Chapter VII: Cluster analysis

Knowledge Discovery in Databases

Luciano Melodia M.A.
Evolutionary Data Management, Friedrich-Alexander University Erlangen-Nürnberg
Summer semester 2021

Chapter VII: Cluster analysis

What is cluster analysis?

Cluster: A collection of data objects within a larger set that are.

Similar (or related) to one another within the same group and,
dissimilar (or unrelated) to the objects outside the group.

Cluster analysis (or clustering, data segmentation, . . .).

Define similarities among data based on the characteristics found in the data (input from user!).
Group similar data objects into clusters.

Unsupervised learning:

No predefined classes.
I.e., learning by observation (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.

Earthquake studies:

Observed earthquake epicenters should be clustered along continent faults.

Climate:

Understanding earth climate, find patterns of atmosphere and ocean.

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 a metric: $d(x, y)$.
The definitions of distance functions are usually rather different for interval-scaled, Boolean, categorical, ordinal, ratio, and vector variables (see chapter 2).
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).

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.

Major clustering approaches

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, CLARA, CLARANS.

Hierarchical approach:

Create a hierarchical decomposition of the set of data (or objects) using some criterion.
Typical methods: AGNES, DIANA, BIRCH, CHAMELEON.

Density-based approach:

Based on connectivity and density functions.
Typical methods: DBSCAN, OPTICS, DENCLUE.

Grid-based approach:

Based on a multiple-level granularity structure.
Typical methods: STING, WaveCluster, CLIQUE.

Major clustering approaches (II)

Model-based approach:

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 approach:

Based on the analysis of frequent patterns.
Typical methods: p-Cluster.

User-guided or constraint-based approach:

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 VII: Cluster analysis

Partitioning algorithms: basic concept

Partitioning method:

Partition a database D of n objects $o_j, j \in \{1, \ldots, n\}$ into a set of k-clusters C_i, $1 \leq i \leq k$ such that the sum of squared distances to c_i is minimized
(where c_i is the **centroid** or **medoid** of cluster C_i):

$$\min \sum_{i=1}^{k} \sum_{o \in C_i} d(o, c_i)^2. \tag{1}$$

Given k, find a partition of k clusters that optimizes the chosen partitioning criterion.

Globally 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:

1. Partition the database into k non-empty subsets.

   E.g. the first $\frac{n}{k}$ objects, then the next $\frac{n}{k}$ objects, ...

2. Compute the centroids of the **clusters** of the current partitioning.

   The centroid is the center, i.e. mean point, of the cluster.
   For each attribute (or dimension), calculate the average value.

3. Assign each object to the cluster with the nearest centroid.

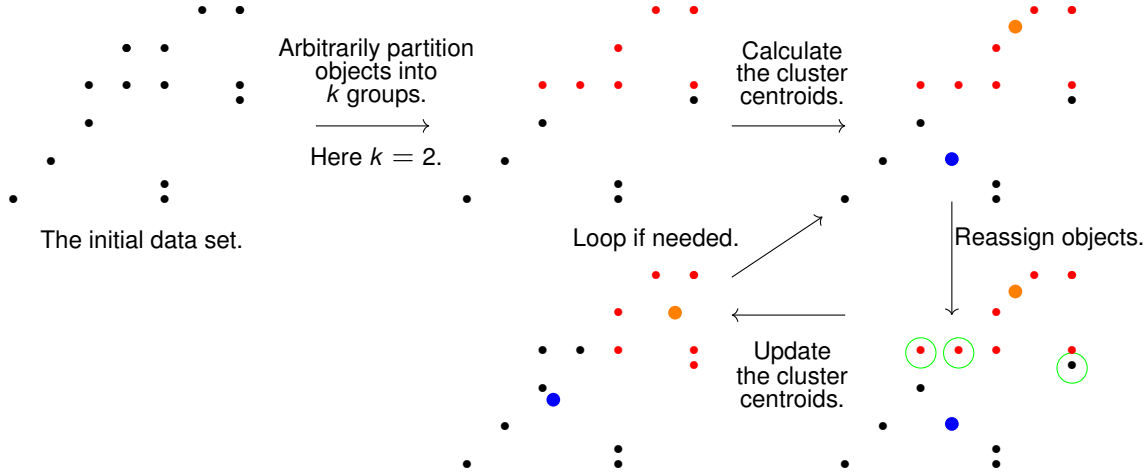
   That is, for each object calculate distance to each of the k centroids and pick the one with the smallest distance.

4. If any object has changed its cluster, go back to step 2. Otherwise stop.

Variant:

Start with arbitrarily chosen k objects as initial centroids in step 1.
Continue with step 3.

An example of k-means clustering



Comments on the k-means method

Strength:

Efficient: $\mathcal{O}(tkn)$, where n is # objects, k is # of clusters, and t is the # of iterations.
Normally, $k, t \ll n$.
Comparing: PAM: $\mathcal{O}(k(n-k)^2)$, CLARA: $\mathcal{O}(ks^2 + k(n-k))$.
Comment: Often terminates at a local optimum.

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 differ in:

- Selection of the initial k subsets (or centroids).
- Dissimilarity calculations.
- Strategies to calculate cluster centroids.

Handling categorical data: k-modes:

- Replacing centroids with modes.
  - See Chapter 2: mode = value that occurs most frequently in the data.
- 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.

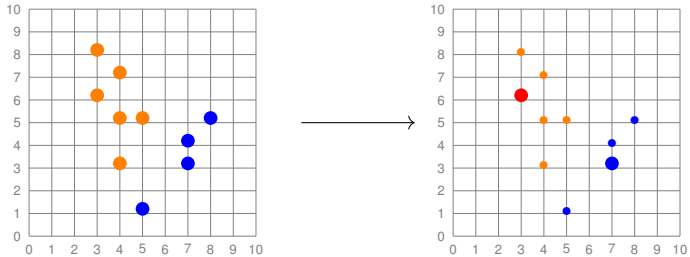
What is the problem of the *k*-means method?

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 objects in a cluster as a reference point, medoids can be used, which is the most centrally located object in a cluster.



The k-medoids clustering method

k-medoids clustering:

Find representative objects (medoids) in clusters.

PAM (Partitioning Around Medoids, Kaufmann & Rousseeuw, 1987):

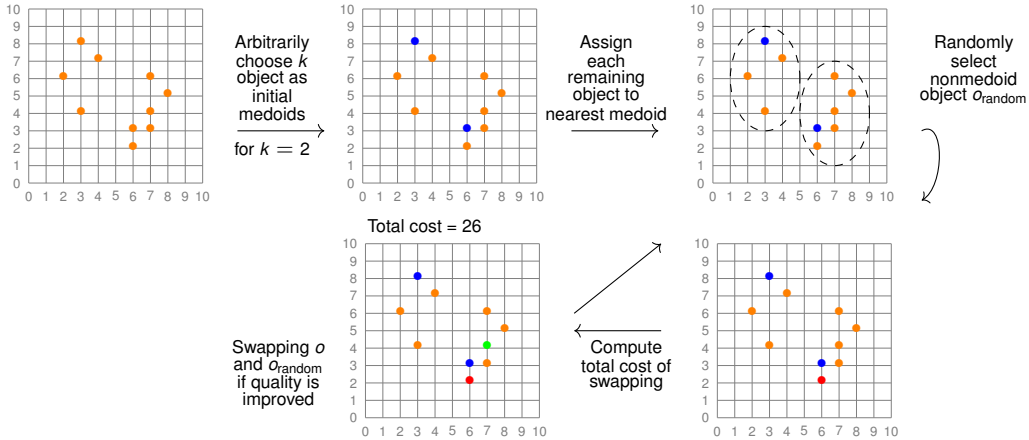
Starts from an initial set of k medoids and iteratively replaces one of the medoids by one of the non-medoids, if it improves the total distance of the resulting clustering. PAM works effectively for small data sets, but does not scale well for large data sets (due to the computational complexity).

Efficiency improvement on PAM:

CLARA (Kaufmann & Rousseeuw, 1990): PAM on samples.

CLARANS (Ng & Han, 1994): Randomized re-sampling.

PAM: A typical k-medoids algorithm



PAM (Partitioning Around Medoids)

Use real objects to represent the clusters.

Algorithm:

1. Arbitrarily choose k objects as the initial mediods.
2. Repeat.
3. Assign each remaining object to the cluster with the nearest mediod o_i.
4. Randomly select a non-medoid object o_h.
5. Compute the total cost TC_{ih} of swapping o_i with o_h.
6. If $\mathrm{TC}_{ih} < 0$ then swap o_i with o_h to form the new set of k medoids.
7. Until no change.

$$\mathrm{TC}_{ih} = \sum_j C_{jih}$$

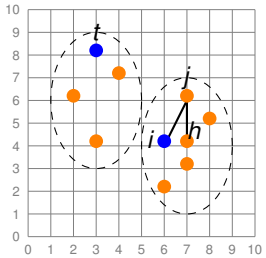
with C_{jih} as the cost for object o_j if o_i is swapped with o_h.
That is, distance to new medoid minus distance to old medoid.

PAM Clustering (II)

Case 1:

o_j currently belongs to medoid o_i. If o_i is replaced with o_h as a medoid, and o_j is closest to o_h, then o_j is reassigned to o_h (same cluster, different distance).

$$C_{jih} = d(o_j, o_h) - d(o_j, o_i). \tag{2}$$

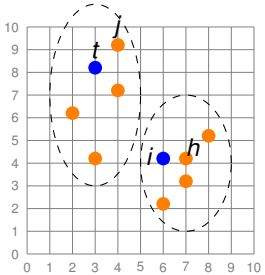


PAM Clustering (III)

Case 2:

o_j currently belongs to medoid o_t, $t \neq j$. If o_i is replaced with o_h as a medoid, and o_j is still closest to o_t, then the assignment does not change.

$$C_{jih} = d(o_j, o_h) - d(o_j, o_i) = 0. \tag{3}$$

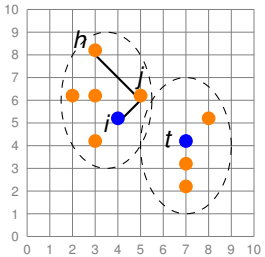


PAM Clustering (IV)

Case 3:

o_j currently belongs to medoid o_i. If o_i is replaced with o_h as a medoid, and o_j is closest to medoid o_t of one of the other clusters, then o_j is reassigned to o_t (new cluster, different distance).

$$C_{jih} = d(o_j, o_t) - d(o_j, o_i). \tag{4}$$

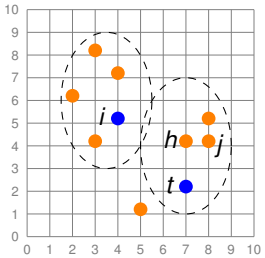


PAM Clustering (V)

Case 4:

o_j currently belongs to medoid o_t, $t \neq j$. If o_i is replaced with o_h as a medoid (from a different cluster!), and o_j is closest to o_h, then o_j is reassigned to o_h (new cluster, different distance).

$$C_{jih} = d(o_j, o_t) - d(o_j, o_i). \tag{5}$$



CLARA (Clustering Large Applications)

(Kaufmann and Rousseeuw, 1990)

Built in statistical-analysis packages, such as S+.

Draws multiple samples of the data set, applies PAM on each sample, and gives the best clustering as the output.

Strength:

Deals with larger data sets than PAM.

Weakness:

Efficiency depends on the sample size.
A good clustering based on samples will not necessarily represent a good clustering of the whole data set if the sample is biased.

CLARANS ("Randomized" CLARA)

A Clustering Algorithm based on Randomized Search (Ng & Han, 1994)

Samples:

Drawn dynamically with some randomness in each step of the search.

Clustering process:

Can be presented as searching a graph where each node is a potential solution, that is, a set of k medoids.

If local optimum found,

start with new randomly selected node in search for a new local optimum.

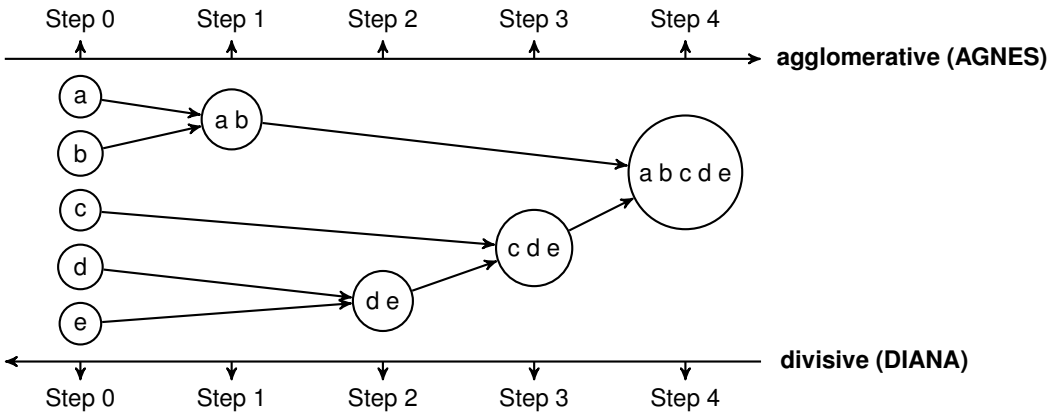
More efficient and scalable than both PAM and CLARA.

Focusing techniques and spatial access structures may further improve its performance. (Ester et al., 1995)

Chapter VII: Cluster analysis

Hierarchical clustering

Does not require the number of clusters k as an input, but needs a termination condition.



AGNES (Agglomerative Nesting)

Introduced in (Kaufmann & Rousseeuw, 1990)

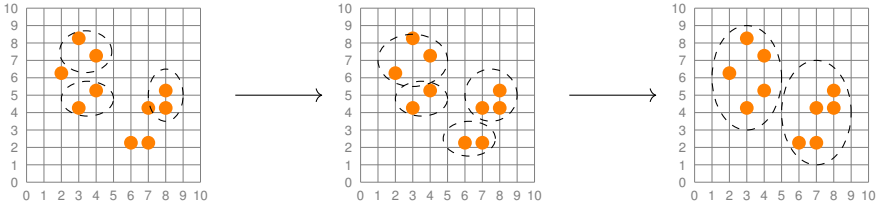
Implemented in statistical packages, e.g., S+.

Use the single-link method. (see below)

Merge nodes that have the least dissimilarity.

Go on in a non-descending fashion.

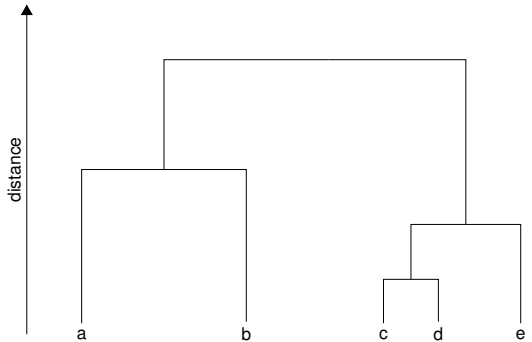
Eventually all nodes belong to the same cluster.



Dendrogram: shows how clusters are merged

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.



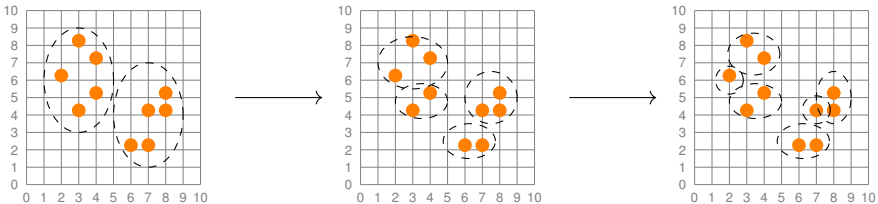
DIANA (Divisive Analysis)

Introduced in (Kaufmann & Rousseeuw, 1990)

Implemented in statistical analysis packages, e.g., S+.

Inverse order of AGNES.

Eventually each node forms a cluster of its own.



Distance between clusters

Minimum distance:

Smallest distance between an object in one cluster and an object in the other, i.e., $\text{dist}_{\text{min}}(C_i, C_j) = \min_{o_{ip} \in C_i, o_{jq} \in C_j} d(o_{ip}, o_{jq})$.

Maximum distance:

Largest distance between an object in one cluster and an object in the other, i.e., $\text{dist}_{\text{max}}(C_i, C_j) = \max_{o_{ip} \in C_i, o_{jq} \in C_j} d(o_{ip}, o_{jq})$.

Average distance:

Average distance between an object in one cluster and an object in the other, i.e., $\text{dist}_{\text{avg}}(C_i, C_j) = \frac{1}{n_i \cdot n_j} \sum_{o_{ip} \in C_i, o_{jq} \in C_j} d(o_{ip}, o_{jq})$.

Mean distance:

Distance between the centroids of two clusters, i.e., $\text{dist}_{\text{mean}}(C_i, C_j) = d(c_i, c_j)$.

Distance between clusters (II)

Nearest-neighbor clustering algorithm:

Uses **minimum distance** to measure distance between clusters.

Single-linkage algorithm:

Terminates if distance between nearest clusters exceeds user-defined threshold.

Minimal spanning-tree algorithm:

View objects (data points) as nodes of a graph.
Edges form a path between nodes in a cluster.
Merging of two clusters corresponds to adding an edge between the nearest pair of nodes.
Because edges linking clusters always go between distinct clusters,
resulting graph will be a tree.
Thus, agglomerative hierarchical clustering that uses minimum distance produces minimal spanning tree.

Distance between clusters (III)

Farthest-neighbor clustering algorithm:

Uses **maximum distance** to measure distance between clusters.

Complete-linkage algorithm:

Terminates if maximum distance between nearest clusters exceeds user-defined threshold.
Good if true clusters are rather compact and approx. equal in size.

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 $\mathcal{O}(n^2)$, where n is the number of objects.

Integration of hierarchical and 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 numerical data, and sensitive to the order of the data records.

Clustering feature in BIRCH

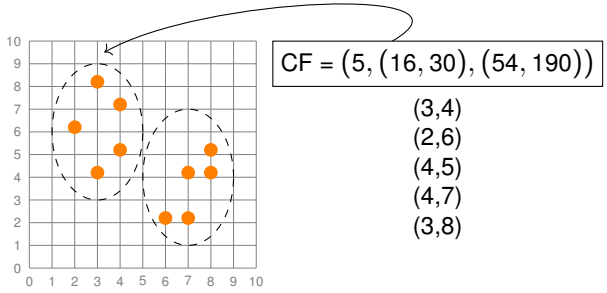
Clustering Feature CF = (n, LS, SS):

3D vector summarizing statistics about clusters.

n: number of data points.

LS: linear sum of N points $\sum_{i=1}^{n} x_i$.

SS: square sum of N points $\sum_{i=1}^{n} x_i^2$.



Clustering feature in BIRCH (II)

Allows to derive many useful statistics of a cluster:

E.g. centroid: $c_i = \frac{\sum_{i=1}^{n} o_i}{n} = \frac{LS}{n}$.

Additive:

For two disjoint clusters C_1 and C_2 with clustering features $CF_1 = (n_1, LS_1, SS_1)$ and $CF_2 = (n_2, LS_2, SS_2)$, the clustering feature of the cluster that is formed by merging C_1 and C_2 is simply: $CF_1 + CF_2 = (n_1 + n_2, LS_1 + LS_2, SS_1 + SS_2)$.

CF-Tree in BIRCH

Height-balanced tree.

Stores the clustering features for a hierarchical clustering.

Non-leaf nodes store sums of the CFs of their children.

Two parameters:

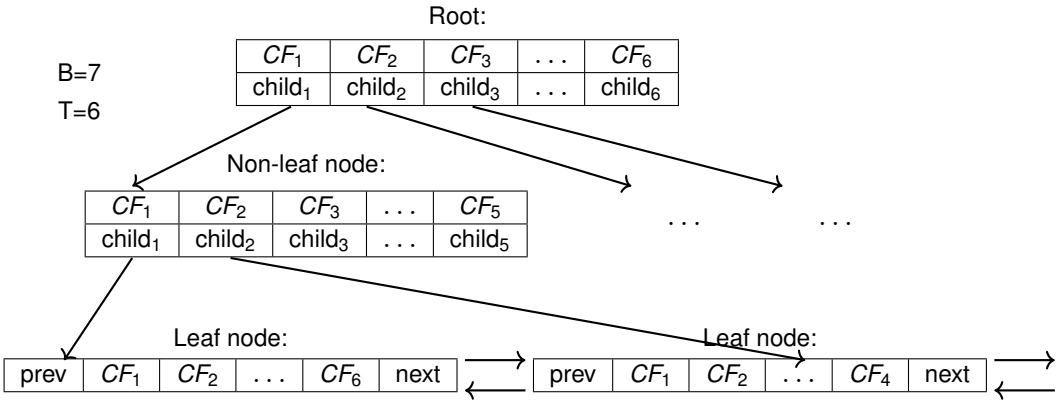
Branching factor B: max # of children.

Threshold T: maximum diameter of sub-clusters stored at leaf nodes.

Diameter D: average pairwise distance within a cluster,
reflects the tightness of the cluster around the centroid:

$$D = \sqrt{\frac{\sum_{i=1}^{n}\sum_{j=1}^{n}(o_i - o_j)^2}{n(n-1)}} = \sqrt{\frac{2nSS - 2LS^2}{n(n-1)}}. \tag{6}$$

CF-Tree structure



The BIRCH algorithm

Phase 1:

For each point in the input:

Find closest leaf-node entry.
Add point to leaf-node entry and update CF.
If *entry_diameter* $>$ *max_diameter*, then split leaf node, and possibly parents.
Information about new point is passed toward the root of the tree.

Algorithm is $\mathcal{O}(n)$ and incremental.

Concerns:

Sensitive to insertion order of data points.
Since we fix the size of leaf nodes, clusters may not be so natural.
Clusters tend to be spherical given the radius and diameter measures.

CHAMELEON: hierarchical clustering using dynamic modeling

(Karypis, Han & Kumar, 1999)

Measures the similarity based on a dynamic model:

Two clusters are merged only if the **interconnectivity** and **closeness (proximity)** between two clusters are high relative to the internal interconnectivity of the clusters and the closeness of items within the clusters.

Graph-based, and a two-phase algorithm.

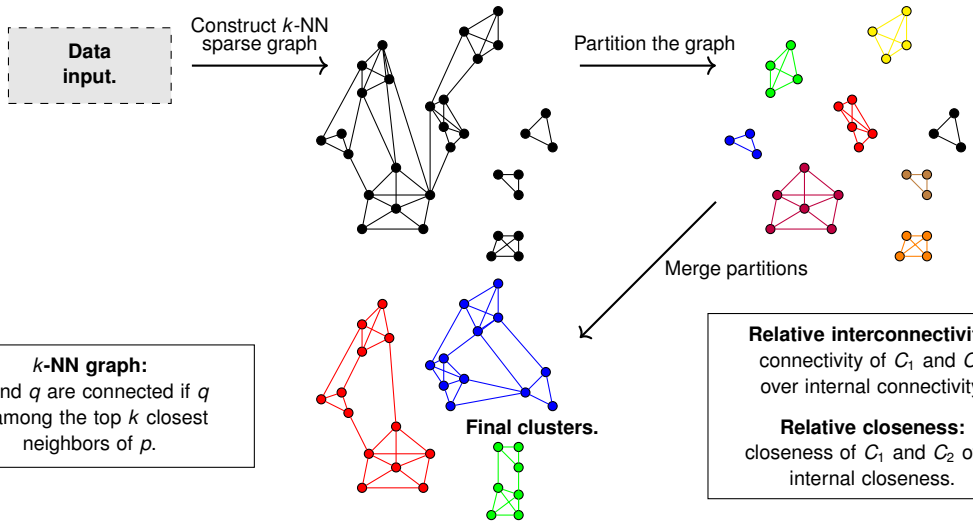
Use a graph-partitioning algorithm:

Cluster objects into a large number of relatively small sub-clusters.

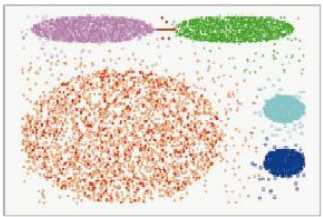
Use an agglomerative hierarchical clustering algorithm:

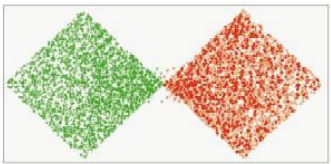
Find the genuine clusters by repeatedly combining these sub-clusters.

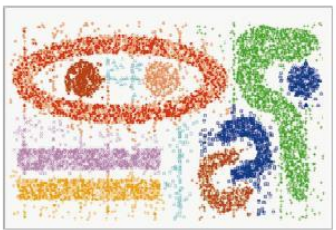
Overall framework of CHAMELEON

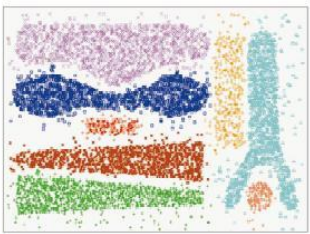


CHAMELEON (Clustering Complex Objects)









Probabilistic hierarchical clustering

Algorithmic hierarchical clustering.

Nontrivial to choose a good distance measure.
Hard to handle missing attribute values.
Optimization goal not clear: heuristic, local search.

Probabilistic hierarchical clustering.

Use probabilistic models to measure distances between clusters.

Generative model:

Regard the set of data objects to be clustered as a sample of the underlying data-generation mechanism to be analyzed.

Easy to understand, same efficiency as algorithmic agglomerative clustering method, can handle partially observed data.
In practice, assume the generative models adopt **common distribution functions**, e.g., Gaussian distribution or Bernoulli distribution, governed by parameters.

Generative model

Given a set of 1-D points $X = \{x_1, \ldots, x_n\}$ for clustering analysis and assuming they are generated by a Gaussian distribution:

$$N(\mu, \sigma) = \frac{1}{\sqrt{2\pi\sigma^2}} \exp\left(\frac{(x-\mu)^2}{2\sigma^2}\right). \tag{7}$$

The probability that a point $x_i \in X$ is generated by the model:

$$P(x_i|\mu, \sigma) = \frac{1}{\sqrt{2\pi\sigma^2}} \exp\left(\frac{(x_i-\mu)^2}{2\sigma^2}\right). \tag{8}$$

Generative model (II)

The likelihood that X is generated by the model:

$$L(N(\mu,\sigma)|X) := P(X|\mu,\sigma) = \prod_{i=1}^{n} \frac{1}{\sqrt{2\pi\sigma^2}} \exp\left(\frac{(x_i-\mu)^2}{2\sigma^2}\right). \tag{9}$$

The task of learning the generative model: find the parameters μ and σ, such that

$$N(\mu_0,\sigma_0) = \arg\max\left(L(N(\mu,\sigma)|X)\right). \tag{10}$$

A probabilistic hierarchical clustering algorithm

For a set of objects partitioned into m clusters $C_1, \ldots, C_m$, the quality can be measured by:

$$Q(\{C_1, \ldots, C_m\}) = \prod_{i=1}^{m} P(C_i), \tag{11}$$

where $P(C_i)$ is the maximum likelihood of C_i.

Distance between clusters C_i and C_j can be computed as a proximity measure:

$$d(C_i, C_j) = -\log \frac{P(C_i \cup C_j)}{P(C_i)P(C_j)}. \tag{12}$$

A probabilistic hierarchical clustering algorithm (II)

Algorithm: Progressively merge points and clusters

Input: $D = \{o_1, \ldots, o_n\}$: a dataset containing n objects.
Output: A hierarchy of clusters.
Method:

`Create cluster for each object` $C_i = \{o_i\}$, `for` $1 \leq i \leq n$.
For $i = 1$ **to** n**:**
`Find a pair of clusters` C_i `and` C_j, `such that`
$C_i, C_j = \arg\max_{i \neq j} \left(\log \left(\frac{P(C_i \cup C_j)}{P(C_i)P(C_j)} \right) \right)$.
If $\log \left(\frac{P(C_i \cup C_j)}{P(C_i)P(C_j)} \right) > 0$ **then** `merge` C_i `and` C_j,
Else stop.

Chapter VII: Cluster analysis

Chapter VII: Cluster analysis

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 & Keim, KDD'98).
- CLIQUE (Agrawal et al., SIGMOD'98) (more grid-based).

Density-dased clustering: basic concepts

Two parameters:

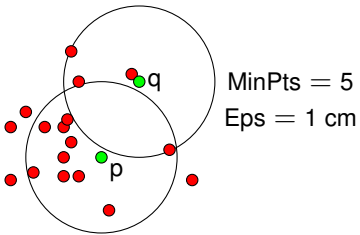
Eps: Maximum radius of a neighborhood.

Defines the neighborhood of a point p: $N_{\text{Eps}}(p) := \{q \in D \mid d(p,q) \leq \text{Eps}\}$.
Distance function is still needed.

MinPts: Minimum number of points in an Eps-neighborhood of a point p.

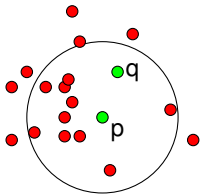
Core-point condition: $|N_{\textbf{Eps}}(p)| \geq$ **MinPts.**

Only these neighborhoods are considered.



Directly density-reachable

A point q is said to be directly density-reachable from a point p w.r.t. Eps, MinPts, if q belongs to $N_{\text{Eps}}(p)$.



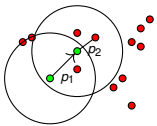
MinPts $= 5$

Eps $= 1$ cm

Density-reachable and density-connected

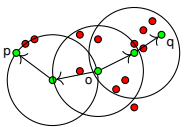
Density-reachable:

A point q is density-reachable from a point p w.r.t. Eps, MinPts, if there is a **chain of points** $p_1, \ldots, p_n$, $p_1 = p$, $p_n = q$ 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.

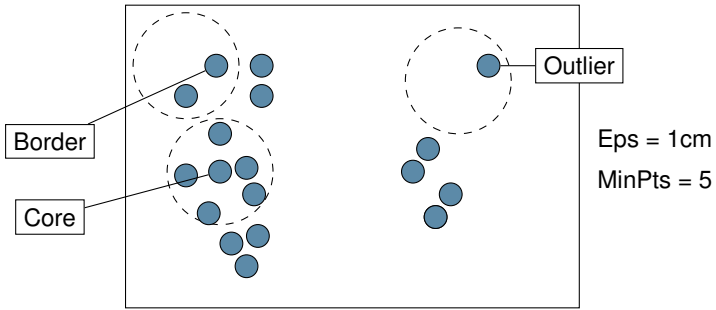


DBSCAN: density-based spatial clustering of applications with noise

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.



DBSCAN: the algorithm

All objects in database D are marked `unvisited`.

Randomly select unvisited object p and mark as `visited`.

If Eps-neighborhood of p contains less than MinPts objects:

- Mark p as noise.

Otherwise (that is, p is core point):

- Create new cluster C for point p.
- Add all objects in Eps-neighborhood of p to candidate set N.
- For each p' in N that does not yet belong to a cluster:
  - Add p' to C.
  - If p' is `unvisited`, mark as `visited`.
  - If p' is core point, add all objects in its Eps-neighborhood to N.
- Ends when N is empty, that is, C can no longer be expanded.

Continue the process (randomly select next unvisited object in D) until all points have been visited.

DBSCAN: sensitive to parameters

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

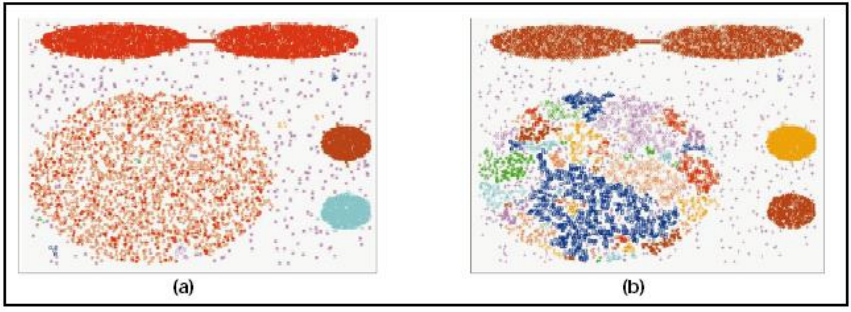
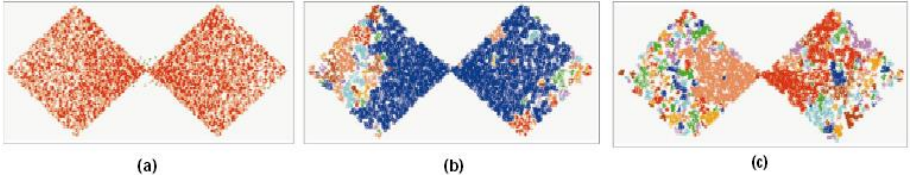


Figure 9. DBScan results for DS2 with MinPts at 4 and Eps at (a) 5.0, (b) 3.5, and (c) 3.0.



OPTICS: a cluster-ordering method

OPTICS: Ordering Points To Identify the Clustering Structure.

(Ankerst, Breunig, Kriegel & Sander, SIGMOD'99)

Avoid difficulty of finding the right parameters:

Only MinPts must be given, Eps remains open.

Produces a special **order of the database** w.r.t. its density-based clustering structure.

This cluster-ordering contains info equivalent 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.

Process point in order: Select a point that is density-reachable w.r.t. the lowest Eps value, so that clusters with higher density (lower Eps) will be finished first.

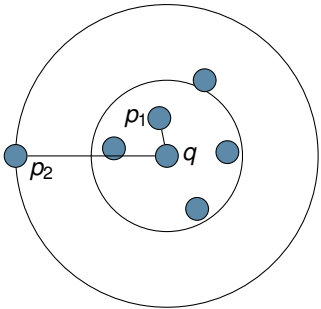
OPTICS: Some Extension of DBSCAN

Core distance:

of a point p.
Min Eps s.t. p is core point.

Reachability Distance r:

of a point p from q.
Minimum radius value that makes p
directly density-reachable from q.
That is, $r(q,p) := \max\{\text{core-distance}(q), d(q,p)\}$.



Example:

$\text{MinPts} = 5$, $\text{core\_distance}(q) = 3cm$.
$d(q,p_1) = 2.8cm \implies r(q,p_1) = 3cm$.
$d(q,p_2) = 4cm \implies r(q,p_2) = 4cm$.

OPTICS: algorithm

Maintains a list of OrderSeeds:

Points sorted by **reachability-distance** from their resp. closest core points.

Begin with arbitrary point from input DB.

For each point *p* under consideration:

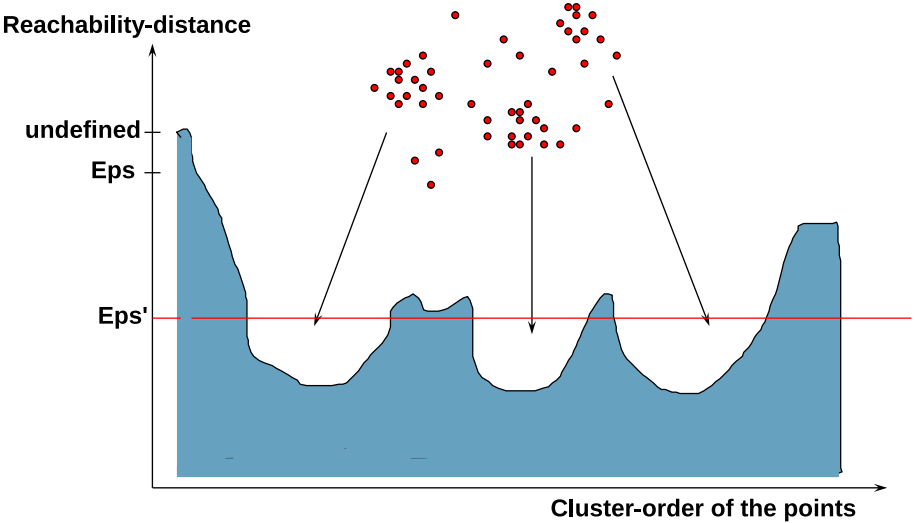
Retrieve the closest MinPts points of *p*, determine core-distance, set reachability-distance to undefined.
Write *p* to output.
If *p* is core point, for each point *q* in the Eps-neighborhood of *p*:

Update reachability-distance from *p*.
Insert *q* into OrderSeeds (if *q* has not yet been processed).

Move to next object in OrderSeeds list with smallest reachability-distance (or input DB if OrderSeeds is empty).

Continue until input DB fully consumed (and OrderSeeds empty).

OPTICS: visualization



DENCLUE: using statistical density functions

DENsity-based CLUstEring (Hinneburg & Keim, KDD'98)

Using statistical density functions:

$f_{\text{Gaussian}}(x, y) = \exp\left(\frac{d(x,y)^2}{2\sigma^2}\right)$, influence of y on x.

$f^D_{\text{Gaussian}}(x, x_i) = \sum_{i=1}^{N} \exp\left(\frac{d(x,y)^2}{2\sigma^2}\right)$, total influence on x.

$\nabla_{x_i} f^D_{\text{Gaussian}}(x, x_i) = \sum_{i=1}^{N} (x_i - x) \cdot \exp\left(\frac{d(x,y)^2}{2\sigma^2}\right)$, gradient of x in direction x_i.

Major features:

Solid mathematical foundation.
Good for data sets with large amounts of noise.
Allows a compact mathematical description of arbitrarily shaped clusters in high-dimensional data sets.
Significantly faster than existing algorithms (e.g., DBSCAN).
But needs a large number of parameters.

DENCLUE: technical essence

Density estimation:

Estimation of an unobservable underlying probability density function based on a set of observed data.
Regarded as a sample.

Kernel density estimation:

Treat an observed object as an indicator of high-probability density in the surrounding region.
Probability density at a point depends on the distances from this point to the observed objects.

Density attractor:

Local maximum of the estimated density function.
Must be above threshold ξ.

DENCLUE: technical essence (II)

Clusters:

Can be determined mathematically by identifying density attractors.
That is, local maxima of the overall density function.

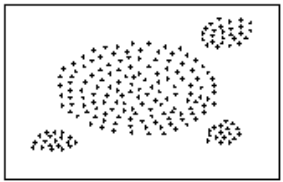
Center-defined clusters:

Assign to each density attractor the points density-attracted to it.

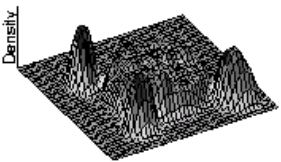
Arbitrary shaped cluster:

Merge density attractors that are connected through paths of high density ($>$ threshold).

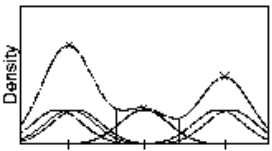
Density attractor



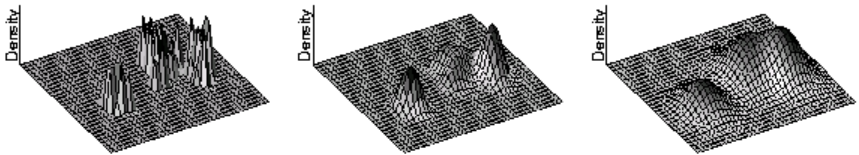
(a) Data Set



(c) Gaussian

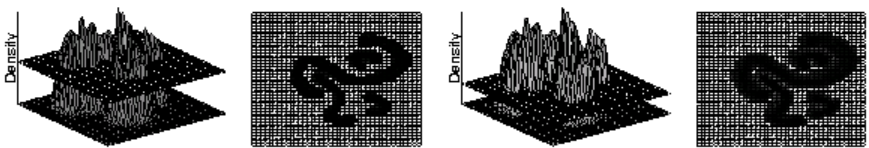


Density attractor



(a) $\sigma = 0.2$ (b) $\sigma = 0.6$ (d) $\sigma = 1.5$

Figure 3: Example of Center-Defined Clusters for different σ



(a) $\xi = 2$ (b) $\xi = 2$ (c) $\xi = 1$ (d) $\xi = 1$

Figure 4: Example of Arbitray-Shape Clusters for different ξ

Chapter VII: Cluster analysis

Grid-based clustering method

Using multi-resolution grid data structure.

Several interesting methods.

STING (a STatistical INformation Grid approach) (Wang, Yang & Muntz, VLDB'97).

WaveCluster (Sheikholeslami, Chatterjee & Zhang, VLDB'98):

A multi-resolution clustering approach using wavelet method.

Not shown here.

CLIQUE (Agrawal et al., SIGMOD'98):

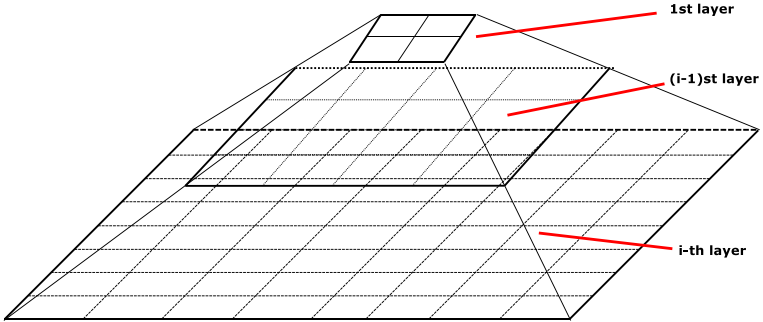
Both grid-based and subspace clustering.

STING: a statistical information grid approach

The spatial area is divided into rectangular cells.

There are several levels of cells corresponding to different levels of resolution.

(Wang, Yang & Muntz, VLDB'97)



The STING clustering method

Each cell at a higher level:

Partitioned into a number of smaller cells at next lower level.

Statistical info of each cell:

Calculated and stored beforehand and used to answer queries.
Count, plus for each attribute: mean, standard dev, min, max and type of distribution: normal, uniform, etc.

Parameters of higher-level cells:

Can easily be calculated from parameters of lower-level cells.

Use a top-down approach:

To answer spatial data queries (or: cluster definitions).

Start from a pre-selected layer:

Typically with a small number of cells.

For each cell at the current level:

Compute the confidence interval.

The STING clustering method (II)

Cells labeled relevant or not relevant.

At the specified confidence level.
Irrelevant cells removed from further consideration.

When finished examining the current layer, proceed to the next lower level.

Only look at cells that are children of relevant cells.

Repeat this process until bottom layer is reached.

Find regions (clusters) that satisfy the density specified.

Breadth-first search, at bottom layer.
Examine cells within a certain distance from center of current cell (often just the neighbors).
If average density within this small area is greater than density specified, mark area and put relevant cells just examined into queue.
Examine next cell from queue and repeat procedure, until end of queue.
Then one region has been identified.

STING algorithm: analysis

Advantages:

Query-independent, easy to parallelize, incremental update.
$\mathcal{O}(K)$, 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)

Using subspaces (lower-dimensional) of a high-dimensional data space that allow better clustering than original space:

(Agrawal, Gehrke, Gunopulos & Raghavan, SIGMOD'98).

CLIQUE can be considered as both density-based and grid-based.

Partitions each dimension into non-overlapping intervals,
thereby partitioning the entire data space into **cells**.
Uses density threshold to identify **dense** cells.

A cell is dense, if the number of data points mapped to it exceeds the density threshold.

CLIQUE: the major steps

Monotonicity of dense cells w.r.t. dimensionality:

Based on the a priori property used in frequent-pattern and association-rule mining (see Chapter 5).

k-dimensional cell c can have at least l points only, if every $(k-1)$-dimensional projection of c (which is a cell in a $(k-1)$-dimensional subspace) has at least l points, too.

Clustering step 1:

Partition each dimension into intervals, identify intervals containing at least l points.

Iteratively join k-dimensional dense cells c_1 and c_2 in subspaces $(D_{i1}, \ldots, D_{ik})$ and $(D_{j1}, \ldots, D_{jk})$ with $D_{i1} = D_{j1}$ and $D_{i2} = D_{j2}$ and $\ldots D_{i(k-1)} = D_{j(k-1)}$ and c_1 and c_2 share the same intervals to those dimensions.

New $(k+1)$-dimensional candidate cell c in space $(D_{i1}, \ldots, D_{ik}, D_{jk})$ tested for density.

CLIQUE: the major steps (II)

Clustering step 1 (cont.):

Iteration terminates when no more candidate cells can be generated or no candidate cells are dense.

Clustering step 2:

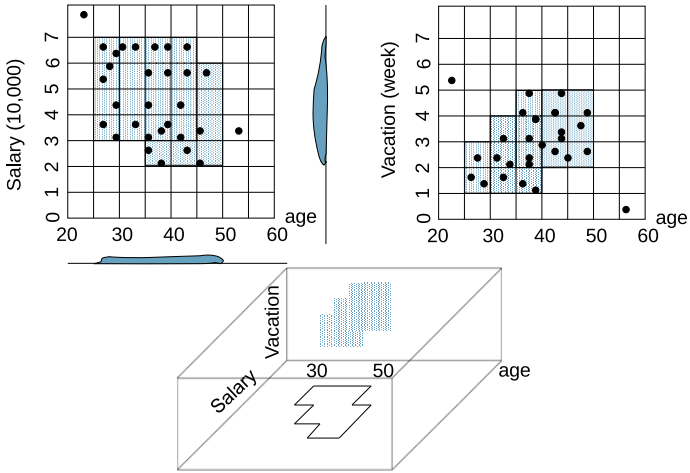
Use dense cells in each subspace to assemble clusters.
Apply Minimum Description Length (MDL) principle to use the maximal regions to cover connected dense cells.

Maximal region: hyper rectangle where every cell falling into the regions is dense, and region cannot be extended further in any dimension.

Simple greedy approach:

Start with arbitrary dense cell.
Find maximum region covering that cell.
Work on remaining dense cells that have not yet been covered.

CLIQUE: example



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 any canonical data distribution.
Scales linearly with the size of input and has good scalability as the number of dimensions in the data is increased.

Weaknesses:

Dependent on proper grid size and density threshold.
Accuracy of clustering result may be degraded at the expense of simplicity of the method.

Chapter VII: Cluster analysis

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.

Data with random structure:

Points uniformly distributed in data space.

Clustering may return clusters, but:

Artificial partitioning.
Meaningless.

Hopkins statistic

Data set:

Let $X = \{x_i \,|i = 1, \ldots, n\}$ be a collection of n patterns in a d-dimensional space such that $x_i = (x_{i1}, x_{i2}, \ldots, x_{id})$.

Random sample of data space:

Let $Y = \{y_j \mid j = 1, \ldots, m\}$ be m sampling points placed at random in the d-dimensional space, with $m \ll n$.

Two types of distances defined:

u_j as minimum distance from y_j to its nearest pattern in X and
w_j as minimum distance from a randomly selected pattern in X to its nearest neighbor in X.

The Hopkins statistic in d dimensions is defined as:

$$H = \frac{\sum_{j=1}^{m} u_j}{\sum_{j=1}^{m} u_j + \sum_{j=1}^{m} w_j}. \tag{13}$$

Hopkins statistic (II)

Compares nearest-neighbor distribution of randomly selected locations (points) to that for randomly selected patterns.

Under the null hypothesis, H_0, of uniform distribution:

Distances from sampling points to nearest patterns should, on the average, **be the same** as the interpattern nearest-neighbor distances, implying randomness.
H should be about 0.5.

When patterns are aggregated or clustered:

Distances from sampling points to nearest patterns should, on the average, **larger** as the interpattern nearest-neighbor distances.
H should be larger than 0.5.
Almost equal to 1.0 for very well clustered data.

Determine the number of clusters

Empirical method:

$\#$ of clusters $\approx \frac{\sqrt{n}}{2}$ 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 distances 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: 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: $Q(C, C_*)$**.**

For a clustering C given the ground truth C_*.

Q **is good, if it satisfies the following four essential criteria:**

Cluster homogeneity:

The purer, the better.

Cluster completeness:

Should assign objects that 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.

Chapter VII: Cluster analysis

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 (k-means and k-medoids).
Hierarchical methods (BIRCH and CHAMELEON; probabilistic hierarchical clustering).
Density-based methods (DBSCAN, OPTICS, and DENCLUE).
Grid-based methods (STING, CLIQUE).
Model-based methods.

Quality of clustering results.

References

R. Agrawal, J. Gehrke, D. Gunopulos, and P. Raghavan: Automatic subspace clustering of high dimensional data for data mining applications. SIGMOD'98.

M. R. Anderberg: Cluster Analysis for Applications. Academic Press, 1973.

M. Ankerst, M. Breunig, H.-P. Kriegel, and J. Sander. OPTICS: Ordering points to identify the clustering structure, SIGMOD'99.

A. Banerjee and R. N. Davé: Validating Clusters using the Hopkins Statistic. Fuzzy Systems 2004.

F. Beil, M. Ester, and X. Xu: Frequent Term-Based Text Clustering. KDD'02.

M. M. Breunig, H.-P. Kriegel, R. Ng, and J. Sander: LOF: Identifying Density-Based Local Outliers. SIGMOD 2000.

M. Ester, H.-P. Kriegel, J. Sander, and X. Xu: A density-based algorithm for discovering clusters in large spatial databases. KDD'96.

M. Ester, H.-P. Kriegel, and X. Xu: Knowledge discovery in large spatial databases: Focusing techniques for efficient class identification. SSD'95.

D. Fisher: Knowledge acquisition via incremental conceptual clustering. Machine Learning, 2:139-172, 1987.

References (II)

L. Fu and E. Medico: FLAME, a novel fuzzy clustering method for the analysis of DNA microarray data. BMC Bioinformatics 2007, 8:3.

D. Gibson, J. Kleinberg, and P. Raghavan: Clustering categorical data: An approach based on dynamic systems. VLDB'98.

V. Ganti, J. Gehrke, and R. Ramakrishan: CACTUS Clustering Categorical Data Using Summaries. KDD'99.

D. Gibson, J. Kleinberg, and P. Raghavan: Clustering categorical data: An approach based on dynamic systems. VLDB'98.

S. Guha, R. Rastogi, and K. Shim. Cure: An efficient clustering algorithm for large databases. SIGMOD'98.

S. Guha, R. Rastogi, and K. Shim. ROCK: A robust clustering algorithm for categorical attributes. ICDE'99.

A. Hinneburg and D. A. Keim: An Efficient Approach to Clustering in Large Multimedia Databases with Noise. KDD'98.

A. K. Jain and R. C. Dubes: Algorithms for Clustering Data. Prentice Hall, 1988.

References (III)

G. Karypis, E.-H. Han, and V. Kumar: CHAMELEON: A Hierarchical Clustering Algorithm Using Dynamic Modeling. IEEE Computer, 32(8): 68-75, 1999.

L. Kaufman and P. J. Rousseeuw: Finding Groups in Data: an Introduction to Cluster Analysis. John Wiley & Sons, 1990.

E. Knorr and R. Ng: Algorithms for mining distance-based outliers in large datasets. VLDB'98.

G. J. McLachlan and K.E. Bkasford: Mixture Models: Inference and Applications to Clustering. John Wiley and Sons, 1988.

R. Ng and J. Han: Efficient and effective clustering method for spatial data mining. VLDB'94.

L. Parsons, E. Haque, and H. Liu: Subspace Clustering for High Dimensional Data: A Review. SIGKDD Explorations, 6(1), June 2004.

E. Schikuta: Grid clustering: An efficient hierarchical clustering method for very large data sets. Pattern Recognition 1996.

G. Sheikholeslami, S. Chatterjee, and A. Zhang: WaveCluster: A multi-resolution clustering approach for very large spatial databases. VLDB'98.

References (IV)

A. K. H. Tung, J. Han, L. V. S. Lakshmanan, and R. T. Ng: Constraint-Based Clustering in Large Databases. ICDT'01.

A. K. H. Tung, J. Hou, and J. Han: Spatial Clustering in the Presence of Obstacles. ICDE'01.

H. Wang, W. Wang, J. Yang, and P.S. Yu: Clustering by pattern similarity in large data sets. SIGMOD'02.

W. Wang, Yang, and R. Muntz: STING: A Statistical Information Grid Approach to Spatial Data Mining. VLDB'97.

T. Zhang, R. Ramakrishnan, and M. Livny: BIRCH : An efficient data clustering method for very large databases. SIGMOD'96.

X. Yin, J. Han, and P. S. Yu: LinkClus: Efficient Clustering via Heterogeneous Semantic Links. VLDB'06.

Thank you for your attention.
Any questions about the seventh chapter?

Ask them now, or again, drop me a line:
luciano.melodia@fau.de.