Machine Learning

Amirkabir University of Technology
(Tehran Polytechnic)

Lecture 10.
Unsupervised learning
Clustering

Alireza Rezvanian

Fall 2023

Last update: Dec. 12, 2022

Outline

- Unsupervised learning
- Clustering
  - Partitioning algorithms
    - K-Means Clustering
    - k-medoids
- Hierarchical algorithms
  - Agglomerative
  - Divisive
- Clustering Validity

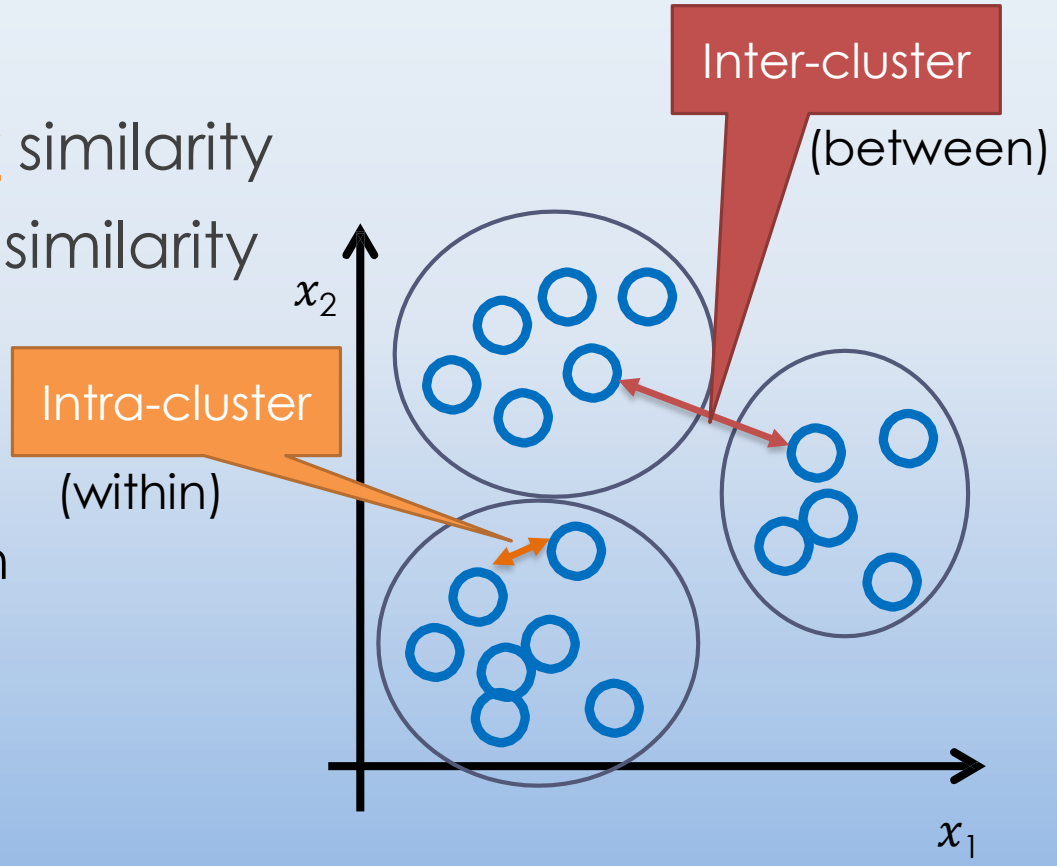
Unsupervised learning

- Clustering
  - Partitioning of data into groups of similar data points.
- Density estimation
  - Parametric & non-parametric density estimation
- Dimensionality reduction
  - Data representation using a smaller number of dimensions while preserving (perhaps approximately) some properties of the data

Clustering: Definition

- We have a set of unlabeled data points {x_1, x_2, …x_n} and we intend to **find groups of similar objects** (based on the observed features)
  - High **intra-cluster** similarity
  - Low **inter-cluster** similarity

Propose
- To group or partition the data when no label is available

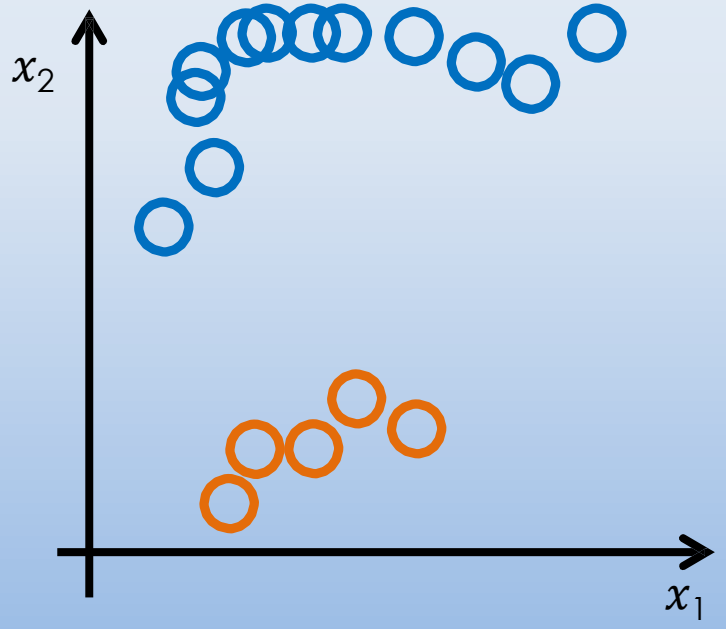


Clustering: Another Definition

- Density-based definition
  - Clusters are regions of high density that are separated from one another by regions of low density

Difficulties:

- Clustering is not as well-defined as classification
- Clustering is subjective
  - Natural grouping may be ambiguous



Clustering Applications

- **Information retrieval** (search and browsing): Cluster text docs or images based on their content (Cluster groups of users based on their access patterns on webpages)
- **Community detection** (Cluster users of social networks by interest)
- **Market segmentation**: Clustering customers based on the their purchase history and their characteristics
- **Image segmentation**: Separate objects from background and identify them or Remove noise
- **Insurance**: Identifying groups of motor insurance policy holders with a high average claim cost.
- **Land use**: Identification of areas of similar land use in an earth observation database.

What is Good Clustering?

- A good clustering method will produce high quality clusters in which:
  - High **intra-cluster** similarity: cohesive within clusters
  - Low **inter-cluster** similarity: distinctive between clusters
- The **quality** of a clustering result also depends on both the similarity measure used by the method and its implementation.
- The **quality** of a clustering method is also measured by its ability to discover some or all of the **hidden** patterns.
- The **quality** of a clustering result also depends on the definition and representation of cluster chosen.

Measure the Quality of Clustering

- Dissimilarity/Similarity metric
  - Similarity is expressed in terms of a distance function, typically metric: $d(i, j)$
- The definitions of distance functions are usually rather different for interval-scaled, Boolean, categorical, ordinal ratio, and vector variables
- Weights should be associated with different variables based on applications and data semantics
- Quality of clustering:
  - There is usually a separate "quality" function that measures the "goodness" of a cluster.
  - It is hard to define "similar enough" or "good enough"
  - The answer is typically highly subjective

Considerations for Cluster Analysis

- **Partitioning criteria**
  - **Single level** vs. **hierarchical** partitioning (often, multi-level hierarchical partitioning is desirable)
- **Separation of clusters**
  - **Exclusive** (e.g., one customer belongs to only one region) vs. **non-exclusive** (e.g., one document may belong to more than one class)
- **Similarity measure**
  - **Distance-based** (e.g., Euclidian, road network, vector) vs. **connectivity-based** (e.g., density or contiguity)
- **Clustering space**
  - **Full space** (often when low dimensional) vs. **subspaces** (often in high-dimensional clustering)

Two main categorization of Clustering Algorithms

- **Partitioning algorithms**: Construct various partitions and then evaluate them by some criterion the desired number of clusters ***k*** must be specified.
  - **k-means** (MacQueen'67): Each cluster is represented by the center of the cluster.
  - **k-medoids** or **PAM** (Partition around medoids) (Kaufman & Rousseeuw '87): Each cluster is represented by one of the objects in the cluster
- **Hierarchical algorithms**: Create a hierarchical decomposition of the set of objects using some criterion

Partitioning Clustering

- $X=\{x_1, x_2, \ldots x_n\}$
- $C = \{c_1, c_2, \ldots, c_k\}$
- $c_i \neq \phi$
- $X = c_1 \cup c_2 \cup \ldots \cup c_k$
- $c_i \cap c_j = \varnothing$ (disjoint partitioning for hard clustering)

Non-hierarchical, each instance is placed in exactly one of K non-overlapping clusters

- **Hard clustering**: Each data can belong to one cluster only
- Since the output is only one set of clusters the user has to specify the desired number of clusters ***k***
- ***k*-means** is the most popular partitioning algorithm

The K-Means Clustering Method

- Given ***k***, the k-means algorithm is implemented in 4 steps:

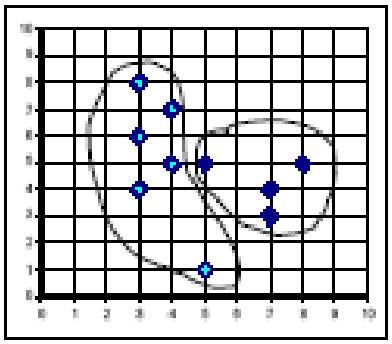
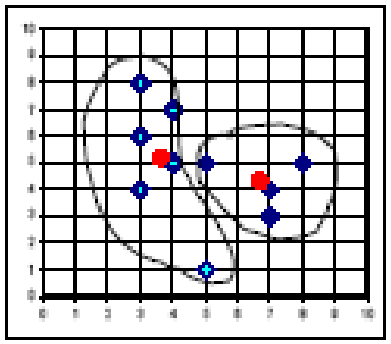
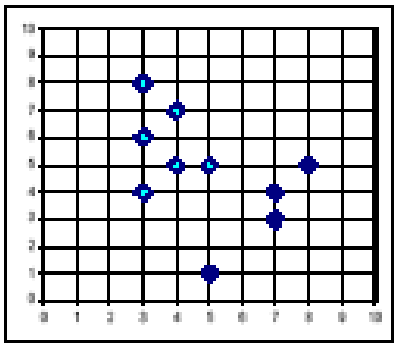
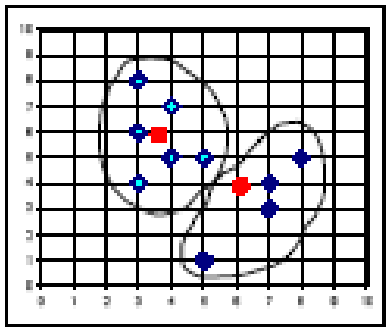
$$\sum_{i=1}^{N} \min_{j \in 1,\ldots,K} \|\boldsymbol{x}^{(i)} - \boldsymbol{c}_j\|^2$$

1. Partition objects into ***k*** non-empty subsets

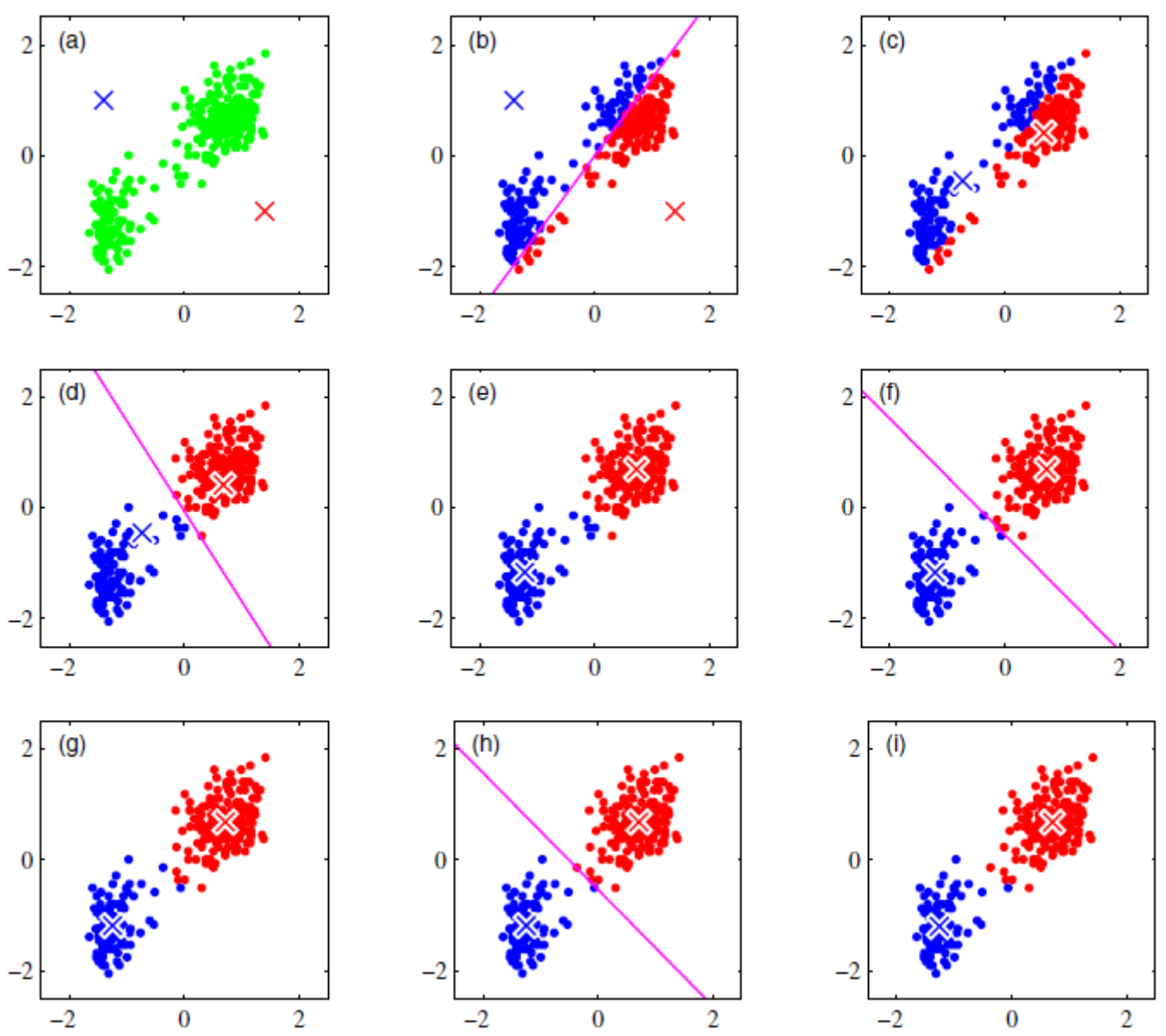
2. Compute seed points as the centroids of the clusters of the current partition. (The centroid is the center, i.e., mean point of the cluster.

3. Assign each object to the cluster with the nearest seed point.

4. Go back to Step 2, stop when no more new assignment.

Example (Assigning data clusters and Updating means)



Intra-cluster similarity view

- *k*-means optimizes intra-cluster similarity

$$J(\mathcal{C}) = \sum_{j=1}^{K} \sum_{\boldsymbol{x}^{(i)} \in \mathcal{C}_j} \left\| \boldsymbol{x}^{(i)} - \boldsymbol{c}_j \right\|^2$$

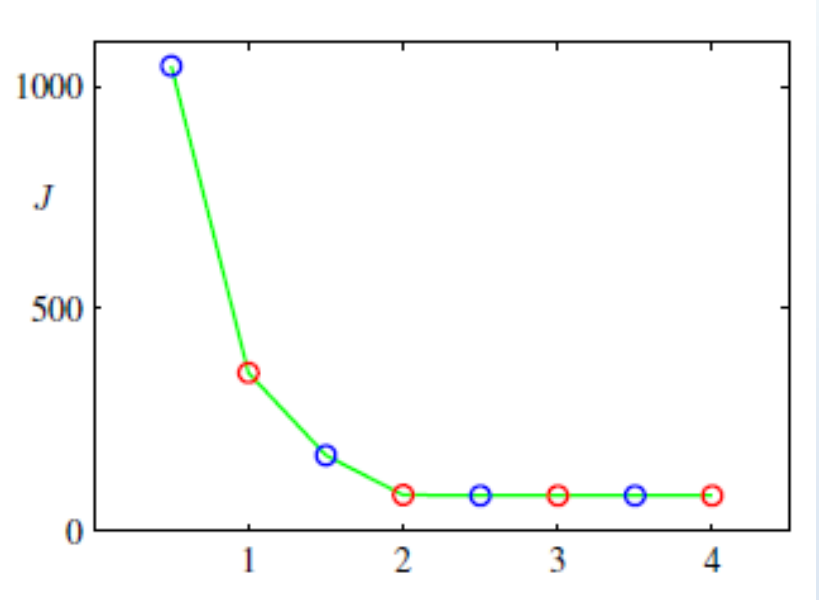
$$\boldsymbol{c}_j = \frac{1}{|\mathcal{C}_j|} \sum_{\boldsymbol{x}^{(i)} \in \mathcal{C}_j} \boldsymbol{x}^{(i)}$$

$$\sum_{\boldsymbol{x}^{(i)} \in \mathcal{C}_j} \left\| \boldsymbol{x}^{(i)} - \boldsymbol{c}_j \right\|^2 = \frac{1}{2|\mathcal{C}_j|} \sum_{\boldsymbol{x}^{(i)} \in \mathcal{C}_j} \sum_{\boldsymbol{x}^{(i')} \in \mathcal{C}_j} \left\| \boldsymbol{x}^{(i)} - \boldsymbol{x}^{(i')} \right\|^2$$

the average distance to members of the same cluster

k-means: Convergence

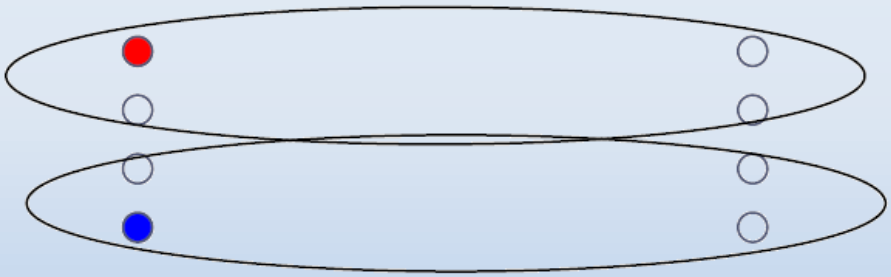
- It always converges.

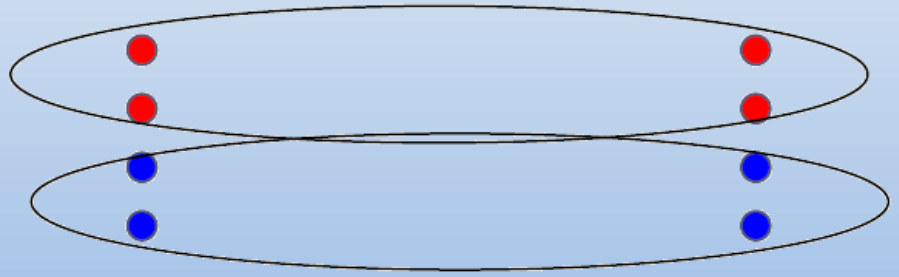


- Why should the *k*-means algorithm ever reach a state in which clustering doesn't change.
  - Reassignment stage monotonically decreases since each vector is assigned to the closest centroid.
  - Centroid update stage also for each cluster minimizes the sum of squared distances of the assigned points to the cluster from its center

Local optimum

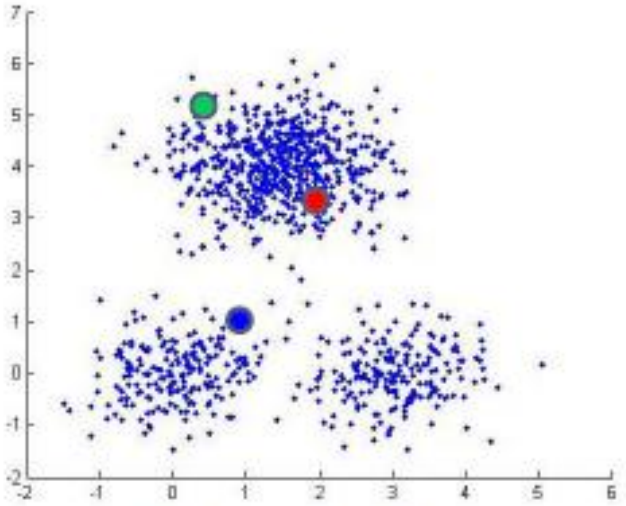
- It always converges
- but it may converge at a local optimum that is different from the global optimum
  - may be arbitrarily worse in terms of the objective score



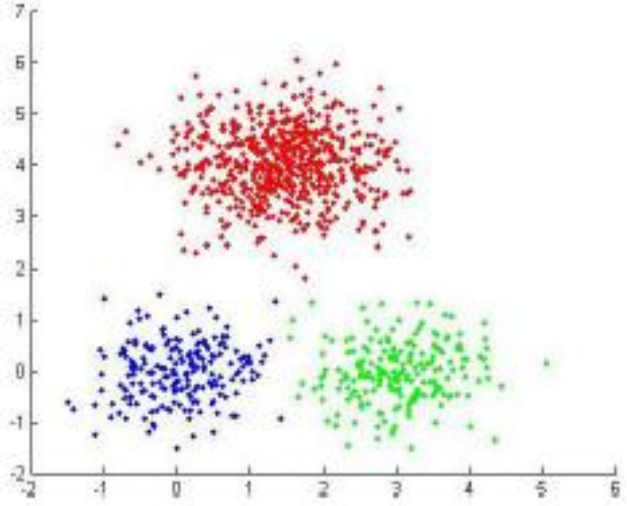


Local optimum: every point is assigned to its nearest center and every center is the mean value of its points.

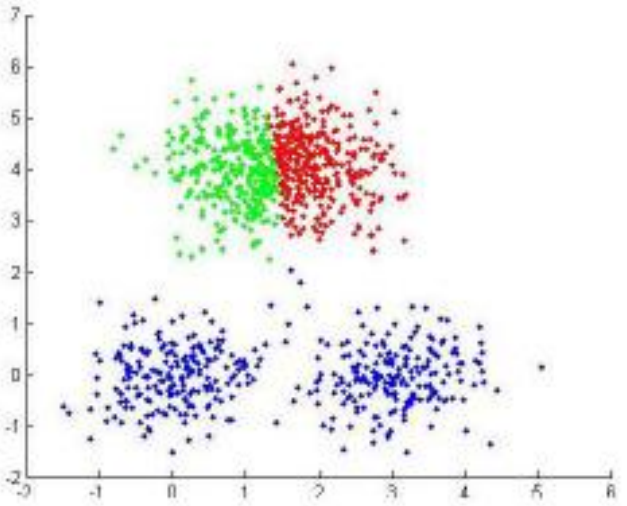
K-means: Local Minimum Problem



Original Data



Optimal Clustering



The obtained Clustering

The Lloyd's method: Initialization

- Initialization is crucial (how fast it converges, quality of clustering)
  - Random centers from the data points
    - Multiple runs and select the best ones
  - Initialize with the results of another method
  - Select good initial centers using a heuristic
    - Furthest traversal (It is sensitive to outliers)
      - Choose c_1 arbitrarily (or at random).
      - For i = 2, … , k
        - Select c_i among datapoints x_1, …, x_n that is farthest from previously chosen c_1, … , c_{i-1}
    - K-means++ (works well and has provable guarantees)

k-means++ Initialization: D2 sampling [D. Arthur and S. Vassilvitskii, 2007]

- Combine random initialization and furthest point initialization ideas
- Let the probability of selection of the point be proportional to the distance between this point and its nearest center.
  - probability of selecting of x is proportional to $D^2(x) = \min_{k<j} ||x - c_k||^2$.

Choose c_1 arbitrarily (or at random).
For j = 2, … , k
Select c_i among data points x_1, … , x_n according to the distribution:

$$\Pr(c_j = x_i) \propto \min_{k<j} ||x - c_k||^2$$

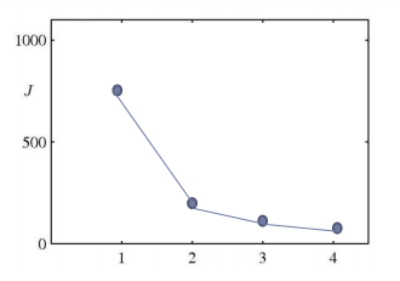
- Theorem: *k*-means++ always attains an $\boldsymbol{O(\log k)}$ approximation to optimal *k*-means solution in expectation

How Many Clusters?

- Number of clusters ***k*** is given in advance in the *k*-means algorithm
  - However, finding the "right" number of clusters is a part of the problem

- Tradeoff between having better focus within each cluster and having too many clusters

How Many Clusters?



- Heuristic:
  - Find large gap between $k-1$-means cost and k-means cost.
  - "knee finding" or "elbow finding".
- Hold-out validation/cross-validation on auxiliary task (e.g., supervised learning task).
- Optimization problem: penalize having lots of clusters
  - some criteria can be used to automatically estimate ***k***
    - Penalize the number of bits you need to describe the extra parameter

$$J'(C) = J(C) + |C| \times \log N$$

- Hierarchical clustering

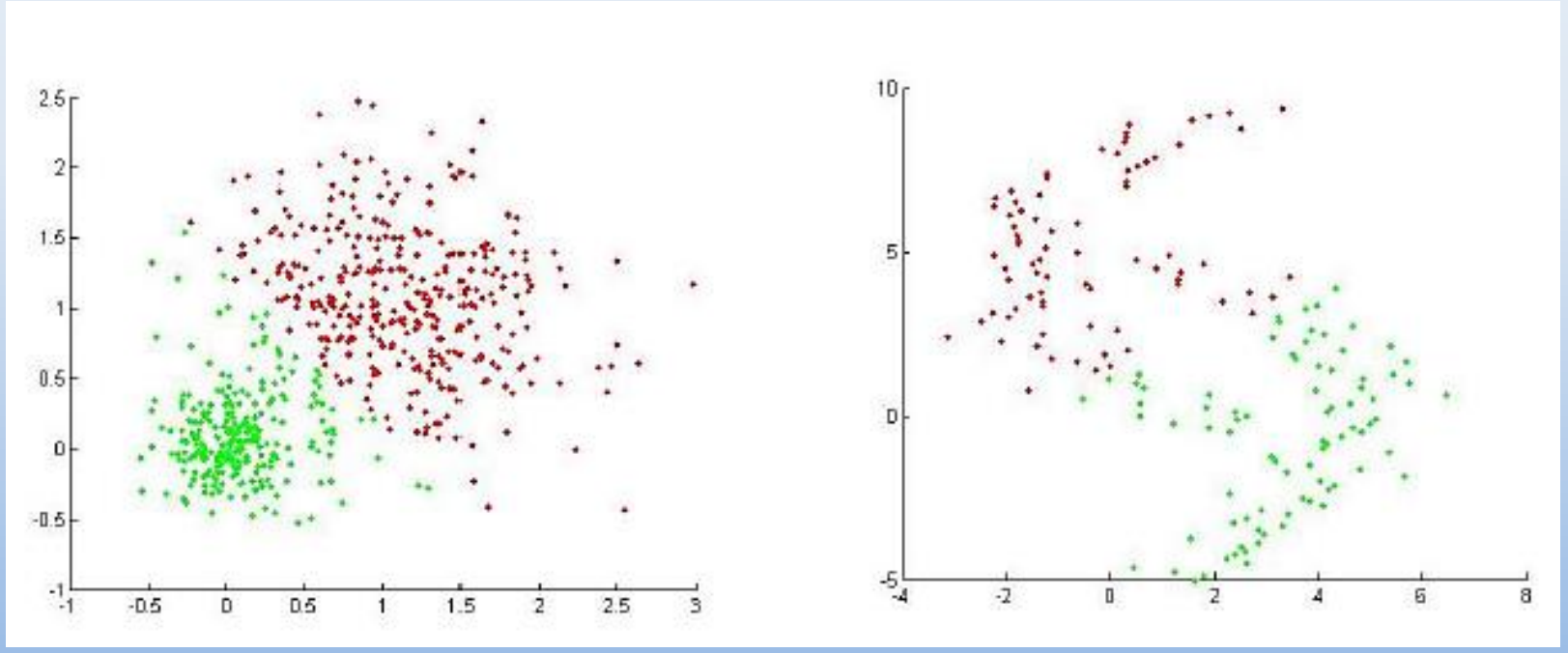
k-means: Advantages and disadvantages

- **Strength**
  - It is a simple method and easy to implement.
  - Relatively efficient: $\boldsymbol{O(tkn)}$, where $\boldsymbol{n}$ is #instances, $\boldsymbol{k}$ is #clusters, and $\boldsymbol{t}$ is #iterations. Normally, $k, t << n$
  - *k*-means typically converges quickly
  - Exponential #of rounds in the worst case [Andrea Vattani 2009].
- **Weakness**
  - Need to specify k, the number of clusters, in advance
  - Often terminates at a local optimum.
  - Applicable only to objects in a continuous n-dimensional space
    - Using the *k*-modes method for categorical data
    - In comparison, *k*-medoids can be applied to a wide range of data
  - Sensitive to noisy data and *outliers*
  - Not suitable to discover clusters with arbitrary data (*non-convex shapes*)

k-means Algorithm: Limitation

- In general, k-means is unable to find clusters of arbitrary shapes, sizes, and densities

- Except to very distant clusters



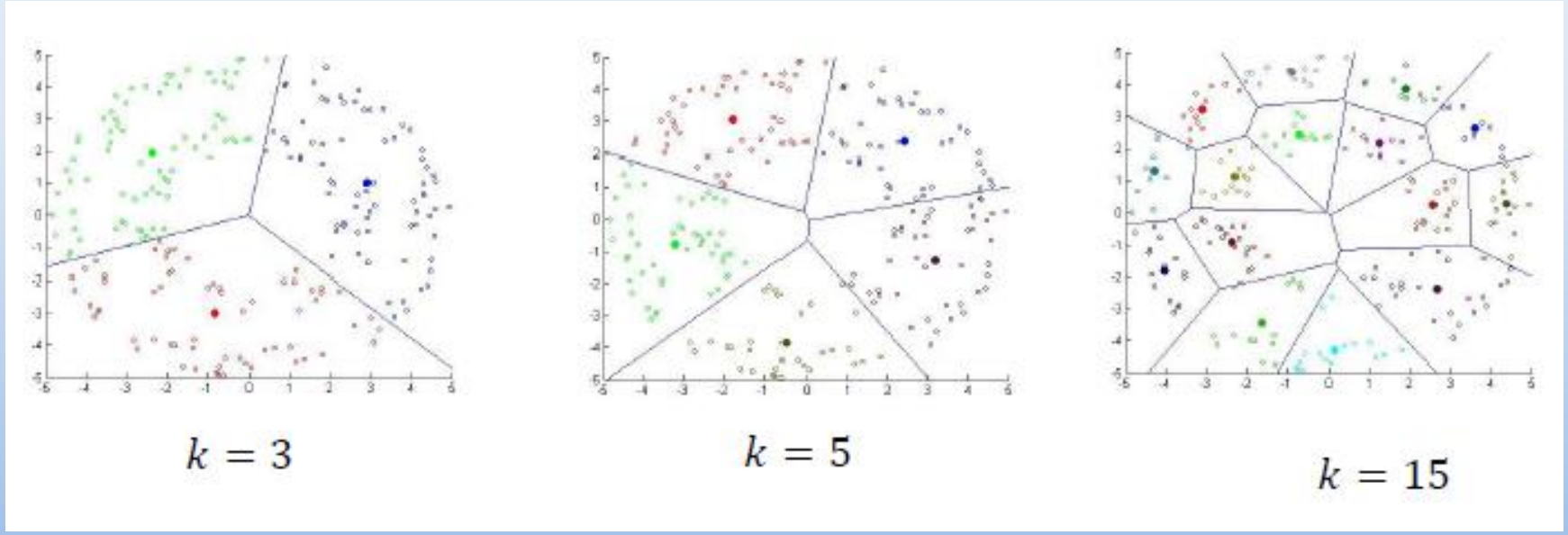
k-means

- *k*-means was proposed near 60 years ago
  - Thousands of clustering algorithms have been published since then
  - However, *k*-means is still widely used.

- This speaks to the difficulty in designing a general purpose clustering algorithm and the ill-posed problem of clustering

k-means: Vector Quantization

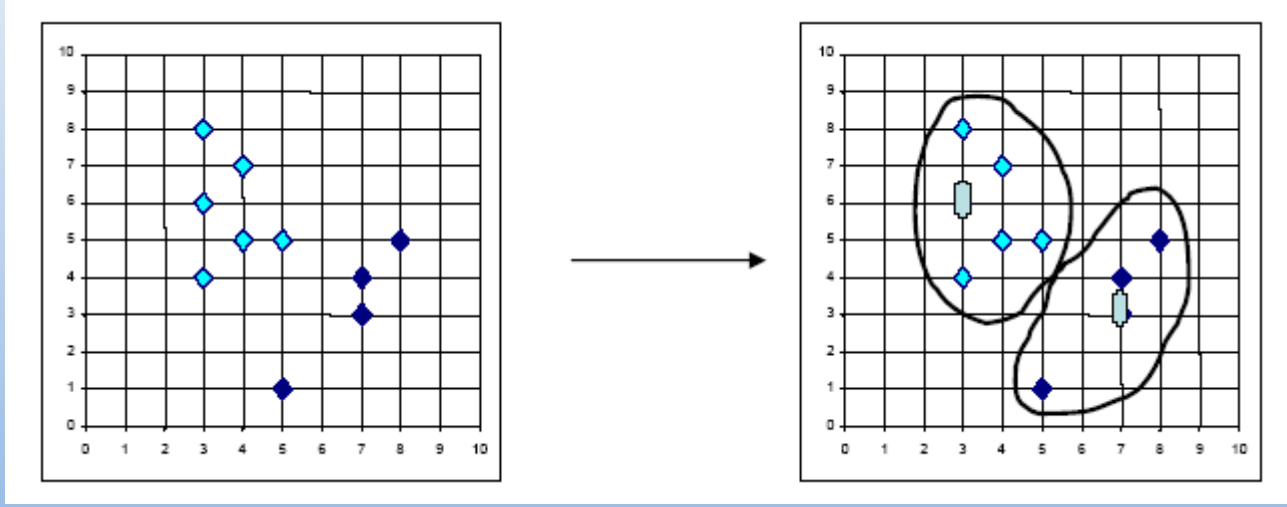
- Data Compression
  - Vector quantization: construct a codebook using *k*-means
    - Cluster means as prototypes representing examples assigned to clusters



k-means: Image Segmentation

What is the problem with the *k*-means Method?

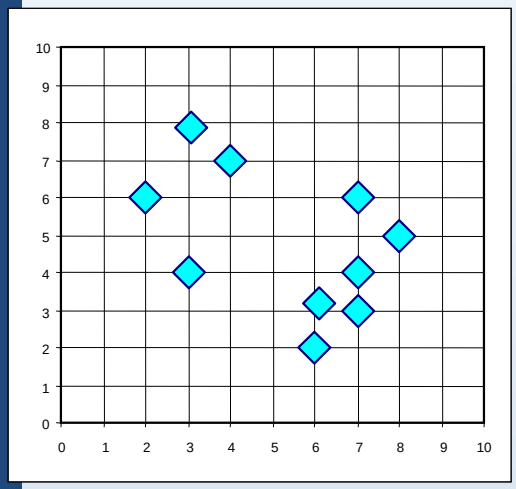
- The *k*-means algorithm is sensitive to outliers !
- Since an object with an extremely large value may substantially distort the distribution of the data.
- ***k*-Medoids:** Instead of taking the mean value of the object in a cluster as a reference point, medoids can be used, which is the most centrally located object in a cluster (i.e., median).



The *k*-medoids clustering method

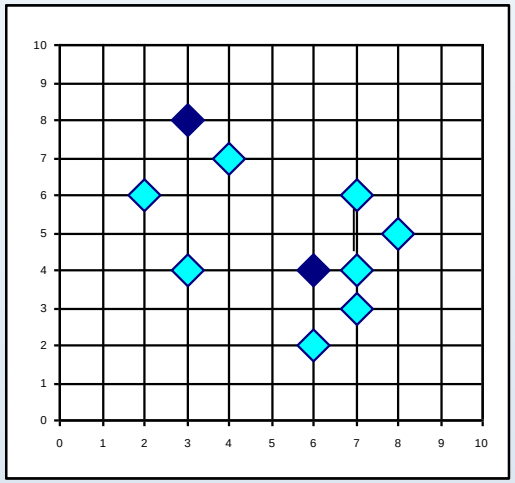
- Find *representative* objects, called medoids, in clusters
- ***PAM*** (Partitioning Around Medoids, 1987)
  - starts from an initial set of medoids and iteratively replaces one of the medoids by one of the non-medoids if it improves the total distance of the resulting clustering
  - *PAM* works effectively for small data sets, but does not scale well for large data sets

PAM: A Typical *k*-Medoids Algorithm



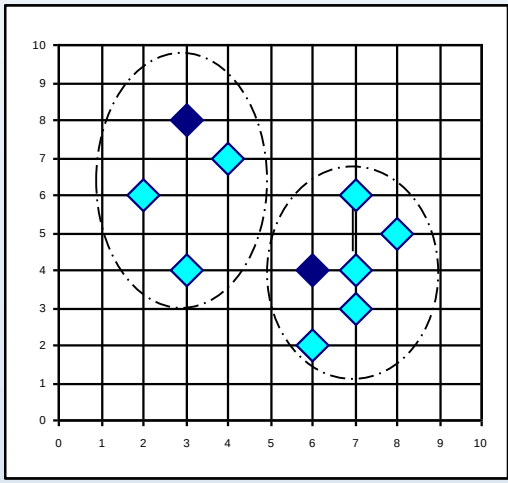
k=2

Arbitrary choose k object as initial medoids

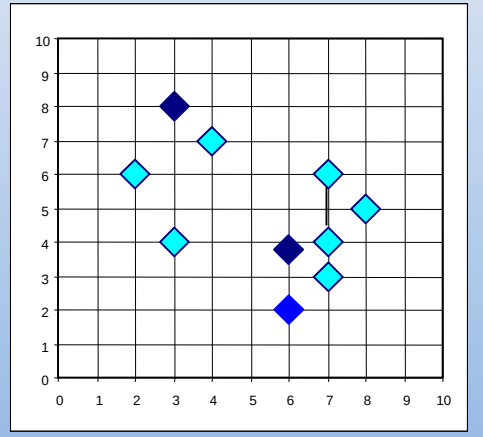


Assign each remaining object to nearest medoids

Total Cost = 20

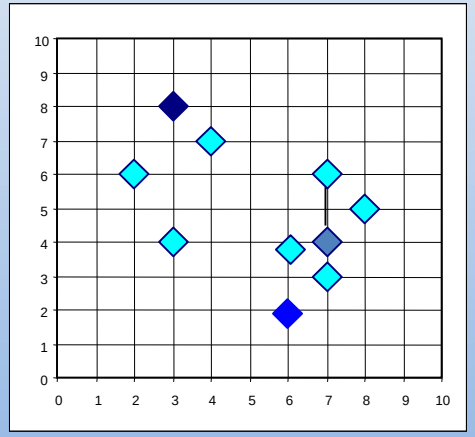


Randomly select a nonmedoid object, O_{ramdom}



Compute total cost of swapping

Total Cost = 26



Swapping O and O_{ramdom}

If quality is improved.

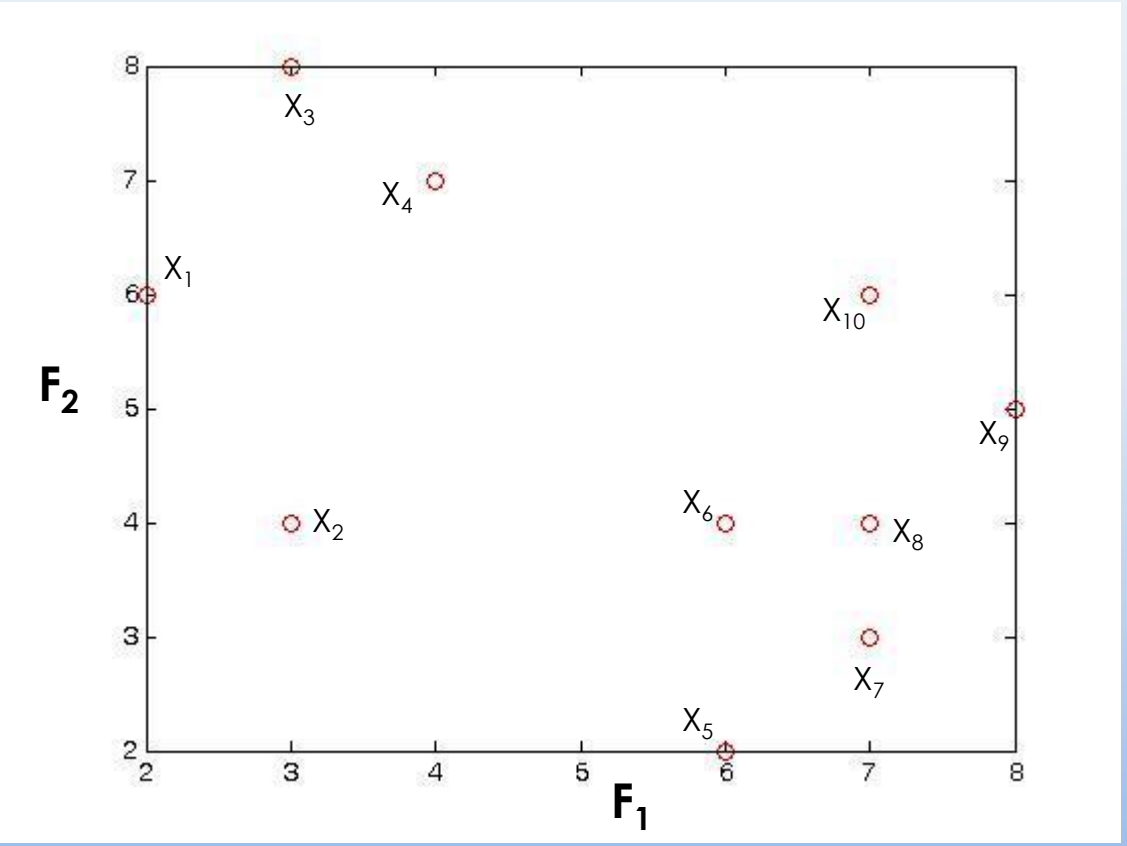
Do loop

Until no change

Example: k-medoids

- Cluster the following data set of ten objects into two clusters i.e., $k = 2$.

	F_1	F_2
X_1	2	6
X_2	3	4
X_3	3	8
X_4	4	7
X_5	6	2
X_6	6	4
X_7	7	3
X_8	7	4
X_9	8	5
X_{10}	7	6



distribution of the data

Example: k-medoids

- 1st step: Initialize *k* centers.
  - Let us assume X_2 and X_8 are selected as medoids, so the centers are $c_1 = (3,4)$ and $c_2 = (7,4)$
  - Calculate distances to each center so as to associate each data object to its nearest medoid. Cost is calculated using Manhattan distance (Minkowski distance metric with $r = 1$).

Example: k-medoids

Costs to the nearest medoid are shown bold in the table

Cost (distance) to c_1					
i	c_1		Data objects (X_i)		Cost (distance)
X_1	3	4	2	6	**3**
X_3	3	4	3	8	**4**
X_4	3	4	4	7	**4**
X_5	3	4	6	2	5
X_6	3	4	6	4	3
X_7	3	4	7	3	5
X_9	3	4	8	5	6
X_{10}	3	4	7	6	6

Cost (distance) to c_2					
i	c_2		Data objects (X_i)		Cost (distance)
X_1	7	4	2	6	7
X_3	7	4	3	8	8
X_4	7	4	4	7	6
X_5	7	4	6	2	**3**
X_6	7	4	6	4	**1**
X_7	7	4	7	3	**1**
X_9	7	4	8	5	**2**
X_{10}	7	4	7	6	**2**

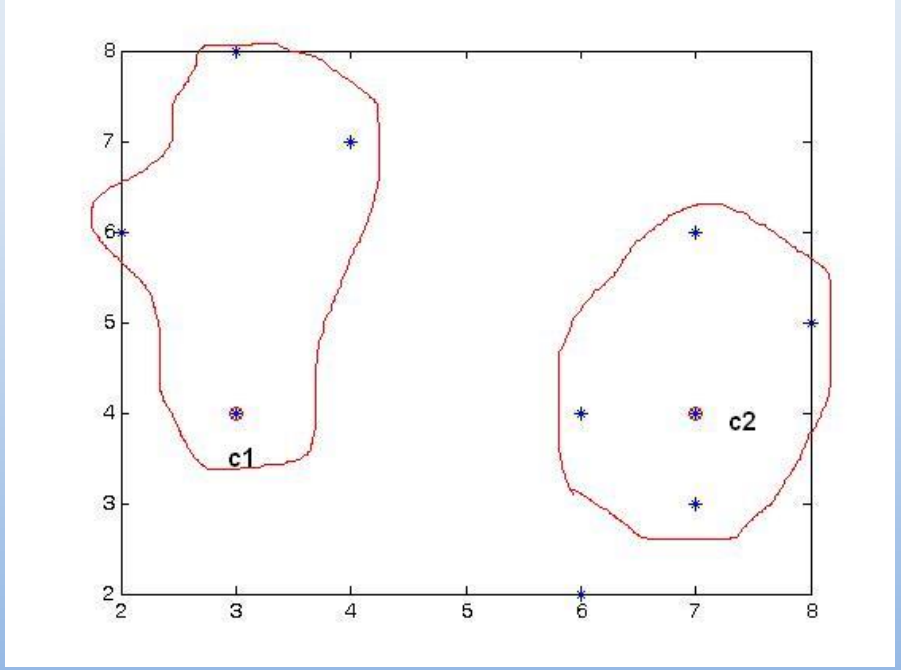
cost between any two points is found using formula

$$Cost(x, c) = \sum_{i=1}^{d} |x_i - c_j|$$

Example: k-medoids

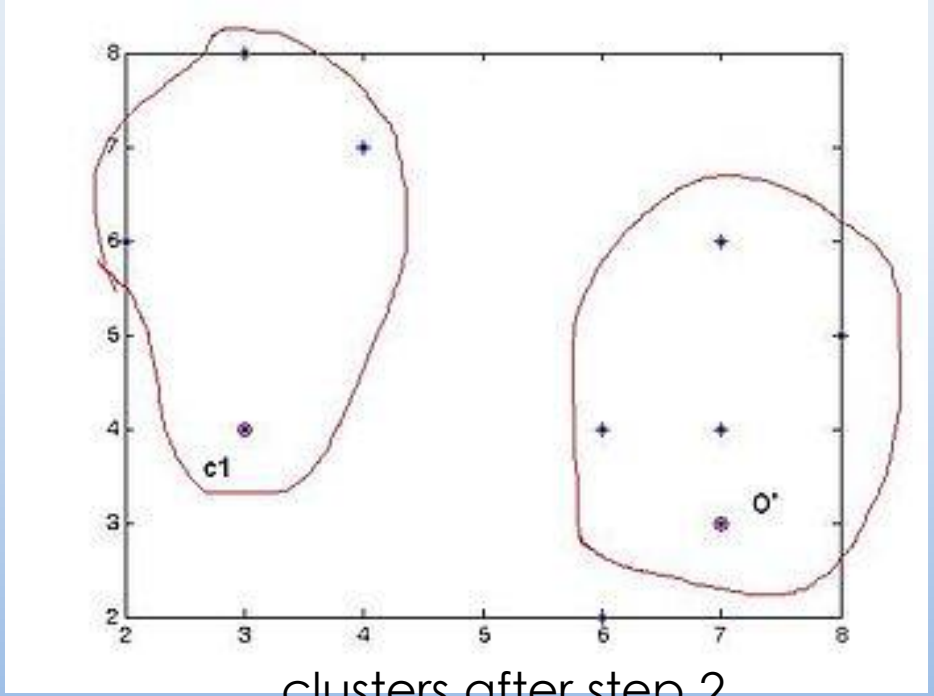
- Since the points (2,6) (3,8) and (4,7) are closer to c_1 hence they form one cluster whilst remaining points form another cluster.
- Then the clusters become:
  - $Cluster_1 = \{(3,4)(2,6)(3,8)(4,7)\}$
  - $Cluster_2 = \{(7,4)(6,2)(6,4)(7,3)(8,5)(7,6)\}$

clusters after step 1



Example: k-medoids

- 2nd step: Select one of the non-medoids O′
  - Let us assume O′ = (7,3), i.e. x_7.
  - So now the medoids are c_1(3,4) and O′(7,3)
  - If c_1 and O′ are new medoids, calculate the total cost involved



clusters after step 2

Example: k-medoids

- By using the formula in the step 1

i	c_1		Data objects (X_i)		Cost (distance)
1	3	4	2	6	**3**
3	3	4	3	8	**4**
4	3	4	4	7	**4**
5	3	4	6	2	5
6	3	4	6	4	3
8	3	4	7	4	4
9	3	4	8	5	6
10	3	4	7	6	6

i	O'		Data objects (X_i)		Cost (distance)
1	7	3	2	6	8
3	7	3	3	8	9
4	7	3	4	7	7
5	7	3	6	2	**2**
6	7	3	6	4	**2**
8	7	3	7	4	**1**
9	7	3	8	5	**3**
10	7	3	7	6	**3**

- Total cost=3+4+4+2+2+1+3+3=22
- So cost of swapping medoid from c_2 to O′ is

S=current cost – past cost

= 22 – 20

= 2>0

Example: k-medoids

- So cost of swapping medoid from c_2 to O′ is

  S=current cost – past cost

  = 22 – 20

  = 2>0

- So moving to O′ would be a bad idea, so the previous choice was good. So we try other non-medoids and found that our first choice was the best. So the configuration does not change and algorithm terminates here (i.e. there is no change in the medoids).
- It may happen some data points may shift from one cluster to another cluster depending upon their closeness to medoid.

Hierarchical Clustering

- Notion of a cluster can be ambiguous?
- How many clusters?
- **Hierarchical Clustering**: Clusters contain sub-clusters and clusters themselves can have sub-sub-clusters, and so on
  - Several levels of details in clustering
- A hierarchy might be more natural.
  - Different levels of granularity

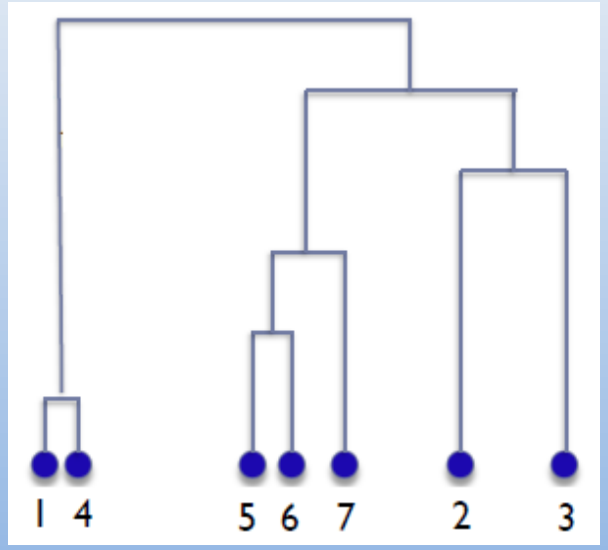
Hierarchical Clustering

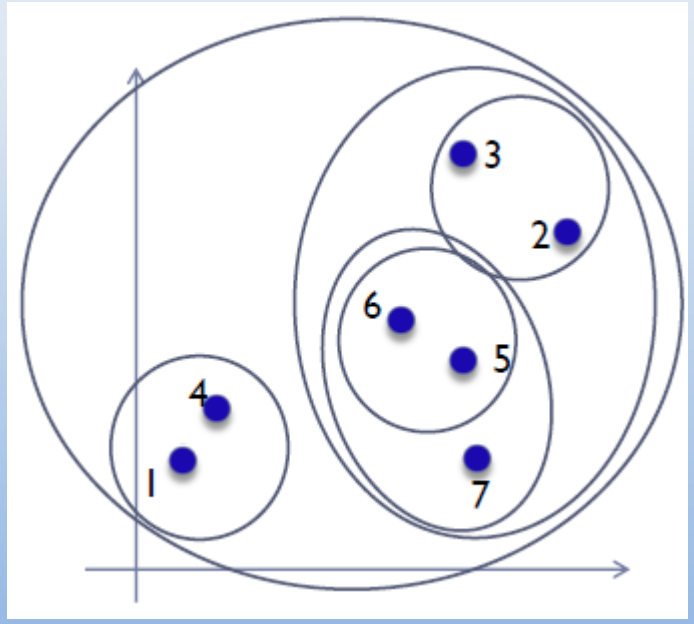
- **Agglomerative** (bottom up): merge clusters iteratively.
  - start by placing each object in its own cluster.
  - merge these atomic clusters into larger and larger clusters.
  - until all objects are in a single cluster.
  - Most hierarchical methods belong to this category. They differ only in their definition of *between-cluster similarity*.
- **Divisive** (top down): split a cluster iteratively.
  - Starts with the whole data as a cluster
  - Repeatedly divide data in one of the clusters until there is only one data in each cluster (or other stopping criteria).

Hierarchical Agglomerative Clustering (HAC)

- Algorithm
  1. Maintain a set of clusters
  2. Initially, each instance forms a cluster
  3. While there are more than one cluster
     - 3.1. Pick the two closest one
     - 3.2. Merge them into a new cluster

Height represents the distance at which the merge occurs

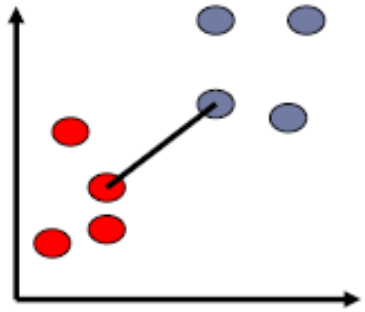




Distances between Cluster Pairs

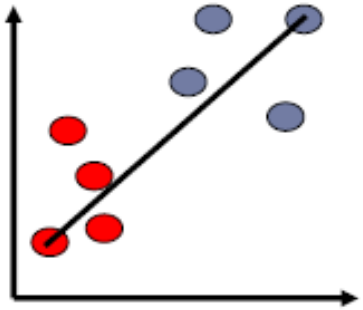
- Many variants to defining distances between pair of clusters
  - Single-link
    - Minimum distance between different pairs of data
  - Complete-link
    - Maximum distance between different pairs of data
  - Average-link
    - Average distance between pairs of elements
  - Centroid (Ward's)
    - Distance between centroids (centers of gravity)

Distances between Cluster Pairs



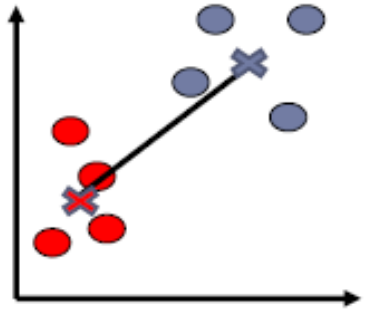
Single-link

$$dist_{SL}(\mathcal{C}_i, \mathcal{C}_j) = \min_{\boldsymbol{x} \in \mathcal{C}_i, \boldsymbol{x}' \in \mathcal{C}_j} dist(\boldsymbol{x}, \boldsymbol{x}')$$



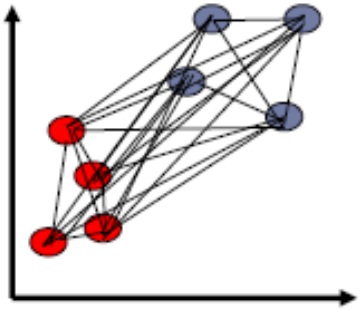
Complete-link

$$dist_{CL}(\mathcal{C}_i, \mathcal{C}_j) = \max_{\boldsymbol{x} \in \mathcal{C}_i, \boldsymbol{x}' \in \mathcal{C}_j} dist(\boldsymbol{x}, \boldsymbol{x}')$$



Ward's

$$dist_{Ward}(\mathcal{C}_i, \mathcal{C}_j) = \frac{|\mathcal{C}_i||\mathcal{C}_j|}{|\mathcal{C}_i| + |\mathcal{C}_j|} dist(\boldsymbol{c}_i, \boldsymbol{c}_j)$$

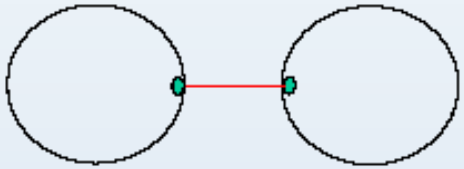


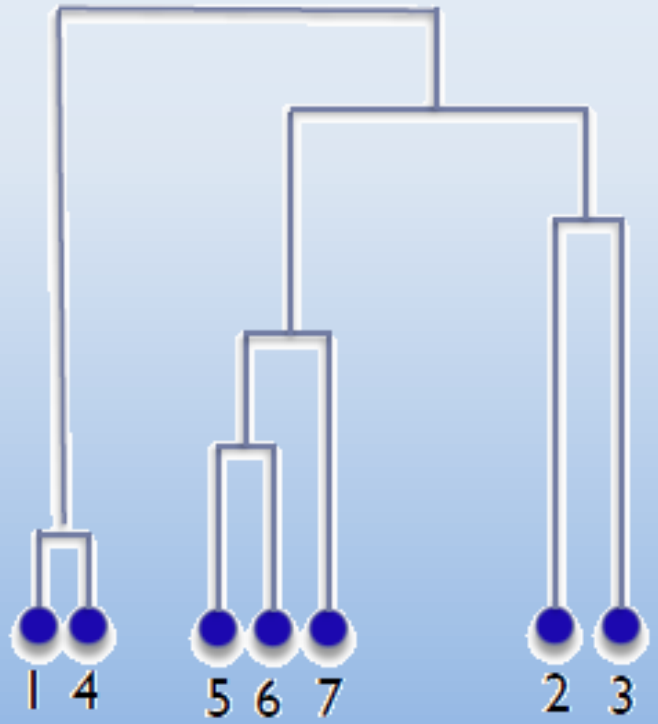
Average-link

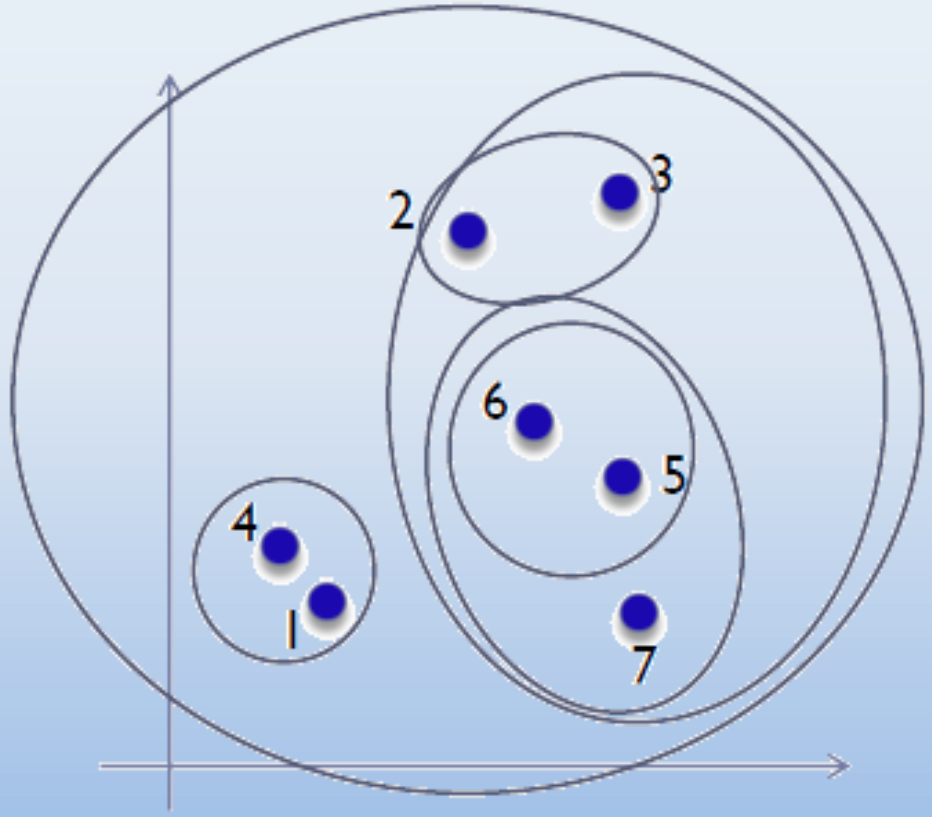
$$dist_{AL}(\mathcal{C}_i, \mathcal{C}_j) = \frac{1}{|\mathcal{C}_i \cup \mathcal{C}_j|} \sum_{\boldsymbol{x} \in \mathcal{C}_i \cup \mathcal{C}_j} \sum_{\boldsymbol{x}' \in \mathcal{C}_i \cup \mathcal{C}_j} dist(\boldsymbol{x}, \boldsymbol{x}'$$

Single-Link

- Minimum distance between different pairs of data
  - $dist(K_i, K_j) = min(t_{ip}, t_{jq})$



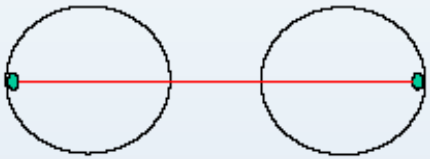


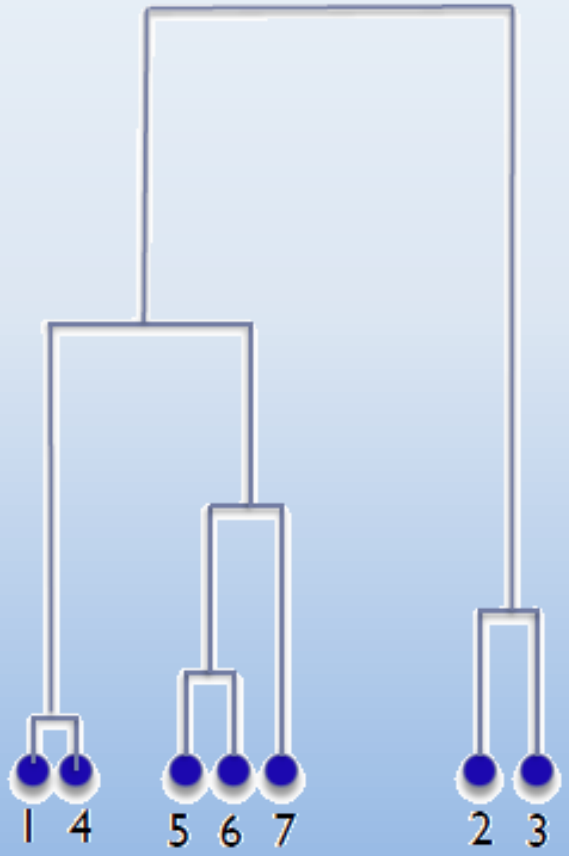


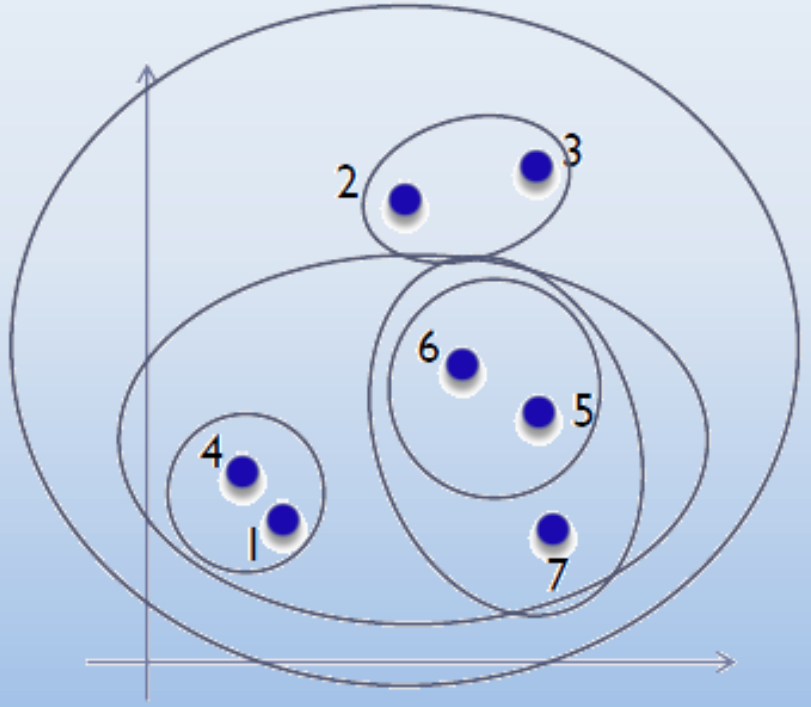
keep max bridge length as small as possible.

Complete Link

- Maximum distance between different pairs of data
  - $dist(K_i, K_j) = max(t_{ip}, t_{jq})$



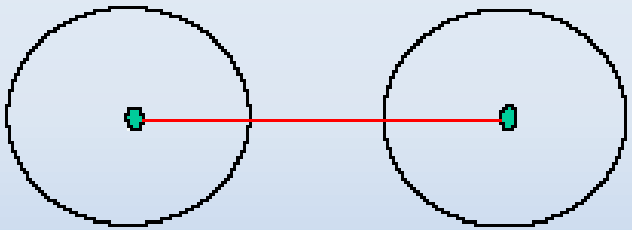




keep max diameter as small as possible

Average link

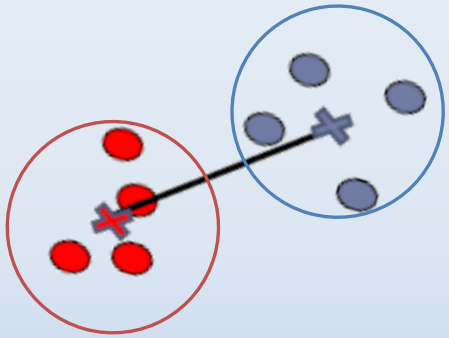
- In *average-link* clustering, the distance between one cluster and another cluster is equal to the average distance from any member of one cluster to any member of the other cluster.
- $dist(K_i, K_j) = avg(t_{ip}, t_{jq})$



Ward's method (Centroid)

- Distance between centroids (centers of gravity)
- The distances between centers of the two clusters (weighted to consider sizes of clusters too)

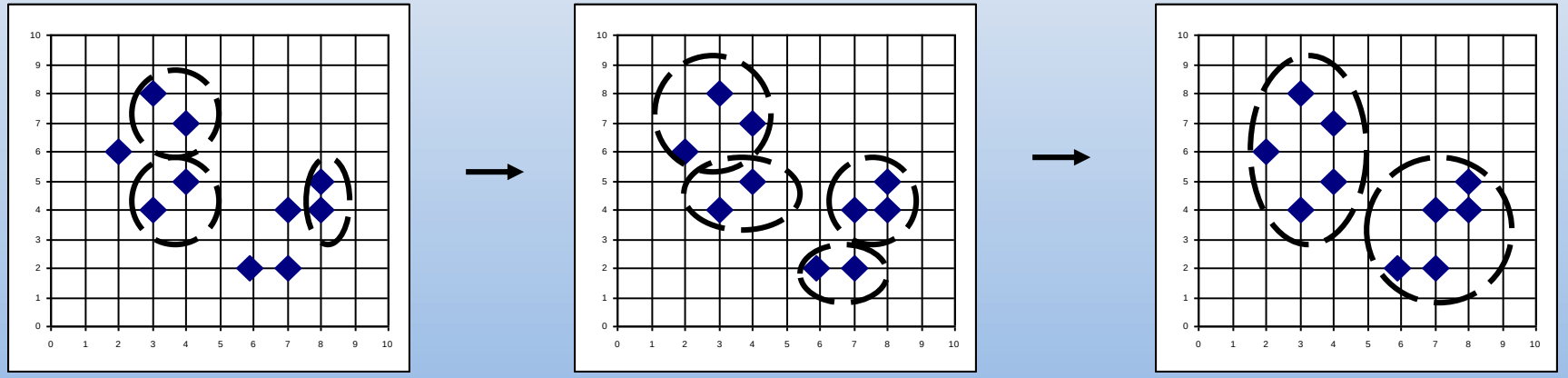
$$dist_{Ward}(\mathcal{C}_i, \mathcal{C}_j) = \frac{|\mathcal{C}_i||\mathcal{C}_j|}{|\mathcal{C}_i| + |\mathcal{C}_j|} dist(\boldsymbol{c}_i, \boldsymbol{c}_j)$$



- Merge the two clusters such that the increase in k-means cost is as small as possible.
- Works well in practice.

AGNES (Agglomerative Nesting)

- Introduced in Kaufmann and Rousseeuw (1990)
- Use the **single-link** method and the dissimilarity matrix
- Initially each object is a cluster
- Merge nodes that have the least dissimilarity
- at the end eventually all nodes belong to the same cluster



Distances between Clusters: Summary

- Which distance is the best?
  - Complete linkage prefers compact clusters.
  - Single linkage can produce long stretched clusters.
- The choice depends on what you need.
  - expert opinion is helpful

Data Matrix vs. Distance Matrix

- Data (or pattern) Matrix: $N \times d$ (features of data):

$$\boldsymbol{X} = \begin{bmatrix} x_1^{(1)} & \cdots & x_d^{(1)} \\ \vdots & \ddots & \vdots \\ x_1^{(N)} & \cdots & x_d^{(N)} \end{bmatrix}$$

- Distance Matrix: $N \times N$ (distances of each pattern pair

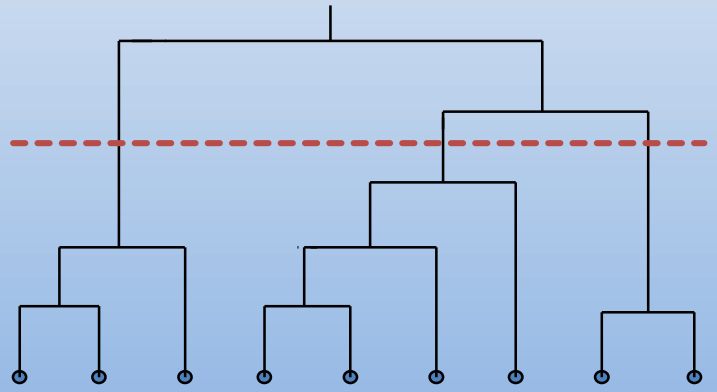
$$\boldsymbol{D} = \begin{bmatrix} d(\boldsymbol{x}^{(1)}, \boldsymbol{x}^{(1)}) & \cdots & d(\boldsymbol{x}^{(1)}, \boldsymbol{x}^{(N)}) \\ \vdots & \ddots & \vdots \\ d(\boldsymbol{x}^{(N)}, \boldsymbol{x}^{(1)}) & \cdots & d(\boldsymbol{x}^{(N)}, \boldsymbol{x}^{(N)}) \end{bmatrix}$$

  - Single-link, complete-link, and average link only needs the distance matrix

Dendrogram: Hierarchical Clustering

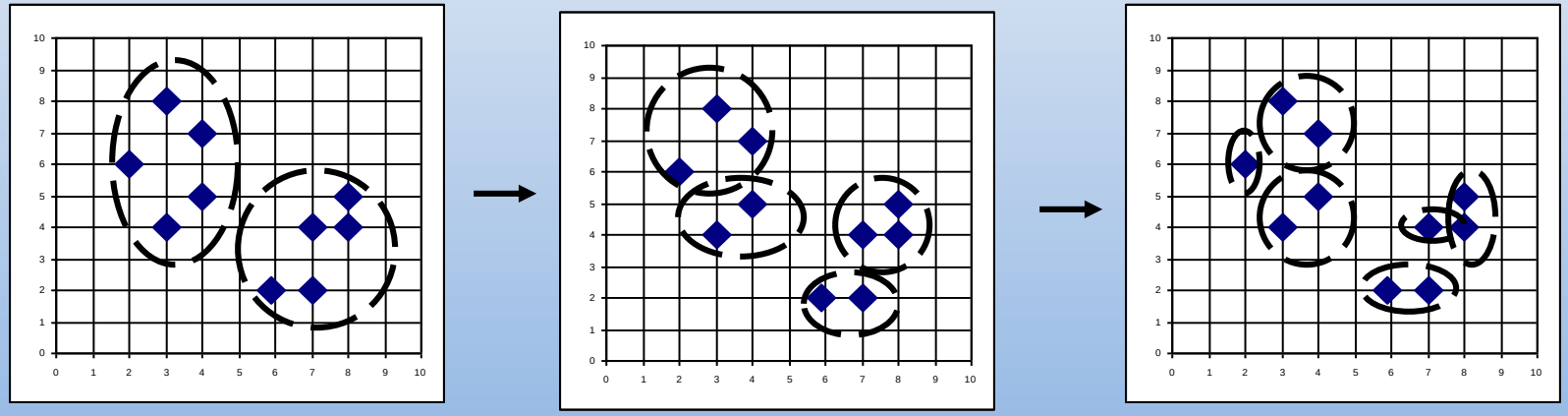
- Decompose data objects into a several levels of nested partitioning (tree of clusters), called a **dendrogram**
- Clustering obtained by cutting the dendrogram at a desired level, then each connected component forms a cluster
  - Cut at a pre-specified level of similarity
  - where the gap between two successive combination similarities is largest
  - select the cutting point that produces K clusters

Where to "cut" the dendrogram is user-determined.



DIANA (Divisive Analysis)

- Introduced in Kaufmann and Rousseeuw (1990)
- Implemented in statistical analysis packages, e.g., Splus
- Inverse order of AGNES
- Eventually each node forms a cluster on its own



Example: Agglomerative clustering with single-link

- Distance matrix

1.

	a	b	c	d	e	f
a	0	4	13	24	12	8
b		0	10	22	11	10
c			0	7	3	9
d				0	6	18
e					0	8.5
f						0

2.

	a	b	{c,e}	d	f
a	0	4	12	24	8
b		0	10	22	10
{c,e}			0	6	8.5
d				0	18
f					0

Example: Agglomerative clustering with single-link

3.

	{a,b}	{c,e}	d	f
{a,b}	0	10	22	8
{c,e}		0	6	8.5
d			0	18
f				0

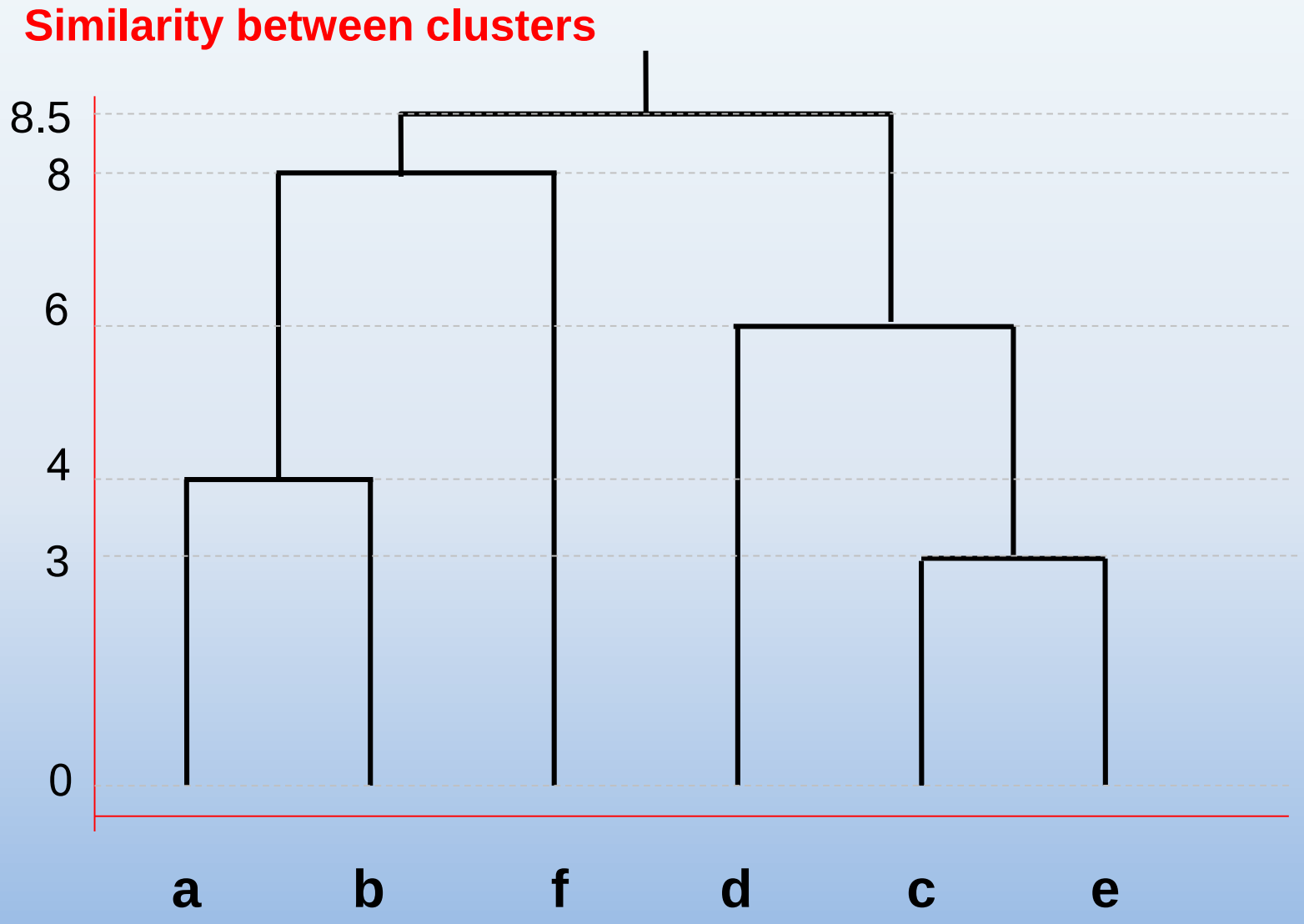
4.

	{a,b}	{c,e,d}	f
{a,b}	0	10	8
{c,e,d}		0	8.5
f			0

5.

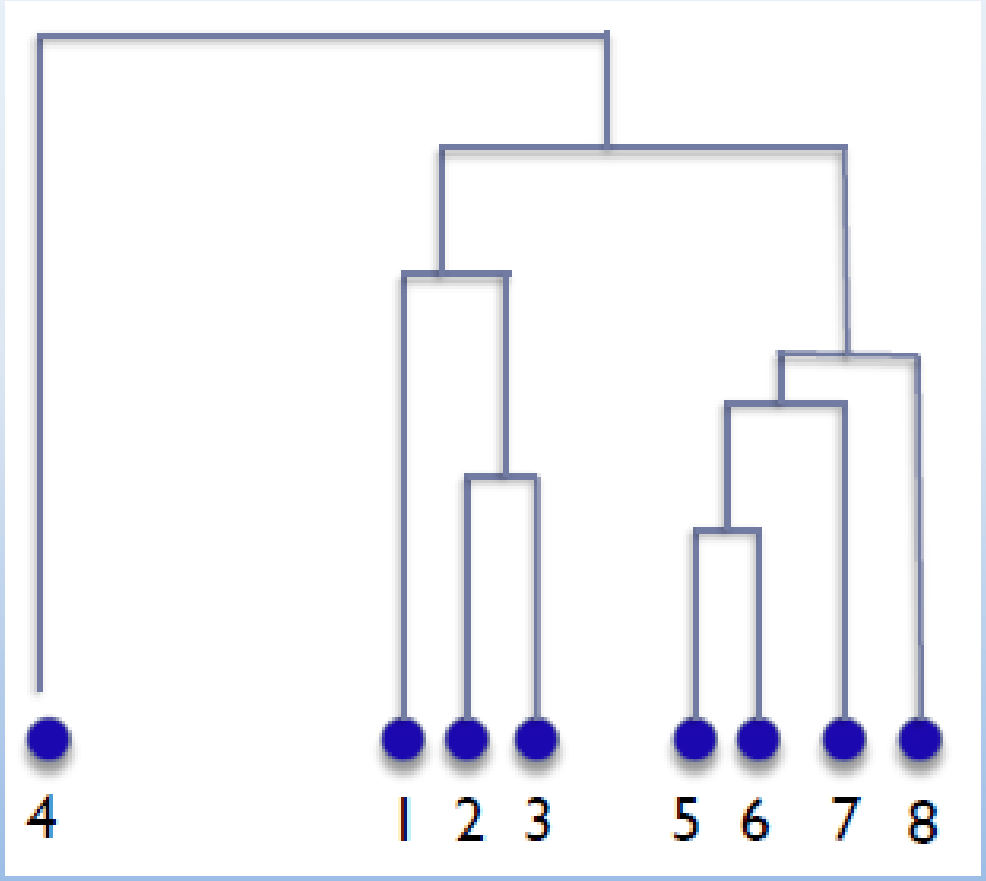
	{a,b,f}	{c,e,d}
{a,b,f}	0	8.5
{c,e,d}		0

Example: Dendrogram



Outliers

- We can detect outliers (that are very different to all others) by finding the isolated branches



k-means vs. Hierarchical

- Time cost:
  - *k*-means is usually fast while hierarchical methods do not scale well
- Human intuition
  - Hierarchical structure provides more natural output compatible with human intuition in some domains
- Local minimum problem
  - It is very common for *k*-means
  - Hierarchical methods like any heuristic search algorithms also suffer from local optima problem.
    - Since they can never undo what was done previously and greedily merge clusters
- Choosing of the number of clusters
  - There is no need to specify the number of clusters in advance for hierarchical methods

Clustering Validity

- We need to determine whether the found clusters are real or compare different clustering methods.
- What is a good clustering?
  - clustering quality measurement
- Main approaches:
  - **Internal index**: evaluate how well the clustering fit the data without reference to an external information.
    - Methods: Dunn index, Davies–Bouldin, Silhouette coefficient
  - **External index**: evaluate how well is the clustering result with respect to known categories.
    - Assumption: Ground truth labels are available
    - Methods: Rand measure, F-measure, Jaccard index, Fowlkes–Mallows index, Confusion matrix

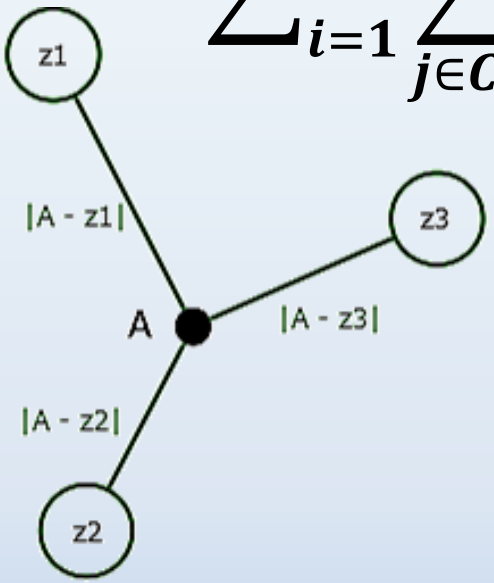
Clustering Error (↓) – SSE, MSE

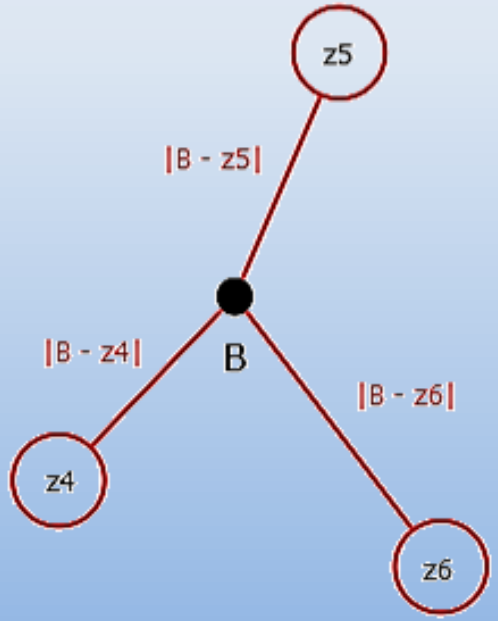
Sum of Squared Error (SSE)

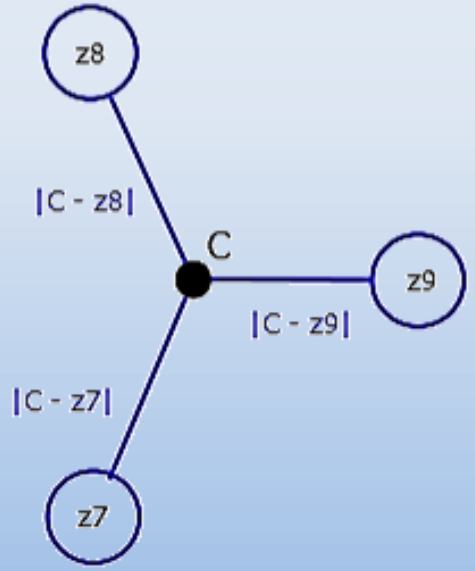
$$\boldsymbol{SSE} = \sum_{\boldsymbol{i=1}}^{\boldsymbol{k}} \sum_{\boldsymbol{j \in C_i}} (\boldsymbol{c_i} - \boldsymbol{x_j})$$

$$\mathrm{MSE} = \frac{1}{|\mathrm{k}|} \Sigma_{\mathrm{i=1}}^{\mathrm{k}} \Sigma_{\mathrm{j} \in \mathrm{C_i}} (\mathrm{j} - \mathrm{c_i})^2$$

: [|A - z1|^2 + |A - z2|^2 + |A - z3|^2 +
|B - z4|^2 + |B - z5|^2 + |B - z6|^2 +
|C - z7|^2 + |C - z8|^2 + |C - z9|^2] / |Z|







Davies-Bouldin index (DB ↓)

- A function of the ratio of the sum of within-cluster (i.e. intra-cluster) scatter to between cluster (i.e. inter-cluster) separation
  - Let $C=\{C_1, \ldots, C_k\}$ be a clustering of a set of N objects:

$$DB = \frac{1}{k} . \sum_{i=1}^{k} R_i$$

with $R_i = \max_{j=1,..k, i \neq j} R_{ij}$ and $R_{i,j} = \frac{S_i + S_j}{M_{i,j}}$

$$M_{i,j} = \|A_i - A_j\|_p = \left(\sum_{k=1}^{n} |a_{k,i} - a_{k,j}|^p \right)^{\frac{1}{p}}$$

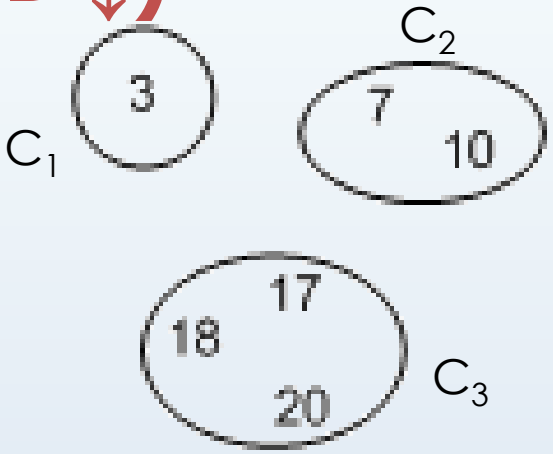
where C_i is the i^{th} cluster and c_i is the centroid for cluster i

Davies-Bouldin index (DB ↓)

for the clusters shown $R_{\substack{ij \\ i\neq j}} = \frac{s^2(C_i)+s^2(C_j)}{\|c_i - c_j\|}$

Compute

$$s^2 = \frac{1}{n-1}\sum_{i=1}^{n}(y_i - \overline{y})^2$$



$s^2(C_1)=0$, $s^2(C_2)=4.5$, $s^2(C_3)=2.33$

Centroid is simply the mean here, so $c_1=3$, $c_2=8.5$, $c_3=18.33$

So, $R_{12}=0.81$, $R_{13}=0.152$, $R_{23}=0.694$

Now, compute $R_i = \max_{j=1,..k, i\neq j} R_{ij}$

$R_1=0.81$ (max of R_{12} and R_{13});

$R_2=0.81$ (max of R_{21} and R_{23});

$R_3=0.69$ (max of R_{31} and R_{32})

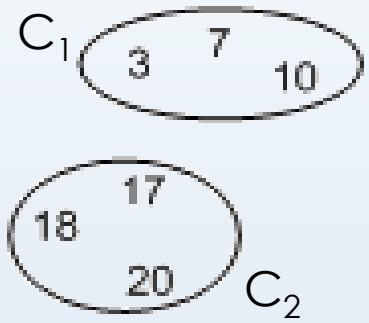
Finally, compute

$DB=0.77$

$$DB = \frac{1}{k}.\sum_{i=1}^{k} R_i$$

Davies-Bouldin index (DB ↓)

Example 2: For the clusters shown



Compute $R_{\substack{ij \\ i\neq j}} = \frac{s^2(C_i)+s^2(C_j)}{\| c_i - c_j \|}$ or $R_{\substack{ij \\ i\neq j}} = \frac{\mathrm{var}(C_i)+\mathrm{var}(C_j)}{\| c_i - c_j \|}$

Only 2 clusters here

Centroid is simply the mean here, so c_1=6.67, c_2=18.33

$s^2(C_1)$=8.22 , $s^2(C_2)$=2.33

$R_{12}=R_{21}$=1.26

Now compute

Since we have only 2 clusters here,

$R_1=R_{12}$=1.26 $\quad R_i = \max_{j=1,..k, i\neq j} R_{ij}$

$R_2=R_{21}$=1.26

Finally, compute $\quad DB = \frac{1}{k}.\sum_{i=1}^{k} R_i$

DB=1.26

Dunn index (D ↑)

- The Dunn index aims to identify **dense and well-separated clusters**. It is defined as the ratio between the minimal inter-cluster distance to maximal intra-cluster distance. For each cluster partition, the Dunn index can be calculated by the following formula

$$D = \frac{d_{\min}}{d_{\max}}$$

Min: Distance between 2 data (inter-cluster)

Max: Distance within 2 data (intra-cluster)

$0<D<\infty$

$$D = \min_{i=1\ldots n_c} \left\{ \min_{j=i+1\ldots n_c} \left(\frac{d(c_i, c_j)}{\max_{k=1\ldots n_c} (diam(c_k))} \right) \right\}$$

$$\mathrm{D}(C) = \frac{\min_{c_k \in C}\{\min_{c_l \in C \setminus c_k}\{\delta(c_k, c_l)\}\}}{\max_{c_k \in C}\{\Delta(c_k)\}},$$

where

$$\delta(c_k, c_l) = \min_{x_i \in c_k} \min_{x_j \in c_l}\{\mathrm{d_e}(x_i, x_j)\},$$

$$\Delta(c_k) = \max_{x_i, x_j \in c_k}\{\mathrm{d_e}(x_i, x_j)\}.$$

$$diam(c_i) = \max_{x, y \in c_i}\{d(x, y)\}$$

$$d(c_i, c_j) = \min_{x \in c_i, y \in c_j}\{d(x, y)\}$$

Dunn index (D ↑)

- Example

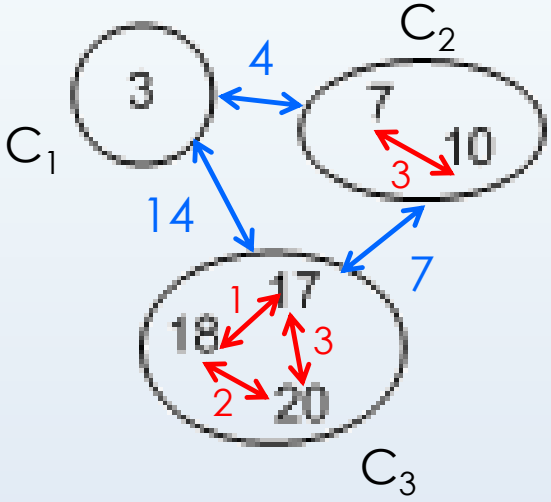
$d(C_1, C_2)=4$, $d(C_1, C_3)=14$ $d(C_2, C_3)=7$

$diam(C_1)=0$, $diam(C_2)=3$, $diam(C_3)=3$,

d_{min}: Min{4, 7, 14}

d_{max}: Max{0, 3, 3}

$$D = \frac{d_{\min}}{d_{\max}} = \frac{4}{3}$$



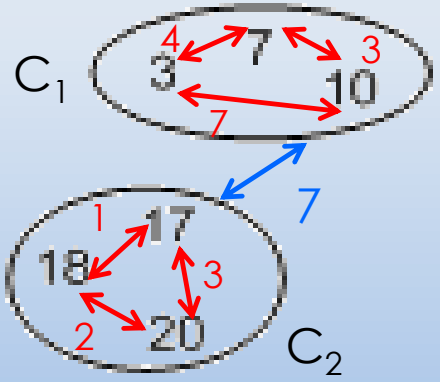
$d(C_1, C_2)=7$

$diam(C_1)=7$, $diam(C_2)=3$

d_{min}: Min{7}

d_{max}: Max{3, 7}

$$D = \frac{d_{\min}}{d_{\max}} = \frac{7}{7}$$



External Index: Rand Index and Clustering F-measure

- $RI = \frac{TP+TN}{TP+TN+FP+FN}$
- $P = \frac{TP}{TP+FP}, R = \frac{TP}{TP+FN}$
- $F_\beta = \frac{(\beta^2+1)PR}{\beta^2 P+R}$
  - *F measure* in addition supports differential weighting of P and R.
- $Jaccard = \frac{TP}{TP+FP+FN}$

TP: # pairs that cluster together in both $\mathcal{C}$ and $\hat{\mathcal{C}}$
TN: # pairs that are in separate clusters in both $\mathcal{C}$ and $\hat{\mathcal{C}}$
FN: # pairs that cluster together in $\mathcal{C}$ but not in $\hat{\mathcal{C}}$
FP: # pairs that cluster together in $\hat{\mathcal{C}}$ but not in $\mathcal{C}$

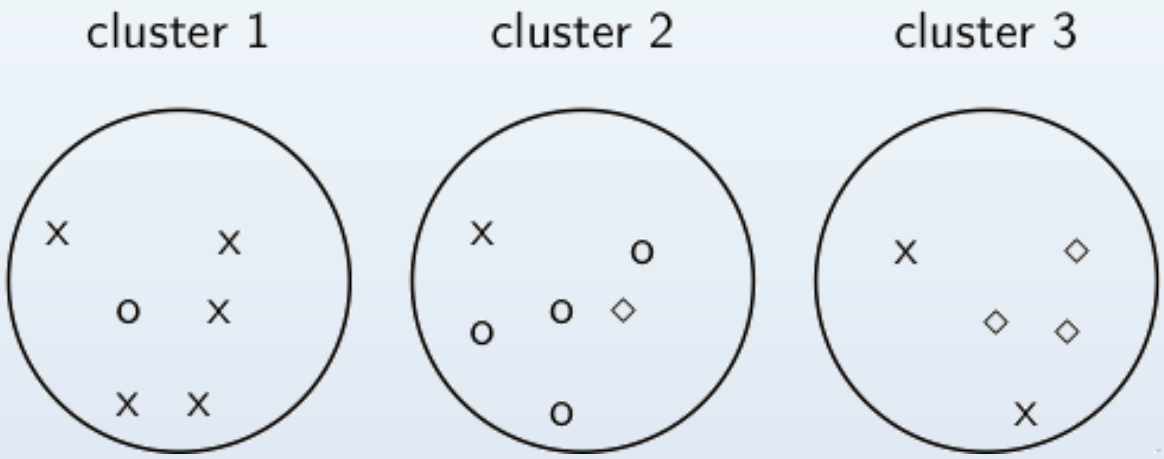
$\mathcal{C}$ \ $\hat{\mathcal{C}}$	Same	Different
Same	TP	FN
Different	FP	TN

Purity (↑)

$$\text{purity}(\Omega, \mathbb{C}) = \frac{1}{N} \sum_k \max_j |\omega_k \cap c_j|$$

- $\Omega = \{\omega_1, \omega_2, \ldots, \omega_K\}$ is the set of clusters and
- $C = \{c_1, c_2, \ldots, c_J\}$ is the set of classes.
- For each cluster ω_k : find class c_j with most members n_{kj} in ω_k
- Sum all n_{kj} and divide by total number of points

Purity (↑)



(class x, cluster 1) $\rightarrow \max_j |\omega_1 \cap c_j| = 5$

(class o, cluster 2) $\rightarrow \max_j |\omega_2 \cap c_j| = 4$

(class ◊, cluster 3) $\rightarrow \max_j |\omega_3 \cap c_j| = 3$

Purity is

$(1/17) \times (5 + 4 + 3) \approx 0.71$

Rand index

- Definition:

$$RI = \frac{TP + TN}{TP + FP + FN + TN}$$

- TP is the number of true positives
- TN is the number of true negatives
- FP is the number of false positives
- FN is the number of false negatives

- TP+FN+FP+TN is the total number of pairs.

Jaccard index

- The Jaccard index is used to quantify the similarity between two datasets. The Jaccard index takes on a value between 0 and 1. An index of 1 means that the two dataset are identical, and an index of 0 indicates that the datasets have no common elements. The Jaccard index is defined by the following formula:

$$J(A,B) = \frac{|A \cap B|}{|A \cup B|} = \frac{TP}{TP + FP + FN}$$

- This is simply the number of unique elements common to both sets divided by the total number of unique elements in both sets.

Fowlkes–Mallows index

- The Fowlkes-Mallows index computes the similarity between the clusters returned by the clustering algorithm and the benchmark classifications. The higher the value of the Fowlkes-Mallows index the more similar the clusters and the benchmark classifications are. It can be computed using the following formula:

$$FM = \sqrt{\frac{TP}{TP+FP} \cdot \frac{TP}{TP+FN}}$$

- The index is the geometric mean of the precision and recall and , while the F-measure is their harmonic mean

Major Dilemma [Jain, 2010]

- What is a cluster?
- What features should be used?
- Should the data be normalized?
- How do we define the pair-wise similarity?
- Which clustering method should be used?
- How many clusters are present in the data?
- Does the data contain any outliers?
- Does the data have any clustering tendency?
- Are the discovered clusters and partition valid?

Reading

- E. Alpaydin, **Introduction to Machine Learning**, 4th ed., The MIT Press, 2020. (ch. 7)
- C. M. Bishop, **Pattern recognition and machine learning**, Springer, 2006. (ch. 9)