Data Mining:

Concepts and Techniques

(3rd ed.)

— Chapter 10 —

Jiawei Han, Micheline Kamber, and Jian Pei

University of Illinois at Urbana-Champaign &

Simon Fraser University

Edited by: Ali Kamandi

Major Clustering Approaches (I)

- Partitioning approach:
  - Construct various partitions and then evaluate them by some criterion, e.g., minimizing the sum of square errors
  - Typical methods: k-means, k-medoids, CLARANS
- Hierarchical approach:
  - Create a hierarchical decomposition of the set of data (or objects) using some criterion
  - Typical methods: Diana, Agnes, BIRCH, CAMELEON
- Density-based approach:
  - Based on connectivity and density functions
  - Typical methods: DBSACN, OPTICS, DenClue
- Grid-based approach:
  - based on a multiple-level granularity structure
  - Typical methods: STING, WaveCluster, CLIQUE

Similarity and Dissimilarity

- **Similarity**
  - Numerical measure of how alike two data objects are
  - Value is higher when objects are more alike
  - Often falls in the range [0,1]
- **Dissimilarity** (e.g., distance)
  - Numerical measure of how different two data objects are
  - Lower when objects are more alike
  - Minimum dissimilarity is often 0
  - Upper limit varies

Data Matrix and Dissimilarity Matrix

- Data matrix
  - n data points with p dimensions
  - Two modes

$$\begin{bmatrix} x_{11} & \cdots & x_{1f} & \cdots & x_{1p} \\ \cdots & \cdots & \cdots & \cdots & \cdots \\ x_{i1} & \cdots & x_{if} & \cdots & x_{ip} \\ \cdots & \cdots & \cdots & \cdots & \cdots \\ x_{n1} & \cdots & x_{nf} & \cdots & x_{np} \end{bmatrix}$$

- Dissimilarity matrix
  - n data points, but registers only the distance
  - A triangular matrix

$$\begin{bmatrix} 0 & & & & \\ d(2,1) & 0 & & & \\ d(3,1) & d(3,2) & 0 & & \\ : & : & : & & \\ d(n,1) & d(n,2) & \ldots & \ldots & 0 \end{bmatrix}$$

Proximity Measure for Nominal Attributes

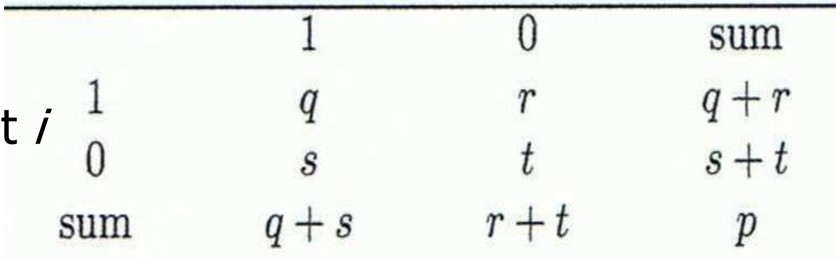
- Can take 2 or more states, e.g., red, yellow, blue, green (generalization of a binary attribute)
- Method 1: Simple matching
  - *m*: # of matches, *p*: total # of variables

$$d(i,j) = \frac{p-m}{p}$$

- Method 2: Use a large number of binary attributes
  - creating a new binary attribute for each of the *M* nominal states

Proximity Measure for Binary Attributes

- A contingency table for binary data



		Object j		
		1	0	sum
Object i	1	q	r	$q+r$
	0	s	t	$s+t$
	sum	$q+s$	$r+t$	p

- Distance measure for symmetric binary variables:

$$d(i, j) = \frac{r+s}{q+r+s+t}$$

- Distance measure for asymmetric binary variables:

$$d(i, j) = \frac{r+s}{q+r+s}$$

- Jaccard coefficient (*similarity* measure for *asymmetric* binary variables):

$$sim_{Jaccard}(i, j) = \frac{q}{q+r+s}$$

  - Note: Jaccard coefficient is the same as “coherence”:

$$coherence(i, j) = \frac{sup(i, j)}{sup(i) + sup(j) - sup(i, j)} = \frac{q}{(q+r) + (q+s) - q}$$

Dissimilarity between Binary Variables

- Example

Name	Gender	Fever	Cough	Test-1	Test-2	Test-3	Test-4
Jack	M	Y	N	P	N	N	N
Mary	F	Y	N	P	N	P	N
Jim	M	Y	P	N	N	N	N

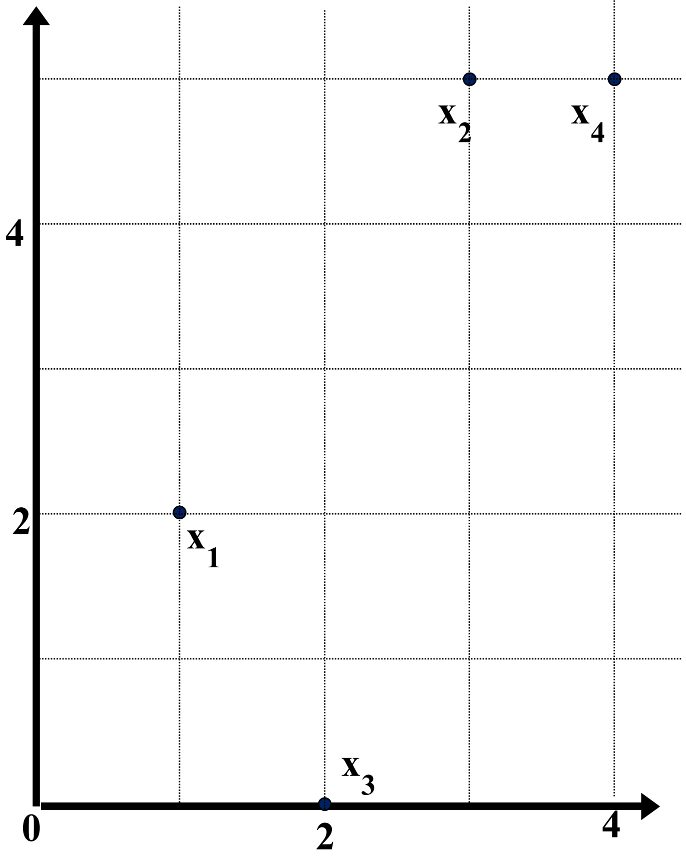
  - Gender is a symmetric attribute
  - The remaining attributes are asymmetric binary
  - Let the values Y and P be 1, and the value N 0

$$d(jack, mary) = \frac{0+1}{2+0+1} = 0.33$$

$$d(jack, jim) = \frac{1+1}{1+1+1} = 0.67$$

$$d(jim, mary) = \frac{1+2}{1+1+2} = 0.75$$

Example:
Data Matrix and Dissimilarity Matrix



Data Matrix

point	attribute1	attribute2
x1	1	2
x2	3	5
x3	2	0
x4	4	5

Dissimilarity Matrix

(with Euclidean Distance)

	x1	*x2*	*x3*	*x4*
x1	0			
x2	3.61	0		
x3	5.1	5.1	0	
x4	4.24	1	5.39	0

Distance on Numeric Data: Minkowski Distance

- *Minkowski distance*: A popular distance measure

$$d(i, j) = \sqrt[h]{|x_{i1} - x_{j1}|^h + |x_{i2} - x_{j2}|^h + \cdots + |x_{ip} - x_{jp}|^h}$$

where $i = (x_{i1}, x_{i2}, \ldots, x_{ip})$ and $j = (x_{j1}, x_{j2}, \ldots, x_{jp})$ are two p-dimensional data objects, and h is the order (the distance so defined is also called L-h norm)

- Properties
  - d(i, j) > 0 if i ≠ j, and d(i, i) = 0 (Positive definiteness)
  - d(i, j) = d(j, i) (Symmetry)
  - d(i, j) ≤ d(i, k) + d(k, j) (Triangle Inequality)
- A distance that satisfies these properties is a metric

Special Cases of Minkowski Distance

- h = 1: Manhattan (city block, L_1 norm) distance
  - E.g., the Hamming distance: the number of bits that are different between two binary vectors

$$d(i,j) = |x_{i_1} - x_{j_1}| + |x_{i_2} - x_{j_2}| + \ldots + |x_{i_p} - x_{j_p}|$$

- h = 2: (L_2 norm) Euclidean distance

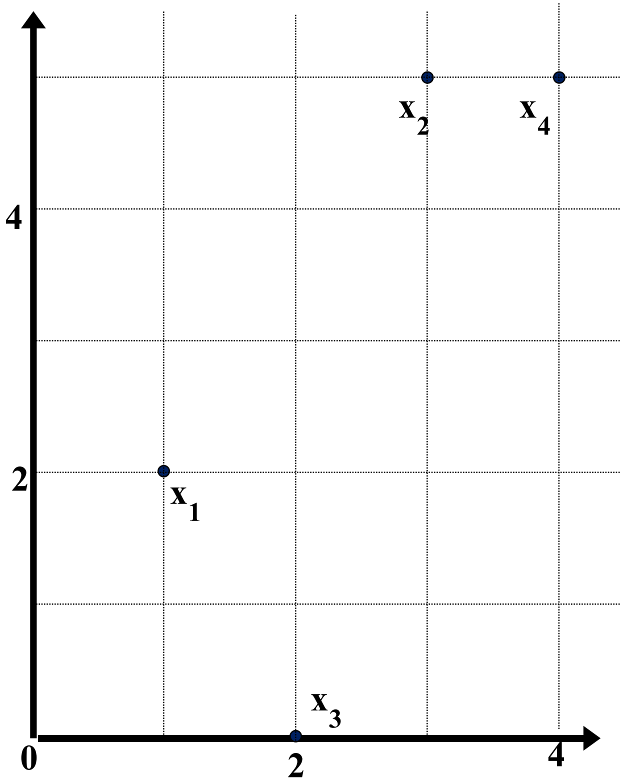
$$d(i,j) = \sqrt{(|x_{i_1} - x_{j_1}|^2 + |x_{i_2} - x_{j_2}|^2 + \ldots + |x_{i_p} - x_{j_p}|^2)}$$

- $h \to \infty$. "supremum" (L_{max} norm, L_∞ norm) distance.
  - This is the maximum difference between any component (attribute) of the vectors

$$d(i,j) = \lim_{h \to \infty} \left(\sum_{f=1}^{p} |x_{if} - x_{jf}|^h \right)^{\frac{1}{h}} = \max_{f}^{p} |x_{if} - x_{jf}|$$

Example: Minkowski Distance

point	attribute 1	attribute 2
x1	1	2
x2	3	5
x3	2	0
x4	4	5



Dissimilarity Matrices

Manhattan (L_1)

L	x1	x2	x3	x4
x1	0			
x2	5	0		
x3	3	6	0	
x4	6	1	7	0

Euclidean (L_2)

L2	x1	x2	x3	x4
x1	0			
x2	3.61	0		
x3	2.24	5.1	0	
x4	4.24	1	5.39	0

Supremum

L_∞	x1	x2	x3	x4
x1	0			
x2	3	0		
x3	2	5	0	
x4	3	1	5	0

Ordinal Variables

- An ordinal variable can be discrete or continuous
- Order is important, e.g., rank
- Can be treated like interval-scaled
  - replace x_{if} by their rank $r_{if} \in \{1, \ldots, M_f\}$
  - map the range of each variable onto [0, 1] by replacing *i*-th object in the *f*-th variable by

$$z_{if} = \frac{r_{if} - 1}{M_f - 1}$$

  - compute the dissimilarity using methods for interval-scaled variables

Attributes of Mixed Type

- A database may contain all attribute types
  - Nominal, symmetric binary, asymmetric binary, numeric, ordinal
- One may use a weighted formula to combine their effects

$$d(i,j) = \frac{\sum_{f=1}^{p} \delta_{ij}^{(f)} d_{ij}^{(f)}}{\sum_{f=1}^{p} \delta_{ij}^{(f)}}$$

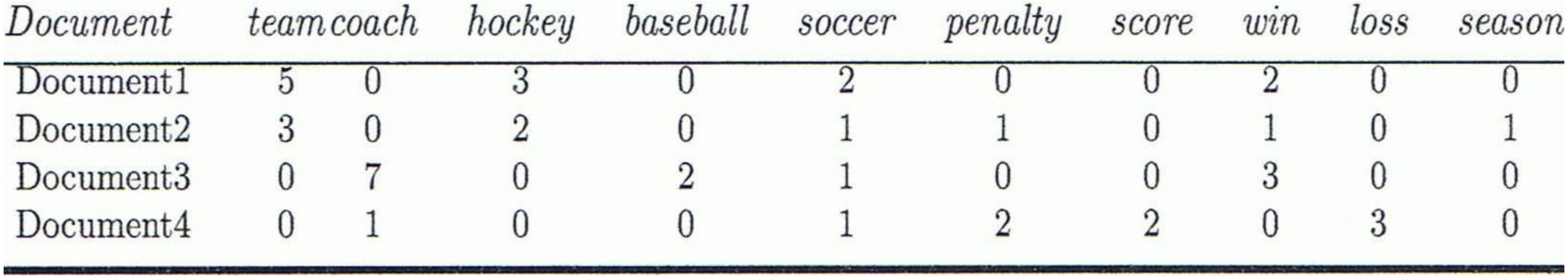
  - *f* is binary or nominal:

    $d_{ij}^{(f)} = 0$ if $x_{if} = x_{jf}$, or $d_{ij}^{(f)} = 1$ otherwise
  - *f* is numeric: use the normalized distance
  - *f* is ordinal
    - Compute ranks r_{if} and
    - Treat z_{if} as interval-scaled

$$z_{if} = \frac{r_{if} - 1}{M_f - 1}$$

Cosine Similarity

- A **document** can be represented by thousands of attributes, each recording the *frequency* of a particular word (such as keywords) or phrase in the document.



Document	*team*	*coach*	*hockey*	*baseball*	*soccer*	*penalty*	*score*	*win*	*loss*	*season*
Document1	5	0	3	0	2	0	0	2	0	0
Document2	3	0	2	0	1	1	0	1	0	1
Document3	0	7	0	2	1	0	0	3	0	0
Document4	0	1	0	0	1	2	2	0	3	0

- Other vector objects: gene features in micro-arrays, …
- Applications: information retrieval, biologic taxonomy, gene feature mapping, ...
- Cosine measure: If d_1 and d_2 are two vectors (e.g., term-frequency vectors), then

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

where • indicates vector dot product, $||d||$: the length of vector d

Example: Cosine Similarity

- $\cos(d_1, d_2) = (d_1 \bullet d_2) / ||d_1||\ ||d_2||$,
  where • indicates vector dot product, ||*d*|: the length of vector *d*

- Ex: Find the **similarity** between documents 1 and 2.

d_1 **=** (5, 0, 3, 0, 2, 0, 0, 2, 0, 0)
d_2 **=** (3, 0, 2, 0, 1, 1, 0, 1, 0, 1)

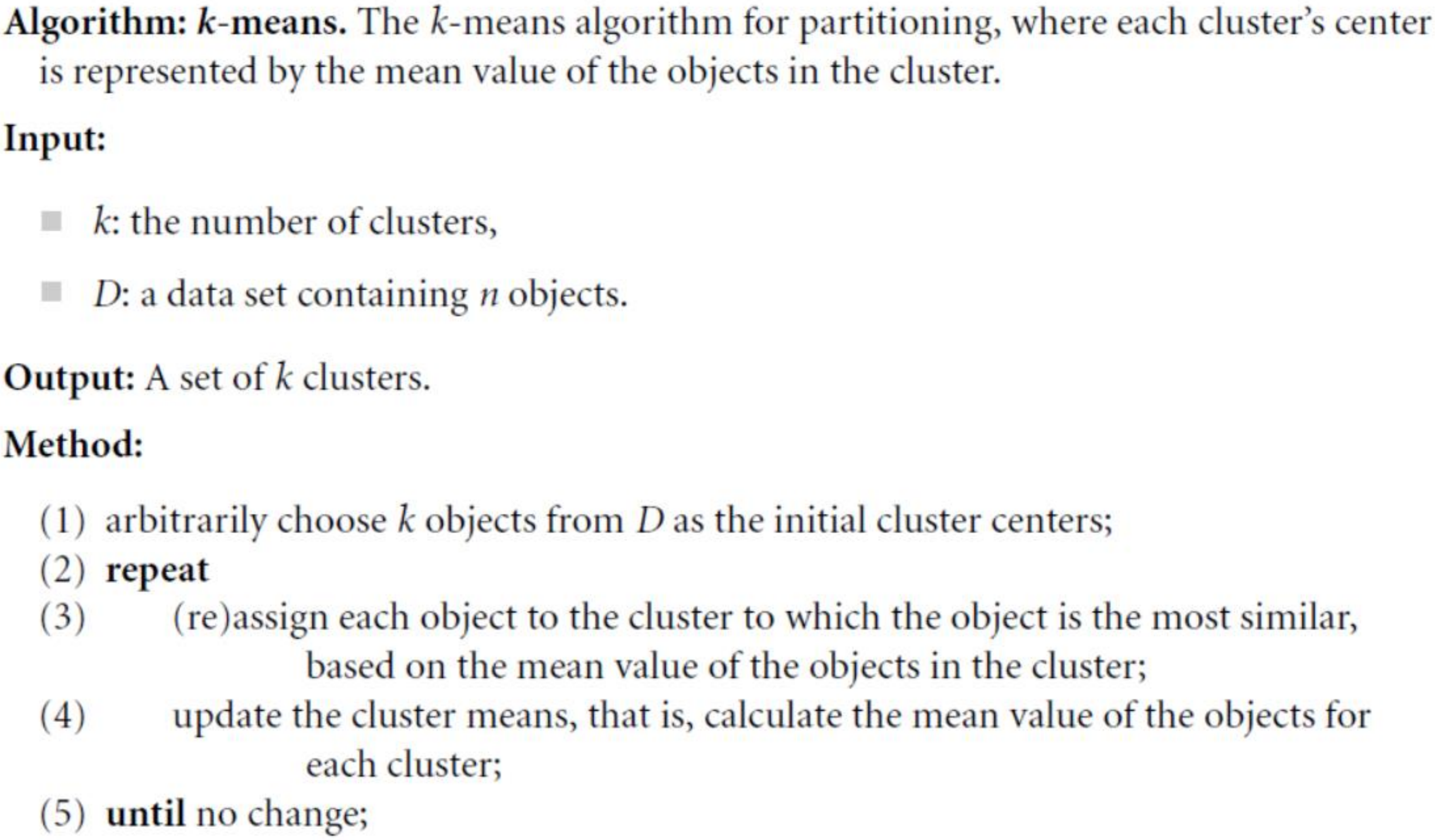
$d_1 \bullet d_2$ = 5*3+0*0+3*2+0*0+2*1+0*1+0*1+2*1+0*0+0*1 = 25
$||d_1||$= $(5*5+0*0+3*3+0*0+2*2+0*0+0*0+2*2+0*0+0*0)^{0.5}$=$(42)^{0.5}$ = 6.481
$||d_2||$= $(3*3+0*0+2*2+0*0+1*1+1*1+0*0+1*1+0*0+1*1)^{0.5}$=$(17)^{0.5}$ = 4.12
$\cos(d_1, d_2)$ = 0.94

Partitioning Algorithms: Basic Concept

- Partitioning method: Partitioning a database ***D*** of ***n*** objects into a set of ***k*** clusters, such that the sum of squared distances is minimized (where c_i is the centroid or medoid of cluster C_i)

$$E = \Sigma_{i=1}^{k} \Sigma_{p \in C_i} (p - c_i)^2$$

- Given *k*, find a partition of *k clusters* that optimizes the chosen partitioning criterion
  - Global optimal: exhaustively enumerate all partitions
  - Heuristic methods: *k-means* and *k-medoids* algorithms
  - *k-means* (MacQueen'67, Lloyd'57/'82): 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



Algorithm: *k*-means. The *k*-means algorithm for partitioning, where each cluster's center is represented by the mean value of the objects in the cluster.

Input:

- *k*: the number of clusters,
- *D*: a data set containing *n* objects.

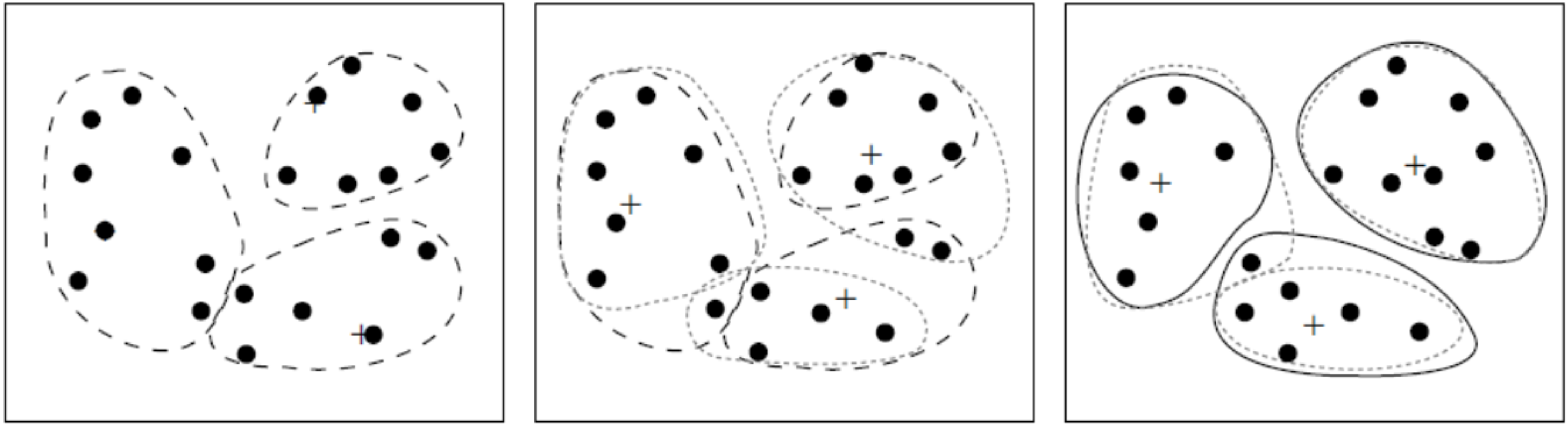
Output: A set of *k* clusters.

Method:

```
arbitrarily choose $k$ objects from $D$ as the initial cluster centers;
repeat
    (re)assign each object to the cluster to which the object is the most similar,
        based on the mean value of the objects in the cluster;
    update the cluster means, that is, calculate the mean value of the objects for
        each cluster;
until no change;
```

The *K-Means* Clustering Method

- Given *k*, the *k-means* algorithm is implemented in four steps:
  - Partition objects into *k* nonempty subsets
  - Compute seed points as the centroids of the clusters of the current partitioning (the centroid is the center, i.e., *mean point*, of the cluster)
  - Assign each object to the cluster with the nearest seed point
  - Go back to Step 2, stop when the assignment does not change



(a) Initial clustering

(b) Iterate

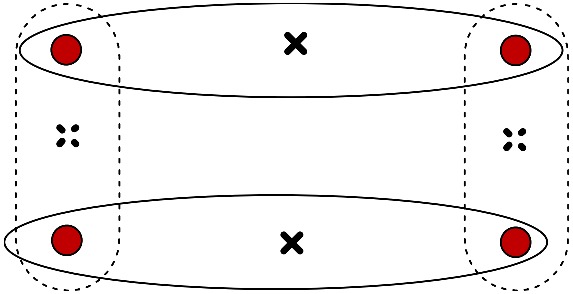
(c) Final clustering

Comments on the *K-Means* Method

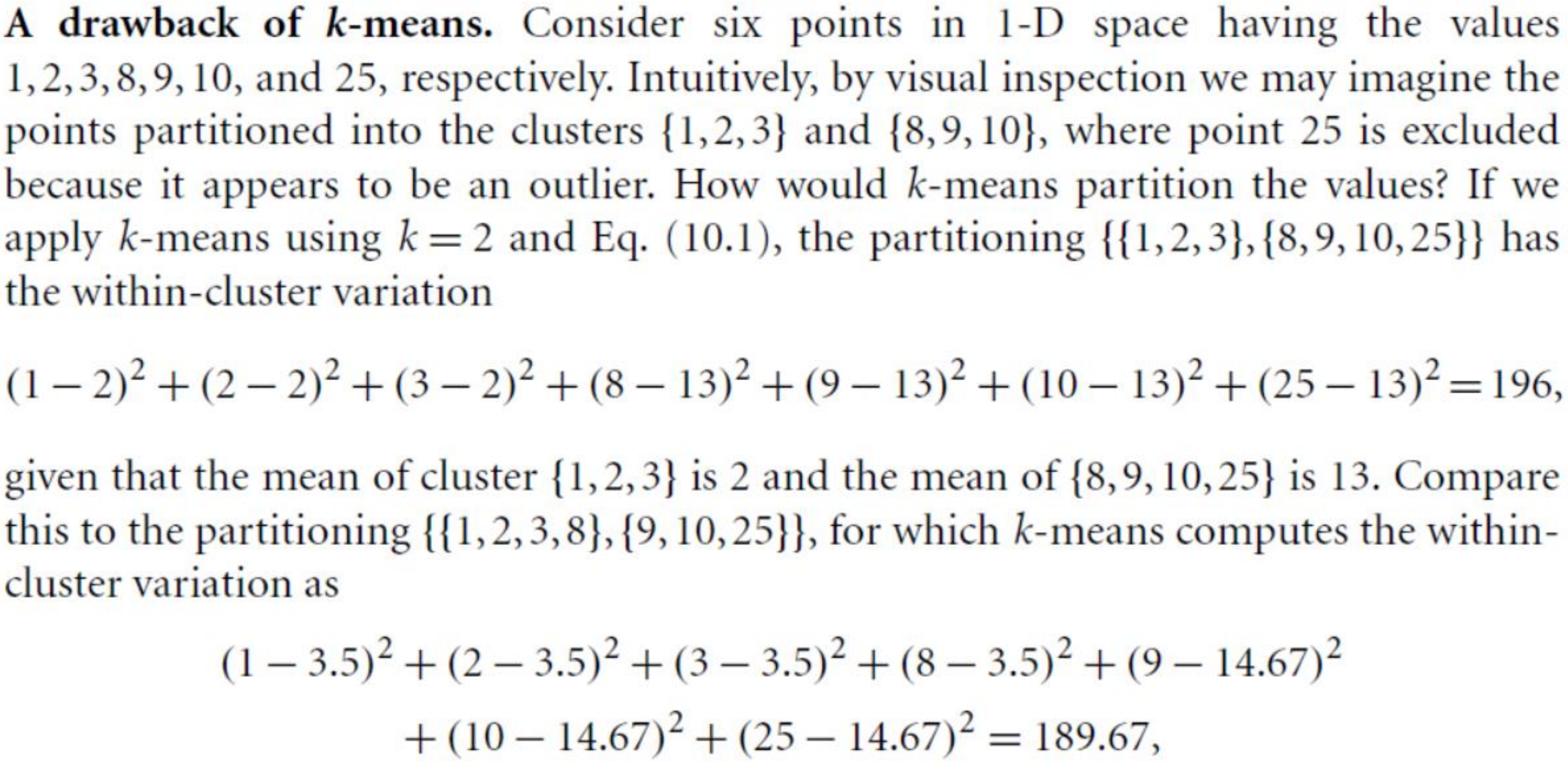
- Strength: *Efficient*: *O*(*tkn*), where *n* is # objects, *k* is # clusters, and *t* is # iterations. Normally, *k*, *t* << *n*.
    - Comparing: PAM: O(k(n-k)^2), CLARA: O(ks^2 + k(n-k))
- Comment: Often terminates at a *local optimal*.
- Weakness
    - Applicable only to objects in a continuous n-dimensional space
        - Using the k-modes method for categorical data
        - In comparison, k-medoids can be applied to a wide range of data
    - Need to specify *k, the number* of clusters, in advance (there are ways to automatically determine the best k (see Hastie et al., 2009)
    - Sensitive to noisy data and *outliers*
    - Not suitable to discover clusters with *non-convex shapes*

Variations of the *K-Means* Method

- Most of the variants of the *k-means* which differ in
  - Selection of the initial *k* means
  - Dissimilarity calculations
  - Strategies to calculate cluster means



- Handling categorical data: *k-modes*
  - Replacing means of clusters with <u>modes</u>
  - Using new dissimilarity measures to deal with categorical objects
  - Using a <u>frequency</u>-based method to update modes of clusters
  - A mixture of categorical and numerical data: *k-prototype* method



A drawback of *k*-means. Consider six points in 1-D space having the values $1, 2, 3, 8, 9, 10$, and 25, respectively. Intuitively, by visual inspection we may imagine the points partitioned into the clusters $\{1, 2, 3\}$ and $\{8, 9, 10\}$, where point 25 is excluded because it appears to be an outlier. How would k-means partition the values? If we apply k-means using $k = 2$ and Eq. (10.1), the partitioning $\{\{1, 2, 3\}, \{8, 9, 10, 25\}\}$ has the within-cluster variation

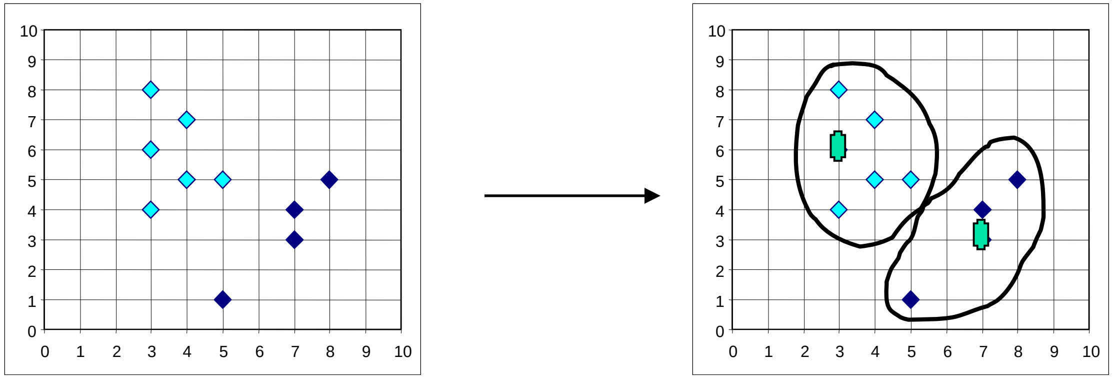
$$(1-2)^2 + (2-2)^2 + (3-2)^2 + (8-13)^2 + (9-13)^2 + (10-13)^2 + (25-13)^2 = 196,$$

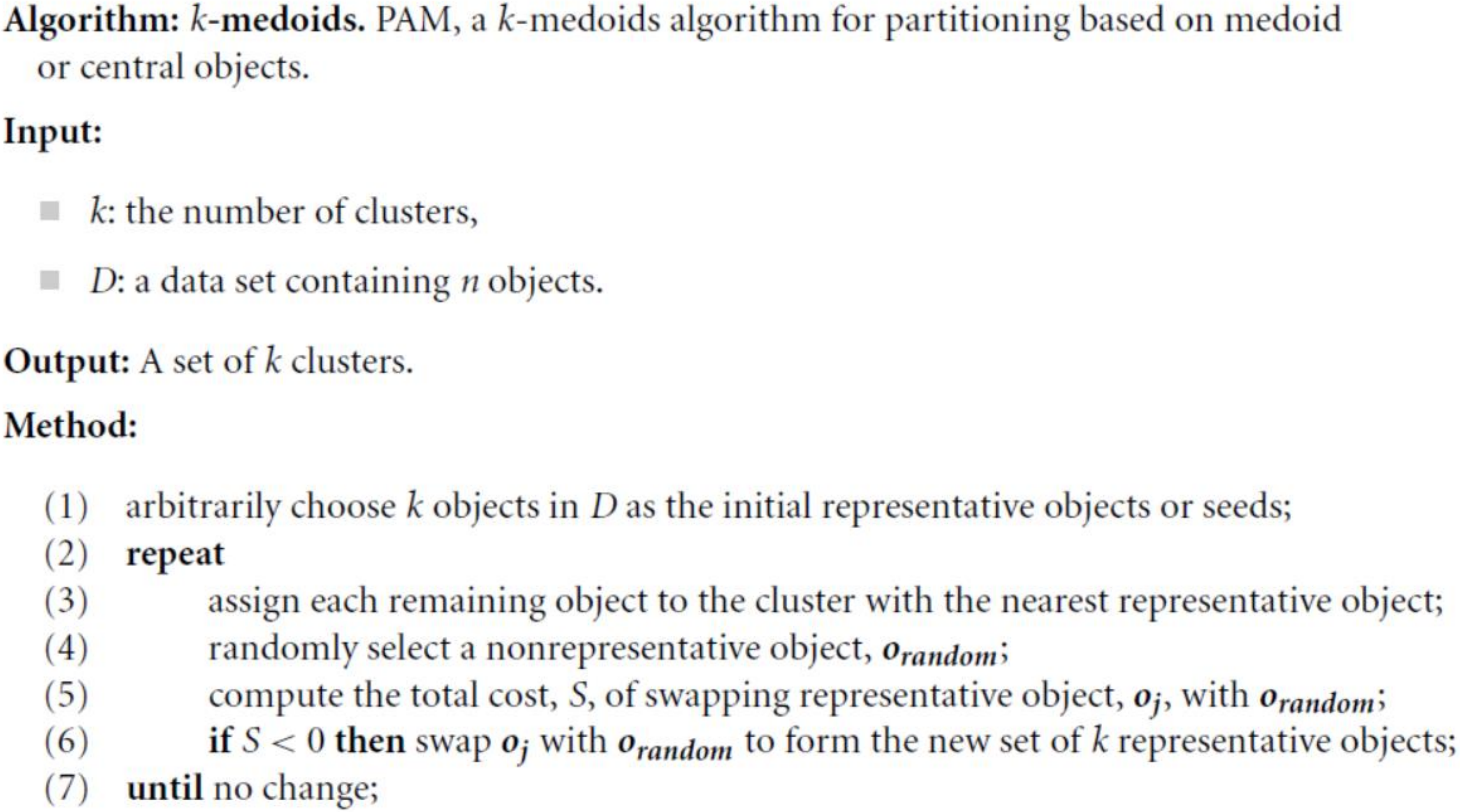
given that the mean of cluster $\{1, 2, 3\}$ is 2 and the mean of $\{8, 9, 10, 25\}$ is 13. Compare this to the partitioning $\{\{1, 2, 3, 8\}, \{9, 10, 25\}\}$, for which k-means computes the within-cluster variation as

$$(1-3.5)^2 + (2-3.5)^2 + (3-3.5)^2 + (8-3.5)^2 + (9-14.67)^2 \\ + (10-14.67)^2 + (25-14.67)^2 = 189.67,$$

What Is the Problem of the K-Means Method?

- The k-means algorithm is sensitive to outliers
- 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.
- PAM: Partition around Medoids





Algorithm: ***k*-medoids.** PAM, a *k*-medoids algorithm for partitioning based on medoid or central objects.

Input:

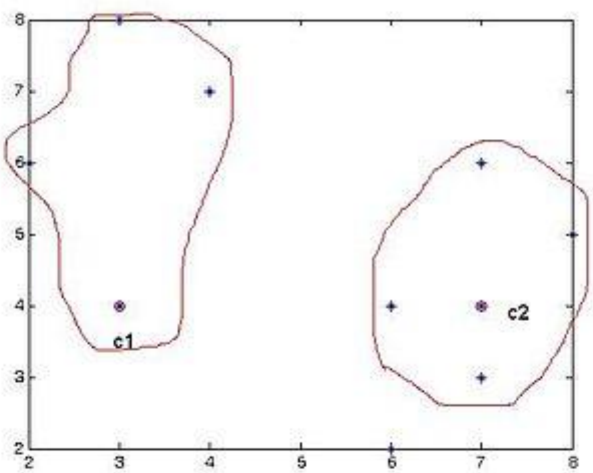
- *k*: the number of clusters,
- *D*: a data set containing *n* objects.

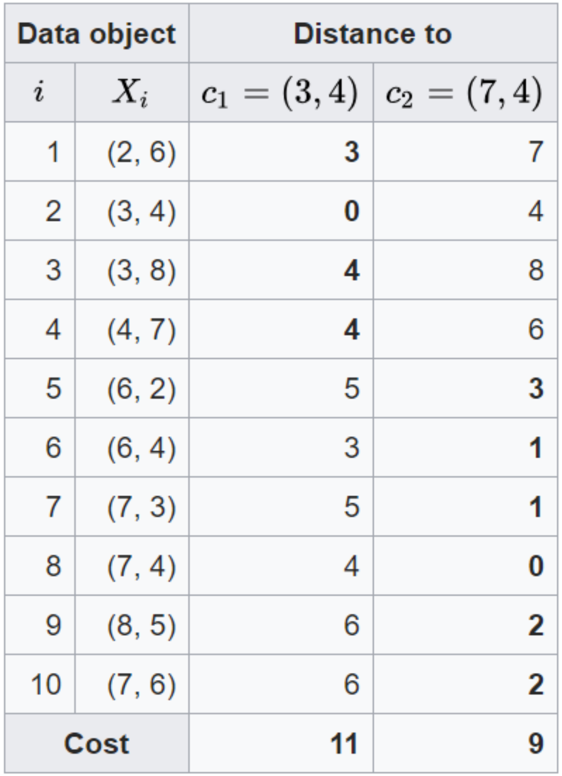
Output: A set of *k* clusters.

Method:

```
arbitrarily choose $k$ objects in $D$ as the initial representative objects or seeds;
repeat
    assign each remaining object to the cluster with the nearest representative object;
    randomly select a nonrepresentative object, $\boldsymbol{o}_{random}$;
    compute the total cost, $S$, of swapping representative object, $\boldsymbol{o}_j$, with $\boldsymbol{o}_{random}$;
    if $S < 0$ then swap $\boldsymbol{o}_j$ with $\boldsymbol{o}_{random}$ to form the new set of $k$ representative objects;
until no change;
```

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





Data object		Distance to	
i	X_i	$c_1 = (3,4)$	$c_2 = (7,4)$
1	(2, 6)	**3**	7
2	(3, 4)	**0**	4
3	(3, 8)	**4**	8
4	(4, 7)	**4**	6
5	(6, 2)	5	**3**
6	(6, 4)	3	**1**
7	(7, 3)	5	**1**
8	(7, 4)	4	**0**
9	(8, 5)	6	**2**
10	(7, 6)	6	**2**
Cost		**11**	**9**

$$\text{Cluster}_1 = \{(3,4)(2,6)(3,8)(4,7)\}$$

$$\text{Cluster}_2 = \{(7,4)(6,2)(6,4)(7,3)(8,5)(7,6)\}$$

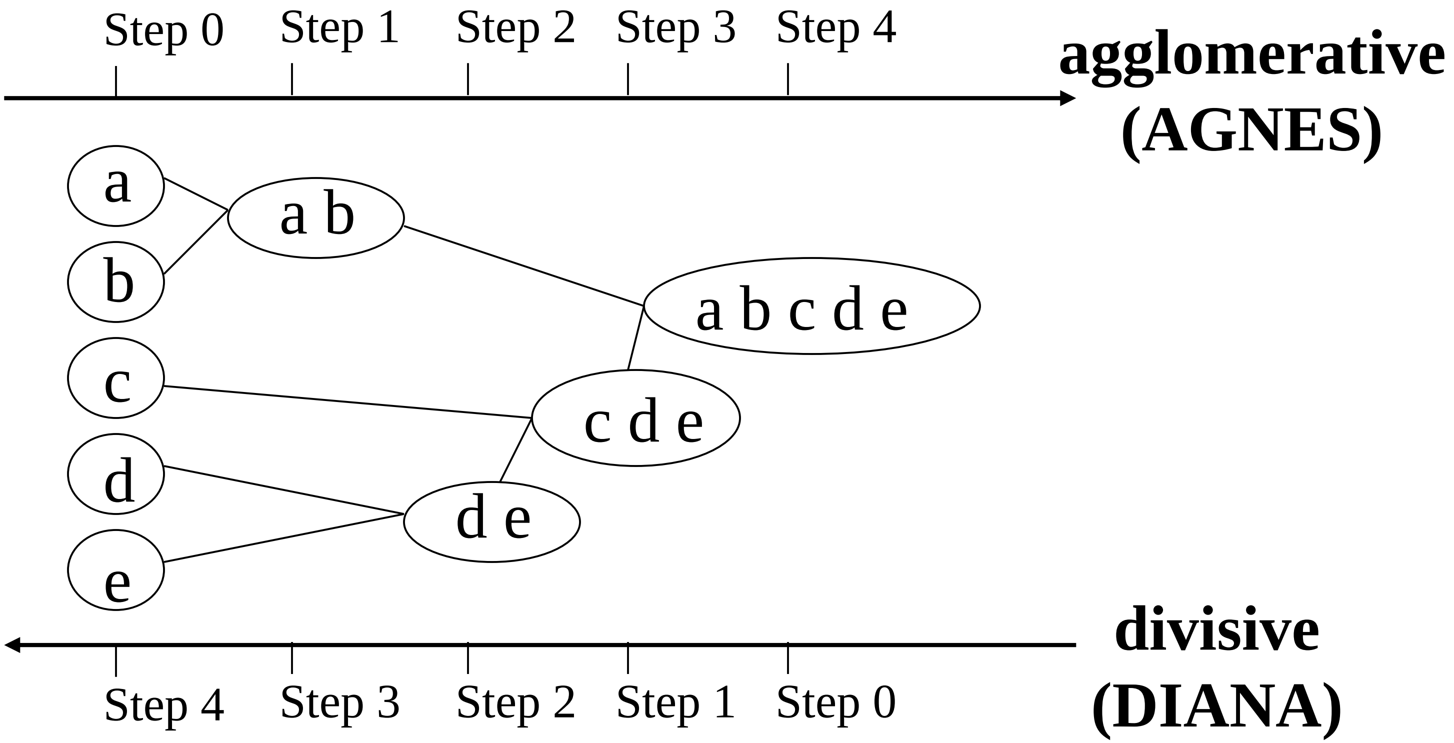
I	**XI**	**C1 = (3,4)**	**O = (7,3)**
1	(2, 6)	**3**	8
2	(3, 4)	**0**	5
3	(3, 8)	**4**	9
4	(4, 7)	**4**	7
5	(6, 2)	5	**2**
6	(6, 4)	3	**2**
7	(7, 3)	5	**0**
8	(7, 4)	4	**1**
9	(8, 5)	6	**3**
10	(7, 6)	6	**3**
Cost		**11**	**11**

Choosing O = (7,3)
instead of C2=(7,4)

Increase cost to 22 from 20

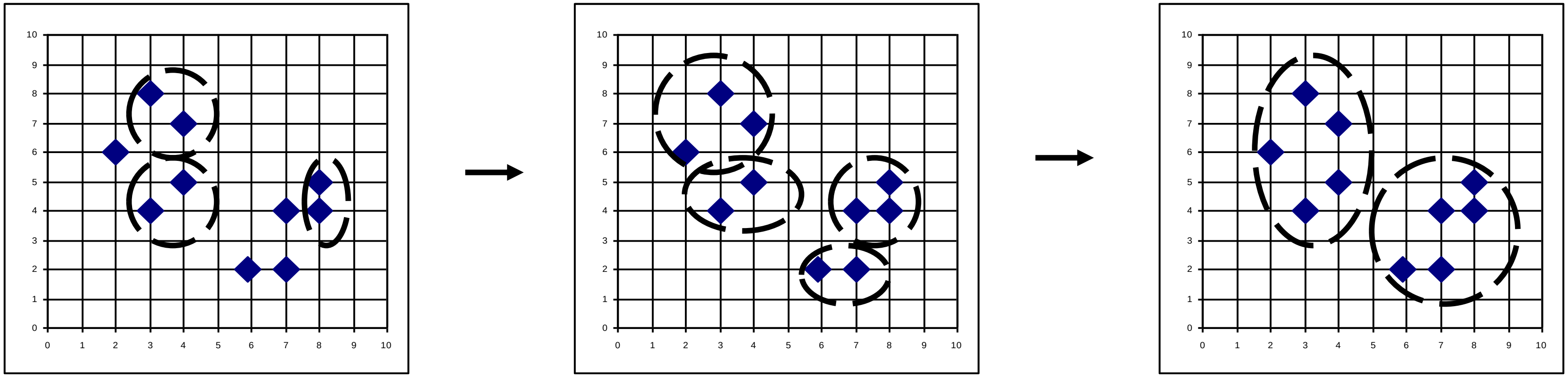
Hierarchical Clustering

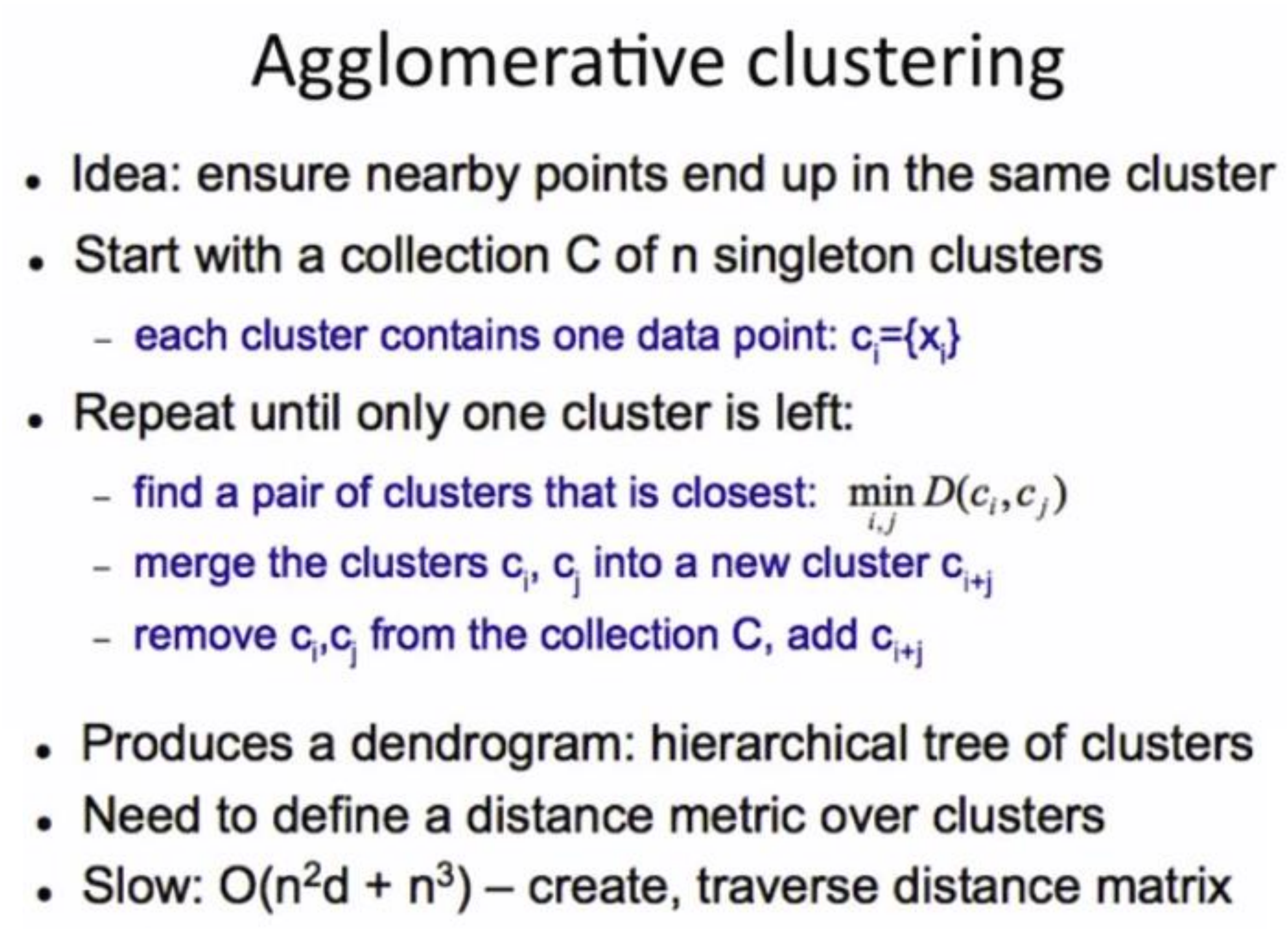
- Use distance matrix as clustering criteria. This method does not require the number of clusters ***k*** as an input, but needs a termination condition



AGNES (Agglomerative Nesting)

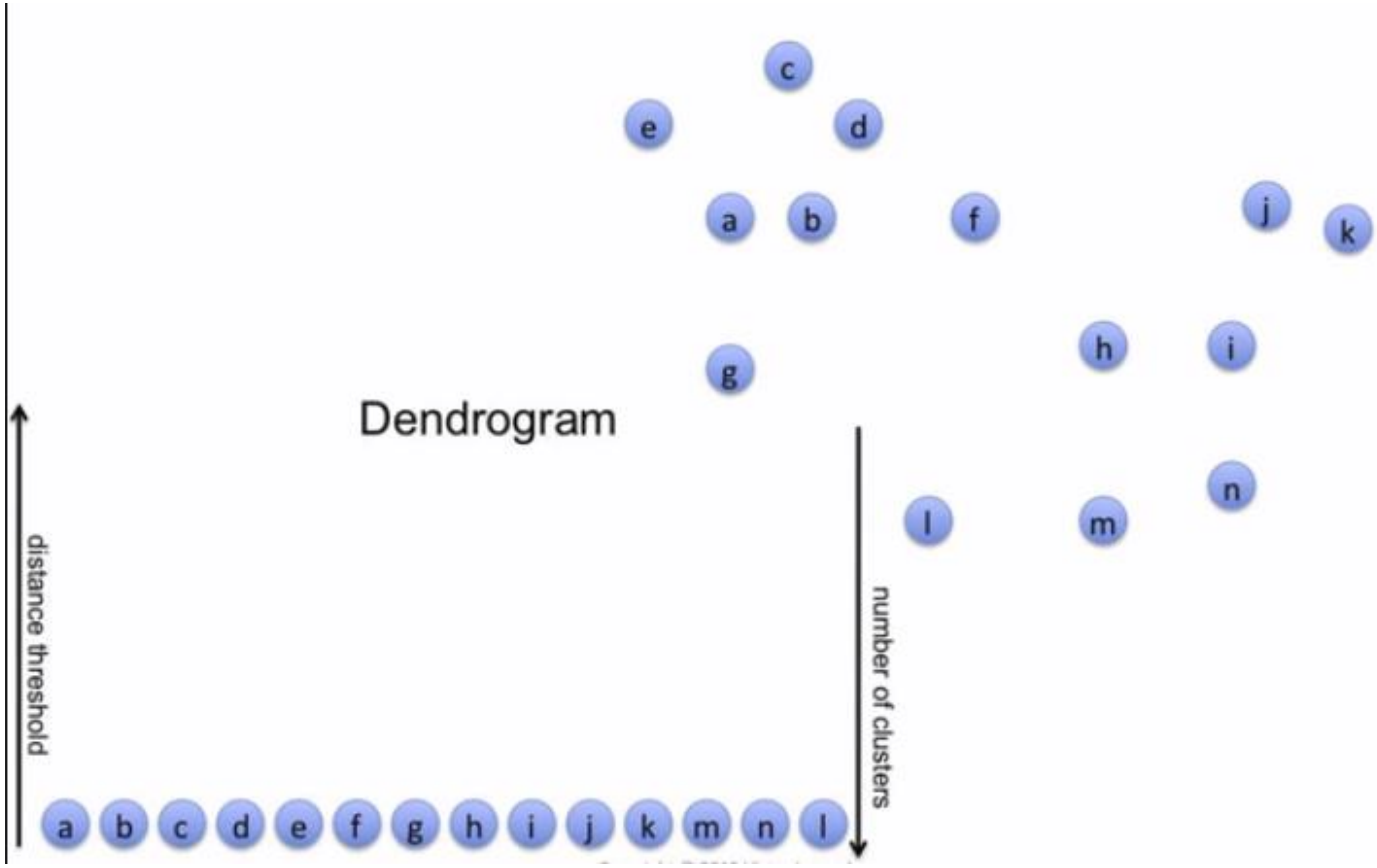
- Introduced in Kaufmann and Rousseeuw (1990)
- Implemented in statistical packages, e.g., Splus
- Use the **single-link** method and the dissimilarity matrix
- Merge nodes that have the least dissimilarity
- Go on in a non-descending fashion
- Eventually all nodes belong to the same cluster

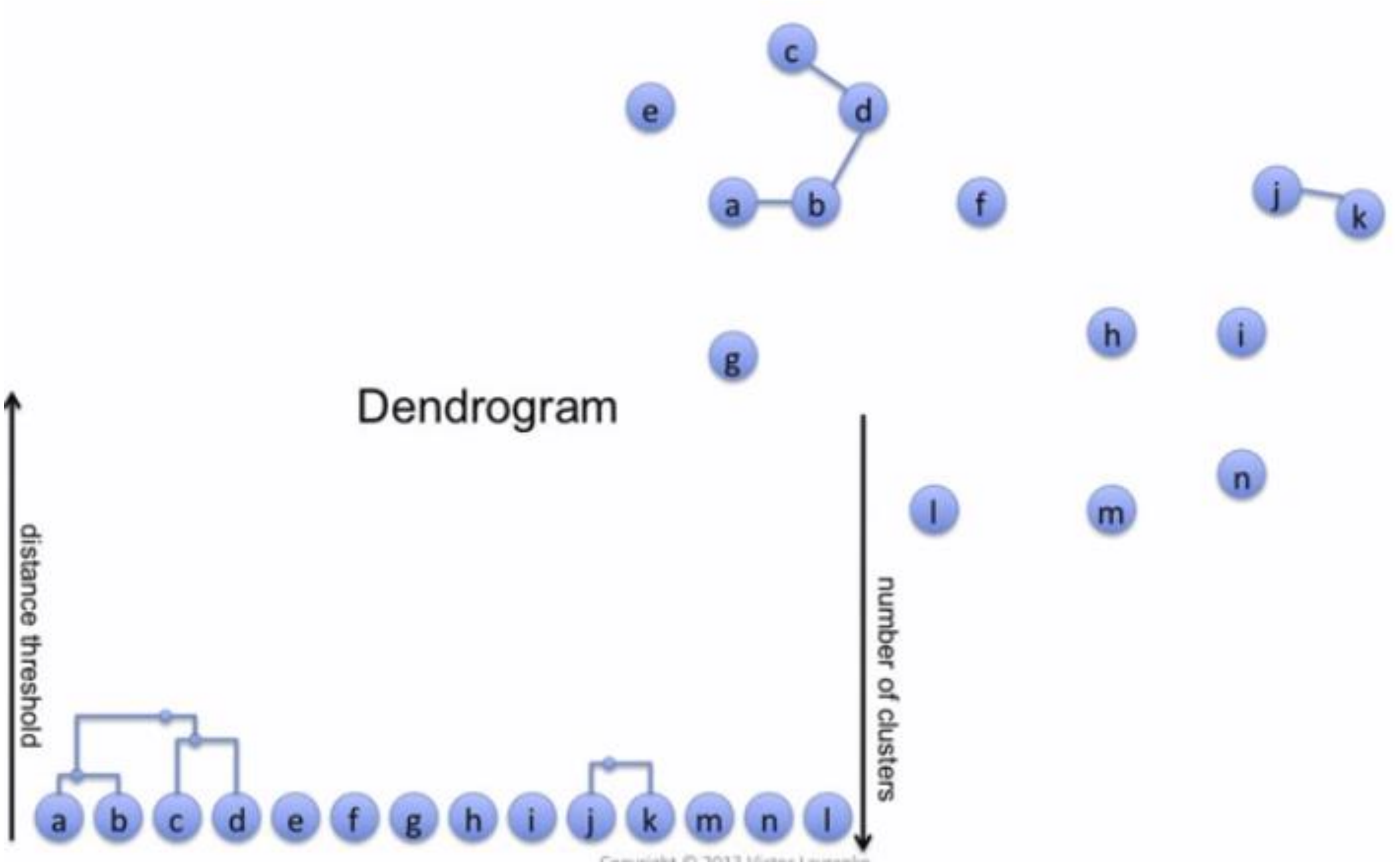




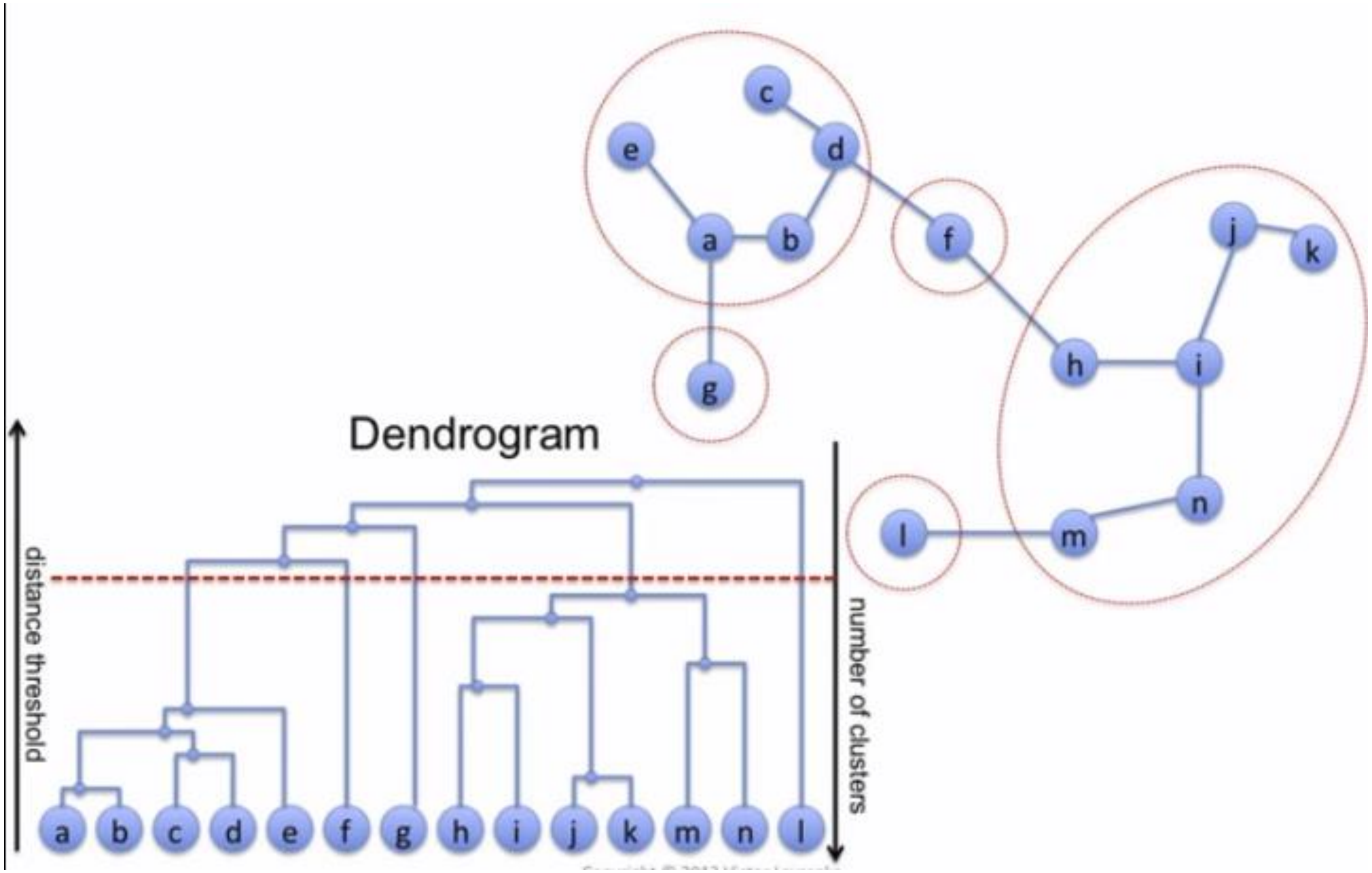
Agglomerative clustering

- Idea: ensure nearby points end up in the same cluster
- Start with a collection C of n singleton clusters
  - each cluster contains one data point: $c_i=\{x_i\}$
- Repeat until only one cluster is left:
  - find a pair of clusters that is closest: $\min_{i,j} D(c_i, c_j)$
  - merge the clusters c_i, c_j into a new cluster c_{i+j}
  - remove c_i,c_j from the collection C, add c_{i+j}
- Produces a dendrogram: hierarchical tree of clusters
- Need to define a distance metric over clusters
- Slow: $O(n^2d + n^3)$ – create, traverse distance matrix

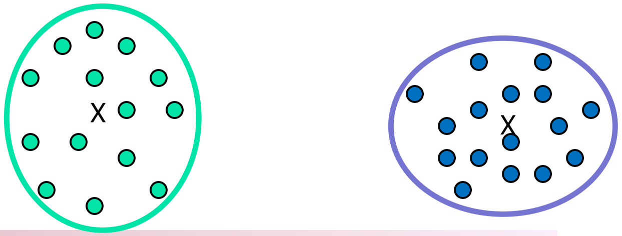

c
e
d
a
b
f
j
k
h
i
g
Dendrogram
n
l
m
distance threshold
number of clusters
a
b
c
d
e
f
g
h
i
j
k
m
n
l



Copyright © 2013 Victor Lavrenko


c
e
d
a
b
f
j
k
g
h
i
n
l
m
Dendrogram
distance threshold
number of clusters
a
b
c
d
e
f
g
h
i
j
k
m
n
l

Distance between Clusters



- Single link: smallest distance between an element in one cluster and an element in the other, i.e., $\text{dist}(K_i, K_j) = \min(t_{ip}, t_{jq})$
- Complete link: largest distance between an element in one cluster and an element in the other, i.e., $\text{dist}(K_i, K_j) = \max(t_{ip}, t_{jq})$
- Average: avg distance between an element in one cluster and an element in the other, i.e., $\text{dist}(K_i, K_j) = \text{avg}(t_{ip}, t_{jq})$
- Centroid: distance between the centroids of two clusters, i.e., $\text{dist}(K_i, K_j) = \text{dist}(C_i, C_j)$
- Medoid: distance between the medoids of two clusters, i.e., $\text{dist}(K_i, K_j) = \text{dist}(M_i, M_j)$
  - Medoid: a chosen, centrally located object in the cluster

Extensions to Hierarchical Clustering

- Major weakness of agglomerative clustering methods
  - Can never undo what was done previously
  - Do not scale well: time complexity of at least $O(n^2)$, where n is the number of total objects
- Integration of hierarchical & distance-based clustering
  - BIRCH (1996): uses CF-tree and incrementally adjusts the quality of sub-clusters
  - CHAMELEON (1999): hierarchical clustering using dynamic modeling

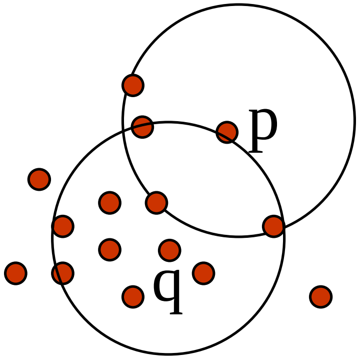
Density-Based Clustering Methods

- Clustering based on density (local cluster criterion), such as density-connected points
- Major features:
  - Discover clusters of arbitrary shape
  - Handle noise
  - One scan
  - Need density parameters as termination condition
- Several interesting studies:
  - DBSCAN: Ester, et al. (KDD'96)
  - OPTICS: Ankerst, et al (SIGMOD'99).
  - DENCLUE: Hinneburg & D. Keim (KDD'98)
  - CLIQUE: Agrawal, et al. (SIGMOD'98) (more grid-based)

Density-Based Clustering: Basic Concepts

- Two parameters*:*
  - *Eps*: Maximum radius of the neighbourhood
  - *MinPts*: Minimum number of points in an Eps-neighbourhood of that point
- $N_{Eps}(p)$: {q belongs to D | dist(p,q) ≤ Eps}
- Directly density-reachable: A point *p* is directly density-reachable from a point *q* w.r.t. *Eps*, *MinPts* if
  - *p* belongs to $N_{Eps}(q)$
  - core point condition:

$$|N_{Eps}(q)| \geq MinPts$$

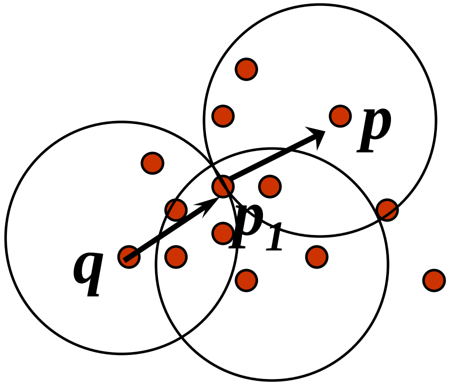


MinPts = 5

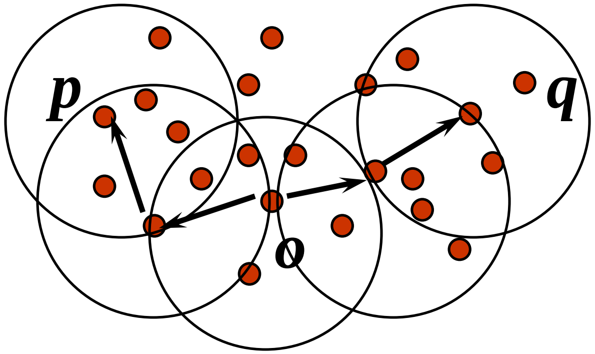
Eps = 1 cm

Density-Reachable and Density-Connected

- Density-reachable:
  - A point p is density-reachable from a point q w.r.t. *Eps*, *MinPts* if there is a chain of points $p_1, \ldots, p_n$, $p_1 = q$, $p_n = p$ such that p_{i+1} is directly density-reachable from p_i

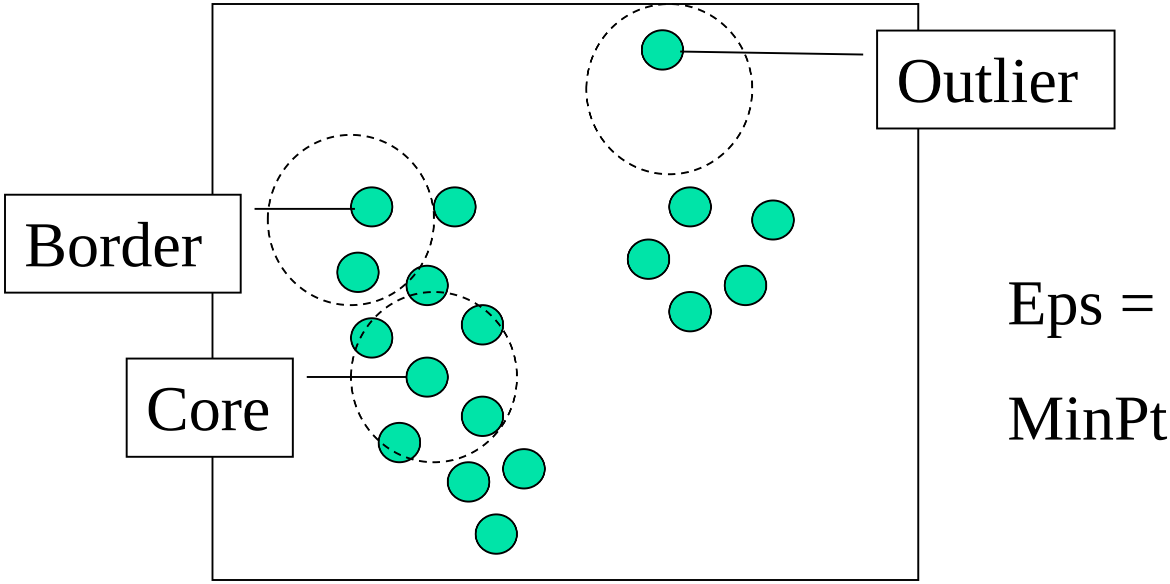


- Density-connected
  - A point p is density-connected to a point q w.r.t. *Eps*, *MinPts* if there is a point o such that both, p and q are density-reachable from o w.r.t. *Eps* and *MinPts*



DBSCAN: Density-Based Spatial Clustering of Applications with Noise

- Relies on a *density-based* notion of cluster: A *cluster* is defined as a maximal set of density-connected points
- Discovers clusters of arbitrary shape in spatial databases with noise

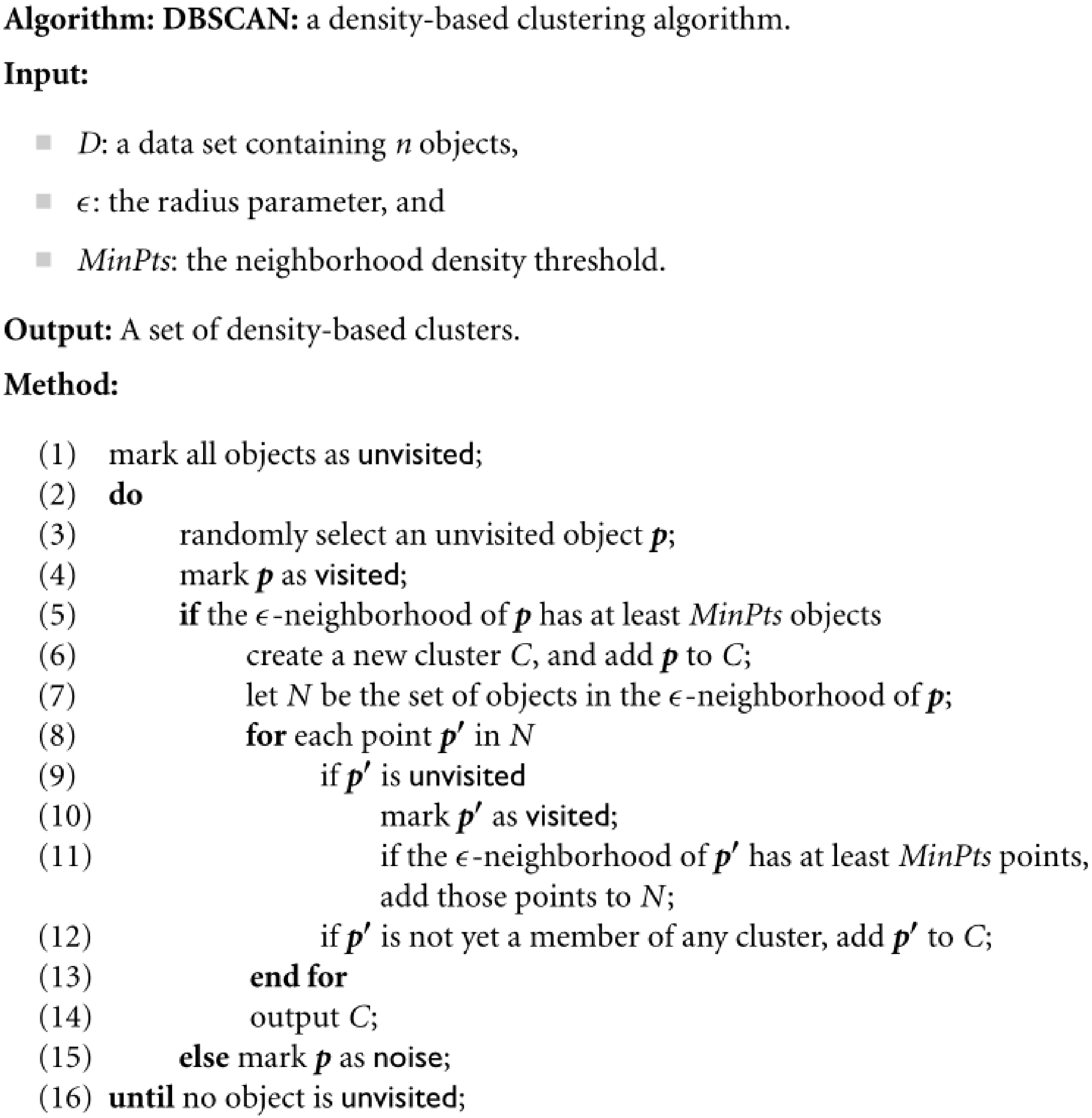


Eps = 1cm

MinPts = 5

DBSCAN: The Algorithm

- Arbitrary select a point *p*
- Retrieve all points density-reachable from *p* w.r.t. *Eps* and *MinPts*
- If *p* is a core point, a cluster is formed
- If *p* is a border point, no points are density-reachable from *p* and DBSCAN visits the next point of the database
- Continue the process until all of the points have been processed



Algorithm: DBSCAN: a density-based clustering algorithm.

Input:

- D: a data set containing n objects,
- ϵ: the radius parameter, and
- $MinPts$: the neighborhood density threshold.

Output: A set of density-based clusters.

Method:

```
mark all objects as unvisited;
do
    randomly select an unvisited object $p$;
    mark $p$ as visited;
    if the $\epsilon$-neighborhood of $p$ has at least $MinPts$ objects
        create a new cluster $C$, and add $p$ to $C$;
        let $N$ be the set of objects in the $\epsilon$-neighborhood of $p$;
        for each point $p'$ in $N$
            if $p'$ is unvisited
                mark $p'$ as visited;
                if the $\epsilon$-neighborhood of $p'$ has at least $MinPts$ points,
                add those points to $N$;
            if $p'$ is not yet a member of any cluster, add $p'$ to $C$;
        end for
        output $C$;
    else mark $p$ as noise;
until no object is unvisited;
```

DBSCAN: Sensitive to Parameters

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

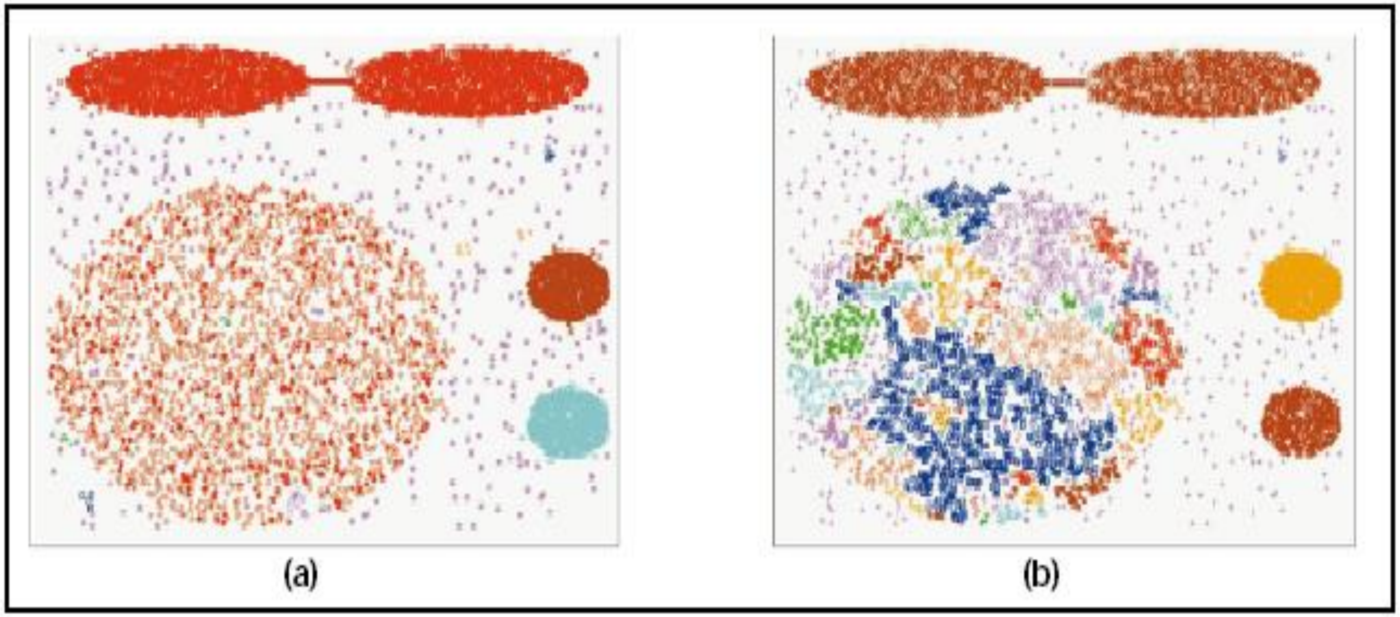
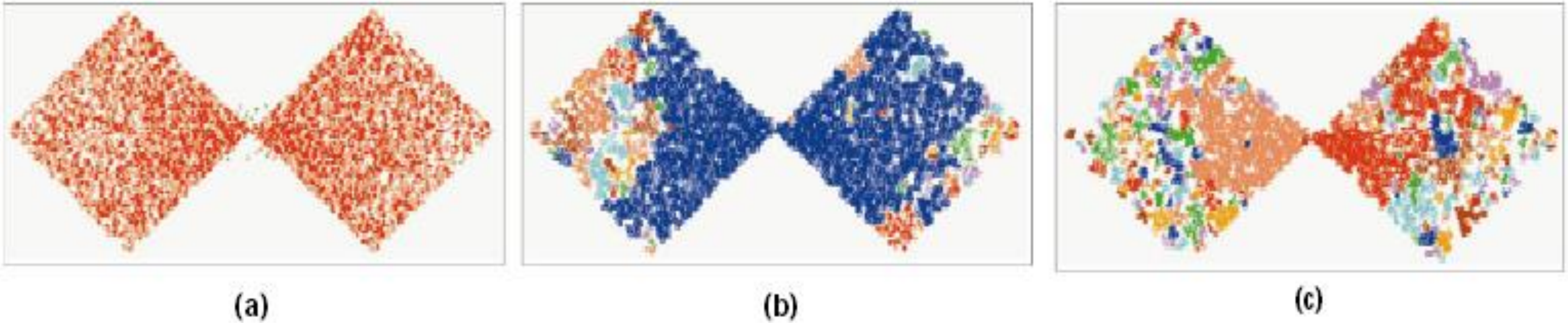


Figure 9. DBScan results for DS2 with MinPts at 4 and Eps at (a) 5.0, (b) 3.5, and (c) 3.0.



OPTICS: A Cluster-Ordering Method (1999)

- OPTICS: Ordering Points To Identify the Clustering Structure
  - Ankerst, Breunig, Kriegel, and Sander (SIGMOD'99)
  - Produces a special order of the database wrt its density-based clustering structure
  - This cluster-ordering contains info equiv to the density-based clusterings corresponding to a broad range of parameter settings
  - Good for both automatic and interactive cluster analysis, including finding intrinsic clustering structure
  - Can be represented graphically or using visualization techniques

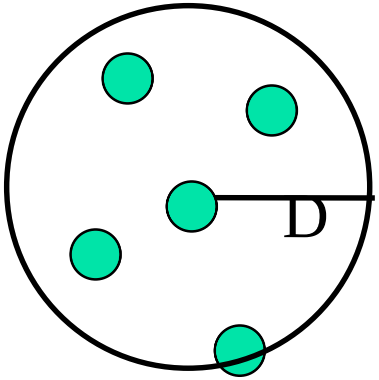
OPTICS: Some Extension from DBSCAN

- **Core Distance:**
  - min eps such that point is core

$$\begin{cases} UNDEFINED, \text{ if } Card(N_{\varepsilon}(p)) < MinPts \\ MinPts\text{-}distance(p), \text{ otherwise} \end{cases}$$

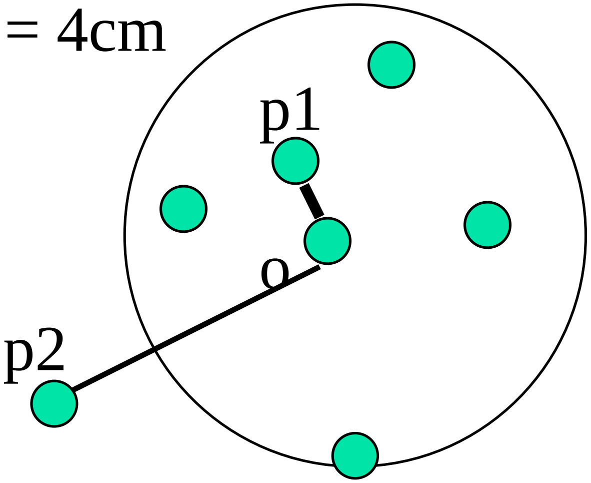
- **Reachability Distanc**

$$\begin{cases} UNDEFINED, \text{ if } |N_{\varepsilon}(o)| < MinPts \\ max(core\text{-}distance(o), distance(o,p)), \text{otherwise} \end{cases}$$



Max (core-distance (o), d (o, p))

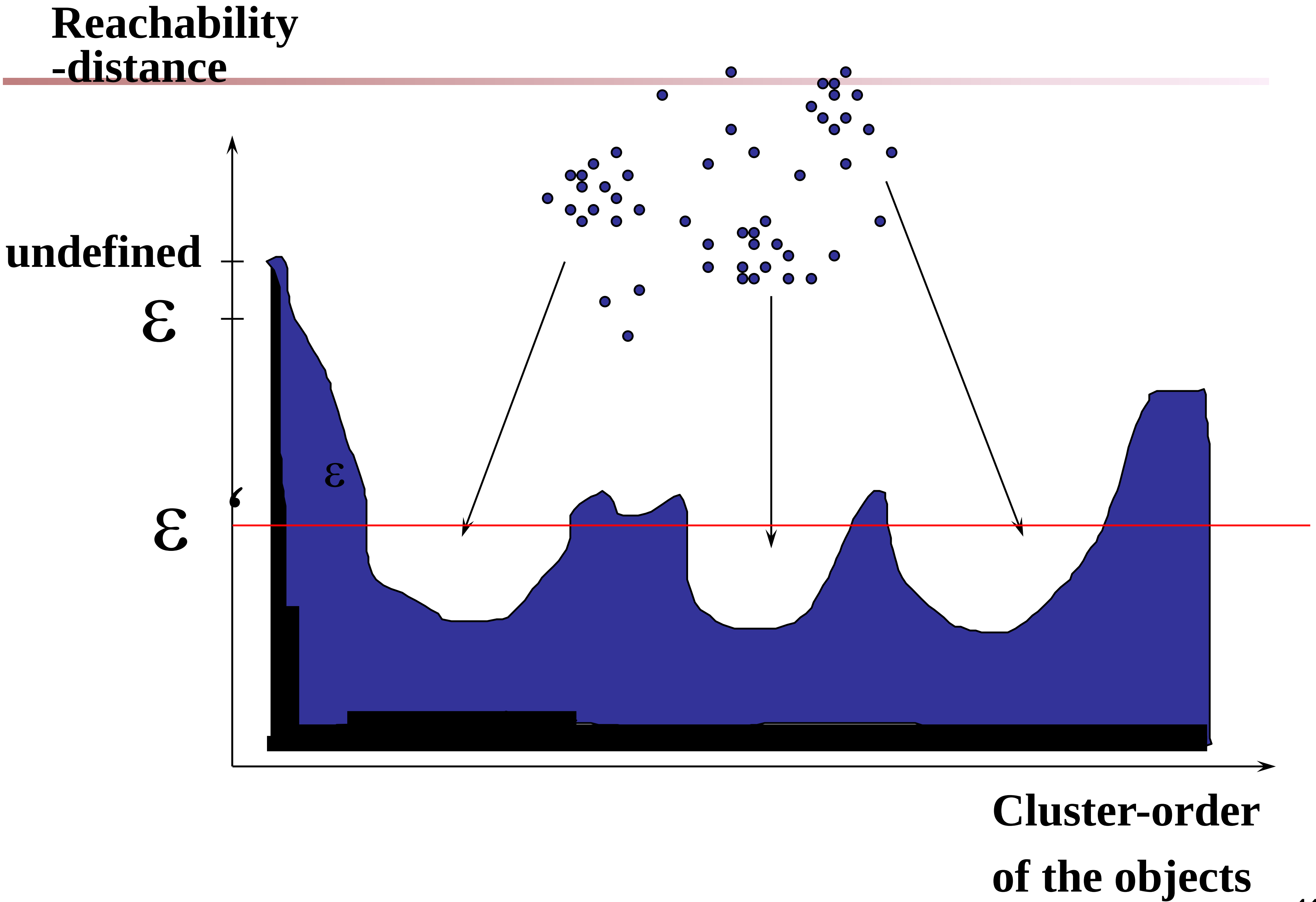
r(p1, o) = 2.8cm. r(p2,o) = 4cm



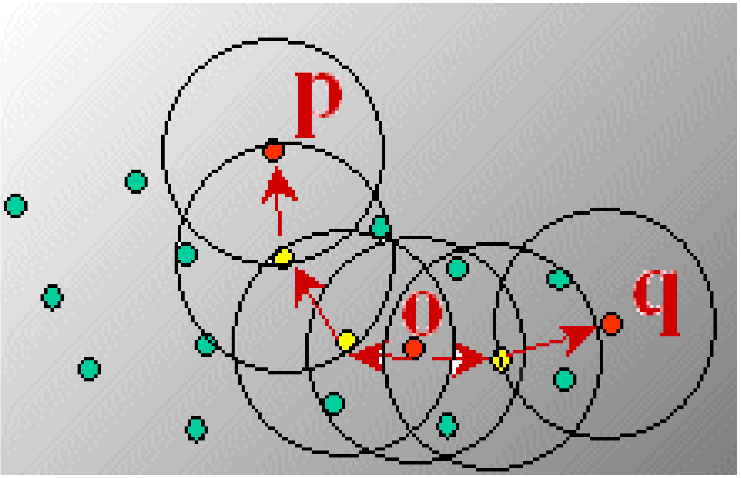
MinPts = 5

ε = 3 cm

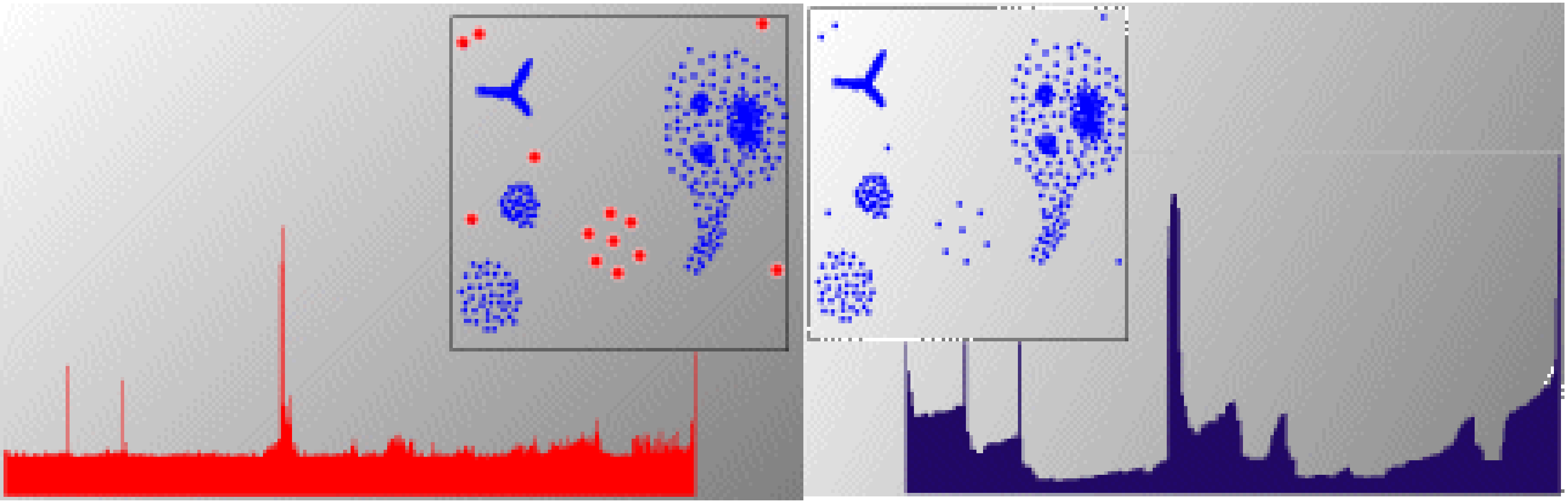
- Index-based:
  - Complexity: O(*NlogN*)


Reachability
-distance
undefined
ε
ε
'
ε
Cluster-order
of the objects

Density-Based Clustering: OPTICS & Its Applications



Change MinPts:

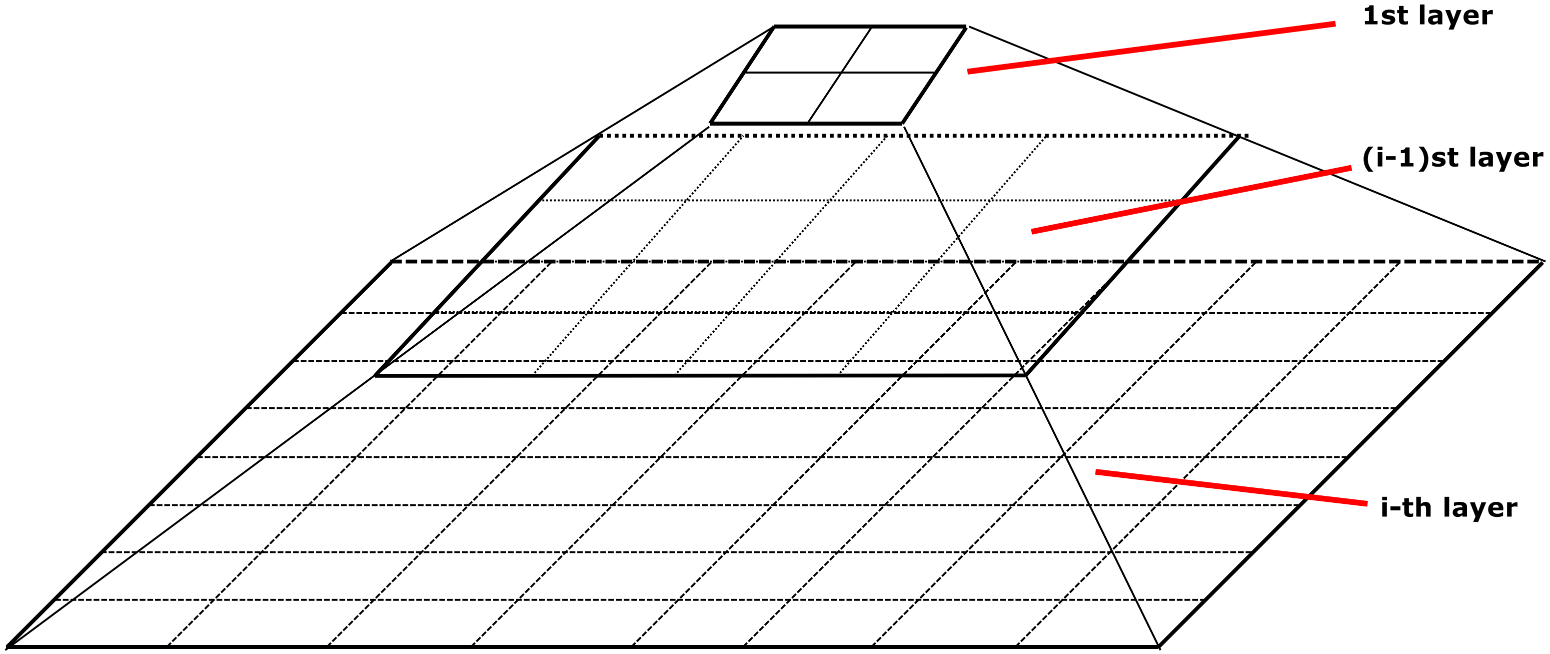


Grid-Based Clustering Method

- Using multi-resolution grid data structure
- Several interesting methods
  - STING (a STatistical INformation Grid approach) by Wang, Yang and Muntz (1997)
  - WaveCluster by Sheikholeslami, Chatterjee, and Zhang (VLDB'98)
    - A multi-resolution clustering approach using wavelet method
  - CLIQUE: Agrawal, et al. (SIGMOD'98)
    - Both grid-based and subspace clustering

STING: A Statistical Information Grid Approach

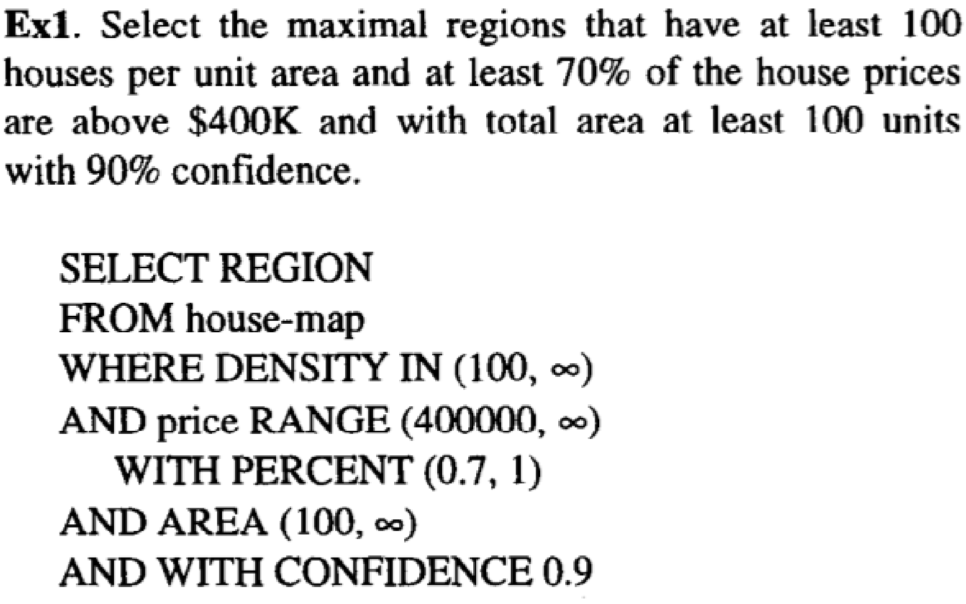
- Wang, Yang and Muntz (VLDB'97)
- The spatial area is divided into rectangular cells
- There are several levels of cells corresponding to different levels of resolution



The STING Clustering Method

- Each cell at a high level is partitioned into a number of smaller cells in the next lower level
- Statistical info of each cell is calculated and stored beforehand and is used to answer queries
- Parameters of higher level cells can be easily calculated from parameters of lower level cell
    - *count*, *mean*, *s*, *min*, *max*
    - type of distribution—*normal*, *uniform*, etc.

STING: spatial data query sample



Ex1. Select the maximal regions that have at least 100 houses per unit area and at least 70% of the house prices are above $400K and with total area at least 100 units with 90% confidence.

```
SELECT REGION
FROM house-map
WHERE DENSITY IN (100, ∞)
AND price RANGE (400000, ∞)
    WITH PERCENT (0.7, 1)
AND AREA (100, ∞)
AND WITH CONFIDENCE 0.9
```

- Use a top-down approach to answer spatial data queries
- Start from a pre-selected layer—typically with a small number of cells
- For each cell in the current level compute the confidence interval

Query Processing in STING

- To process a region query:
  - Start at the root and proceed to the next lower level.
  - Calculate the likelihood that a cell is relevant to the query at some confidence level using the statistical information of the cell.
  - Only children of likely relevant cells are recursively explored.
  - Repeat this process until the bottom layer is reached

Analysis of STING Algorithm

- Advantages:
  - Query-independent, easy to parallelize, incremental update
  - *O(K)*, where *K* is the number of grid cells at the lowest level

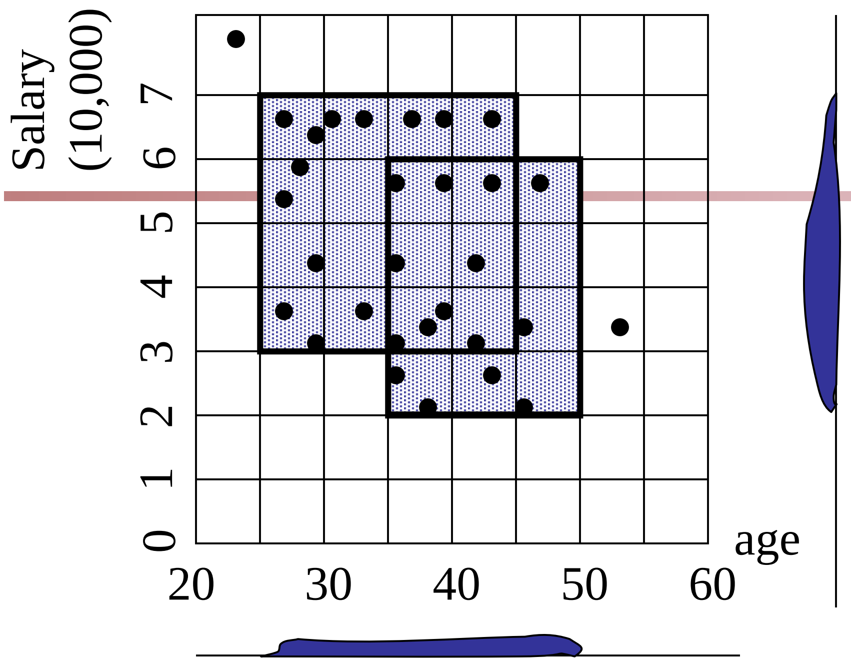
- Disadvantages:
  - All the cluster boundaries are either horizontal or vertical, and no diagonal boundary is detected

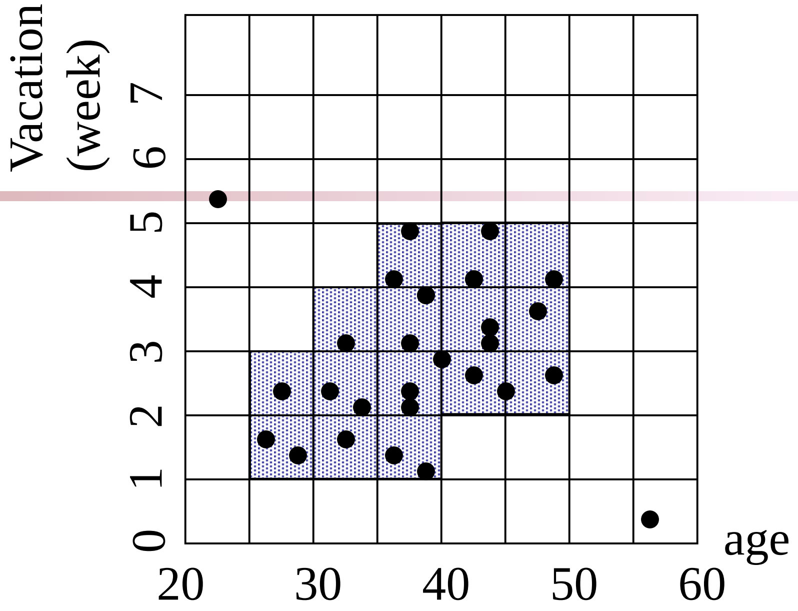
CLIQUE (Clustering In QUEst)

- Agrawal, Gehrke, Gunopulos, Raghavan (SIGMOD'98)
- Automatically identifying subspaces of a high dimensional data space that allow better clustering than original space
- CLIQUE can be considered as both density-based and grid-based
    - It partitions each dimension into the same number of equal length interval
    - It partitions an m-dimensional data space into non-overlapping rectangular units
    - A unit is dense if the fraction of total data points contained in the unit exceeds the input model parameter
    - A cluster is a maximal set of connected dense units within a subspace

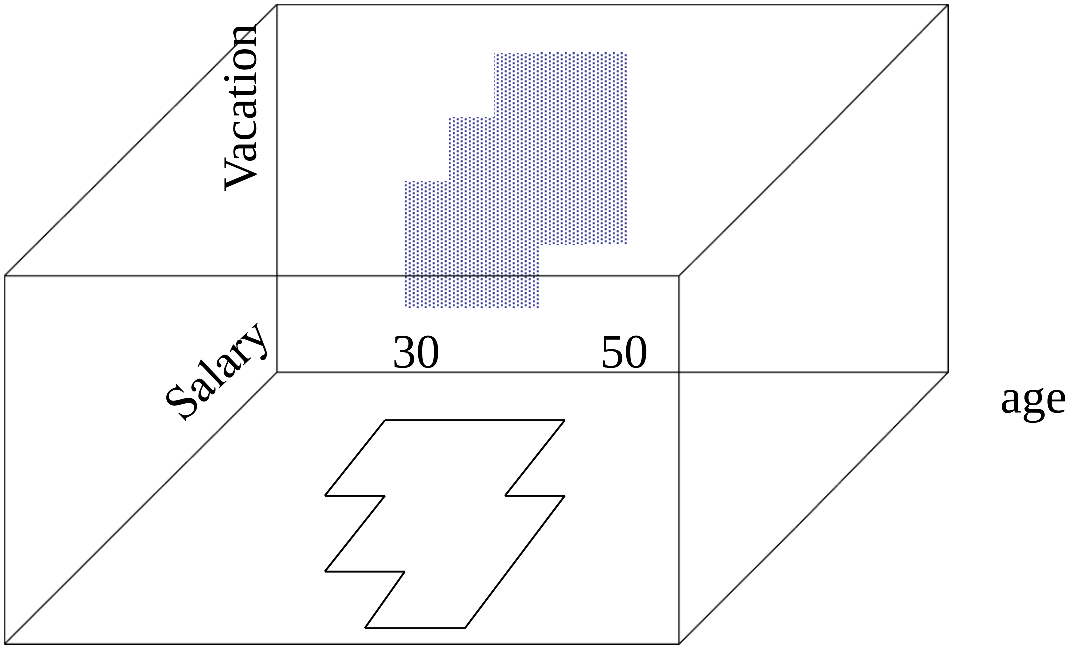
CLIQUE: The Major Steps

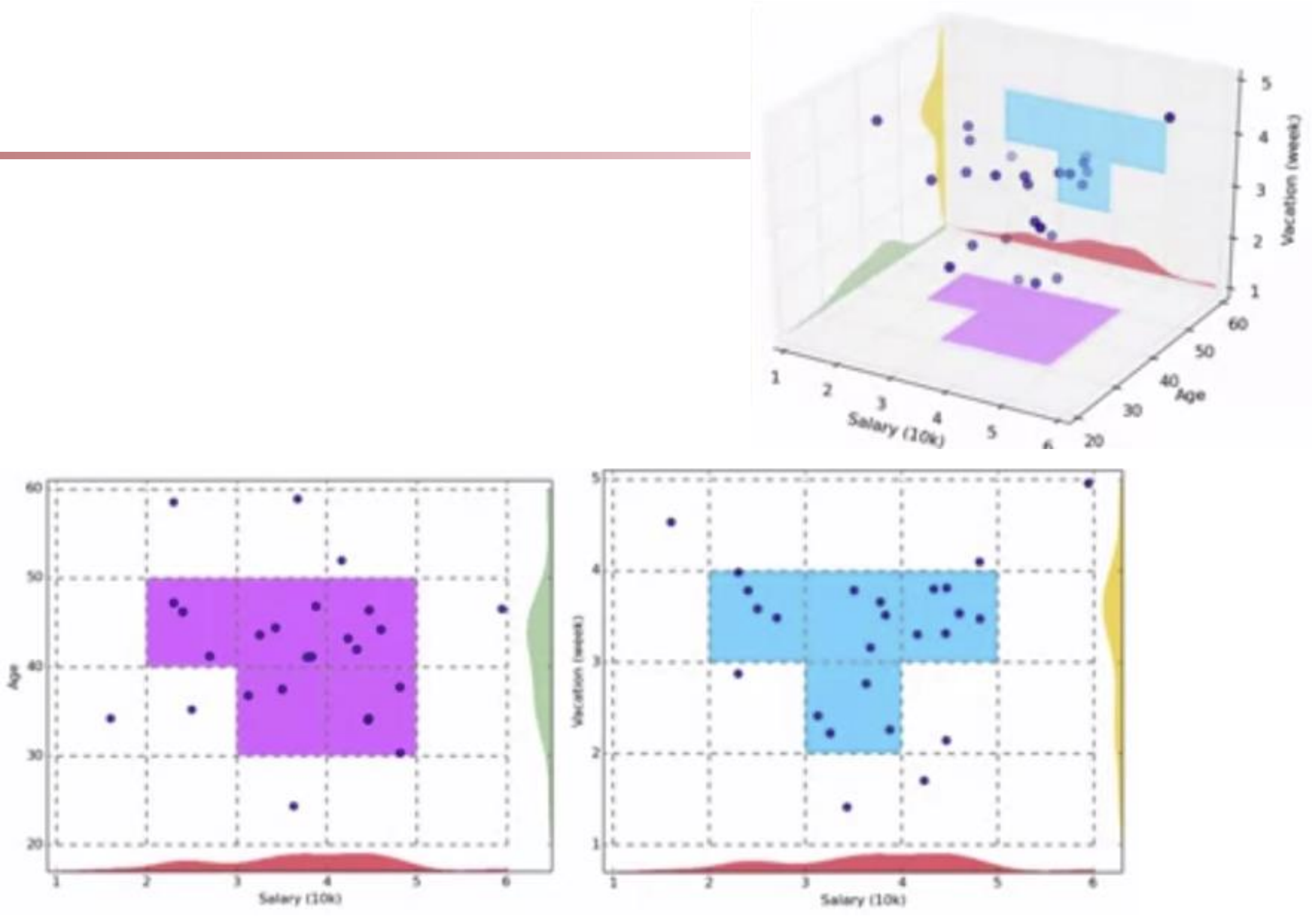
- Partition the data space and find the number of points that lie inside each cell of the partition.
- Identify the subspaces that contain clusters using the Apriori principle
- Identify clusters
  - Determine dense units in all subspaces of interests
  - Determine connected dense units in all subspaces of interests.
- Generate minimal description for the clusters
  - Determine maximal regions that cover a cluster of connected dense units for each cluster
  - Determination of minimal cover for each cluster


Salary
(10,000)
0
1
2
3
4
5
6
7
20
30
40
50
60
age


Vacation
(week)
0
1
2
3
4
5
6
7
20
30
40
50
60
age

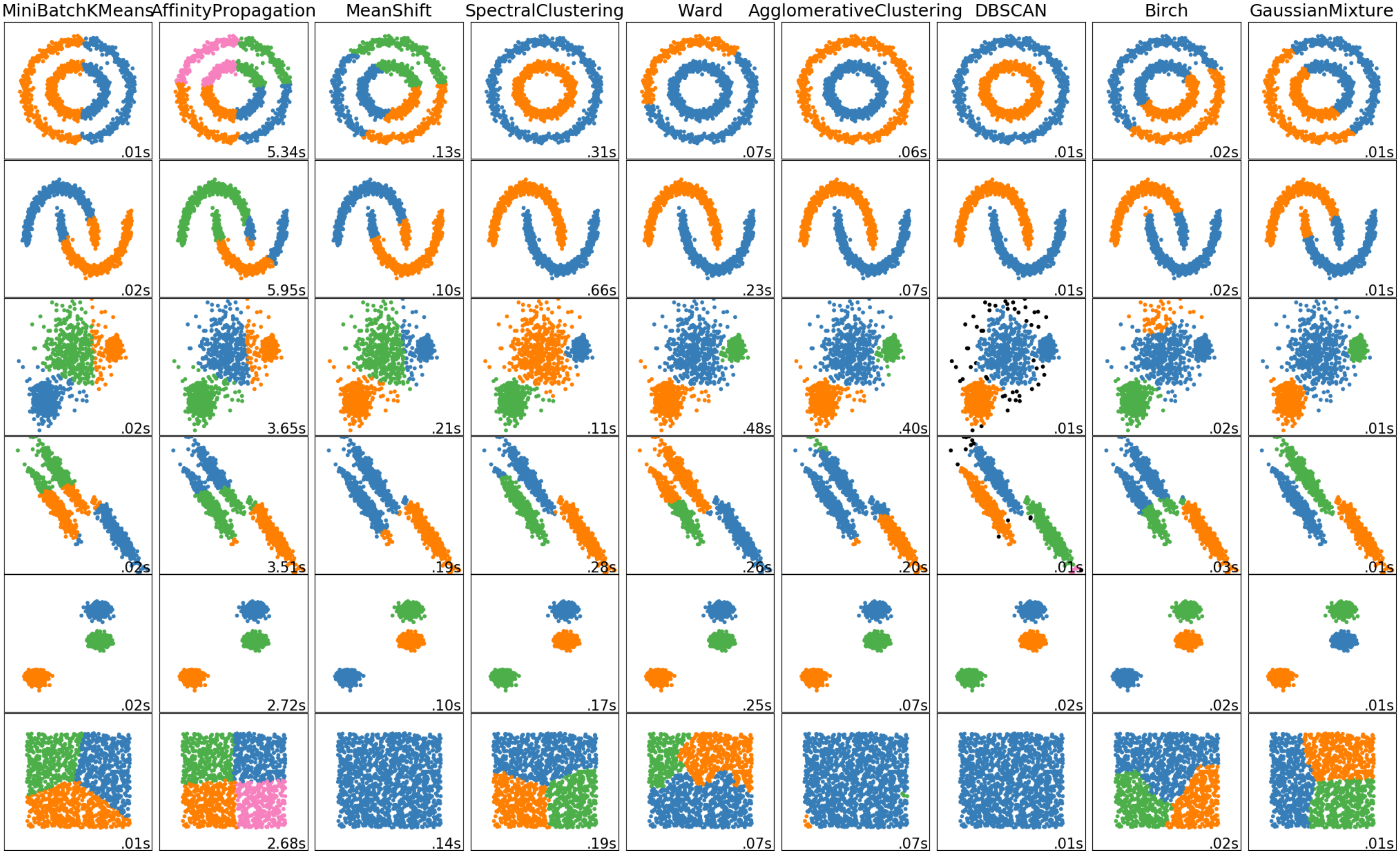
τ = 3


Vacation
Salary
30
50
age

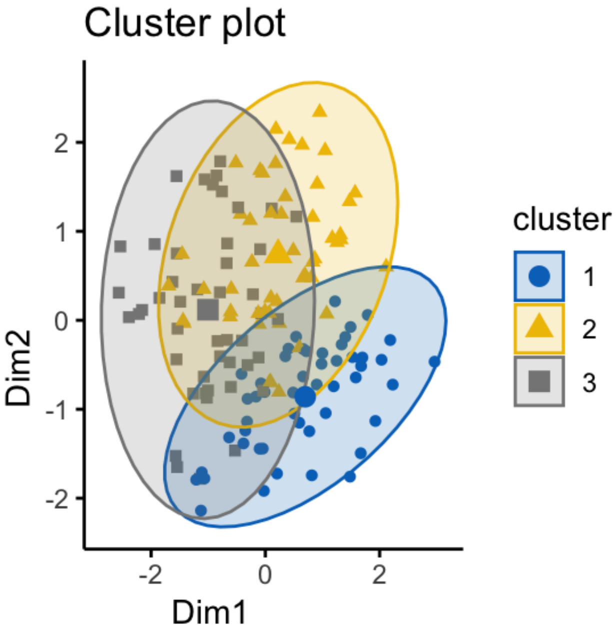

Vacation (week)
5
4
3
2
1
60
50
40
Age
30
20
1
2
3
4
5
6
Salary (10k)
60
50
40
30
20
Age
1
2
3
4
5
6
Salary (10k)
5
4
3
2
1
Vacation (week)
1
2
3
4
5
6
Salary (10k)

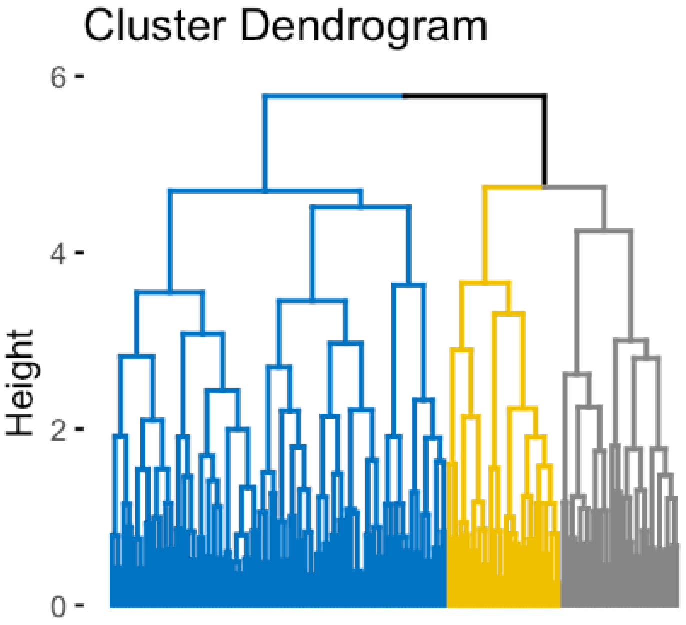
Strength and Weakness of *CLIQUE*

- Strength
  - *automatically* finds subspaces of the highest dimensionality such that high density clusters exist in those subspaces
  - *insensitive* to the order of records in input and does not presume some canonical data distribution
  - scales *linearly* with the size of input and has good scalability as the number of dimensions in the data increases
- Weakness
  - The accuracy of the clustering result may be degraded at the expense of simplicity of the method


MiniBatchKMeans
AffinityPropagation
MeanShift
SpectralClustering
Ward
AgglomerativeClustering
DBSCAN
Birch
GaussianMixture
.01s
5.34s
.13s
.31s
.07s
.06s
.01s
.02s
.01s
.02s
5.95s
.10s
.66s
.23s
.07s
.01s
.02s
.01s
.02s
3.65s
.21s
.11s
.48s
.40s
.01s
.02s
.01s
.02s
3.51s
.19s
.28s
.26s
.20s
.01s
.03s
.01s
.02s
2.72s
.10s
.17s
.25s
.07s
.02s
.02s
.01s
.01s
2.68s
.14s
.19s
.07s
.07s
.01s
.02s
.01s

Assessing Clustering Tendency





Hopkins Static

- **Null hypothesis**: the data set D is uniformly distributed (i.e., no meaningful clusters)
  - Sample *n* points, $p_1, \ldots, p_n$, uniformly from D. For each p_i, find its nearest neighbor in D: $x_i = min\{dist\ (p_i, v)\}$ where *v* in D
  - Generate *n* points, $q_1, \ldots, q_n$, with the same parameters as D (variance). For each q_i, find its nearest neighbor in D – $\{q_i\}$: $y_i = min\{dist\ (q_i, v)\}$ where *v* in D and v ≠ q_i
  - Calculate the Hopkins Statistic:

$$H = \frac{\sum_{i=1}^{n} y_i}{\sum_{i=1}^{n} x_i + \sum_{i=1}^{n} y_i}$$

- If D is uniformly distributed, $\sum x_i$ and $\sum y_i$ will be close to each other and H is close to 0.5.
- A value for H higher than 0.75 indicates a clustering tendency at the 90% confidence level.

Measuring Clustering Quality

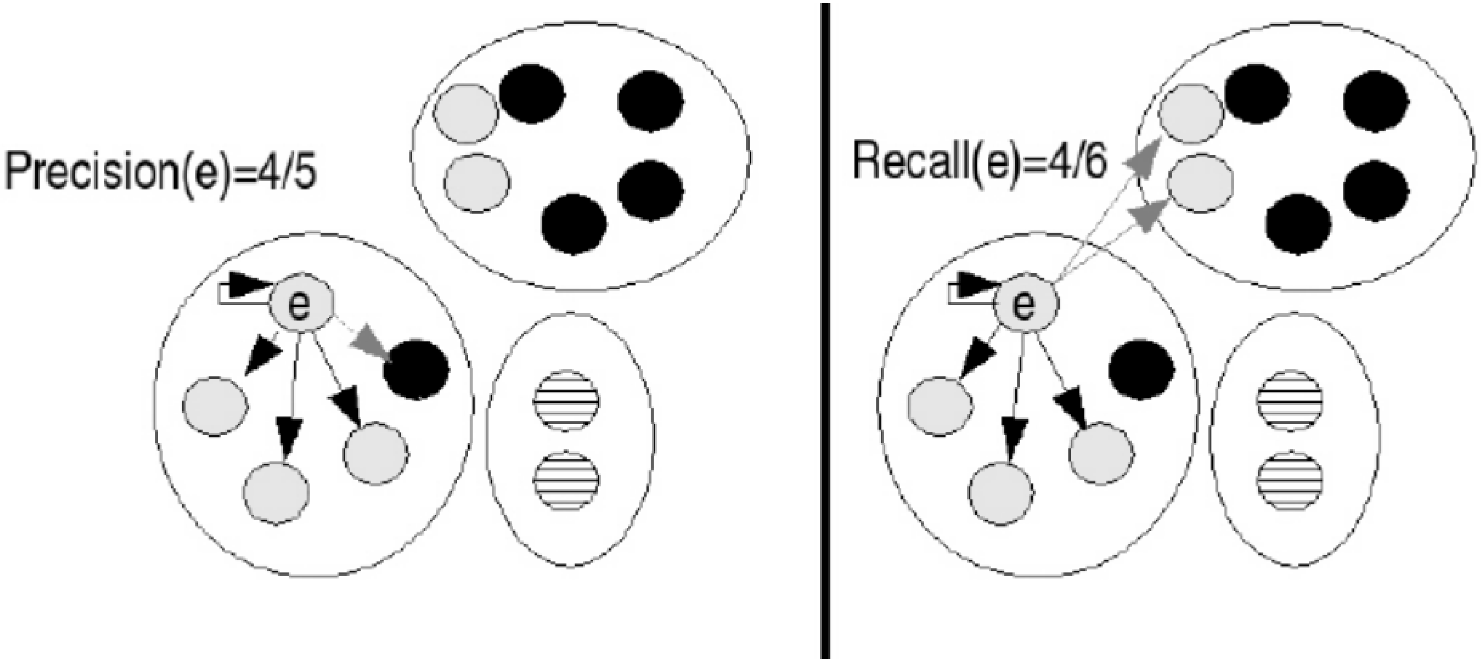
- Two methods: extrinsic vs. intrinsic
- Extrinsic: supervised, i.e., the ground truth is available
    - Compare a clustering against the ground truth using certain clustering quality measure
    - Ex. BCubed precision and recall metrics
- Intrinsic: unsupervised, i.e., the ground truth is unavailable
    - Evaluate the goodness of a clustering by considering how well the clusters are separated, and how compact the clusters are
    - Ex. Silhouette coefficient

Measuring Clustering Quality: Extrinsic Methods

- Clustering quality measure: $Q(C, C_g)$, for a clustering C given the ground truth C_g.
- Q is good if it satisfies the following **4** essential criteria
  - Cluster homogeneity: the purer, the better
  - Cluster completeness: should assign objects belong to the same category in the ground truth to the same cluster
  - Rag bag: putting a heterogeneous object into a pure cluster should be penalized more than putting it into a *rag bag* (i.e., “miscellaneous” or “other” category)
  - Small cluster preservation: splitting a small category into pieces is more harmful than splitting a large category into pieces

BCubed precision and recall metrics

- Let D={o1 , ..., on } be a set of objects, and C be a clustering on D.
- Let L(oi) (1 ≤ i ≤ n) be the category of oi given by ground truth.
- C(oi) be the cluster ID of oi in C.
- The **precision** of an object indicates how many other objects in the same cluster belong to the same category as the object.
- The **recall** of an object reflects how many objects of the same category are assigned to the same cluster.


Precision(e)=4/5
e
Recall(e)=4/6
e

$$\text{Correctness}(\boldsymbol{o_i}, \boldsymbol{o_j}) = \begin{cases} 1 & \text{if } L(\boldsymbol{o_i}) = L(\boldsymbol{o_j}) \Leftrightarrow C(\boldsymbol{o_i}) = C(\boldsymbol{o_j}) \\ 0 & \text{otherwise.} \end{cases}$$

$$\textbf{BCubed precision} = \frac{\displaystyle\sum_{i=1}^{n} \frac{\displaystyle\sum_{\boldsymbol{o_j}: i \neq j, C(\boldsymbol{o_i}) = C(\boldsymbol{o_j})} \text{Correctness}(\boldsymbol{o_i}, \boldsymbol{o_j})}{\| \{\boldsymbol{o_j} | i \neq j, C(\boldsymbol{o_i}) = C(\boldsymbol{o_j})\} \|}}{n}.$$

$$\textbf{BCubed recall} = \frac{\displaystyle\sum_{i=1}^{n} \frac{\displaystyle\sum_{\boldsymbol{o_j}: i \neq j, L(\boldsymbol{o_i}) = L(\boldsymbol{o_j})} \text{Correctness}(\boldsymbol{o_i}, \boldsymbol{o_j})}{\| \{\boldsymbol{o_j} | i \neq j, L(\boldsymbol{o_i}) = L(\boldsymbol{o_j})\} \|}}{n}$$

Intrinsic Method: Silhouette coefficient

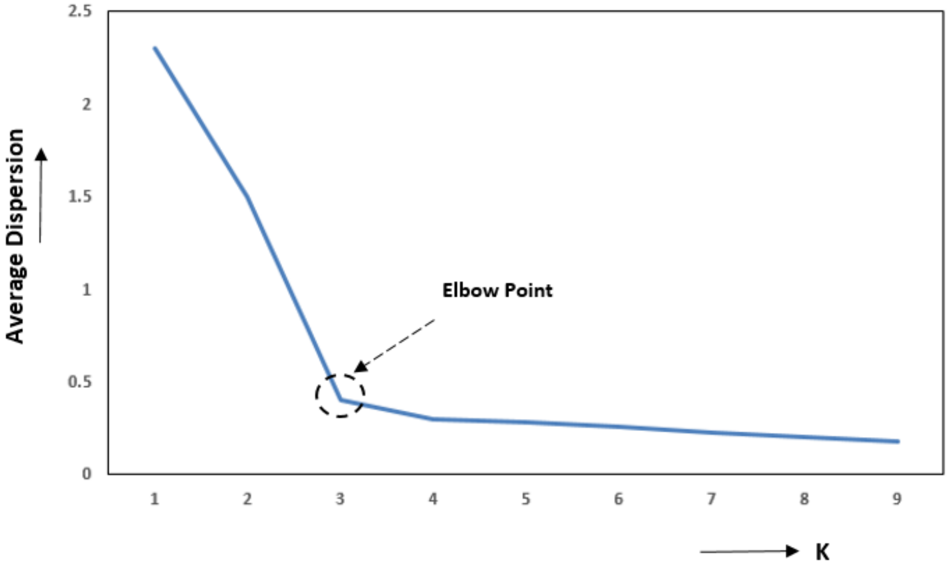
- a(o): the average distance between o and all other objects in the cluster to which o belongs.
- b(o) is the minimum average distance from o to all clusters to which o does not belong.
- **Compactness**: a(o)
- the degree to which o is **separated** from other clusters: b(o)

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

- 1: ☺   -1 ☹

Determine the Number of Clusters

- **Empirical method**
  - # of clusters ≈√n/2 for a dataset of n points
- **Elbow method**
  - Use the turning point in the curve of sum of within cluster variance w.r.t the # of clusters
- **Cross validation method**



Elbow Method for selection of optimal "K" clusters

Determine the Number of Clusters

- **Cross validation method**
  - Divide a given data set into *m* parts
  - Use *m* – 1 parts to obtain a clustering model
  - Use the remaining part to test the quality of the clustering
    - E.g., For each point in the test set, find the closest centroid, and use the sum of squared distance between all points in the test set and the closest centroids to measure how well the model fits the test set
  - For any k > 0, repeat it *m* times, compare the overall quality measure w.r.t. different *k's*, and find # of clusters that fits the data the best