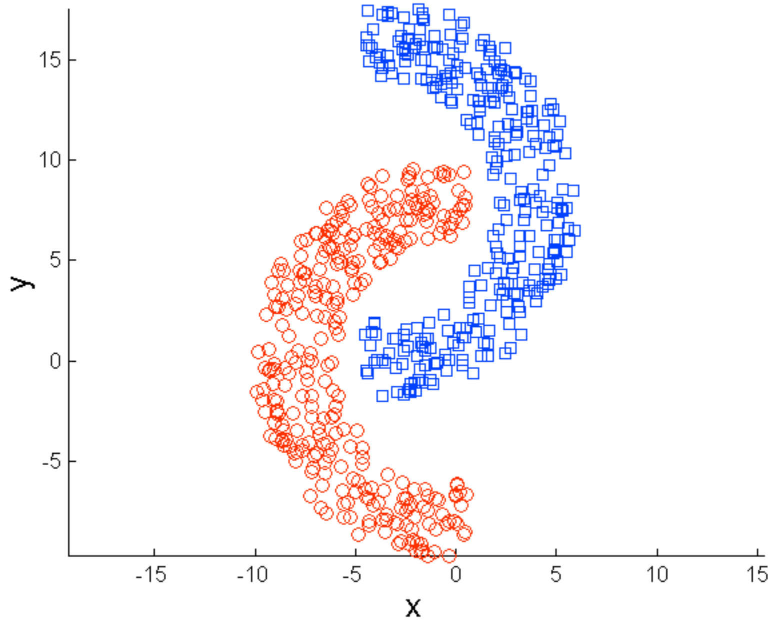


Original Points

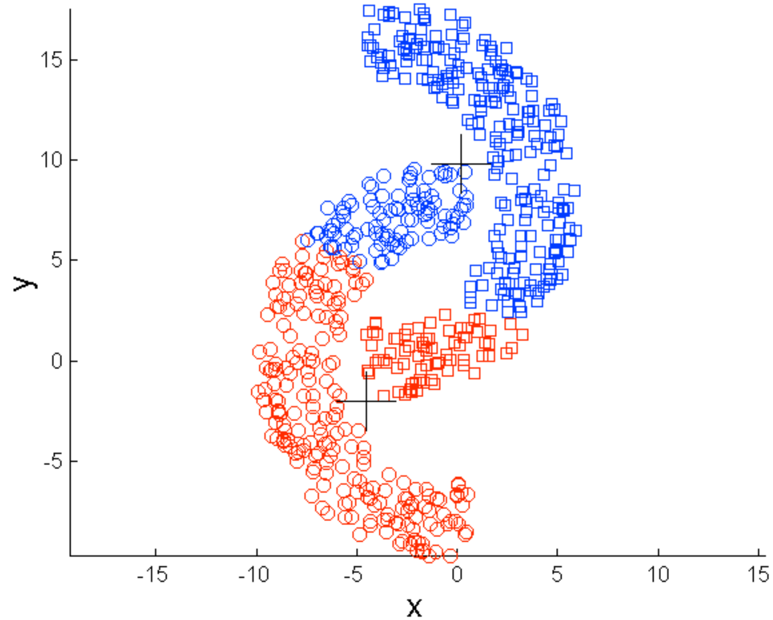


K-means (3 Clusters)

Limitations of K-means: Non-globular Shapes

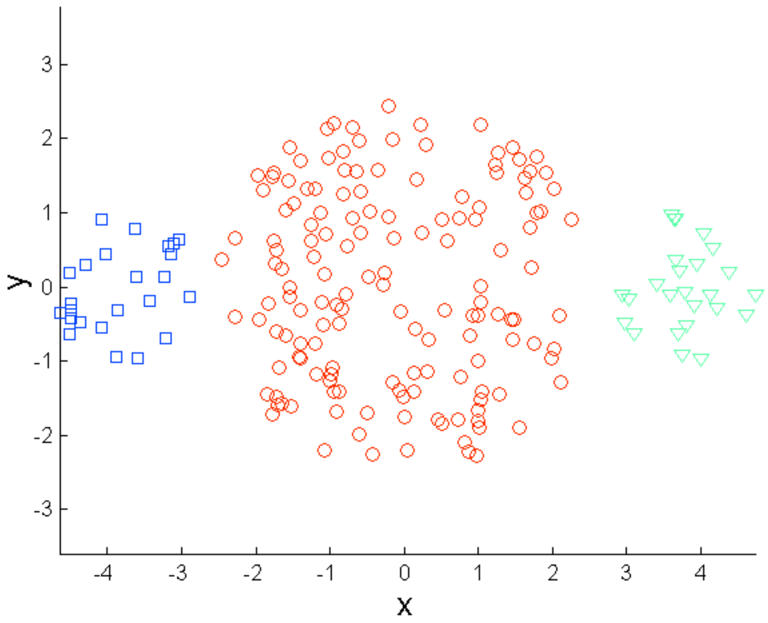


Original Points

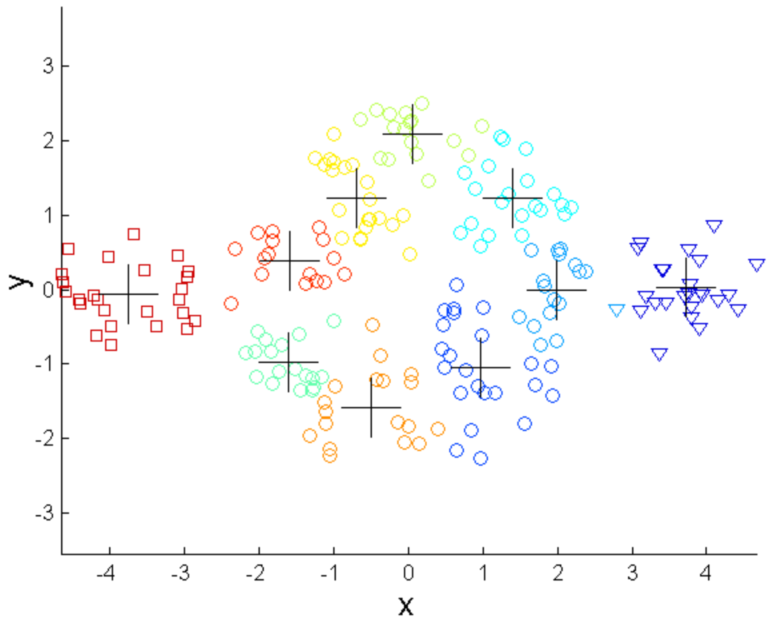


K-means (2 Clusters)

Overcoming K-means Limitations



Original Points



K-means Clusters

많은 수의 작은 군집들을 1차적으로 탐색한 후,
군집들로부터 2차적으로 큰 군집을 찾는 과정을 수행함