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

Chapter 10. Cluster Analysis: Basic Concepts and Methods

- Cluster Analysis: Basic Concepts
- Partitioning Methods
- Hierarchical Methods
- Density-Based Methods
- Grid-Based Methods
- Evaluation of Clustering
- Summary

What is Cluster Analysis?

- Cluster: A collection of data objects
  - similar (or related) to one another within the same group
  - dissimilar (or unrelated) to the objects in other groups
- Cluster analysis (or *clustering*, *data segmentation, …*)
  - Finding similarities between data according to the characteristics found in the data and grouping similar data objects into clusters
- Unsupervised learning: no predefined classes (i.e., *learning by observations* vs. learning by examples: supervised)
- Typical applications
  - As a stand-alone tool to get insight into data distribution
  - As a preprocessing step for other algorithms

Clustering for Data Understanding and Applications

- Biology: taxonomy of living things: kingdom, phylum, class, order, family, genus and species
- Information retrieval: document clustering
- Land use: Identification of areas of similar land use in an earth observation database
- Marketing: Help marketers discover distinct groups in their customer bases, and then use this knowledge to develop targeted marketing programs
- City-planning: Identifying groups of houses according to their house type, value, and geographical location
- Earth-quake studies: Observed earth quake epicenters should be clustered along continent faults
- Climate: understanding earth climate, find patterns of atmospheric and ocean
- Economic Science: market resarch

Clustering as a Preprocessing Tool (Utility)

- Summarization:
  - Preprocessing for regression, PCA, classification, and association analysis
- Compression:
  - Image processing: vector quantization
- Finding K-nearest Neighbors
  - Localizing search to one or a small number of clusters
- Outlier detection
  - Outliers are often viewed as those “far away” from any cluster

Quality: What Is Good Clustering?

- A good clustering method will produce high quality clusters
  - high intra-class similarity: cohesive within clusters
  - low inter-class similarity: distinctive between clusters
- The quality of a clustering method depends on
  - the similarity measure used by the method
  - its implementation, and
  - Its ability to discover some or all of the hidden patterns

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

Requirements and Challenges

- Scalability
  - Clustering all the data instead of only on samples
- Ability to deal with different types of attributes
  - Numerical, binary, categorical, ordinal, linked, and mixture of these
- Constraint-based clustering
  - User may give inputs on constraints
  - Use domain knowledge to determine input parameters
- Interpretability and usability
- Others
  - Discovery of clusters with arbitrary shape
  - Ability to deal with noisy data
  - Incremental clustering and insensitivity to input order
  - High dimensionality

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)

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

Major Clustering Approaches (II)

- Model-based:
  - A model is hypothesized for each of the clusters and tries to find the best fit of that model to each other
  - Typical methods: EM, SOM, COBWEB
- Frequent pattern-based:
  - Based on the analysis of frequent patterns
  - Typical methods: p-Cluster
- User-guided or constraint-based:
  - Clustering by considering user-specified or application-specific constraints
  - Typical methods: COD (obstacles), constrained clustering
- Link-based clustering:
  - Objects are often linked together in various ways
  - Massive links can be used to cluster objects: SimRank, LinkClus

Chapter 10. Cluster Analysis: Basic Concepts and Methods

- Cluster Analysis: Basic Concepts
- Partitioning Methods
- Hierarchical Methods
- Density-Based Methods
- Grid-Based Methods
- Evaluation of Clustering
- Summary

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)

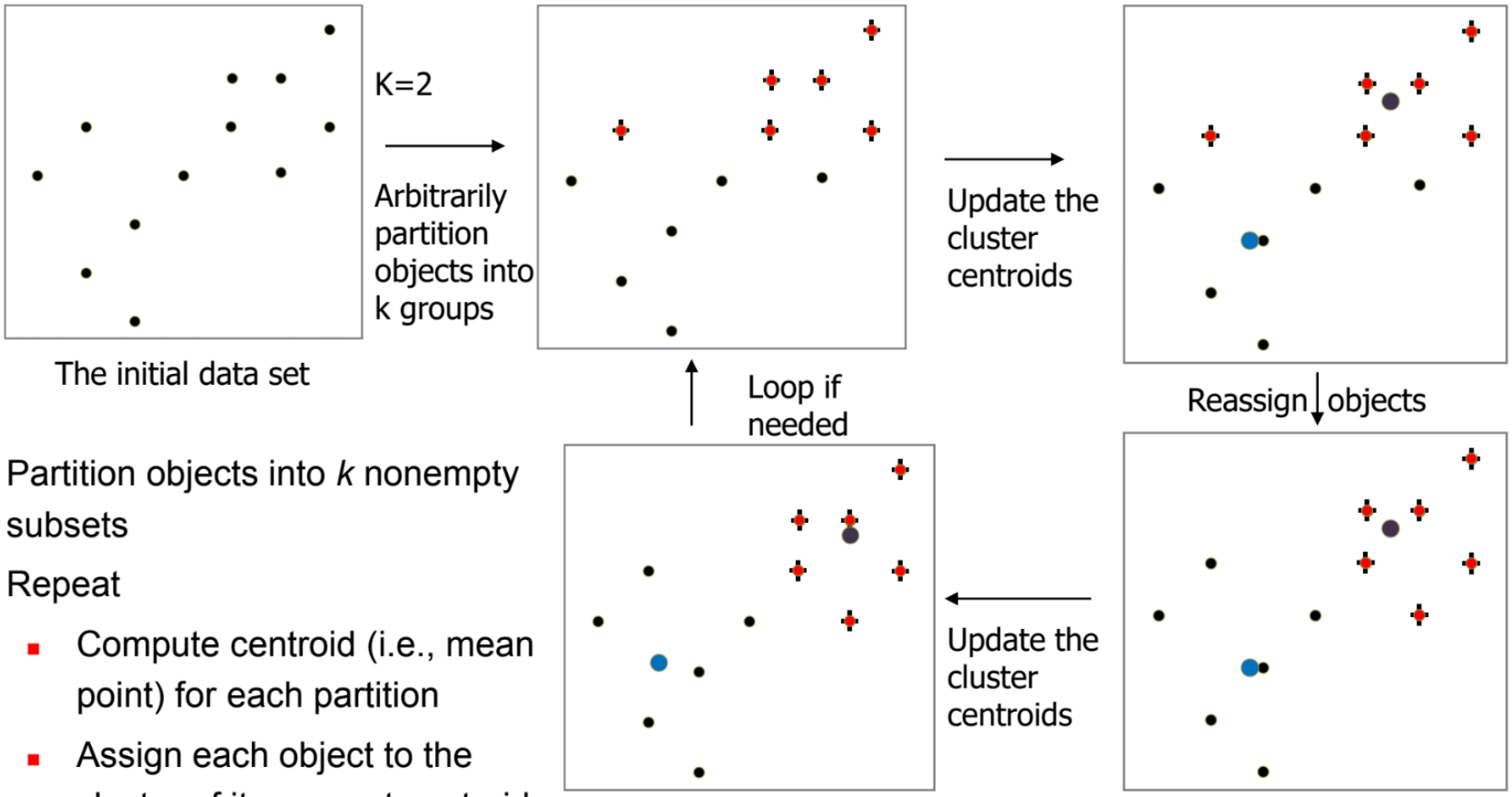
$$E = \Sigma_{i=1}^{k} \Sigma_{p \in C_i} (p - c_i)^2$$

- 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

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

An Example of *K-Means* Clustering



- Partition objects into *k* nonempty subsets
- Repeat
  - Compute centroid (i.e., mean point) for each partition
  - Assign each object to the cluster of its nearest centroid
- Until no change

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:

(1) arbitrarily choose *k* objects from *D* as the initial cluster centers;
(2) **repeat**
(3) (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;
(4) update the cluster means, that is, calculate the mean value of the objects for each cluster;
(5) **until** no change;

Figure 10.2 The *k*-means partitioning algorithm.

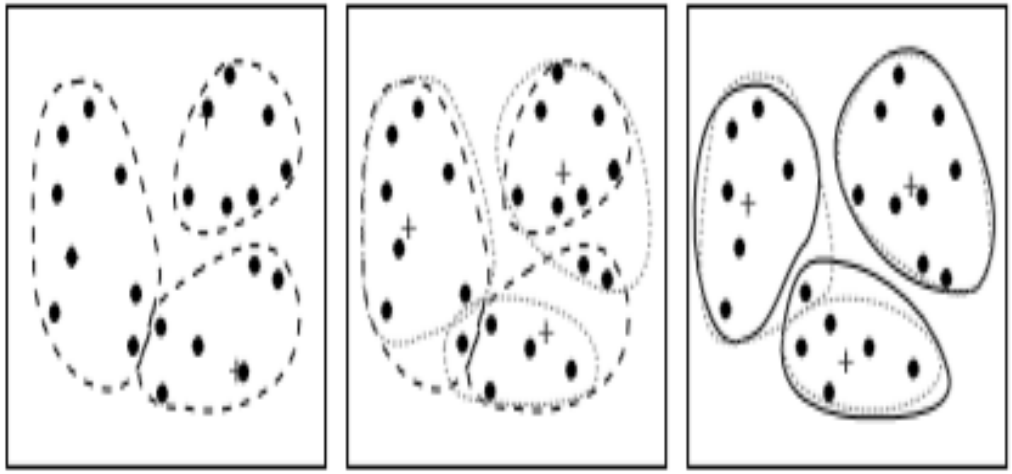


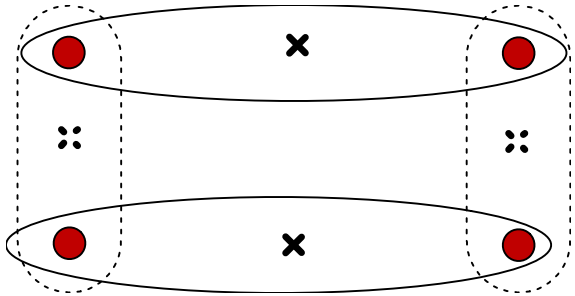
Figure 10.3 Clustering of a set of objects using the *k*-means method; for (b) update cluster centers and reassign objects accordingly (the mean of each cluster is marked by a +).

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

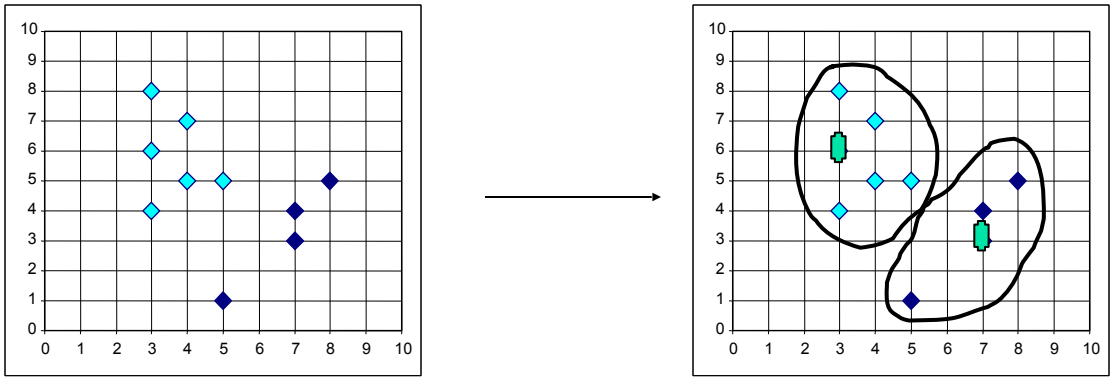
- Most of the variants of the *k-means* which differ in
  - Selection of the initial *k* means
  - Dissimilarity calculations
  - Strategies to calculate cluster means



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

What Is the Problem of 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



Exercise

- Cluster points 1, 2, 3, 8,9,10 into two clusters
- Repeat the clustering when there is an outlier like 25 in the list. 1, 2, 3, 8,9,10, 25.

Example 10.2 **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

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

Application-Discussion

- Suppose that you are to allocate a number of automatic teller machines (ATMs) in a given region so as to satisfy a number of constraints. Households or workplaces may be clustered so that typically one ATM is assigned per cluster. The clustering, however, may be constrained by two factors:(1) obstacle objects (i.e.,the reare bridges,rivers, and high ways that can affect ATM accessibility),and

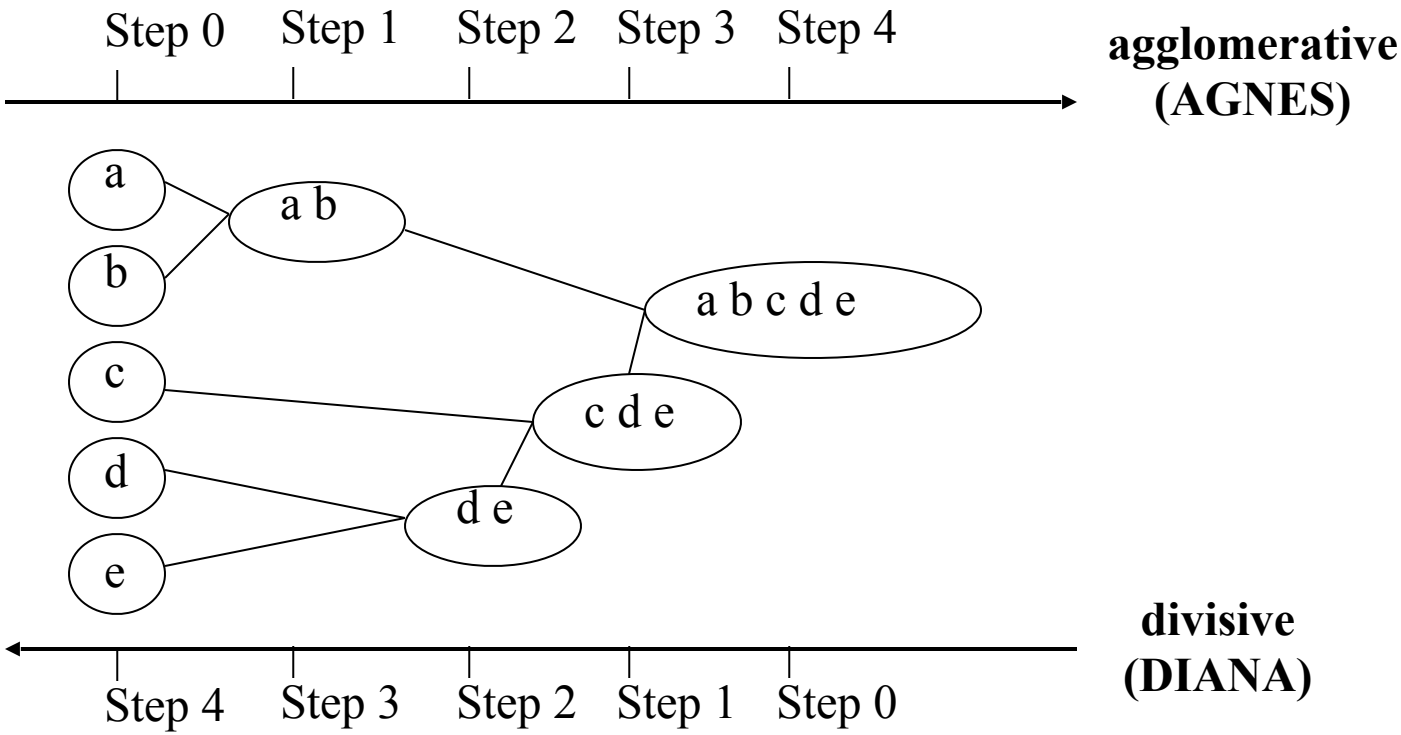
  (2)additional user-specified constraints such as that each ATM should serve at least 10,000 households. How can a clustering algorithm such as k-means be modified for quality clustering under both constraints?

Chapter 10. Cluster Analysis: Basic Concepts and Methods

- Cluster Analysis: Basic Concepts
- Partitioning Methods
- Hierarchical Methods
- Density-Based Methods
- Grid-Based Methods
- Evaluation of Clustering
- Summary

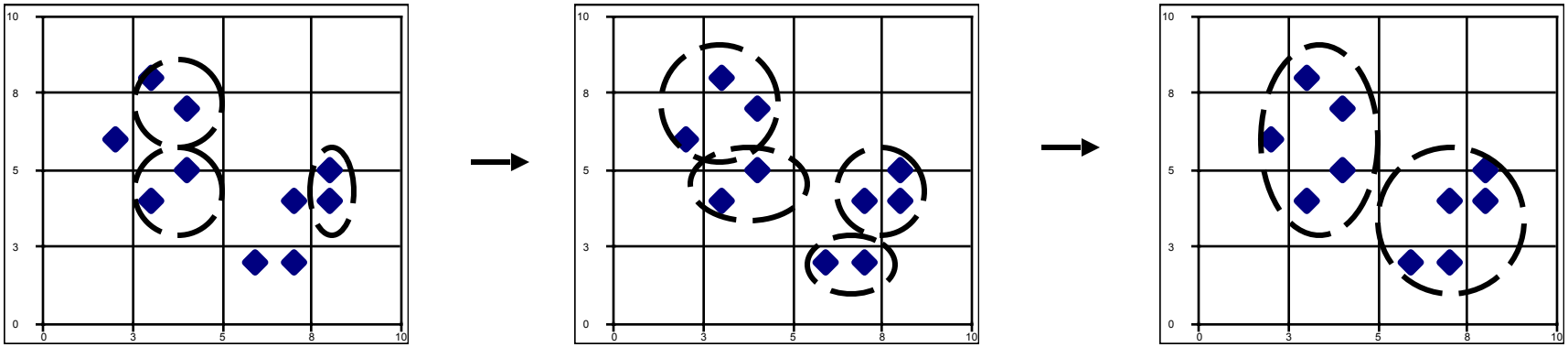
Hierarchical Clustering

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

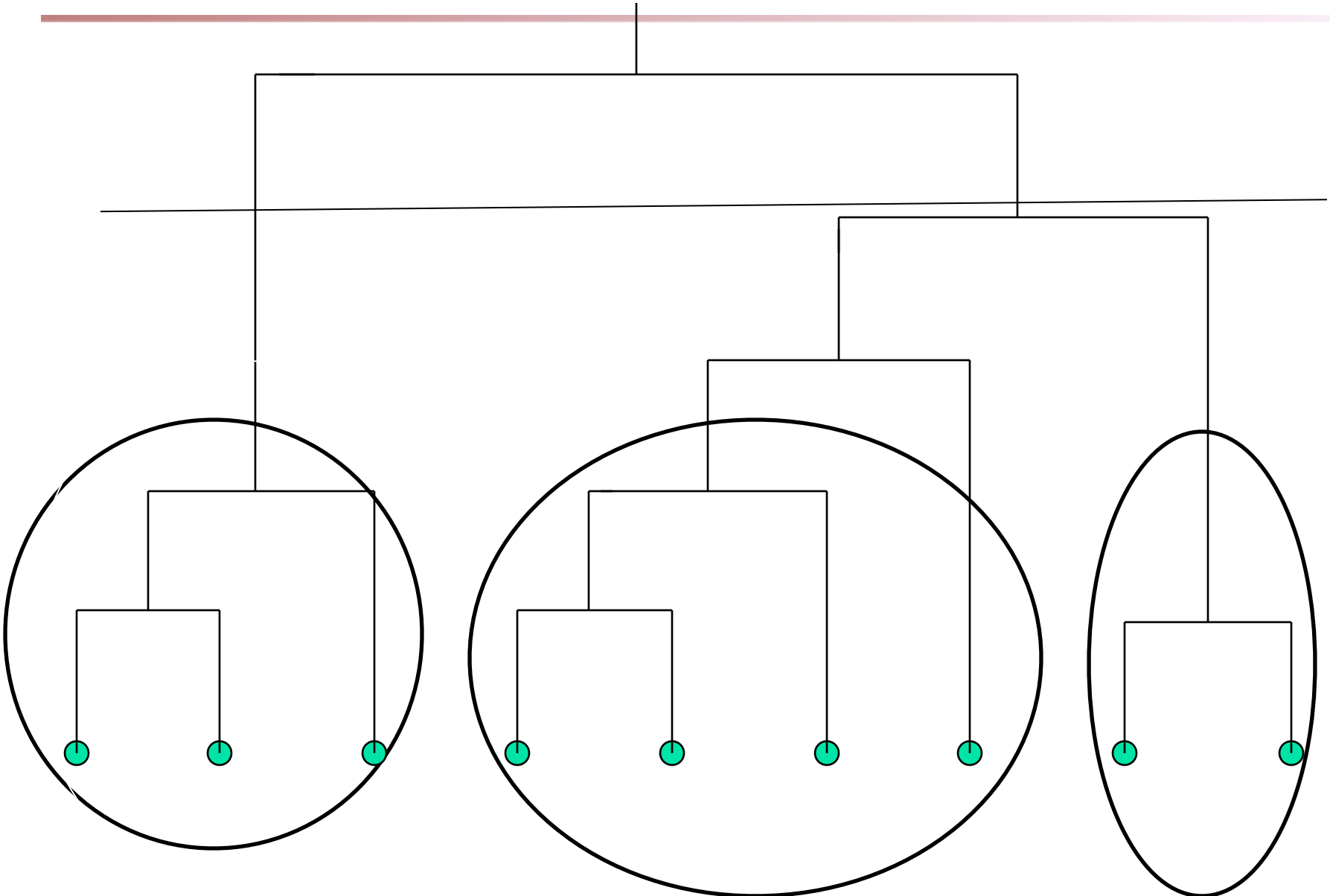


AGNES (Agglomerative Nesting)

- 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



Dendrogram: Shows How Clusters are Merged



Dendrogram: Shows How Clusters are Merged

Decompose data objects into a several levels of nested partitioning (tree of clusters), called a dendrogram

A clustering of the data objects is obtained by cutting the dendrogram at the desired level, then each connected component forms a cluster

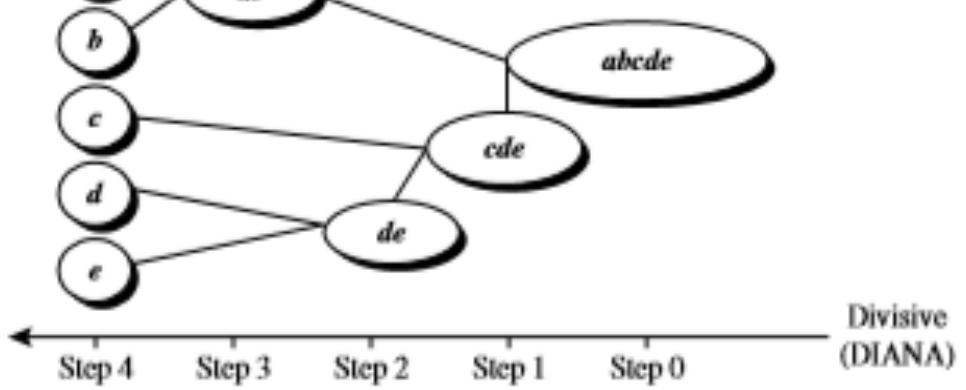


Figure 10.6 Agglomerative and divisive hierarchical clustering on data objects $\{a, b, c, d, e\}$.

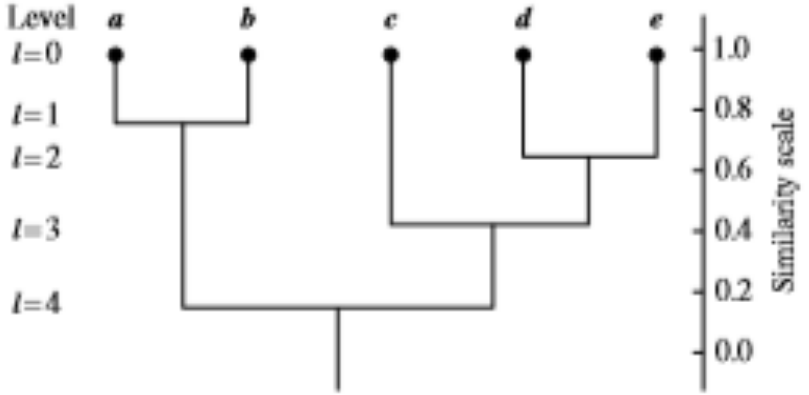
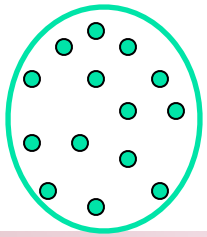
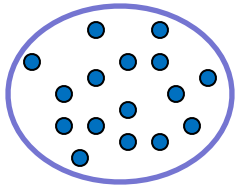


Figure 10.7 Dendrogram representation for hierarchical clustering of data objects $\{a, b, c, d, e\}$.

Distance between Clusters





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

Centroid, Radius and Diameter of a Cluster (for numerical data sets)

- Centroid: the "middle" of a cluster

$$C_m = \frac{\sum_{i=1}^{N} (t_{ip})}{N}$$

- Radius: square root of average distance from any point of the cluster to its centroid

$$R_m = \sqrt{\frac{\sum_{i=1}^{N} (t_{ip} - c_m)^2}{N}}$$

- Diameter: square root of average mean squared distance between all pairs of points in the cluster

$$D_m = \sqrt{\frac{\sum_{i=1}^{N} \sum_{i=1}^{N} (t_{ip} - t_{iq})^2}{N(N-1)}}$$

Exercise

- C1 {2,3,7} C2 {20, 21, 25}
- Compute the following:
- Centroids
- Medoids
- Single link distance
- Complete Link distance
- Average Distance
- Radius of the clusters
- Diameter of the clusters

Distance matrix

C2

	20	21	25
2	18	19	23
C1 3	17	18	22
7	13	14	18

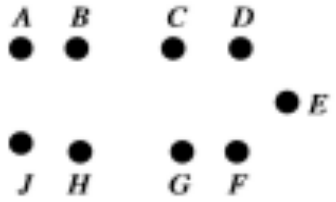
Centroid - C1 = 4 C2 = 22
Medoid – C1 = 3 C2 = 21
Single Link(C1. C2) – 13
Complete Link(C1,C2)- 23

	2	3	7
2	0	1	25
3	1	0	16
7	25	16	0

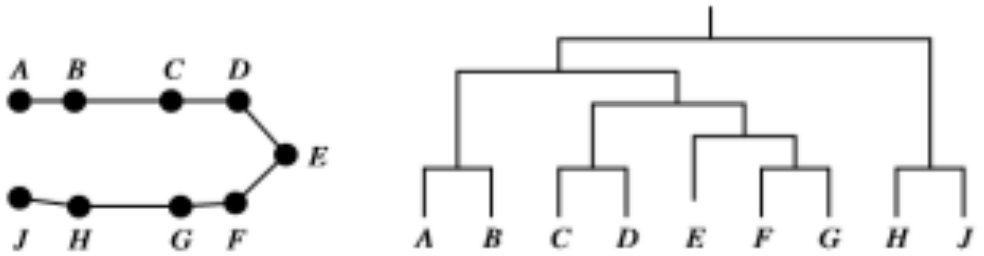
Radius – C1 = sqrt((4+1+9)/3)
C2 = sqrt((4+1+9)/3)
Diameter- C1 = sqrt(84/6)
C2 = sqrt(84/6)

	20	21	25
20	0	1	25
21	1	0	16
25	25	16	0

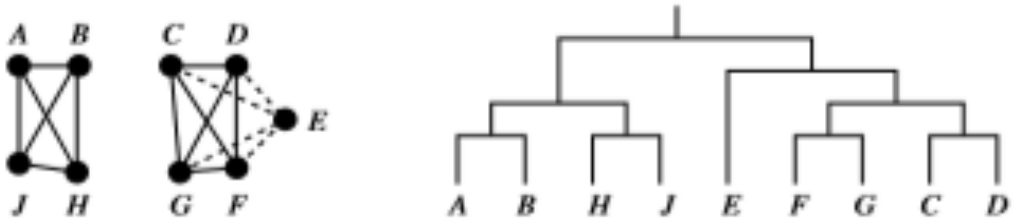
Use of Single Linkage/Compete Linkage in AGNES



(a) Data set



(b) Clustering using single linkage

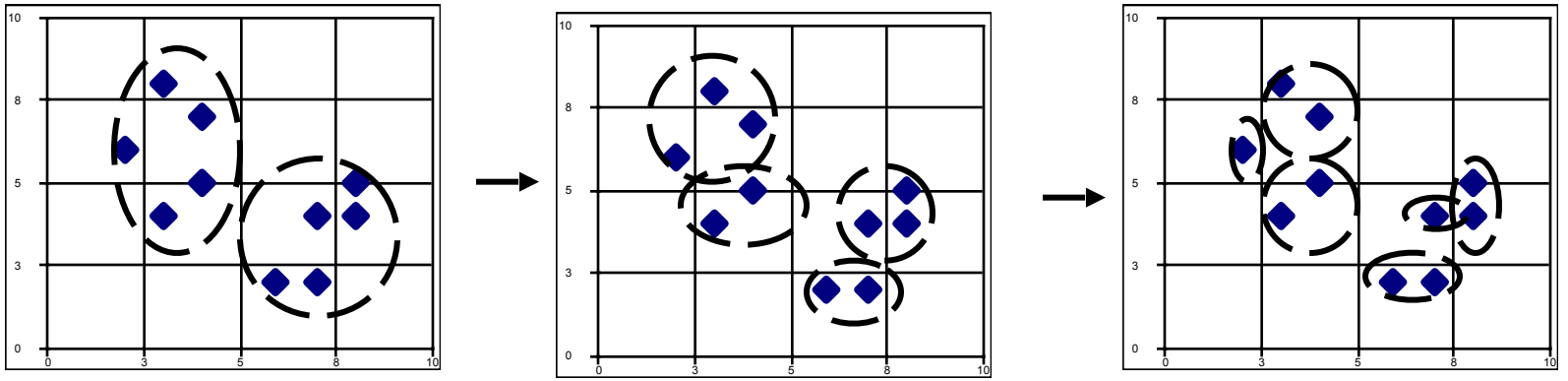


(c) Clustering using complete linkage

Figure 10.8 Hierarchical clustering using single and complete linkages.

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



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

BIRCH (Balanced Iterative Reducing and Clustering Using Hierarchies)

- Zhang, Ramakrishnan & Livny, SIGMOD'96
- Incrementally construct a CF (Clustering Feature) tree, a hierarchical data structure for multiphase clustering
  - Phase 1: scan DB to build an initial in-memory CF tree (a multi-level compression of the data that tries to preserve the inherent clustering structure of the data)
  - Phase 2: use an arbitrary clustering algorithm to cluster the leaf nodes of the CF-tree
- *Scales linearly*: finds a good clustering with a single scan and improves the quality with a few additional scans
- *Weakness:* handles only numeric data, and sensitive to the order of the data record

Clustering Feature Vector in BIRCH

Clustering Feature (CF): ***CF = (N, LS, SS)***

N: **Number of data points**

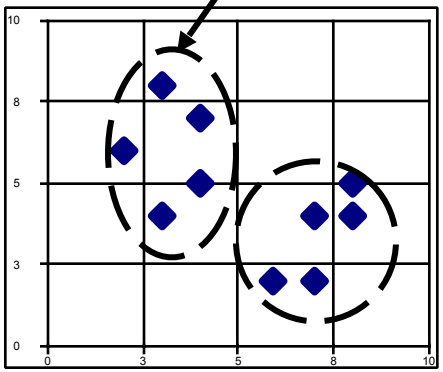
LS: linear sum of N points:

$$\sum_{i=1}^{N} X_i$$

SS: square sum of N points

$$\sum_{i=1}^{N} X_i^2$$

CF = (5, (16,30),(54,190))



(3,4)
(2,6)
(4,5)
(4,7)
(3,8)

Cluster Properties

AVERAGE

$$\mathbf{x_0} = \frac{\sum_{i=1}^{n} \mathbf{x_i}}{n} = \frac{LS}{n},$$

$$R = \sqrt{\frac{\sum_{i=1}^{n} (x_i - x_0)^2}{n}} = \sqrt{\frac{nSS - 2LS^2 + nLS}{n^2}}, \qquad (10.9)$$

$$D = \sqrt{\frac{\sum_{i=1}^{n}\sum_{j=1}^{n} (x_i - x_j)^2}{n(n-1)}} = \sqrt{\frac{2nSS - 2LS^2}{n(n-1)}}. \qquad (10.10)$$

Exercise

a. Compute CF feature for the cluster { (2,5), (3,2), (4,3)}

CF = (3, (9, 10), (29, 38))

b. Compute the CF feature for the cluster got by merging the above cluster, with another cluster with CF (3, (35,36), (417,440))

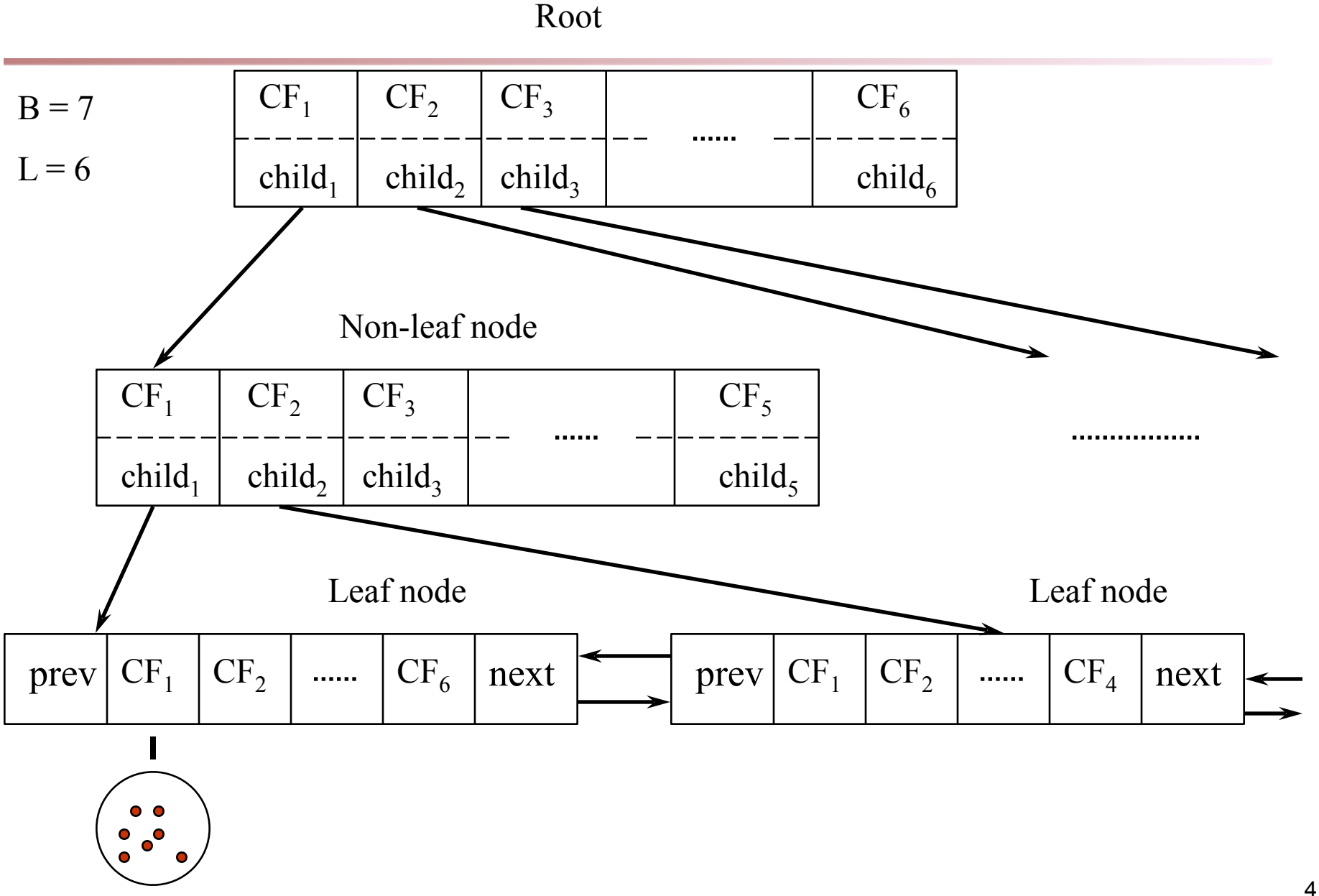
.CF of merged clusters = (6, (44,46), (446, 478))

(3,(9,10),(29,38)) sum of c1 and c2 = c3

CF-Tree in BIRCH

- Clustering feature:
  - Summary of the statistics for a given subcluster: the 0-th, 1st, and 2nd moments of the subcluster from the statistical point of view
  - Registers crucial measurements for computing cluster and utilizes storage efficiently
- A CF tree is a height-balanced tree that stores the clustering features for a hierarchical clustering
  - A nonleaf node in a tree has descendants or "children"
  - The nonleaf nodes store sums of the CFs of their children
- A CF tree has two parameters
  - Branching factor: max # of children
  - Threshold: max diameter of sub-clusters stored at the leaf nodes

The CF Tree Structure



The Birch Algorithm

- Cluster Diameter

$$\sqrt{\frac{1}{n(n-1)}\sum(x_i - x_j)^2}$$

- For each point in the input
  - Find closest leaf entry
  - Add point to leaf entry and update CF
  - If entry diameter > max_diameter, then split leaf, and possibly parents
- Algorithm is O(n)
- Concerns
  - Sensitive to insertion order of data points
  - Since we fix the size of leaf nodes, so clusters may not be so natural
  - Clusters tend to be spherical given the radius and diameter measures

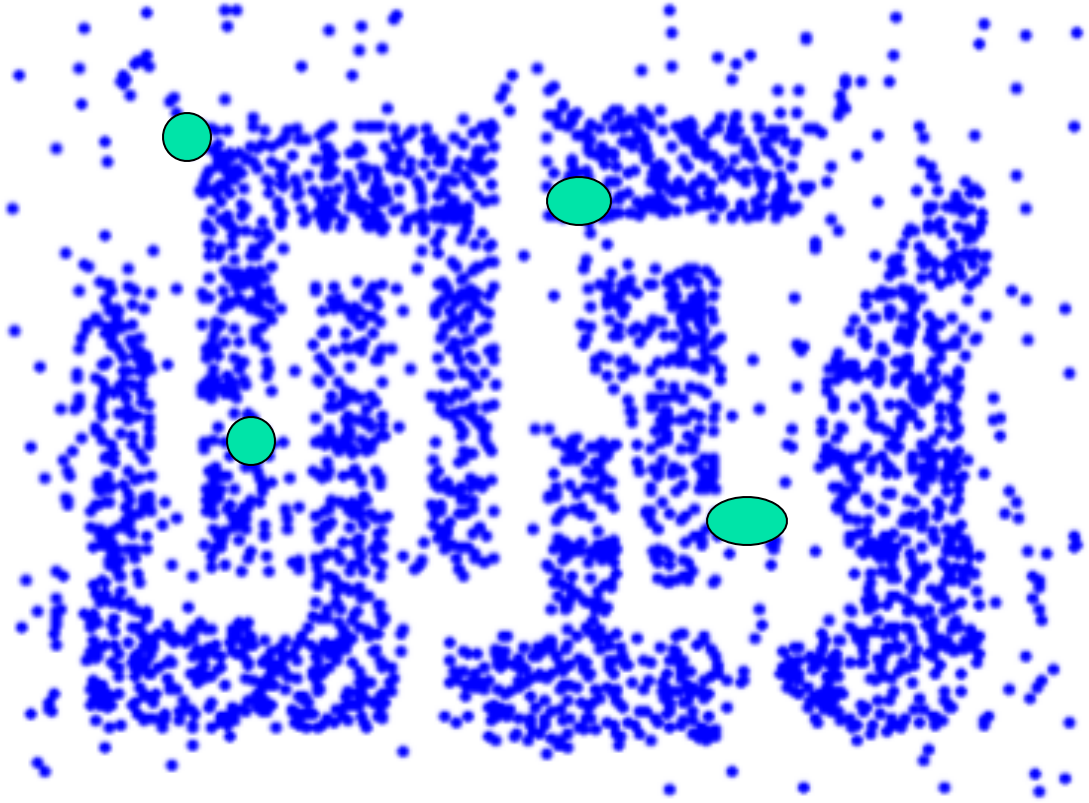
Chapter 10. Cluster Analysis: Basic Concepts and Methods

- Cluster Analysis: Basic Concepts
- Partitioning Methods
- Hierarchical Methods
- Density-Based Methods
- Grid-Based Methods
- Evaluation of Clustering
- Summary

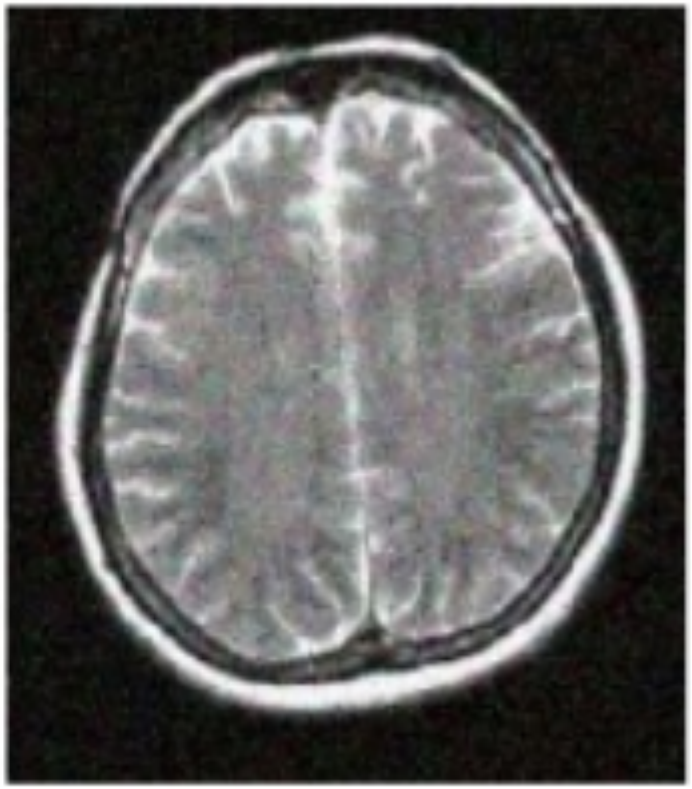
Density-Based Clustering Methods

- 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)

How will K-means work on this?



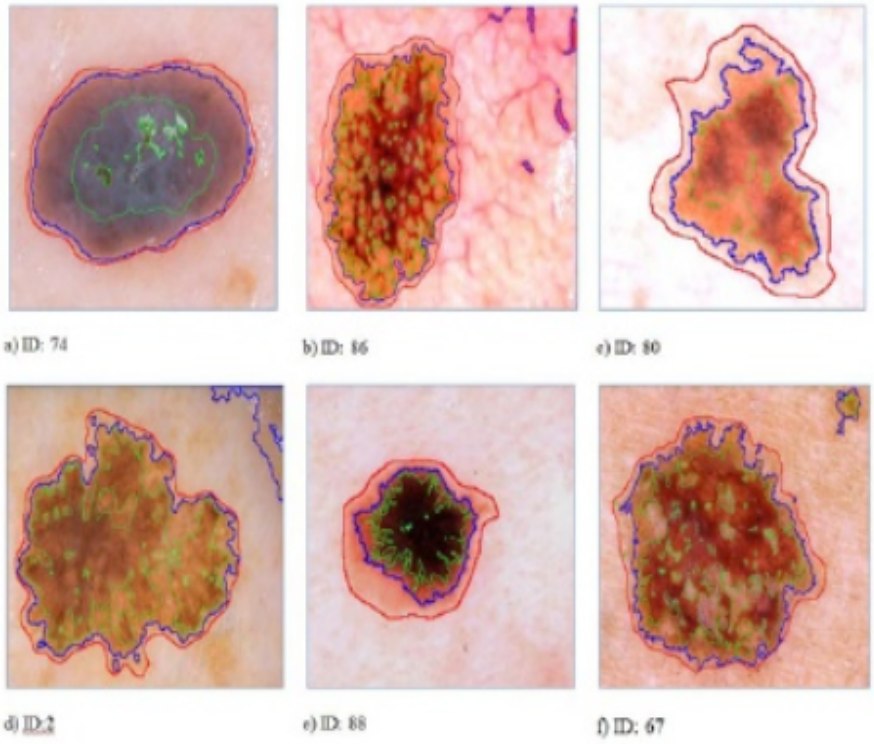
Density Based Clustering application



ed.

Density Based Clustering application

Automatic border detection in dermoscopy images

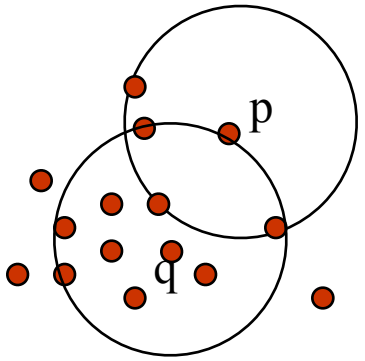


Sample images showing assessments of the dermatologist (red), automated frameworks DBSCAN (blue) and FCM (green).
Kockara *et al. BMC Bioinformatics* 2010 **11**(Suppl 6):S26 doi:10.1186/1471-2105-11-S6-S26

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:

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

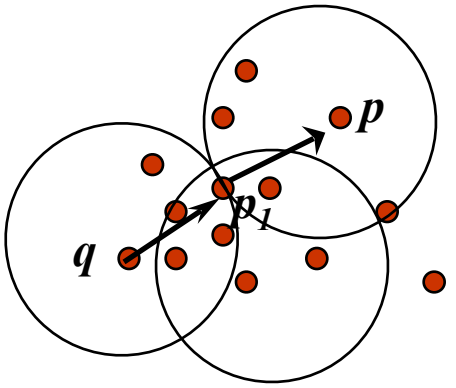


MinPts = 5

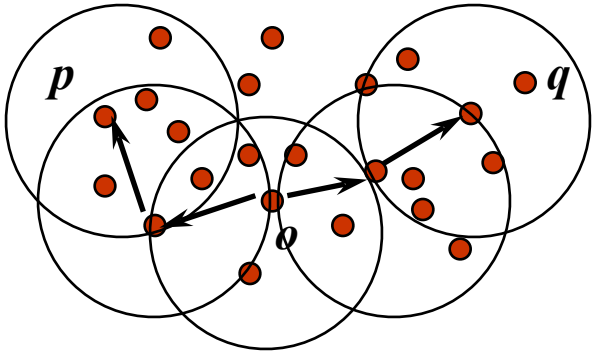
Eps = 1 cm

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

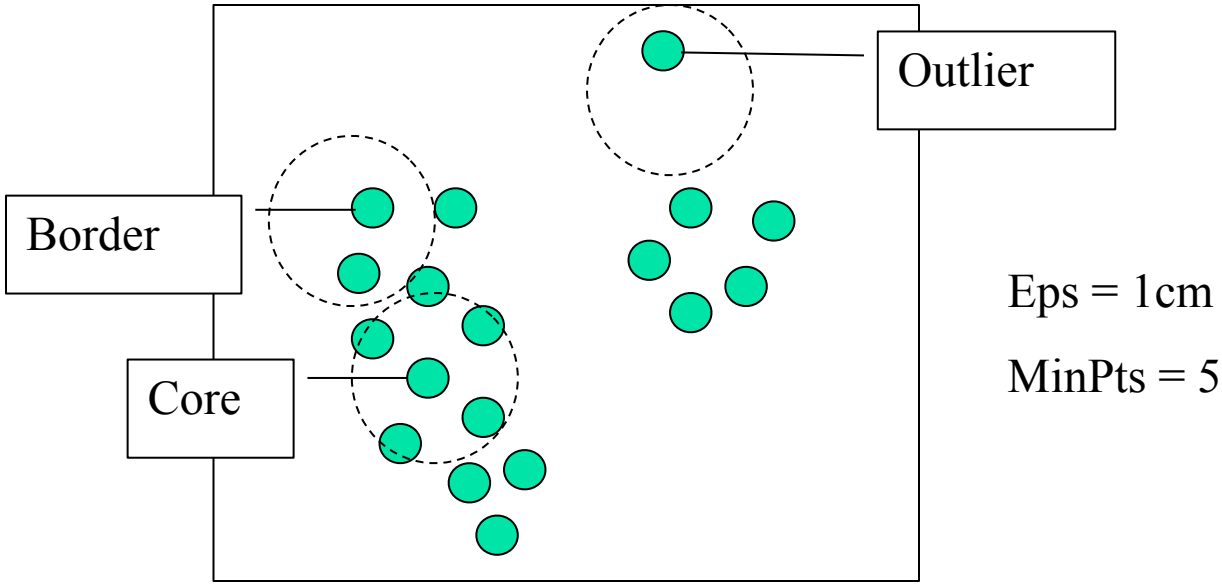


- Density-connected
  - A point *p* is density-connected to a point *q* w.r.t. *Eps*, *MinPts* if there is a point *o* such that both, *p* and *q* are density-reachable from *o* w.r.t. *Eps* and *MinPts*



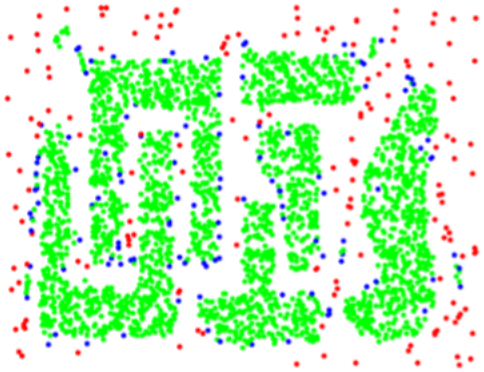
DBSCAN: Density-Based Spatial Clustering of Applications with Noise

- 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



Example

Original Points



Point types: core, border and outliers

$\varepsilon = 10$, MinPts = 4

Exercise

1 a	2 b	3 c	4 d	5 e	6 f		8 g			11 h	12 i	13 j	14 k	19 l

Points :
Above is a list of 1D points a through k,
and their co-ordinates/values are marked above them.

For Eps = 2.5, MinPts = 2,

a.Identify the core points

b.Identify the border points

c.Identify the outliers/noise points

d.Density reachable (g,a)? (a,g)?

(k,g)? (g,k)?

(l,g) (g,l)?

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:

```
(1)  mark all objects as unvisited;
(2)  do
(3)      randomly select an unvisited object p;
(4)      mark p as visited;
(5)      if the ε-neighborhood of p has at least MinPts objects
(6)          create a new cluster C, and add p to C;
(7)          let N be the set of objects in the ε-neighborhood of p;
(8)          for each point p' in N
(9)              if p' is unvisited
(10)                 mark p' as visited;
(11)                 if the ε-neighborhood of p' has at least MinPts points,
                     add those points to N;
(12)             if p' is not yet a member of any cluster, add p' to C;
(13)         end for
(14)         output C;
```

Exercise- DBSCAN walkthrough

1 a	2 b	3 c	4 d	5 e	6 f	7	8 g	9	10	11 h	12 i	13 j	14 k	19 l

N-<a. Visited - a. N-< a, b,c Cluster C1 - {a }

N-< a, b,c. Visited - a, b. N-< a, b,c,d. Cluster C1 -{ a ,b}

N-< a, b,c,d. Visited - a, b,c N-< a, b,c,d,e Cluster C1 - {a ,b,c}

N-< a, b,c,d,e. Visited - a, b,c,d N-< a, b,c,d,e,f Cluster C1 - {a ,b,c,d}

N-< a, b,c,d,e,f. Visited - a, b,c,d ,e N-< a, b,c,d,e,f Cluster C1 - {a ,b,c,d,e}

N-< a, b,c,d,e,f Visited - a, b,c,d ,e,f N-< a, b,c,d,e,f,g Cluster C1 - {a ,b,c,d,e,f}

N-< a, b,c,d,e,f,g Visited - a, b,c,d ,e,f,g N-< a, b,c,d,e,f,g Cluster C1 -{a ,b,c,d,e,f,g}

No unvisited point in N

Randomly select another point say ‘h’ and continue processing

U can get the next cluster C2 - {h,i,j,k}

Point ‘l’ is an outlier, it will get visited, but not assigned a cluster label

DBSCAN: Sensitive to Parameters

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

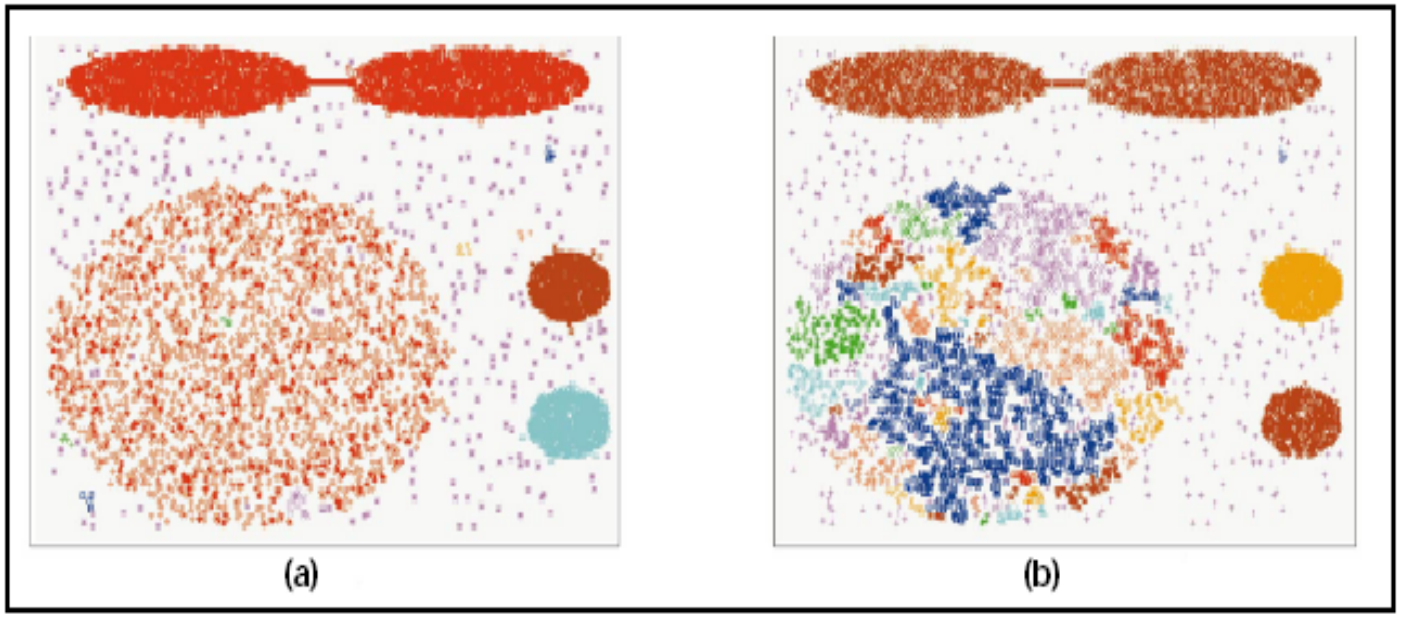
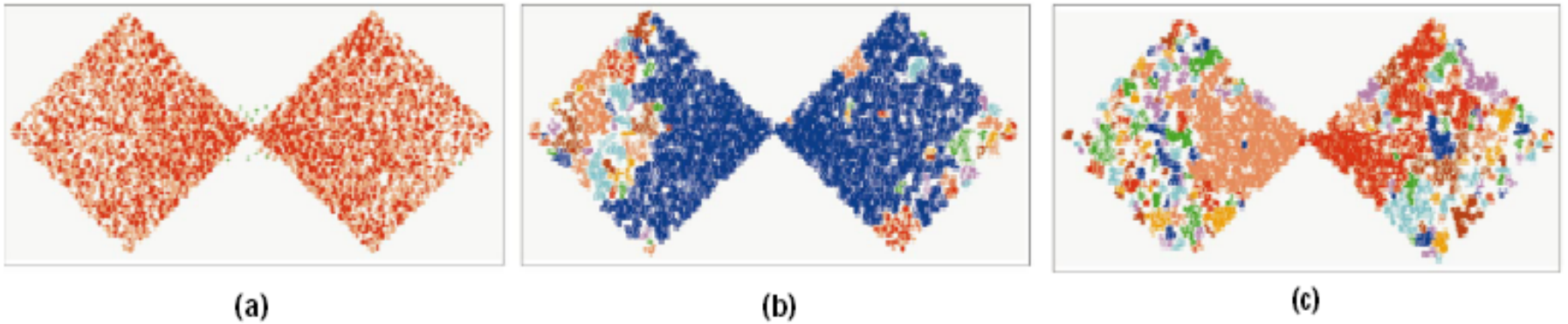
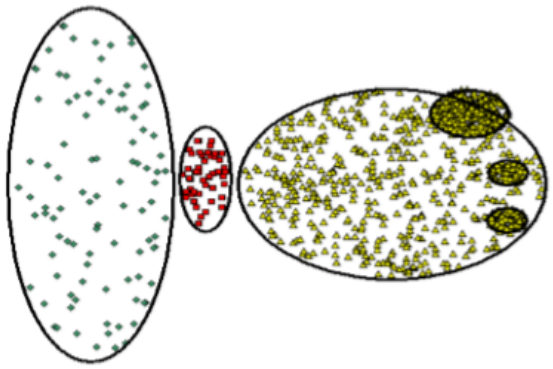


Figure 9. DBScan results for DS2 with MinPts at 4 and Eps at (a) 5.0, (b) 3.5, and (c) 3.0.

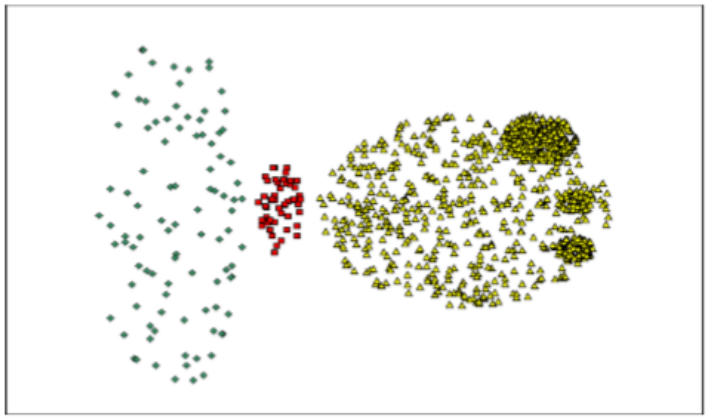


When DBSCAN Does Not Work Well

- Cannot handle varying densities



Original Points



(ε =9.92, MinPts=4)

DBSCAN : Advantages

- Does not require one to specify the number of clusters in the data
- Can find arbitrarily shaped clusters. even find a cluster completely surrounded by a different cluster.
- Has a notion of noise, and is robust to outliers.
- Requires just two parameters and is mostly insensitive to the ordering of the points in the database.
- Designed for accelerate region queries.
- minPts and ε can be set by a domain expert

DBSCAN : Disadvantages

- DBSCAN is not entirely deterministic: Border points that are reachable from more than one cluster can be part of either cluster, depending on the order the data is processed.

- The quality of DBSCAN depends on the distance measure used in the function regionQuery. (such as Euclidean distance)

- If the data and scale are not well understood, choosing a meaningful distance threshold ε can be difficult.

DBSCAN : Complexity

- **Time Complexity:** $O(n^2)$
  - for each point it has to be determined if it is a core point.
  - can be reduced to O(n*log(n)) in lower dimensional spaces by using efficient data structures (n is the number of objects to be clustered);
- **Space Complexity:** O(n).

Chapter 10. Cluster Analysis: Basic Concepts and Methods

- Cluster Analysis: Basic Concepts
- Partitioning Methods
- Hierarchical Methods
- Density-Based Methods
- Grid-Based Methods
- Evaluation of Clustering
- Summary

Assessing Clustering Tendency

- Assess if non-random structure exists in the data by measuring the probability that the data is generated by a uniform data distribution
- Test spatial randomness by statistic test: Hopkins Static
  - Given a dataset D regarded as a sample of a random variable o, determine how far away o is from being uniformly distributed in the data space
  - 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
  - Sample n points, $q_1, \ldots, q_n$, uniformly from D. For each q_i, find its nearest neighbor in D – $\{q_i\}$: $y_i = min\{dist(q_i, v)\}$ where v in D and $v \neq q_i$
  - 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. If D is highly skewed, H is close to 0

Hopkins statistic interpretation

- Null Hypothesis : Points are uniformly distributed - No clusters
- Alternate hypothesis: Points are not homogeneous, there is clustering
- H value > 0.5 , unlikely that there are clusters

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
  - 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

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

Clustering Quality – Extrinsic Methods

- Ground truth C_g says that the objects in a data set, D, can belong to categories L1,...,Ln.
- Clustering Goodness properties
  - Homogenity
  - Completeness
  - Rag Bag
  - Preserving small clusters

- Homogenity
  - Consider Two clustering outputs C1 and C2.
    - C1 contains objects from both Li and Lj in same cluster,
    - but C2 contains one cluster with objects from Li and another cluster with objects from L2, then
      Q(C2, Cg) > Q(C1, Cg)

- Cluster Completeness {inverse of Homogenity}
  - Consider clustering *C*1, which contains clusters c1 and c2,of which the members belong to the same category according to ground truth.
  - Let clustering *C*2 be identical to *C*1 except that c1 and c2 are merged into one cluster in *c*2.
  - Then, a clustering quality measure, Q, respecting cluster completeness should give a higher score to C2, that is, Q(*C*2,*C*g) > Q(*C*1,*C*g).

- Rag Bag
  - Miscellaneous cluster
  - A clustering output C2 with miscellaneous clustering is better than a clustering output which accommodates dissimilar objects in a cluster
- Small cluster preservation.
  - Suppose clustering C1 has three clusters, c1 ={o1,..., on}, c2 ={on+1}, and c3 ={on+2}.
  - Let clustering C2 have three clusters, too, namely c1 ={o1,..., on−1}, c2 ={on}, and c3 ={on+1,on+2}.
  - In other words, C1 splits the small category o_{n+1}, o_{n+2} and C2 splits the big category o_1 thru o_n. A clustering quality measure Q preserving small clusters should give a higher score toC2, that is, Q(C2,Cg) > Q(C1,Cg).

Extrinsic Quality - Measures

$$\text{Correctness}(o_i, o_j) = \begin{cases} 1 & \text{if } L(o_i) = L(o_j) \Leftrightarrow C(o_i) = C(o_j) \\ 0 & \text{otherwise.} \end{cases} \tag{10.28}$$

BCubed precision is defined as

$$\text{Precision BCubed} = \frac{\sum_{i=1}^{n} \frac{\sum_{o_j: i \neq j, C(o_i)=C(o_j)} \text{Correctness}(o_i, o_j)}{\|\{o_j | i \neq j, C(o_i) = C(o_j)\}\|}}{n}. \tag{10.29}$$

BCubed recall is defined as

$$\text{Recall BCubed} = \frac{\sum_{i=1}^{n} \frac{\sum_{o_j: i \neq j, L(o_i)=L(o_j)} \text{Correctness}(o_i, o_j)}{\|\{o_j | i \neq j, L(o_i) = L(o_j)\}\|}}{n}. \tag{10.30}$$

Intrinsic Measures

- No ground truth cluster data available
- Measures
  - Cluster homogenity/Compactness
  - Inter cluster distance

data set.

The **silhouette coefficient** is such a measure. For a data set, D, of n objects, suppose D is partitioned into k clusters, $C_1, \ldots, C_k$. For each object $o \in D$, we calculate $a(o)$ as the average distance between o and all other objects in the cluster to which o belongs. Similarly, $b(o)$ is the minimum average distance from o to all clusters to which o does not belong. Formally, suppose $o \in C_i$ $(1 \leq i \leq k)$; then

$$a(o) = \frac{\sum_{o' \in C_i, o \neq o'} dist(o, o')}{|C_i| - 1} \tag{10.31}$$

Cluster Compactness measure

a should be low for good clustering tech

and

$$b(o) = \min_{C_j : 1 \le j \le k, j \ne i} \left\{ \frac{\sum_{o' \in C_j} dist(o, o')}{|C_j|} \right\}.$$

The **silhouette coefficient** of o is then defined as

Inter Cluster Distance

b should be high for good clustering tech

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

Chapter 10. Cluster Analysis: Basic Concepts and Methods

- Cluster Analysis: Basic Concepts
- Partitioning Methods
- Hierarchical Methods
- Density-Based Methods
- Grid-Based Methods
- Evaluation of Clustering
- Summary

Summary

- Cluster analysis groups objects based on their similarity and has wide applications
- Measure of similarity can be computed for various types of data
- Clustering algorithms can be categorized into partitioning methods, hierarchical methods, density-based methods, grid-based methods, and model-based methods
- K-means and K-medoids algorithms are popular partitioning-based clustering algorithms
- Birch and Chameleon are interesting hierarchical clustering algorithms, and there are also probabilistic hierarchical clustering algorithms
- DBSCAN, OPTICS, and DENCLU are interesting density-based algorithms
- STING and CLIQUE are grid-based methods, where CLIQUE is also a subspace clustering algorithm
- Quality of clustering results can be evaluated in various ways