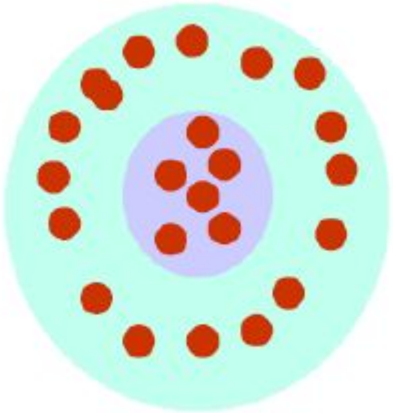
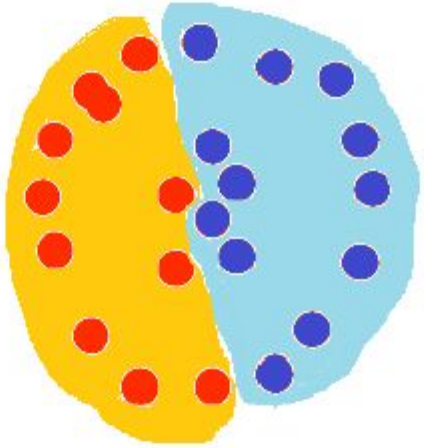
Introduction to Clustering

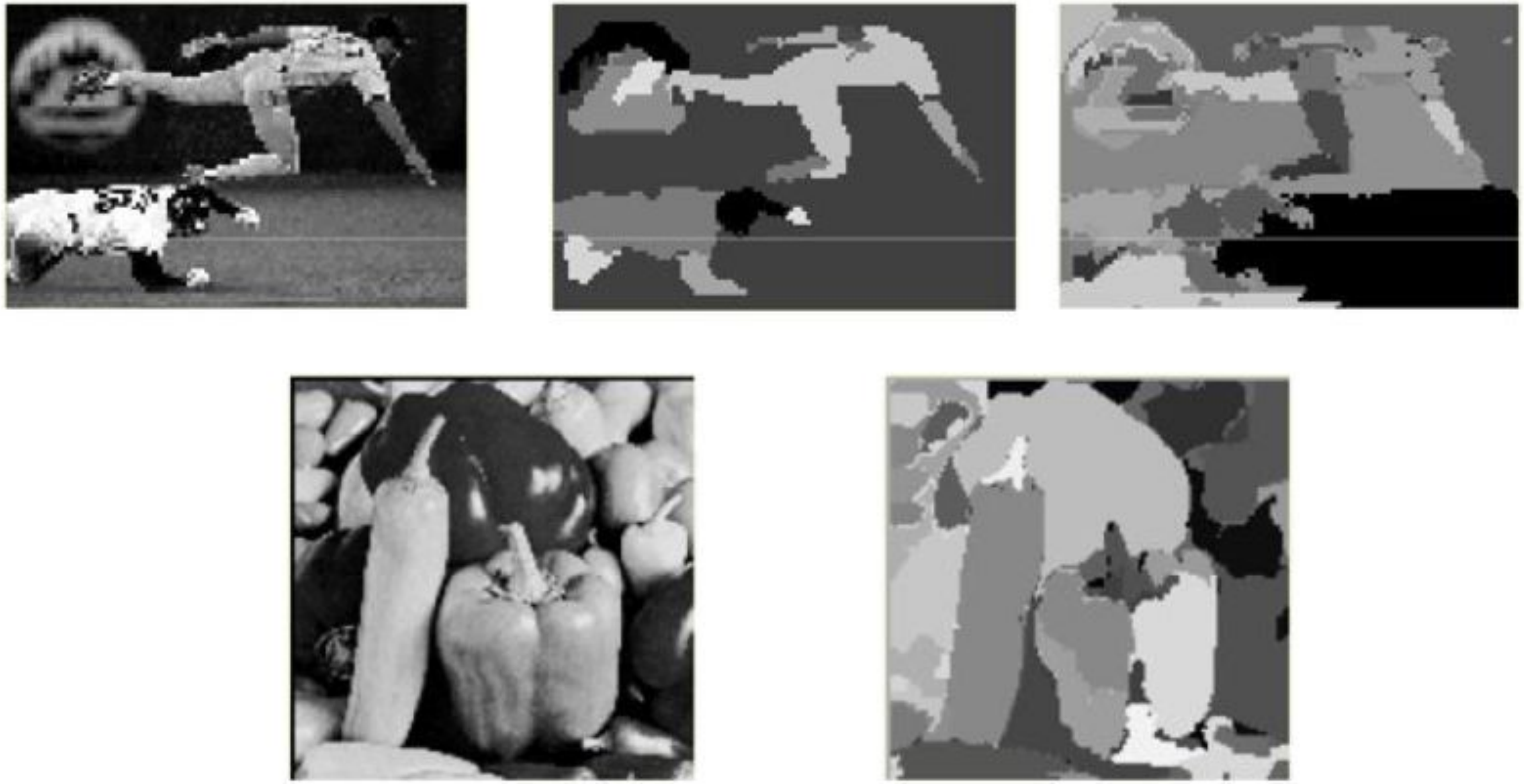
Clustering

- Classify objects (cases) into homogeneous groups called clusters.
- Objects in each cluster tend to be similar and dissimilar to objects in the other clusters.





Computer vision application: Image segmentation

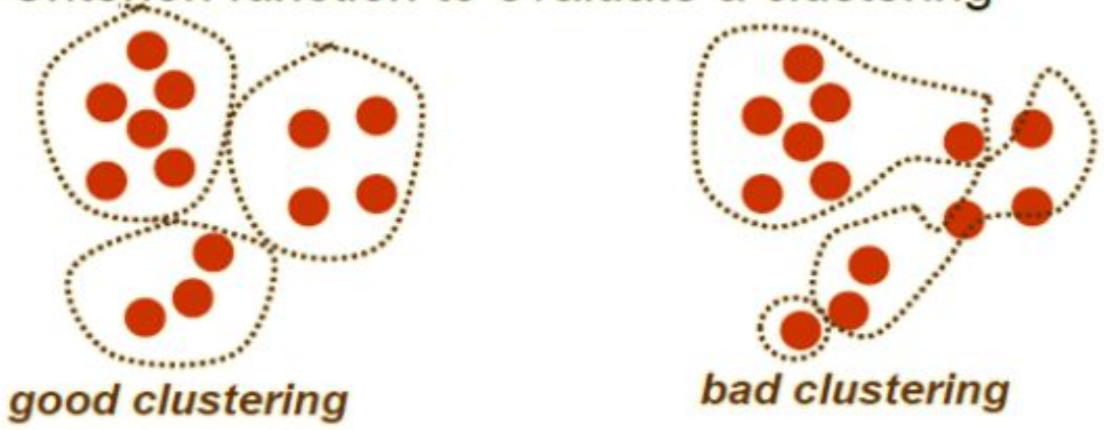


From: Image Segmentation by Nested Cuts, O. Veksler, CVPR2000

What do we need for clustering?

1. Proximity measure, *either*
   - similarity measure $s(x_i,x_k)$: large if x_i,x_k are similar
   - dissimilarity(or distance) measure $d(x_i,x_k)$: small if x_i,x_k are similar

2. Criterion function to evaluate a clustering



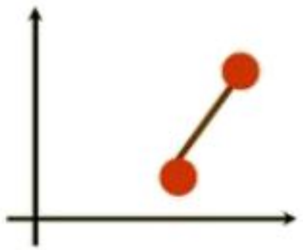
3. Algorithm to compute clustering
   - For example, by optimizing the criterion function

Distance (dissimilarity) measures

- Euclidean distance

$$d(x_i, x_j) = \sqrt{\sum_{k=1}^{d} \left(x_i^{(k)} - x_j^{(k)}\right)^2}$$

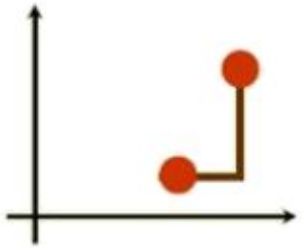
  - translation invariant



- Manhattan (city block) distance

$$d(x_i, x_j) = \sum_{k=1}^{d} \left|x_i^{(k)} - x_j^{(k)}\right|$$

  - approximation to Euclidean distance, cheaper to compute



- They are special cases of **Minkowski distance**:

$$d_p(\mathbf{x}_i, \mathbf{x}_j) = \left(\sum_{k=1}^{m} \left| x_{ik} - x_{jk} \right|^p \right)^{\frac{1}{p}}$$

(p is a positive integer)

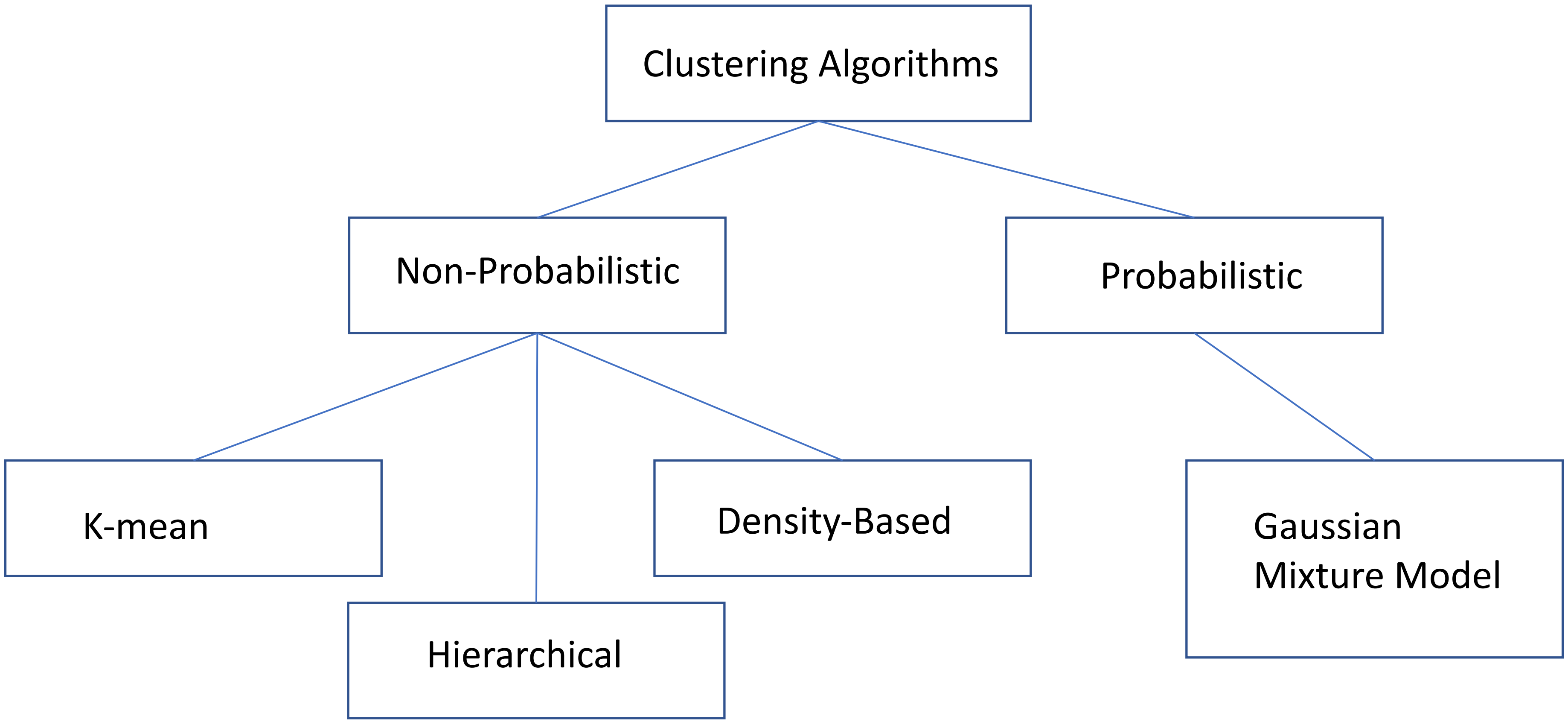
How about measures for mixed types?

Clustering Algorithms

- Clustering Problem is am NP-Complete Problem

- Algorithms are Heuristic

Clustering Algorithms (Cont.)



K-Means Algorithm

- K-means (MacQueen, 1967) is a partitional clustering algorithm
- Let the set of data points D be $\{\mathbf{x}_1, \mathbf{x}_2, \ldots, \mathbf{x}_n\}$,

  where $\mathbf{x}_i = (x_{i1}, x_{i2}, \ldots, x_{ir})$ is a vector in $X \subseteq R^r$, and r is the number of dimensions.
- The k-means algorithm partitions the given data into k clusters:
  - Each cluster has a cluster **center**, called **centroid**.
  - k is specified by the user

K-Means Algorithm (Cont.)

- Given *k*, the *k-means* algorithm works as follows:
    1. Choose *k* (random) data points (seeds) to be the initial centroids, cluster centers
    2. Assign each data point to the closest centroid
    3. Re-compute the centroids using the current cluster memberships
    4. If a convergence criterion is not met, repeat steps 2 and 3

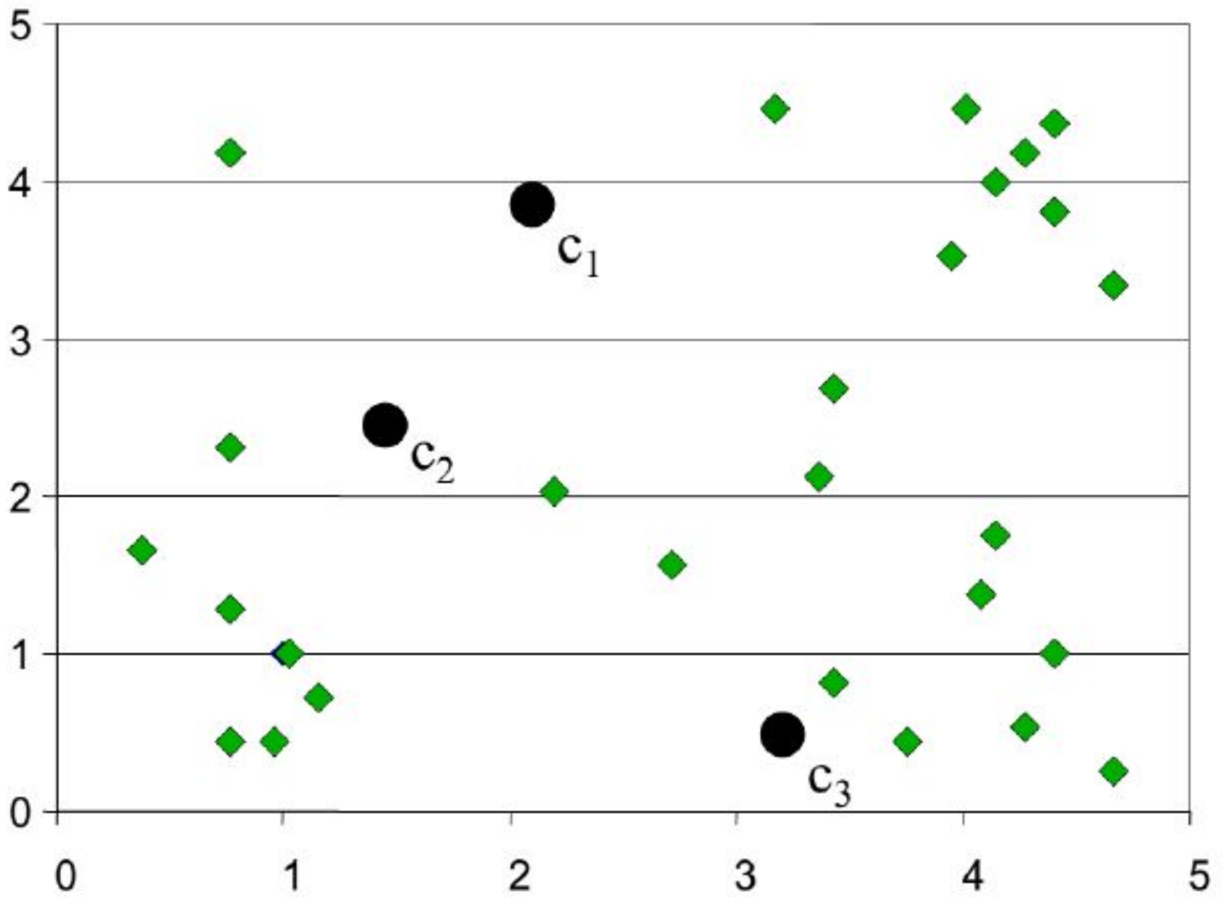
K-Means Convergence Criterion

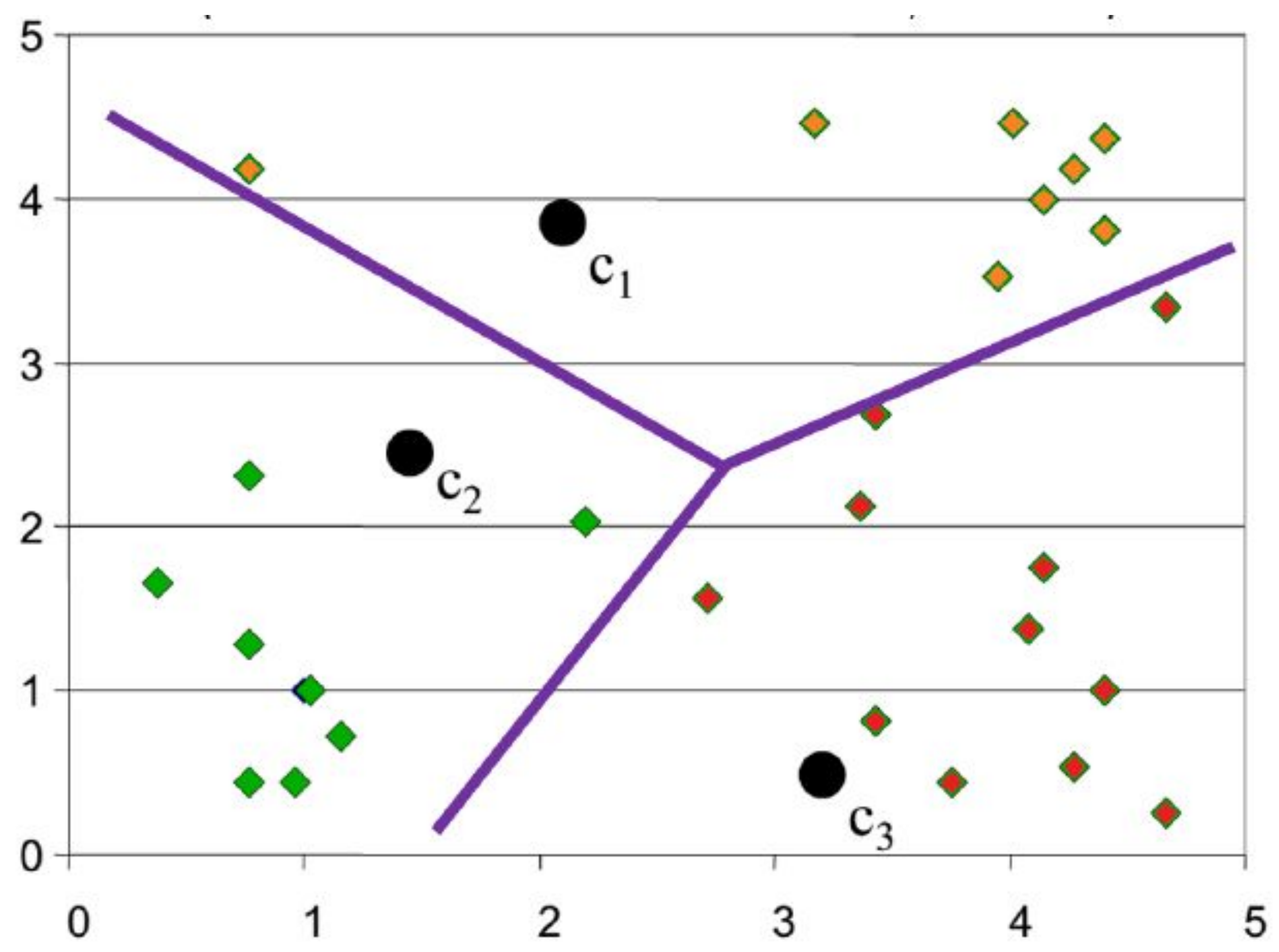
- no (or minimum) re-assignments of data points to different clusters, *or*
- no (or minimum) change of centroids, *or*
- minimum decrease in the **sum of squared error** (SSE),

$$SSE = \sum_{j=1}^{k} \sum_{\mathbf{x} \in C_j} d(\mathbf{x}, \mathbf{m}_j)^2$$

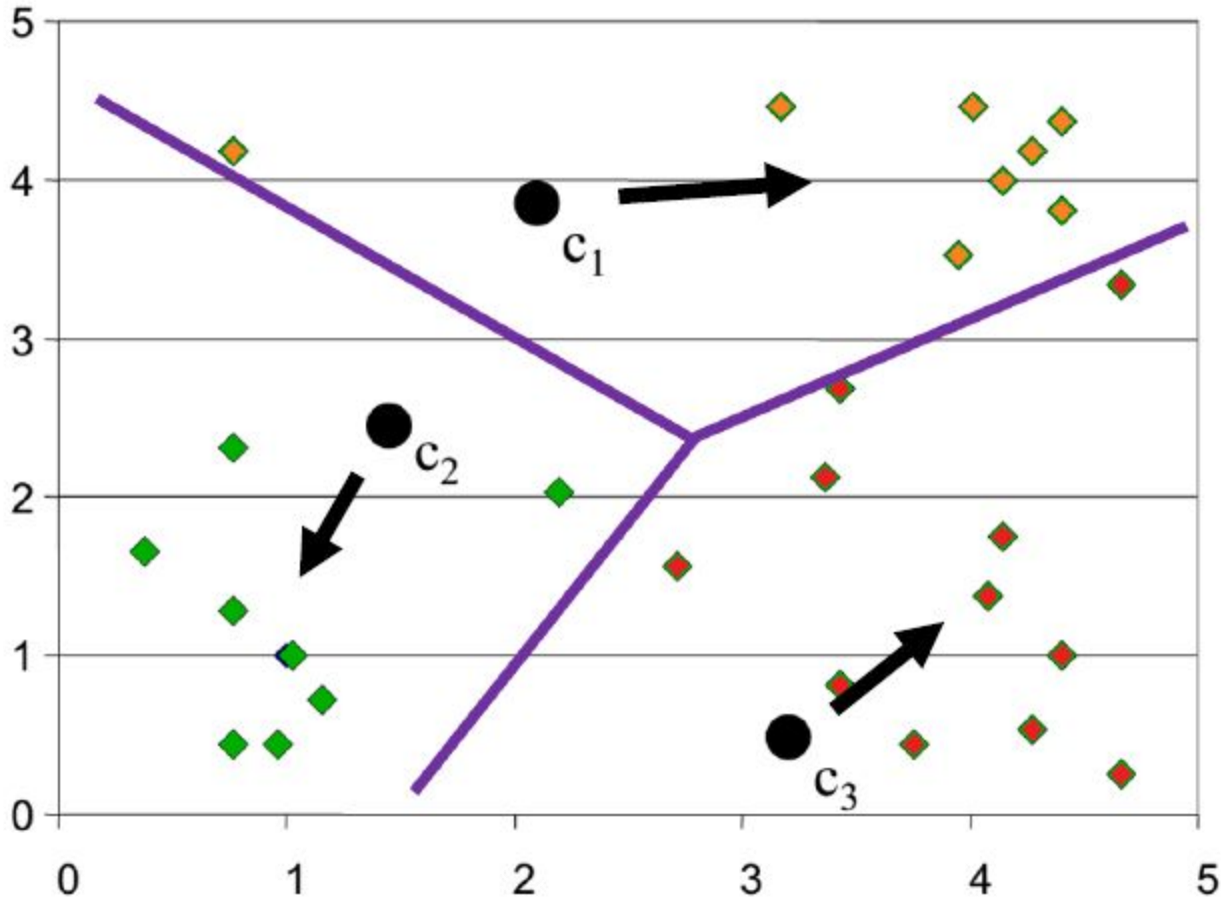
  - C_j is the *j*th cluster,
  - $\mathbf{m}_j$ is the centroid of cluster C_j (the mean vector of all the data points in C_j),
  - $d(\mathbf{x}, \mathbf{m}_j)$ is the (Eucledian) distance between data point **x** and centroid $\mathbf{m}_j$.

K-Means Example



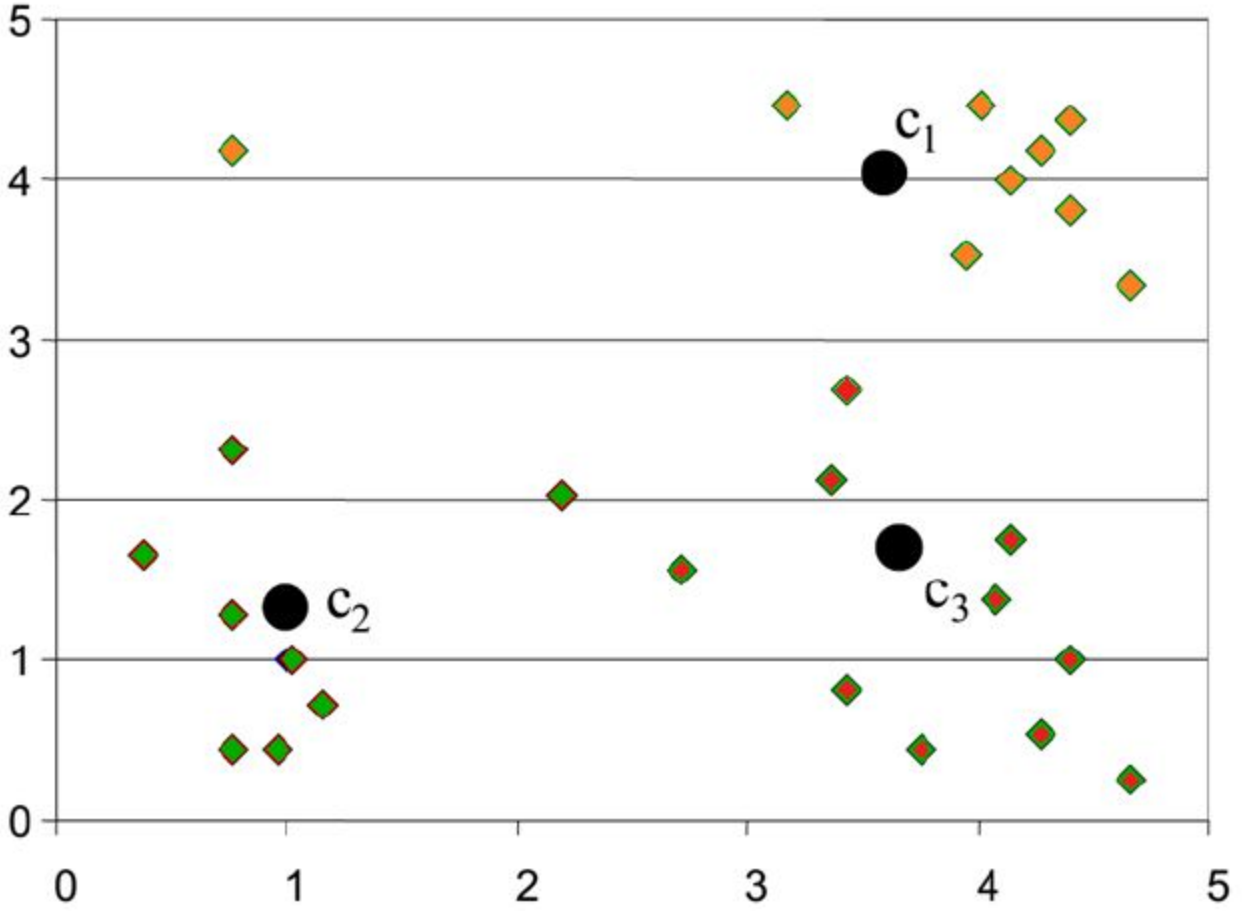


5
4
3
2
1
0
0
1
2
3
4
5
c_1
c_2
c_3

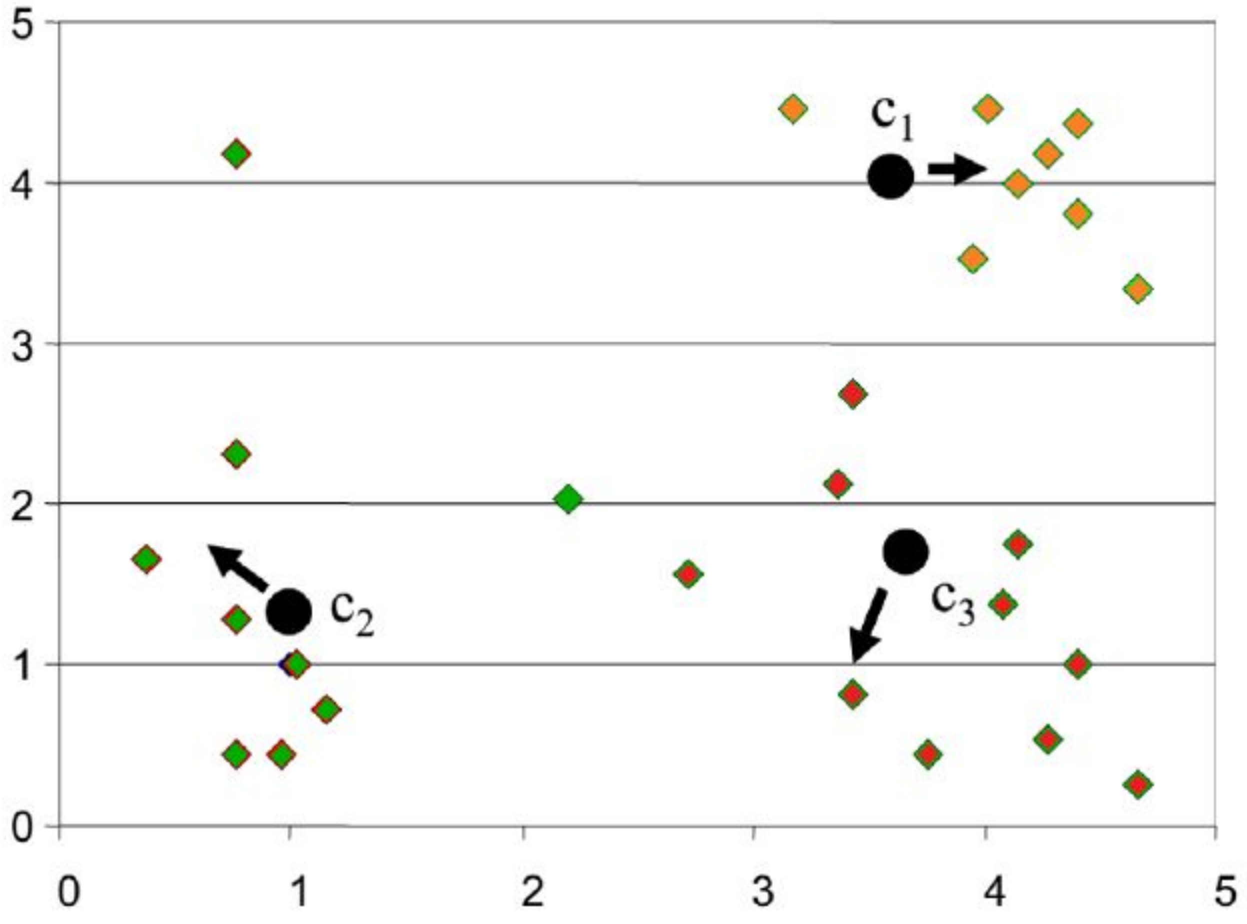


c1
c2
c3
0
1
2
3
4
5
0
1
2
3
4
5

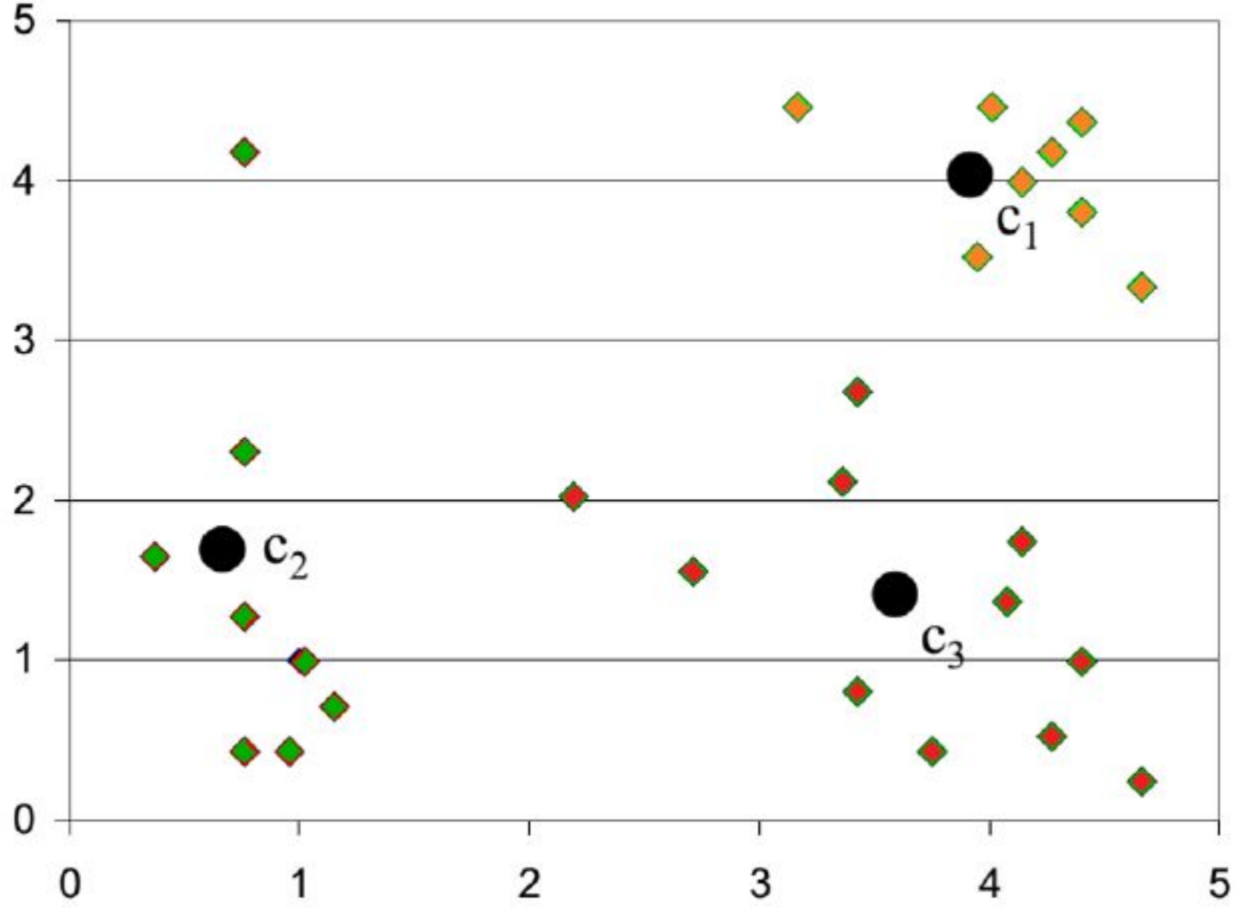
Result of first iteration



Second iteration



Result of second iteration



Why K-Means Algorithm?

- Strengths:
  - Simple: easy to understand and to implement
  - Efficient: Time complexity: $O(tkn)$,

    where n is the number of data points,

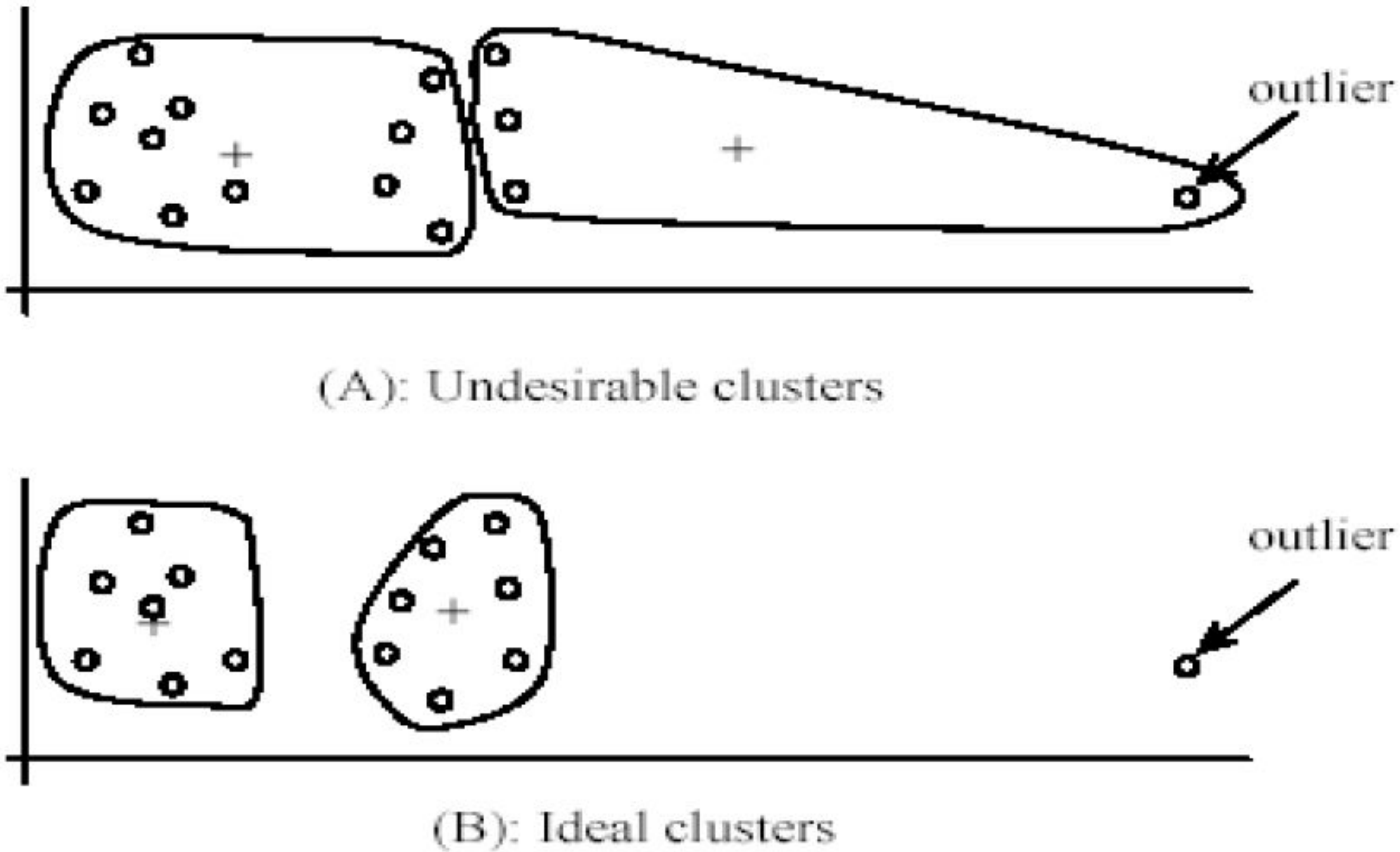
    k is the number of clusters, and

    t is the number of iterations.
  - Since both k and t are small. k-means is considered a linear algorithm.
- K-means is the most popular clustering algorithm.
- Note that: it terminates at a local optimum if SSE is used. The global optimum is hard to find due to complexity.

Weaknesses of K-Means Algorithm

- The algorithm is only applicable if the mean is defined.
  - For categorical data, *k*-mode - the centroid is represented by most frequent values.
- The user needs to specify *k*.
- The algorithm is sensitive to **outliers**
  - Outliers are data points that are very far away from other data points.
  - Outliers could be errors in the data recording or some special data points with very different values.

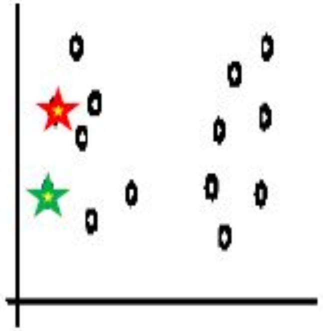
Outliers



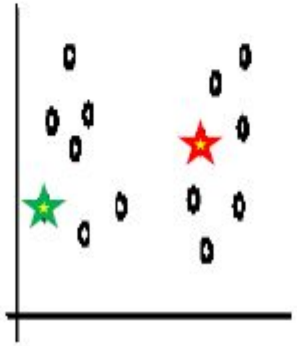
Outliers (Cont.)

- Remove some data points that are much further away from the centroids than other data points
  - To be safe, we may want to monitor these possible outliers over a few iterations and then decide to remove them.
- Perform random sampling: by choosing a small subset of the data points, the chance of selecting an outlier is much smaller
  - Assign the rest of the data points to the clusters by distance or similarity comparison, or classification

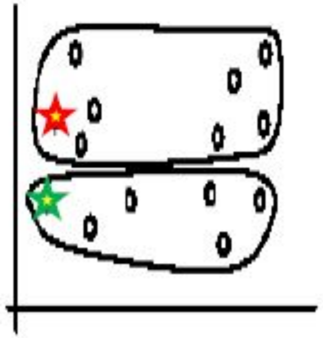
Sensitivity to initial seeds



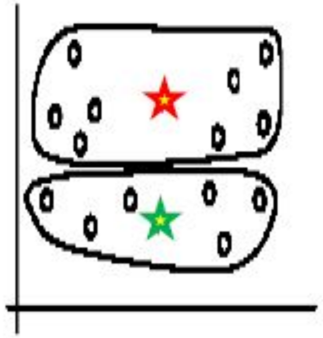
Random selection of seeds (centroids)



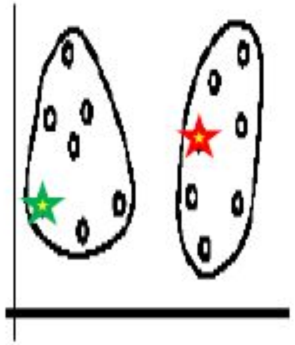
Random selection of seeds (centroids)



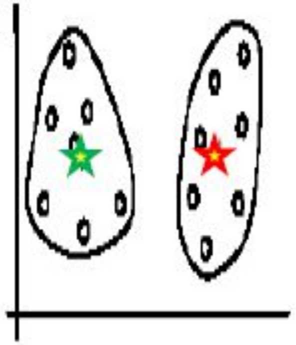
Iteration 1



Iteration 2



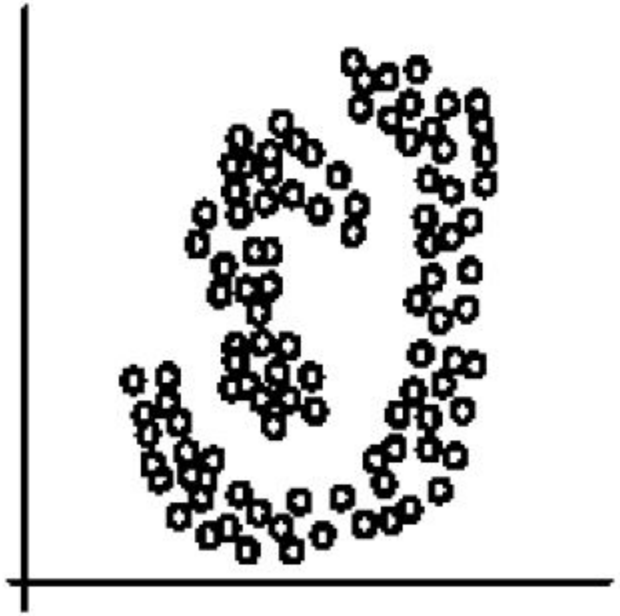
Iteration 1



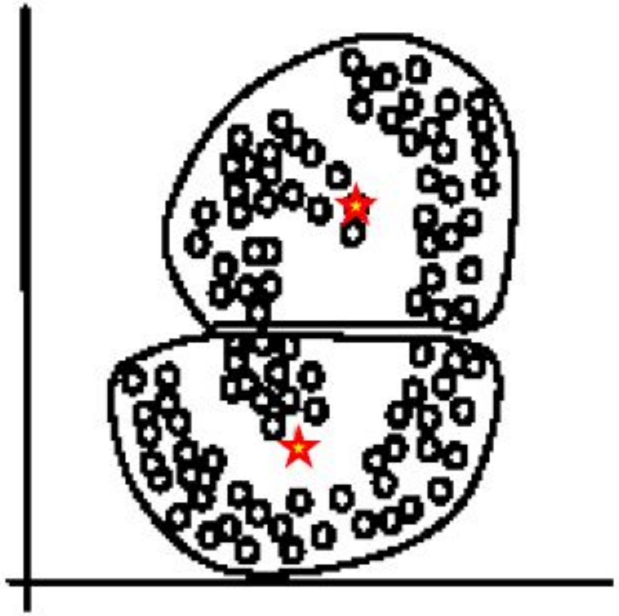
Iteration 2

Special data structures

- The *k*-means algorithm is not suitable for discovering clusters that are not hyper-ellipsoids (or hyper-spheres).



(A): Two natural clusters



(B): *k*-means clusters

K-means summary

- Despite weaknesses, *k*-means is still the most popular algorithm due to its simplicity and efficiency
- No clear evidence that any other clustering algorithm performs better in general
- Comparing different clustering algorithms is a difficult task. No one knows the correct clusters!

Hierarchical Clustering Methods

- **Hierarchical clustering** is characterized by the development of a hierarchy or tree-like structure.

  -**Agglomerative clustering** starts with each object in a separate cluster. Clusters are formed by grouping objects into bigger and bigger clusters.

  -**Divisive clustering** starts with all the objects grouped in a single cluster. Clusters are divided or split until each object is in a separate cluster.

- Agglomerative methods are commonly used in marketing research. They consist of linkage methods, variance methods, and centroid methods.

Hierarchical Clustering Methods

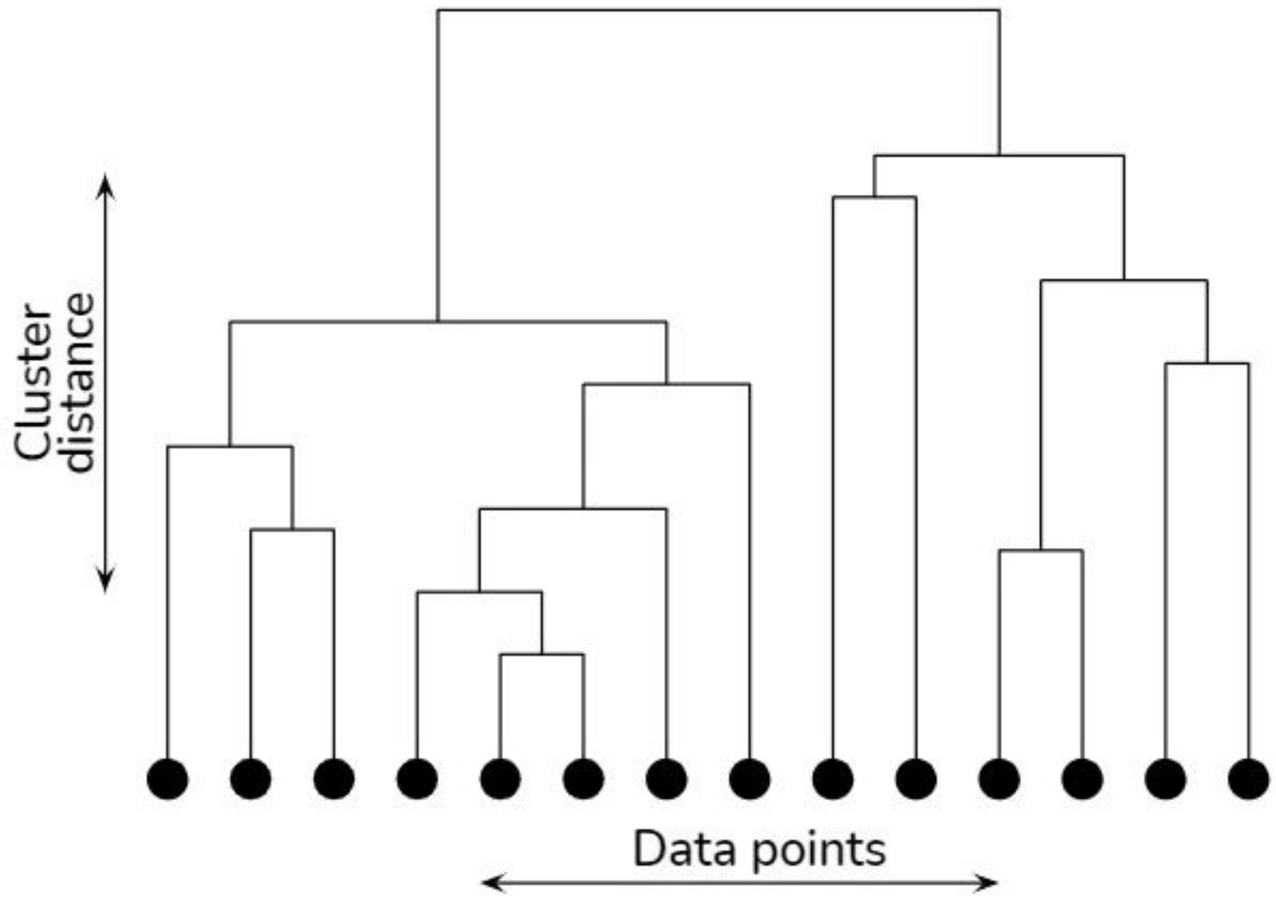
- **Hierarchical clustering** is characterized by the development of a hierarchy or tree-like structure.

  -**Agglomerative clustering** starts with each object in a separate cluster. Clusters are formed by grouping objects into bigger and bigger clusters.

  -**Divisive clustering** starts with all the objects grouped in a single cluster. Clusters are divided or split until each object is in a separate cluster.

- Agglomerative methods are commonly used in marketing research. They consist of linkage methods, variance methods, and centroid methods.

Dendrogram

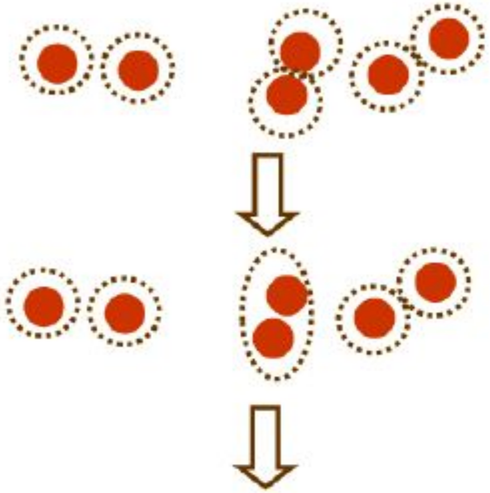


Agglomerative hierarchical clustering

initialize with each example in singleton cluster

while there is more than ***1*** cluster

1. find 2 nearest clusters
2. merge them



- Four common ways to measure cluster distance
  1. minimum distance $d_{\min}(D_i, D_j) = \min_{x \in D_i, y \in D_j} \| x - y \|$
  2. maximum distance $d_{\max}(D_i, D_j) = \max_{x \in D_i, y \in D_j} \| x - y \|$
  3. average distance $d_{avg}(D_i, D_j) = \frac{1}{n_i n_j} \sum_{x \in D_i} \sum_{y \in D_j} \| x - y \|$
  4. mean distance $d_{mean}(D_i, D_j) = \| \mu_i - \mu_j \|$

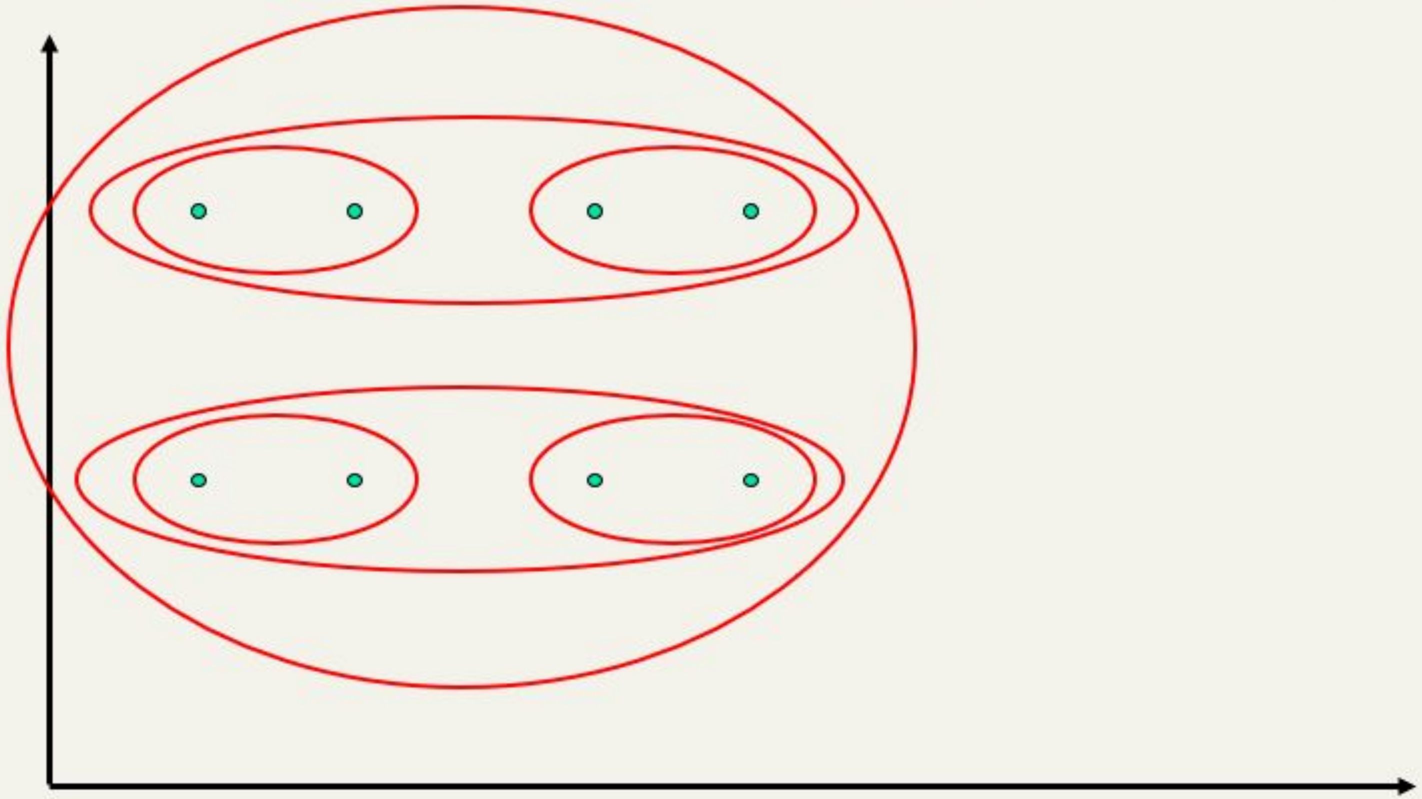
Hierarchical Agglomerative Clustering-Linkage Method

- The **single linkage** method is based on minimum distance, or the nearest neighbor rule.

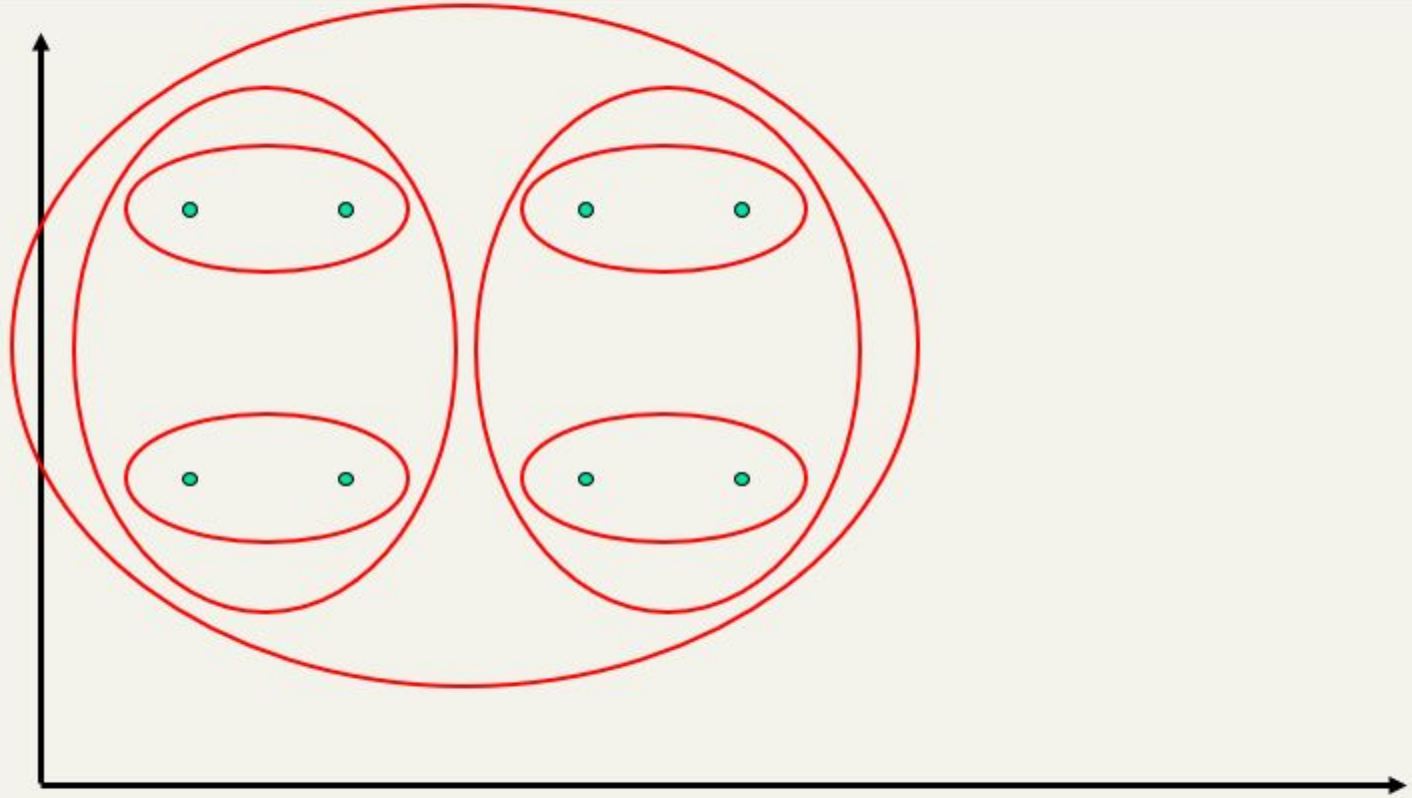
- The **complete linkage** method is based on the maximum distance or the furthest neighbor approach.

- The **average linkage** method the distance between two clusters is defined as the average of the distances between all pairs of objects

Single Link Example



Complete Link Example



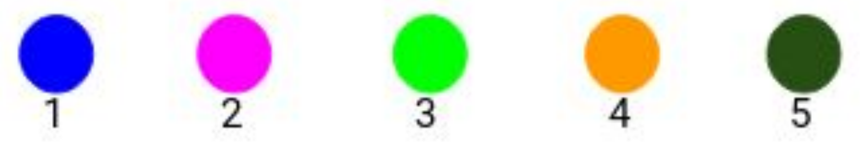
Hierarchical Clustering Example
- Dataset -

Student_ID	Marks
1	10
2	7
3	28
4	20
5	35

Hierarchical Clustering Example
- Initial Step -

ID	1	2	3	4	5
1	0	3	18	10	25
2	3	0	21	13	28
3	18	21	0	8	7
4	10	13	8	0	15
5	25	28	7	15	0

Proximity Matrix



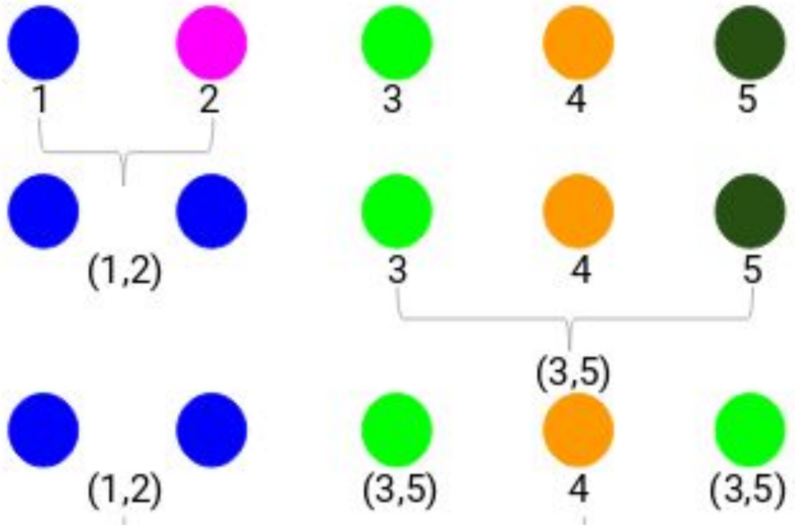
Initial Cluster

Hierarchical Clustering Example
- 1st Merge Step -

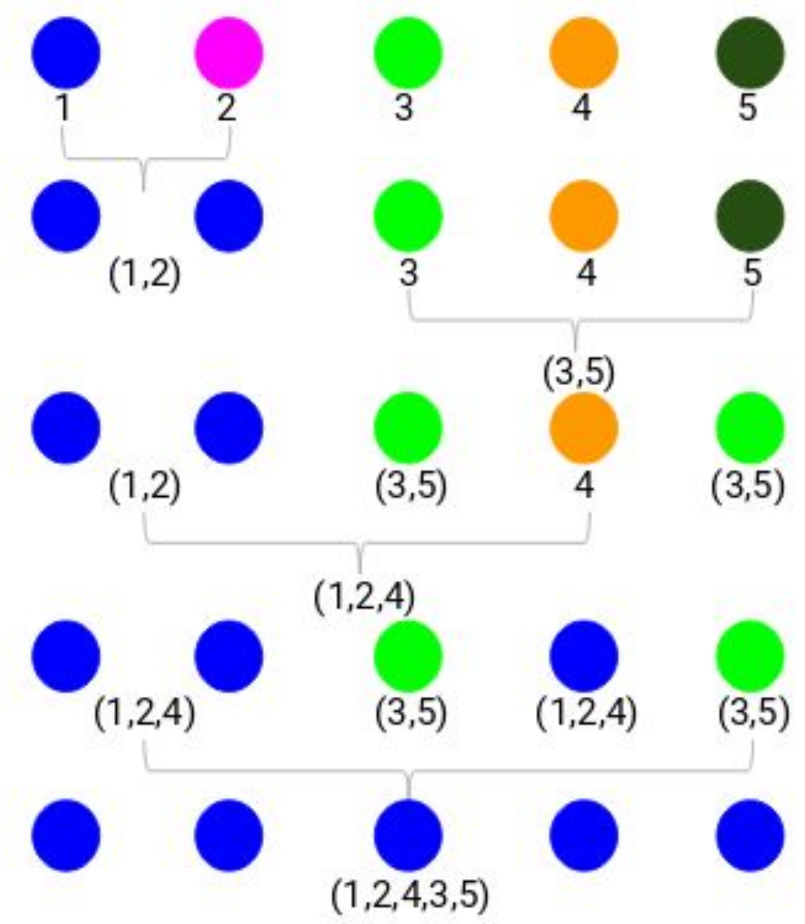
ID	1	2	3	4	5
1	0	3	18	10	25
2	3	0	21	13	28
3	18	21	0	8	7
4	10	13	8	0	15
5	25	28	7	15	0

Hierarchical Clustering Example
- 2nd Merge Step -

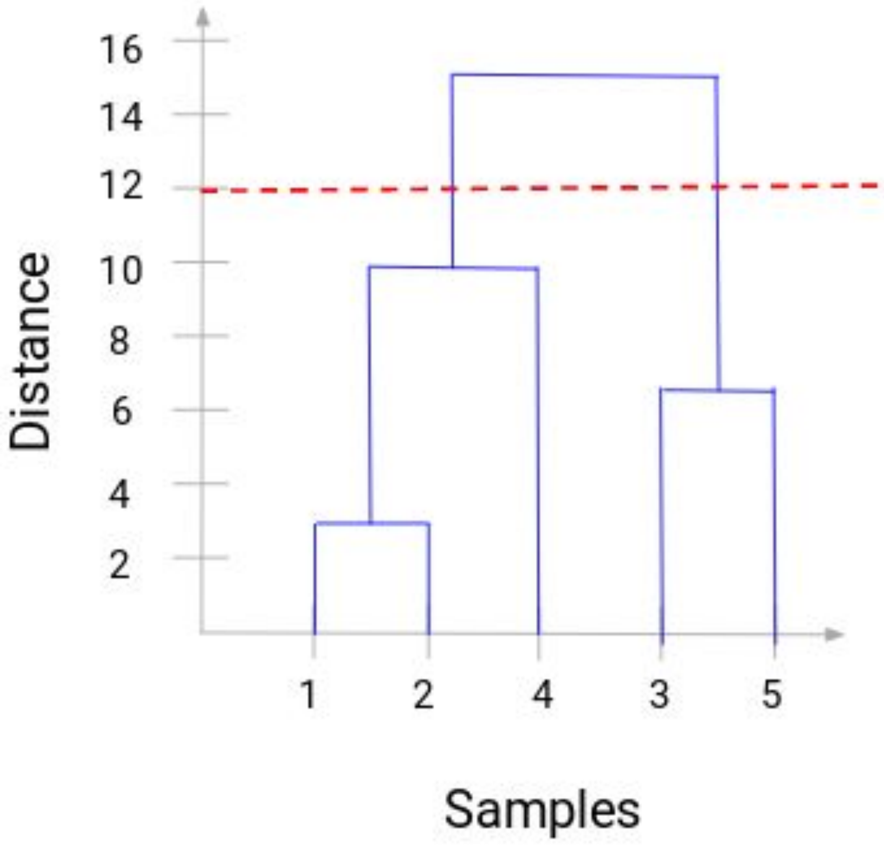
ID	(1,2)	3	4	5
(1,2)	0	18	10	25
3	18	0	8	7
4	10	8	0	15
5	25	7	15	0



Hierarchical Clustering Example
- Final Clustering -

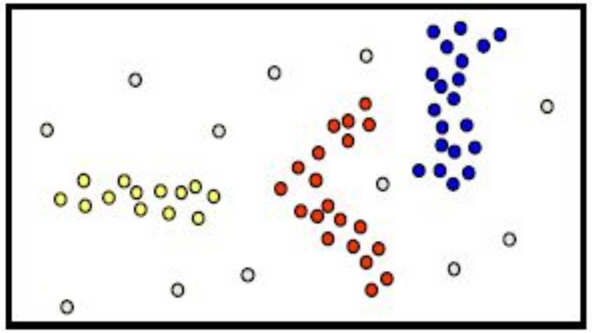


Number of Clusters



Density-based Clustering

- **Basic idea**
  - Clusters are dense regions in the data space, separated by regions of lower object density
  - A cluster is defined as a maximal set of density-connected points
  - Discovers clusters of arbitrary shape
- **Method**
  - DBSCAN

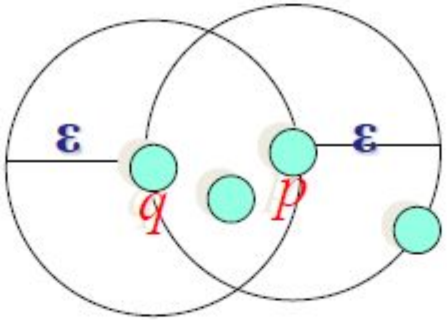


Density Definition

- ε-Neighborhood – Objects within a radius of ε from an object.

$$N_{\varepsilon}(p):\{q \mid d(p,q) \le \varepsilon\}$$

- "High density" - ε-Neighborhood of an object contains at least *MinPts* of objects.



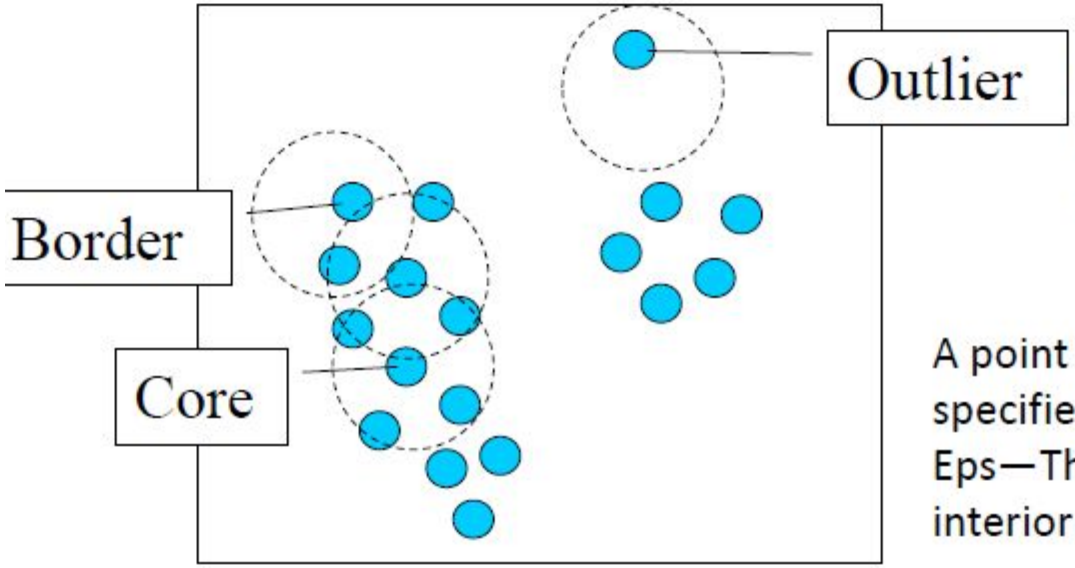
ε-Neighborhood of *p*

ε-Neighborhood of *q*

Density of p is "high" (MinPts = 4)

Density of q is "low" (MinPts = 4)

Core, Border & Outlier



ε = 1unit, MinPts = 5

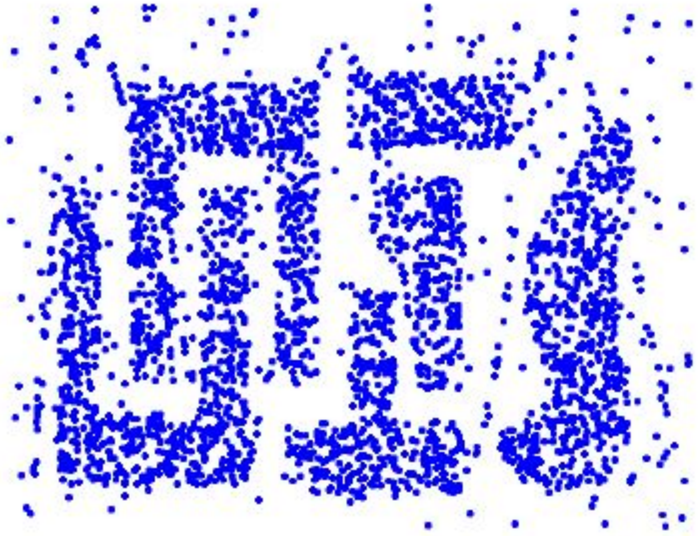
Given ε and *MinPts*, categorize the objects into three exclusive groups.

A point is a core point if it has more than a specified number of points (MinPts) within Eps—These are points that are at the interior of a cluster.

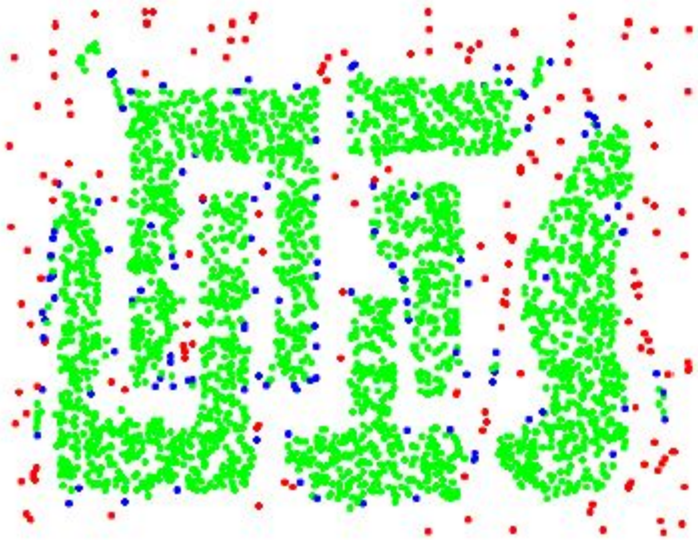
A border point has fewer than MinPts within Eps, but is in the neighborhood of a core point.

A noise point is any point that is not a core point nor a border point.

Example



Original Points

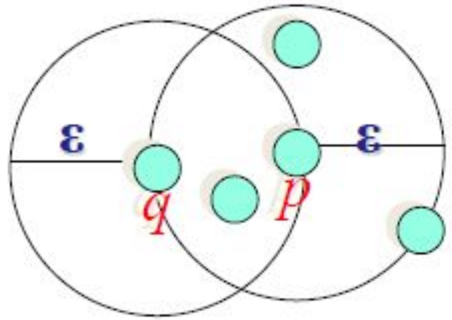


Point types: core, border and outliers

ε = 10, MinPts = 4

Density-reachability

- Directly density-reachable
  - An object *q* is directly density-reachable from object *p* if *p* is a core object and *q* is in p's ε-neighborhood.

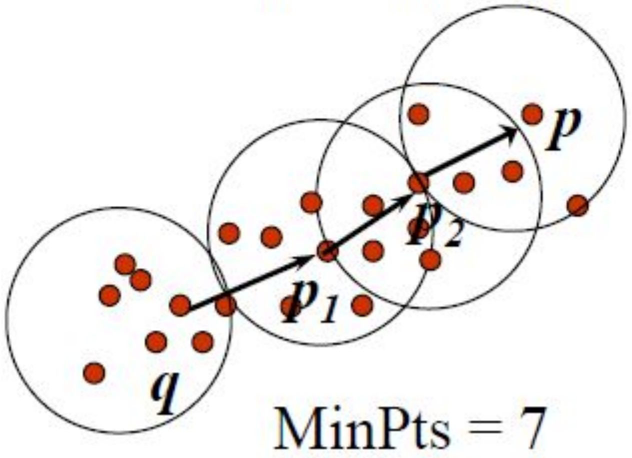


MinPts = 4

- *q* is directly density-reachable from *p*
- *p* is not directly density-reachable from *q*
- Density-reachability is asymmetric

Density-reachability

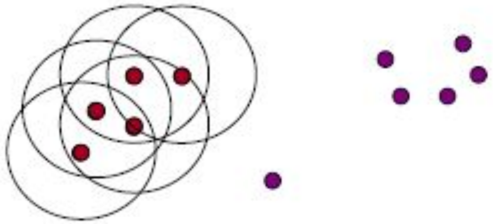
- Density-Reachable (directly and indirectly):
  - A point p is directly density-reachable from p_2
  - p_2 is directly density-reachable from p_1
  - p_1 is directly density-reachable from q
  - $p \leftarrow p_2 \leftarrow p_1 \leftarrow q$ form a chain



- p is (indirectly) density-reachable from q
- q is not density-reachable from p

DBSCAN Algorithm: Example

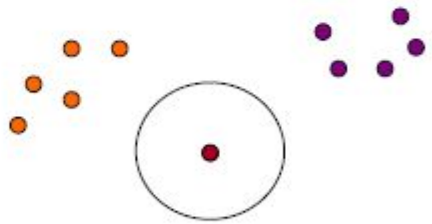
- **Parameter**
  - ε = 2 cm
  - *MinPts* = 3



```
for each o ∈ D do
    if o is not yet classified then
        if o is a core-object then
            collect all objects density-reachable from o
            and assign them to a new cluster.
        else
            assign o to NOISE
```

DBSCAN Algorithm: Example

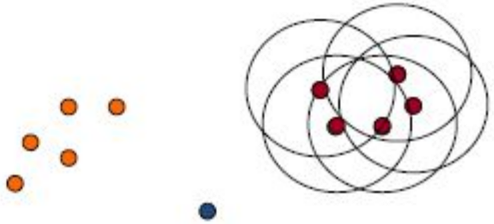
- **Parameter**
  - ε = 2 cm
  - *MinPts* = 3



for each $o \in D$ **do**
    if o is not yet classified **then**
        if o is a core-object **then**
            collect all objects density-reachable from o
            and assign them to a new cluster.
        else
            assign o to NOISE

DBSCAN Algorithm: Example

- **Parameter**
  - ε = 2 cm
  - *MinPts* = 3



for each $o \in D$ **do**
    if o is not yet classified **then**
        if o is a core-object **then**
            collect all objects density-reachable from o
            and assign them to a new cluster.
        else
            assign o to NOISE

DBSCAN: Sensitive to Parameters

Figure 8. DBScan results for DS1 with MinPts at 4 and Eps at (a) 0.5 and (b) 0.4.

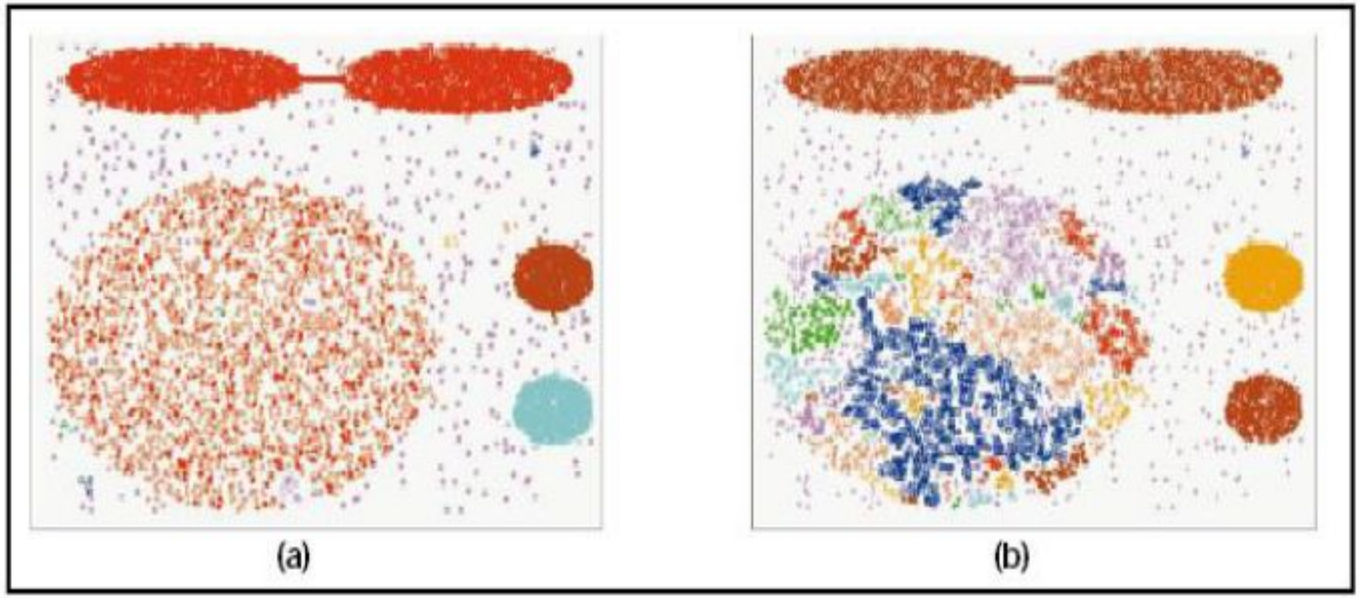
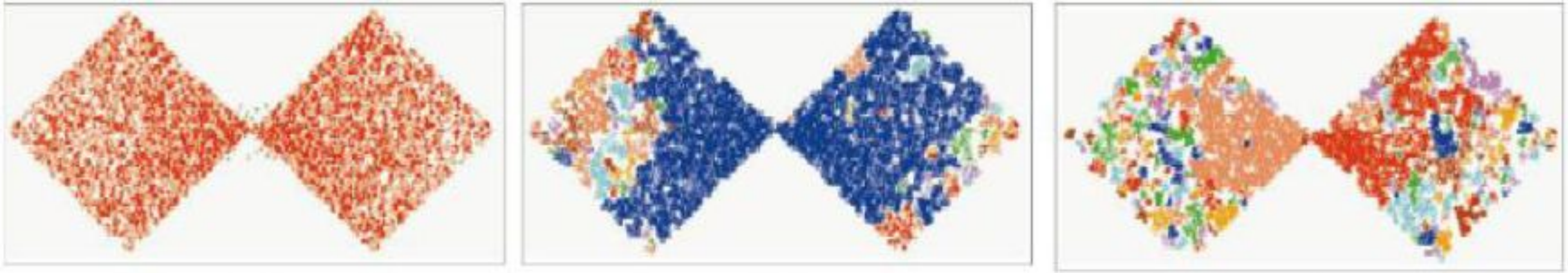


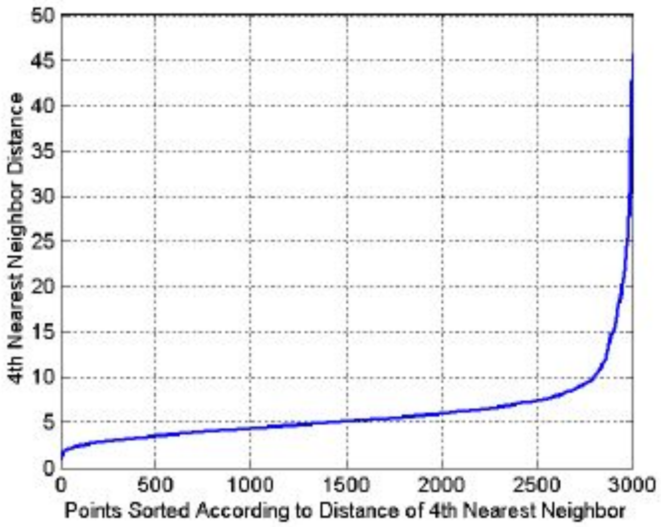
Figure 9. DBScan results for DS2 with MinPts at 4 and Eps at (a) 5.0, (b) 3.5, and (c) 3.0.



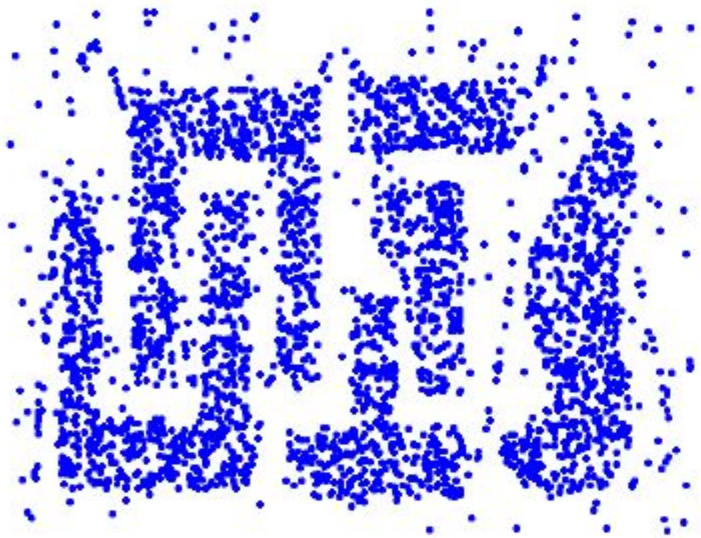
(a) (b) (c)

DBSCAN: Determining EPS and MinPts

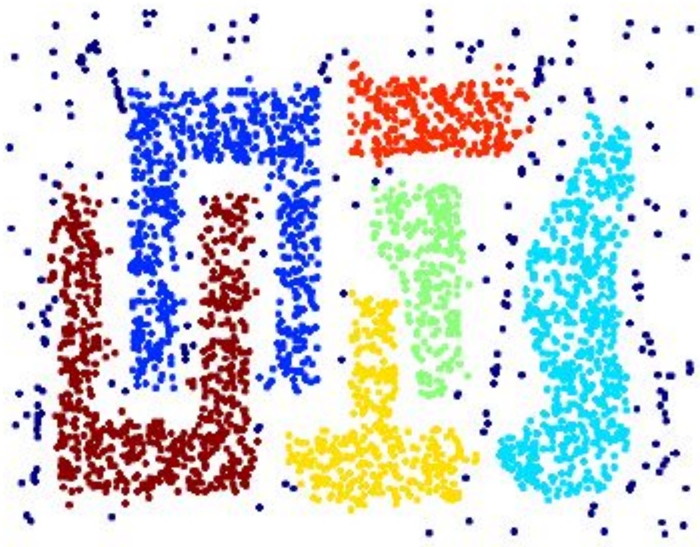
- Idea is that for points in a cluster, their k^{th} nearest neighbors are at roughly the same distance
- Noise points have the k^{th} nearest neighbor at farther distance
- So, plot sorted distance of every point to its k^{th} nearest neighbor



When DBSCAN Works Well



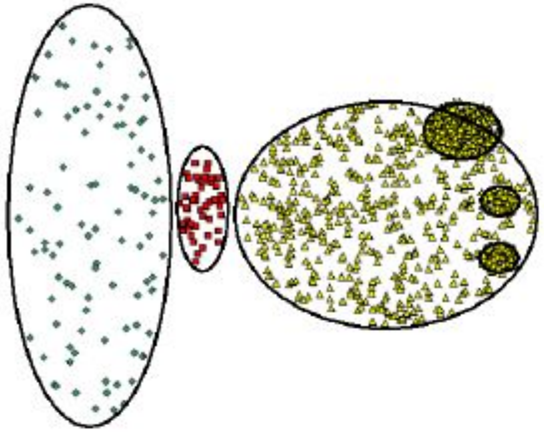
Original Points



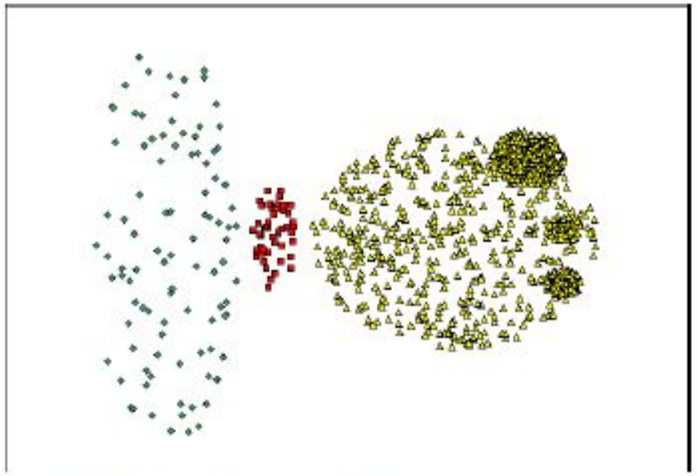
Clusters

- Resistant to Noise
- Can handle clusters of different shapes and sizes

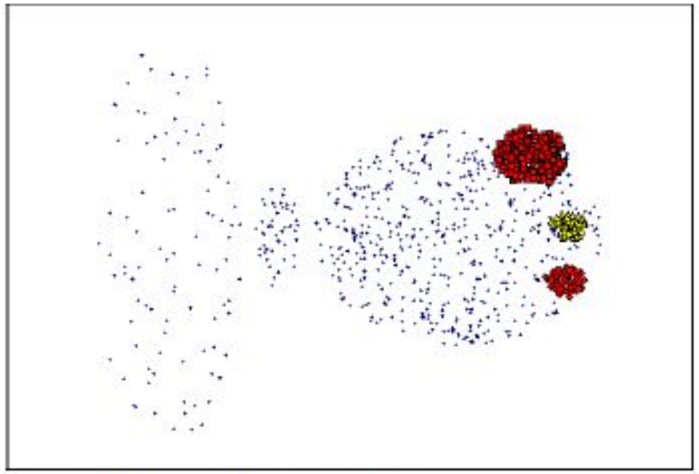
When DBSCAN Does NOT Work Well



Original Points



(MinPts=4, Eps=9.92).

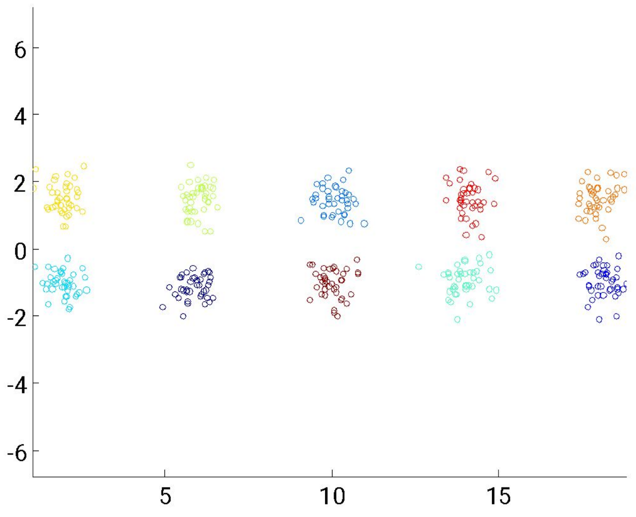


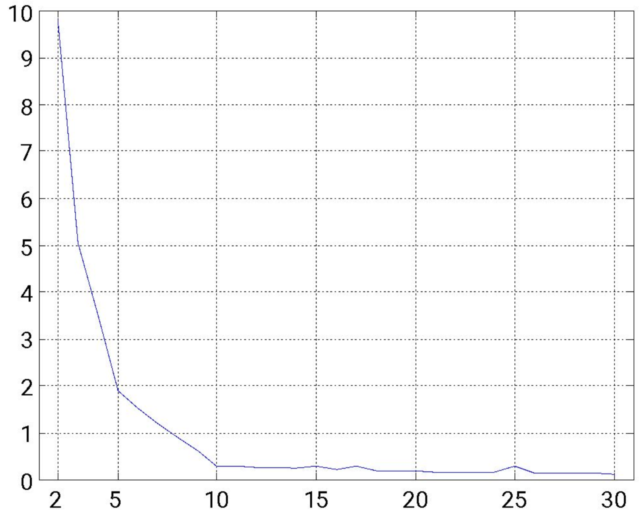
(MinPts=4, Eps=9.75)

- Cannot handle varying densities
- sensitive to parameters—hard to determine the correct set of parameters

Internal Measures: SSE

- Clusters in more complicated figures aren't well separated
- Internal Index: Used to measure the goodness of a clustering structure without respect to external information
    - SSE
- SSE is good for comparing two clusterings or two clusters (average SSE).
- Can also be used to estimate the number of clusters





Measure II: Cohesion and Separation

- Cluster Cohesion: Measures how closely related are objects in a cluster
  - Example: SSE
- Cluster Separation: Measure how distinct or well-separated a cluster is from other clusters

- Example: Squared Error
  - Cohesion is measured by the within cluster sum of squares (SSE)

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

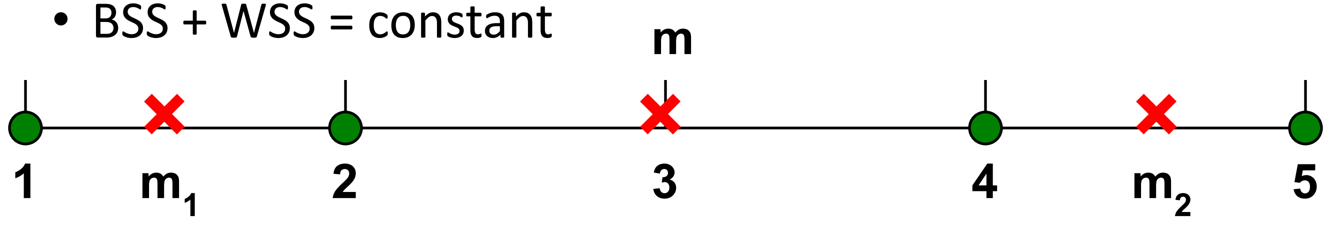
  - Separation is measured by the between cluster sum of squares

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

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

Internal Measures: Cohesion and Separation

- Example: SSE
  - BSS + WSS = constant



K=1 cluster:

$$WSS = (1-3)^2 + (2-3)^2 + (4-3)^2 + (5-3)^2 = 10$$

$$BSS = 4 \times (3-3)^2 = 0$$

$$Total = 10 + 0 = 10$$

K=2 clusters:

$$WSS = (1-1.5)^2 + (2-1.5)^2 + (4-4.5)^2 + (5-4.5)^2 = 1$$

$$BSS = 2 \times (3-1.5)^2 + 2 \times (4.5-3)^2 = 9$$

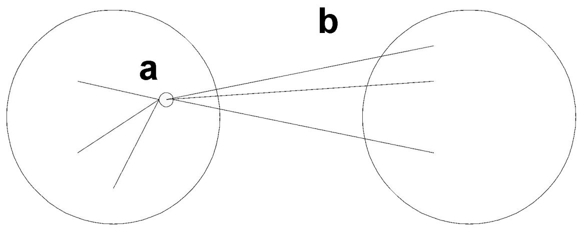
$$Total = 1 + 9 = 10$$

Measure III: Silhouette Coefficient

- Silhouette Coefficient combine ideas of both cohesion and separation, but for individual points, as well as clusters and clusterings
- For an individual point, *i*
  - Calculate ***a(i)*** = average distance of *i* to the points in its cluster
  - Calculate ***b(i)*** = min (average distance of *i* to points in another cluster)
  - The silhouette coefficient for a point is then given by

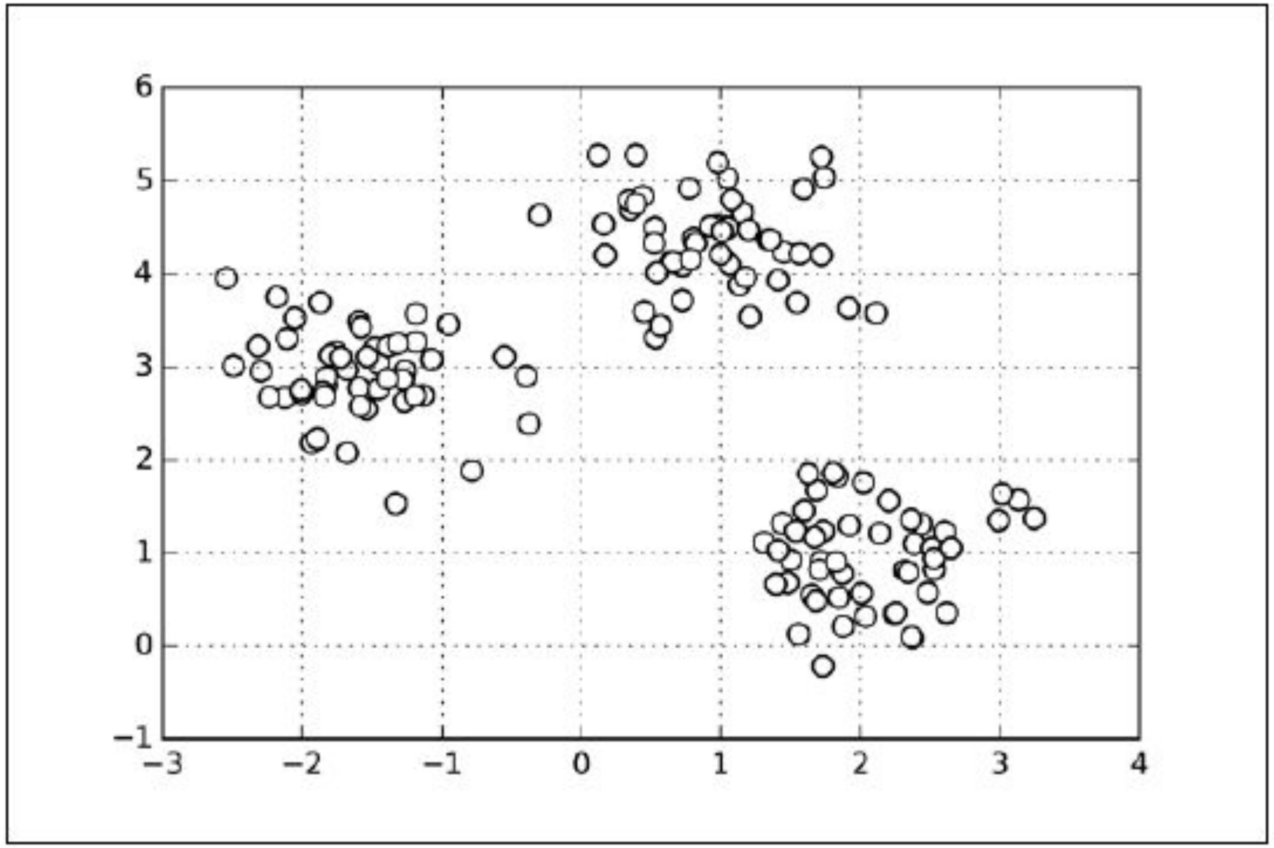
$$s(i) = \frac{b(i) - a(i)}{\max\{a(i), b(i)\}}$$

  - Typically between 0 and 1.
  - The closer to 1 the better.

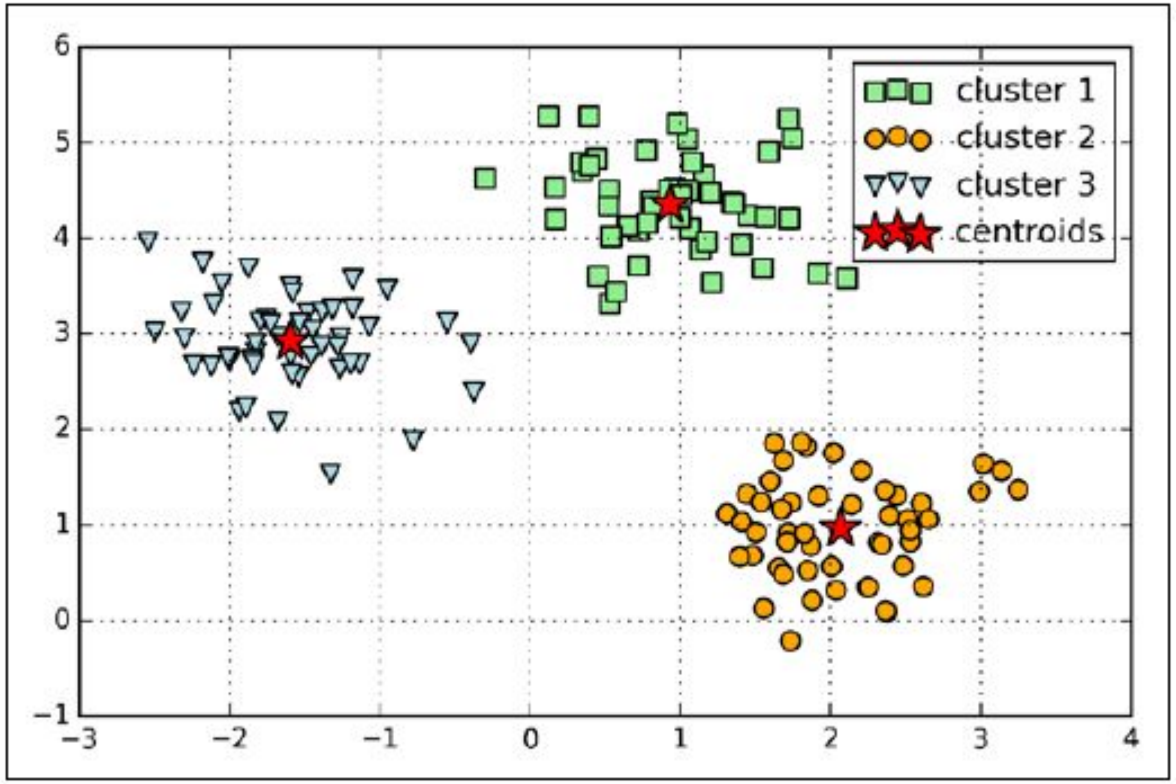


- Can calculate the Average Silhouette width for a cluster or a clustering

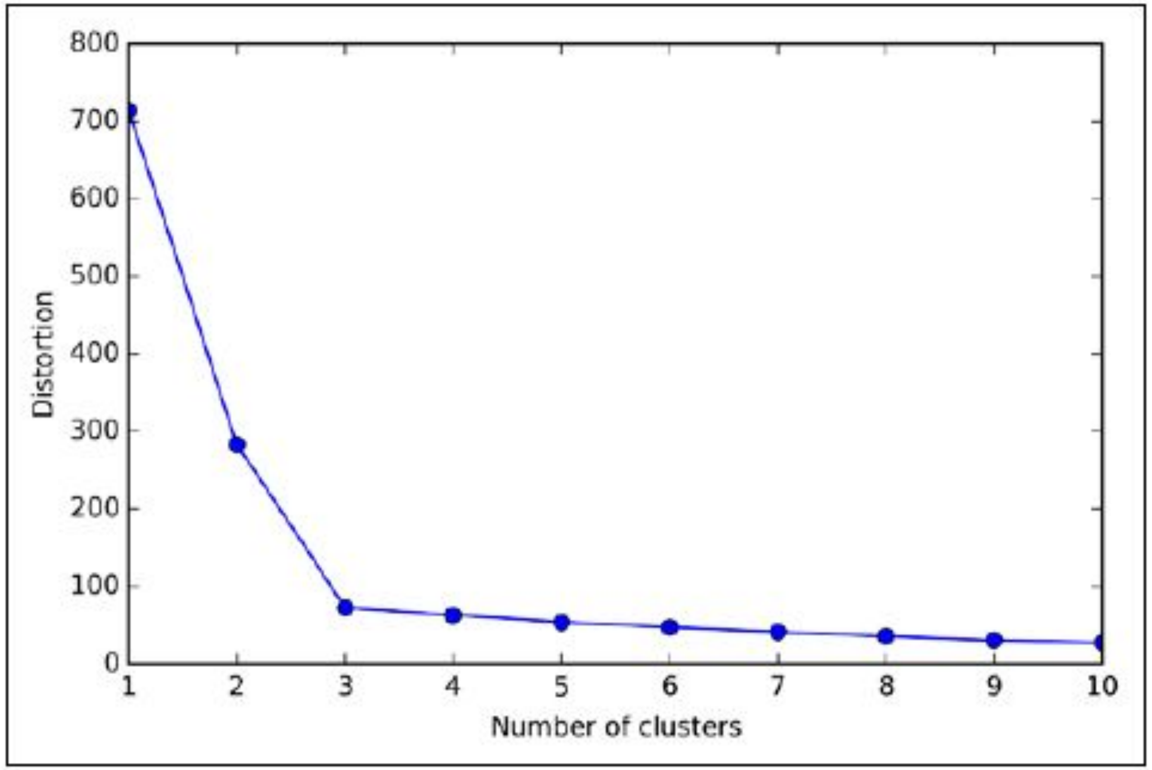
Example:



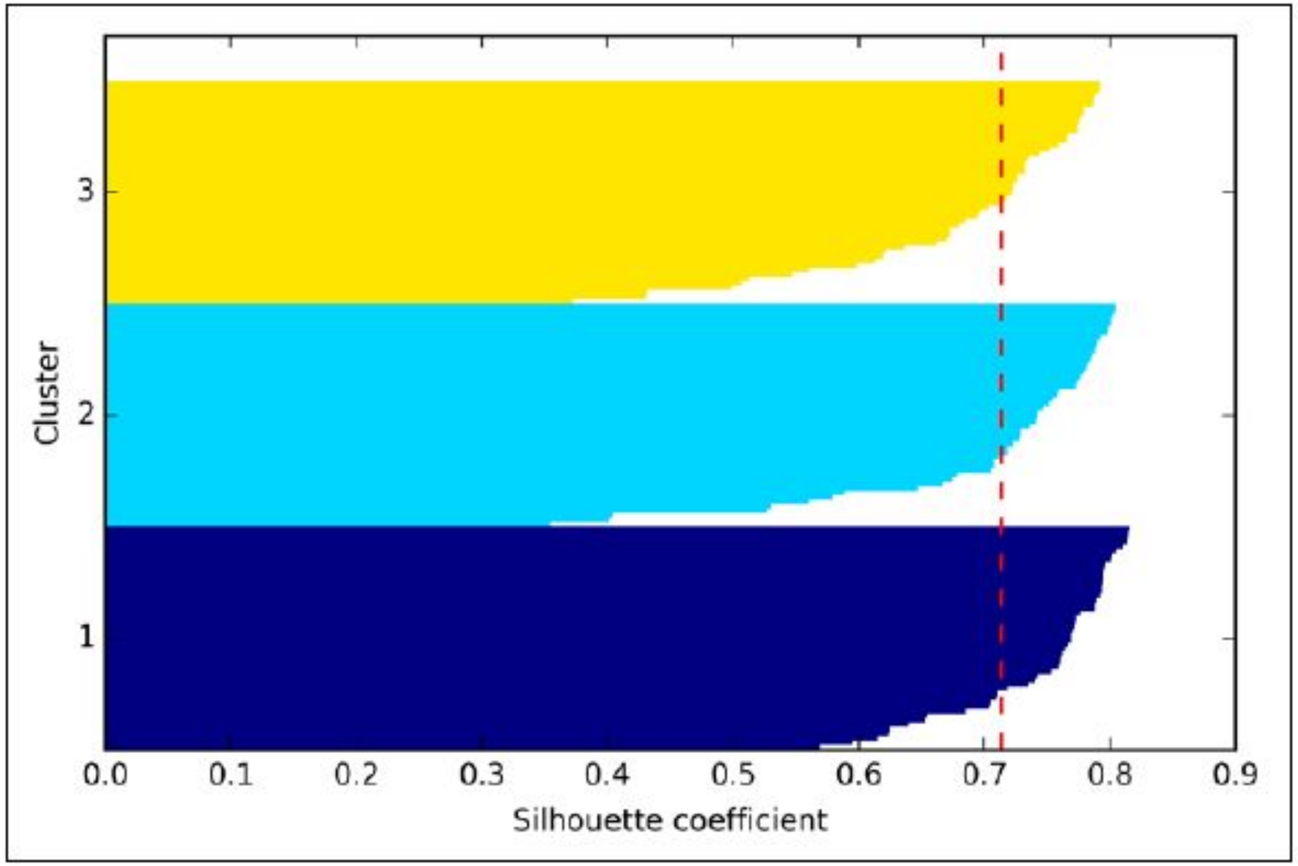
K-Means (K=3)



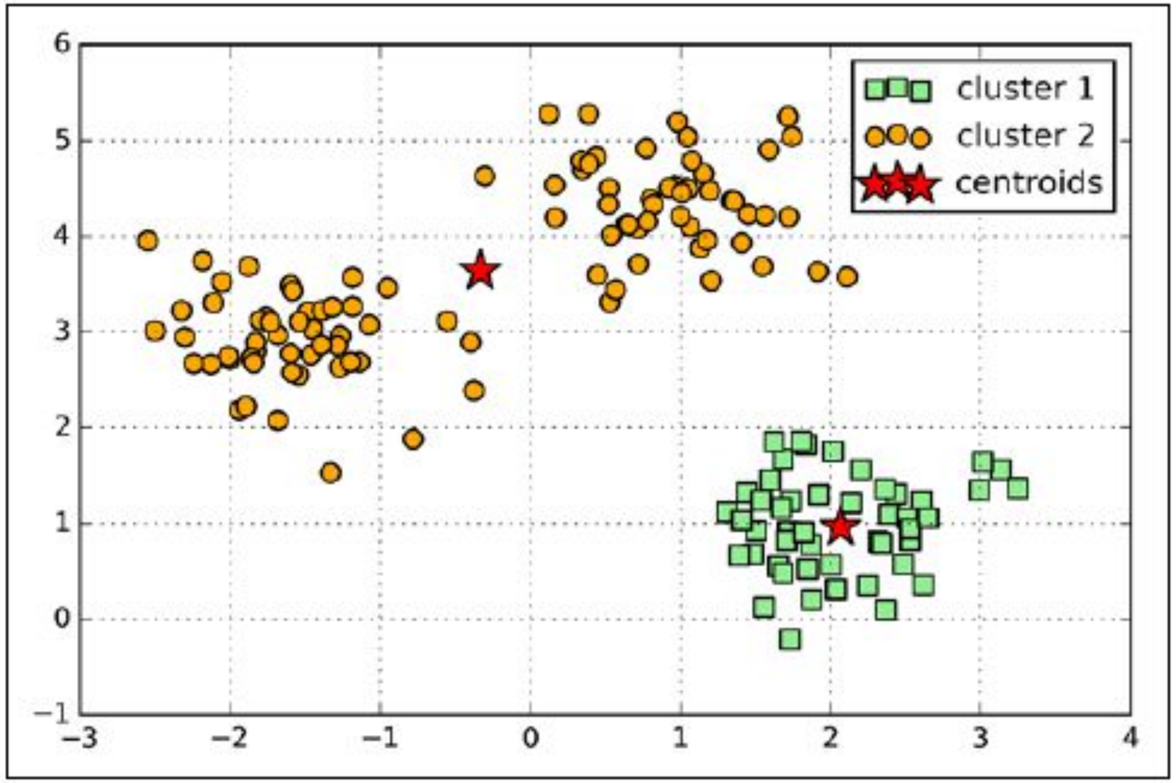
SSE vs Number of Clusters (Elbow Method)



Silhouette Coefficient Plot (k=3)



K-Means (K=2)



Silhouette Coefficient Plot (k=2)