Data Science

Clustering

Themis Palpanas
University of Paris

Thanks for slides to:

- Jiawei Han
- Eamonn Keogh
- Jeff Ullman
- Anand Rajaraman

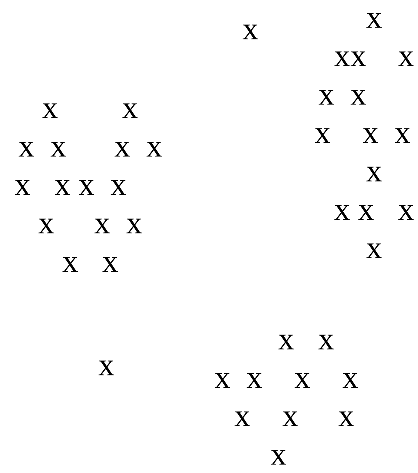
Roadmap

1. What is Cluster Analysis?
2. Types of Data in Cluster Analysis
3. A Categorization of Major Clustering Methods
4. Partitioning Methods
5. Hierarchical Methods
6. Density-Based Methods
7. Grid-Based Methods
8. Model-Based Methods
9. Clustering High-Dimensional Data
10. Constraint-Based Clustering
11. Summary

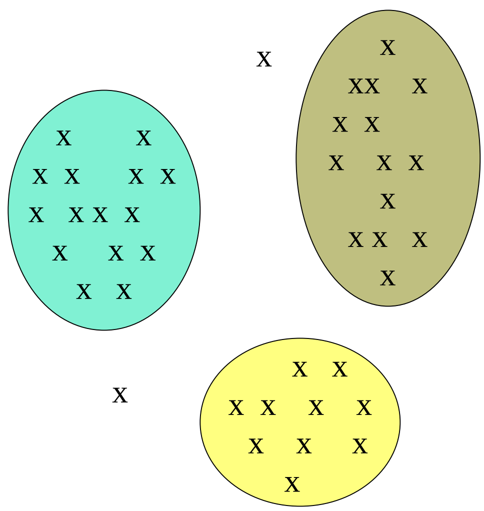
What is Cluster Analysis?

- Cluster: a collection of data objects
  - Similar to one another within the same cluster
  - Dissimilar to the objects in other clusters
- Cluster analysis
  - Finding similarities between data according to the characteristics found in the data and grouping similar data objects into clusters

Example: Clusters



Example: Clusters



What is Cluster Analysis?

- Cluster: a collection of data objects
  - Similar to one another within the same cluster
  - Dissimilar to the objects in other clusters
- Cluster analysis
  - Finding similarities between data according to the characteristics found in the data and grouping similar data objects into clusters
- Unsupervised learning: no predefined classes
- Typical applications
  - As a stand-alone tool to get insight into data distribution
  - As a preprocessing step for other algorithms

Clustering: Rich Applications and Multidisciplinary Efforts

- Pattern Recognition
- Spatial Data Analysis
  - Create thematic maps in GIS by clustering feature spaces
  - Detect spatial clusters or for other spatial mining tasks
- Image Processing
- Economic Science (especially market research)
- WWW
  - Document classification
  - Cluster Weblog data to discover groups of similar access patterns

Examples of Clustering Applications

- Marketing: Help marketers discover distinct groups in their customer bases, and then use this knowledge to develop targeted marketing programs
- Land use: Identification of areas of similar land use in an earth observation database
- Insurance: Identifying groups of motor insurance policy holders with a high average claim cost
- 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

Quality: What Is Good Clustering?

- A good clustering method will produce high quality clusters with
  - high intra-class similarity
  - low inter-class similarity
- The quality of a clustering result depends on both the similarity measure used by the method and its implementation
- The quality of a clustering method is also measured by its ability to discover some or all of the hidden patterns

Measure the Quality of Clustering

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

Problems With Clustering

- Clustering in two dimensions looks easy.
- Clustering small amounts of data looks easy.
- And in most cases, looks are *not* deceiving.

The Curse of Dimensionality

- Many applications involve not 2, but 10 or 10,000 dimensions.
- High-dimensional spaces look different: almost all pairs of points are at about the same distance.
  - Example: assume random points within a bounding box, e.g., values between 0 and 1 in each dimension.

Example: SkyCat

- A catalog of 2 billion "sky objects" represents objects by their radiation in 9 dimensions (frequency bands).
- Problem: cluster into similar objects, e.g., galaxies, nearby stars, quasars, etc.
- Sloan Sky Survey is a newer, better version.

Example: Clustering CD's (Collaborative Filtering)

- Intuitively: music divides into categories, and customers prefer a few categories.
  - But what are categories really?
- Represent a CD by the customers who bought it.
- Similar CD's have similar sets of customers, and vice-versa.

The Space of CD's

- Think of a space with one dimension for each customer.
  - Values in a dimension may be 0 or 1 only.
- A CD's point in this space is $(x_1, x_2,\dots, x_k)$, where x_i = 1 iff the i^{th} customer bought the CD.
  - Compare with the "shingle/signature" matrix: rows = customers; cols. = CD's.
- For Amazon, the dimension count is tens of millions.

Example: Clustering Documents

- Represent a document by a vector $(x_1, x_2,\ldots, x_k)$, where $x_i = 1$ iff the i^{th} word (in some order) appears in the document.
  - It actually doesn't matter if k is infinite; i.e., we don't limit the set of words.
- Documents with similar sets of words may be about the same topic.

Example: Gene Sequences

- Objects are sequences of {C,A,T,G}.
- Distance between sequences is *edit distance*, the minimum number of inserts and deletes needed to turn one into the other.
- Note there is a "distance," but no convenient space in which points "live."

Requirements of Clustering in Data Mining

- Scalability
- Ability to deal with different types of attributes
- Ability to handle dynamic data
- Discovery of clusters with arbitrary shape
- Minimal requirements for domain knowledge to determine input parameters
- Able to deal with noise and outliers
- Insensitive to order of input records
- High dimensionality
- Incorporation of user-specified constraints
- Interpretability and usability

Roadmap

1. What is Cluster Analysis?
2. Types of Data in Cluster Analysis
3. A Categorization of Major Clustering Methods
4. Partitioning Methods
5. Hierarchical Methods
6. Density-Based Methods
7. Grid-Based Methods
8. Model-Based Methods
9. Clustering High-Dimensional Data
10. Constraint-Based Clustering
11. Summary

Type of data in clustering analysis

- Interval-scaled variables
- Binary variables
- Categorical (or Nominal), ordinal, and ratio variables
- Variables of mixed types

Interval-valued variables

- Standardize data
  - Calculate the mean absolute deviation:

$$s_f = \frac{1}{n}(|x_{1f} - m_f| + |x_{2f} - m_f| + ... + |x_{nf} - m_f|)$$

where $m_f = \frac{1}{n}(x_{1f} + x_{2f} + ... + x_{nf})$.

  - Calculate the standardized measurement (*z-score*)

$$z_{if} = \frac{x_{if} - m_f}{s_f}$$

- Using mean absolute deviation is more robust than using standard deviation

Similarity and Dissimilarity Between Objects

- Distances are normally used to measure the similarity or dissimilarity between two data objects
- Some popular ones include: *Minkowski distance*:

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

where $i = (x_{i1}, x_{i2}, \ldots, x_{ip})$ and $j = (x_{j1}, x_{j2}, \ldots, x_{jp})$ are two p-dimensional data objects, and q is a positive integer

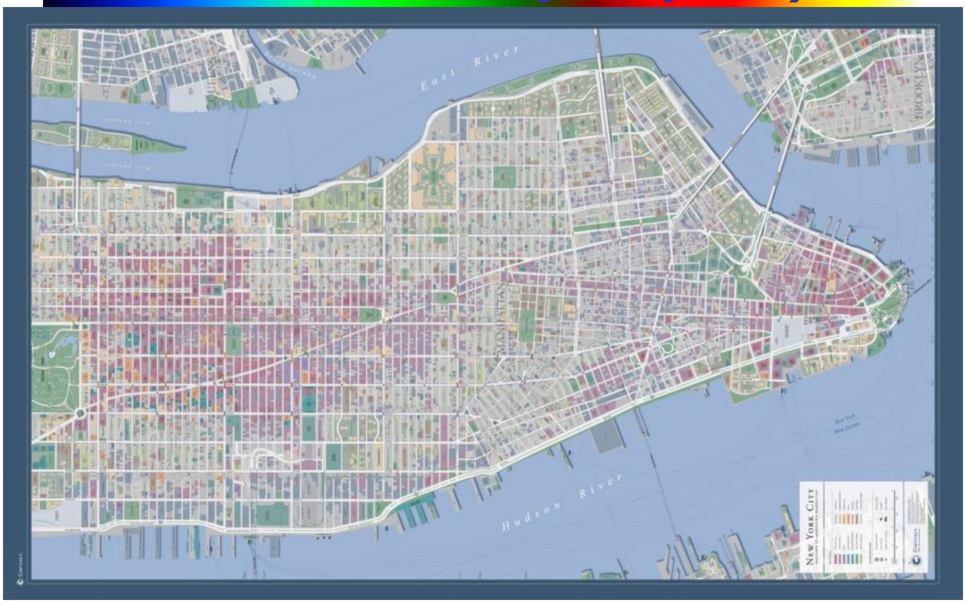
- Also, one can use weighted distance, parametric Pearson product moment correlation, or other dissimilarity measures

Similarity and Dissimilarity Between Objects (Cont.)

- *If q = 1, d is Manhattan distance*

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

Similarity and Dissimilarity Between Objects (Cont.)



25

Similarity and Dissimilarity Between Objects (Cont.)

26

Similarity and Dissimilarity Between Objects (Cont.)

- *If q = 1, d is Manhattan distance*

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

- *If q = 2, d* is Euclidean distance:

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

Metric Distances

- Is distance *d(i,j)* a metric?

Metric Distances

- Is distance $d(i,j)$ a metric?
- Axioms of a metric
  - d is a metric if it is a function from pairs of points to real numbers such that:
    - $d(i,j) \geq 0$
    - $d(i,i) = 0$
    - $d(i,j) = d(j,i)$
    - $d(i,j) \leq d(i,k) + d(k,j)$ (triangle inequality)

Binary Variables

- A contingency table for binary data

		Object j		
		1	0	*sum*
Object i	1	a	b	$a+b$
	0	c	d	$c+d$
	sum	$a+c$	$b+d$	p

Binary Variables

- A contingency table for binary data

		Object j		
		1	0	*sum*
Object i	1	a	b	$a+b$
	0	c	d	$c+d$
	sum	$a+c$	$b+d$	p

- Distance measure for symmetric binary variables:

$$d(i,j) = \frac{b+c}{a+b+c+d}$$

Binary Variables

- A contingency table for binary data

		Object j		
		1	0	*sum*
Object i	1	a	b	$a+b$
	0	c	d	$c+d$
	sum	$a+c$	$b+d$	p

- Distance measure for symmetric binary variables:

$$d(i,j) = \frac{b+c}{a+b+c+d}$$

- Distance measure for asymmetric binary variables:

$$d(i,j) = \frac{b+c}{a+b+c}$$

Binary Variables

- A contingency table for binary data

		Object j		
		1	0	*sum*
Object *i*	1	a	b	$a+b$
	0	c	d	$c+d$
	sum	$a+c$	$b+d$	p

- Distance measure for symmetric binary variables:

$$d(i,j) = \frac{b+c}{a+b+c+d}$$

- Distance measure for asymmetric binary variables:

$$d(i,j) = \frac{b+c}{a+b+c}$$

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

$$sim_{Jaccard}(i,j) = \frac{a}{a+b+c}$$

  - equals to: size of intersection over size of union
  - ($1\text{-}sim_{Jaccard}$) is a distance measure

Dissimilarity between Binary Variables

- Example

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

  - gender is a symmetric attribute
  - the remaining attributes are asymmetric binary
  - let the values Y and P be set to 1, and the value N be set to 0
    - then, if we only take into account the asymmetric variables:

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

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

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

Categorical (Nominal) Variables

- A generalization of the binary variable in that it can take more than 2 states, e.g., red, yellow, blue, green
- Method 1: Simple matching
  - *m*: # of matches, *p*: total # of variables

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

- Method 2: use a large number of binary variables
  - creating a new binary variable for each of the *M* nominal states

Ordinal Variables

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

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

  - compute the dissimilarity using methods for interval-scaled variables

Ratio-Scaled Variables

- Ratio-scaled variable: a positive measurement on a nonlinear scale, approximately at exponential scale,
  such as Ae^{Bt} or Ae^{-Bt}
- Methods:
  - treat them like interval-scaled variables—*not a good choice!* (why?—the scale can be distorted)
  - apply logarithmic transformation

$$y_{if} = \log(x_{if})$$

  - treat them as continuous ordinal data treat their rank as interval-scaled

Variables of Mixed Types

- A database may contain all the six types of variables
  - symmetric binary, asymmetric binary, categorical, ordinal, interval and ratio
- One may use a weighted formula to combine their effects

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

  - f is binary or nominal:
    $d_{ij}^{(f)} = 0$ if $x_{if} = x_{jf}$, or $d_{ij}^{(f)} = 1$ otherwise
  - f is interval-based: use the normalized distance
  - f is ordinal or ratio-scaled
    - compute ranks r_{if} and
    - and treat z_{if} as interval-scaled $z_{if} = \frac{r_{if} - 1}{M_f - 1}$

Vector Objects

- Vector objects: keywords in documents, gene features in micro-arrays, etc.
- Broad applications: information retrieval, biologic taxonomy, etc.
- Cosine distance $s(\vec{X}, \vec{Y}) = \frac{\vec{X}^t \cdot \vec{Y}}{|\vec{X}||\vec{Y}|}$,

  $\vec{X}^t$ is a transposition of vector $\vec{X}$, $|\vec{X}|$ is the Euclidean normal of vector $\vec{X}$,
  - cosine distance is a distance measure
- A variant: Tanimoto coefficient $s(\vec{X}, \vec{Y}) = \frac{\vec{X}^t \cdot \vec{Y}}{\vec{X}^t \cdot \vec{X} + \vec{Y}^t \cdot \vec{Y} - \vec{X}^t \cdot \vec{Y}}$,
  - expresses the ratio of number of attributes shared by *x* and *y* to the number of total attributes of *x* and *y*

String Objects

- string objects: words of a document, genes, etc.
- Edit distance
  - number of inserts and deletes to change one string into another.
  - edit distance is a distance measure
- example:
  - *x* = *abcde* ; *y* = *bcduve*.
  - Turn *x* into *y* by deleting *a*, then inserting *u* and *v* after *d*.
    - Edit-distance = 3.

Roadmap

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, ROCK, CAMELEON
- Density-based approach:
  - Based on connectivity and density functions
  - Typical methods: DBSACN, OPTICS, DenClue

Major Clustering Approaches (II)

- Grid-based approach:
  - based on a multiple-level granularity structure
  - Typical methods: STING, WaveCluster, CLIQUE
- 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: pCluster
- User-guided or constraint-based:
  - Clustering by considering user-specified or application-specific constraints
  - Typical methods: COD (obstacles), constrained clustering

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

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

Typical Alternatives to Calculate the Distance between Clusters

- Single link: smallest distance between an element in one cluster and an element in the other, i.e., $dis(K_i, K_j) = min(t_{ip}, t_{jq})$

- Complete link: largest distance between an element in one cluster and an element in the other, i.e., $dis(K_i, K_j) = max(t_{ip}, t_{jq})$

- Average: avg distance between an element in one cluster and an element in the other, i.e., $dis(K_i, K_j) = avg(t_{ip}, t_{jq})$

Typical Alternatives to Calculate the Distance between Clusters

- Centroid: distance between the centroids of two clusters, i.e., $dis(K_i, K_j) = dis(C_i, C_j)$

- Medoid: distance between the medoids of two clusters, i.e., $dis(K_i, K_j) = dis(M_i, M_j)$
  - Medoid: one chosen, centrally located object in the cluster
    - medoid is the object (of a cluster) whose average dissimilarity to all the other objects in the cluster is minimal

Roadmap

1. What is Cluster Analysis?
2. Types of Data in Cluster Analysis
3. A Categorization of Major Clustering Methods
4. Partitioning Methods
5. Hierarchical Methods
6. Density-Based Methods
7. Grid-Based Methods
8. Model-Based Methods
9. Clustering High-Dimensional Data
10. Constraint-Based Clustering
11. Summary

Partitioning Algorithms: Basic Concept

- Partitioning method: Construct a partition of a database ***D*** of ***n*** objects into a set of ***k*** clusters, s.t., min sum of squared distance

$$\Sigma_{m=1}^{k}\Sigma_{t_{mi}\in Km}(C_m - t_{mi})^2$$

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

Partitioning Algorithms: Basic Concept

- Partitioning method: Construct a partition of a database ***D*** of ***n*** objects into a set of ***k*** clusters, s.t., min sum of squared distance

$$\Sigma_{m=1}^{k}\Sigma_{t_{mi}\in Km}(C_m - t_{mi})^2$$

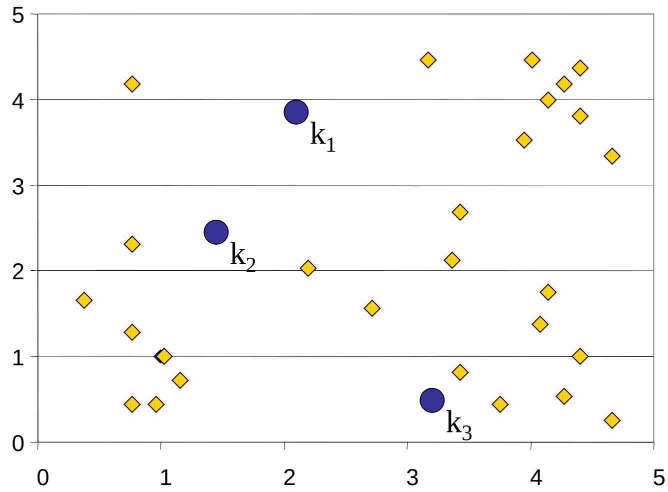
- Given a *k*, find a partition of *k clusters* that optimizes the chosen partitioning criterion
  - Global optimal: exhaustively enumerate all partitions
  - Heuristic methods: *k-means* and *k-medoids* algorithms
  - *k-means* (MacQueen'67): Each cluster is represented by the center of the cluster
  - *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

- 1. Decide on a value for *k*.
- 2. Initialize the *k* cluster centers (randomly, if necessary).
- 3. Decide the class memberships of the *N* objects by assigning them to the nearest cluster center.
- 4. Re-estimate the *k* cluster centers, by assuming the memberships found above are correct.
- 5. If none of the *N* objects changed membership in the last iteration, exit. Otherwise goto 3.

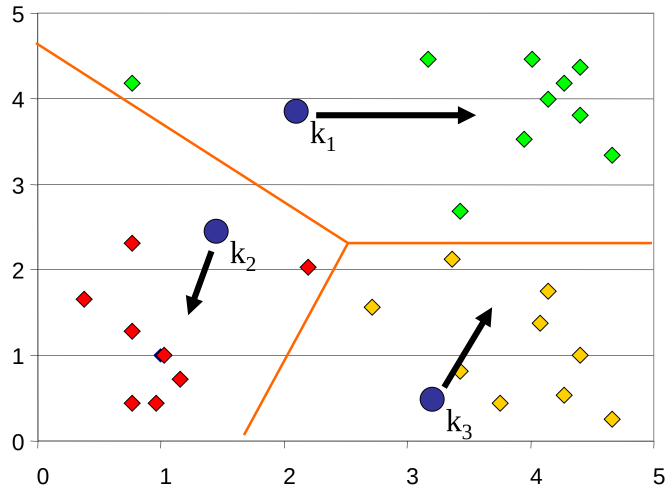
K-means Clustering: Step 1

Algorithm: k-means, Distance Metric: Euclidean Distance



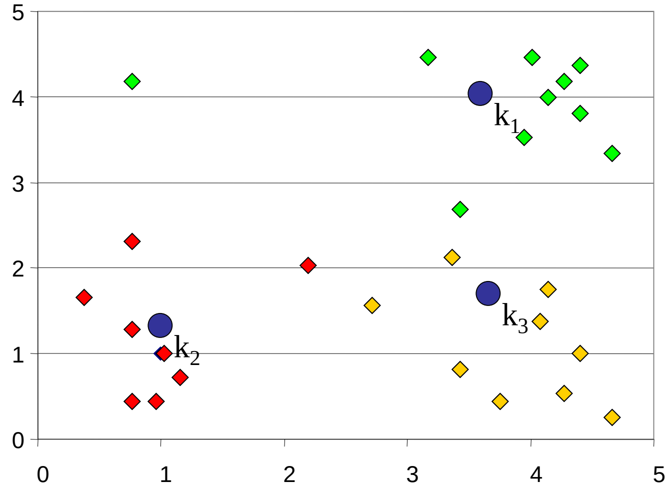
K-means Clustering: Step 2

Algorithm: k-means, Distance Metric: Euclidean Distance



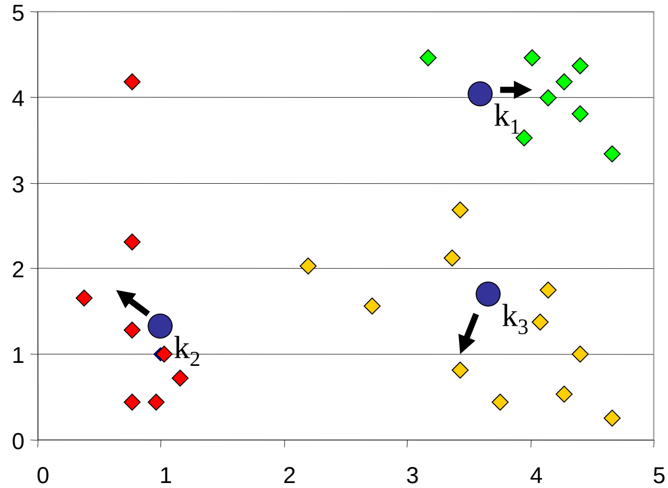
K-means Clustering: Step 3

Algorithm: k-means, Distance Metric: Euclidean Distance



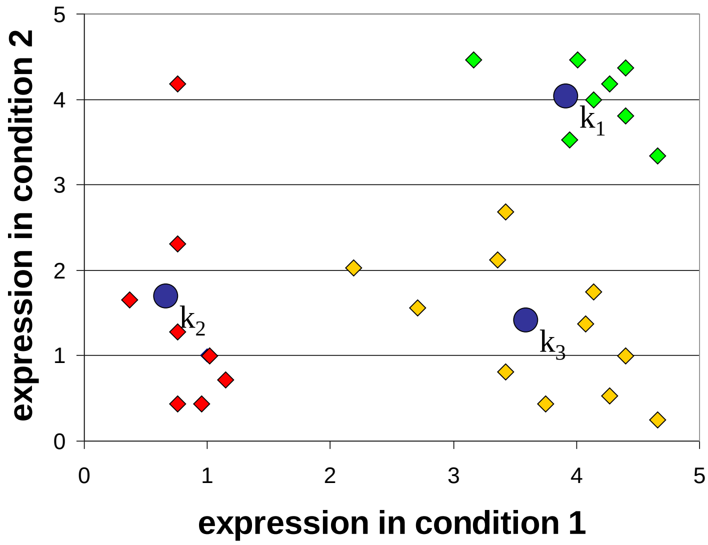
K-means Clustering: Step 4

Algorithm: k-means, Distance Metric: Euclidean Distance



K-means Clustering: Step 5

Algorithm: k-means, Distance Metric: Euclidean Distance



Comments on the *K-Means* Method

- Strength: *Relatively 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: $O(ks^2 + k(n-k))$

Comments on the *K-Means* Method

- Strength: *Relatively 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: Optimality?

Comments on the *K-Means* Method

- Strength: *Relatively 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 optimum*. The *global optimum* may be found using techniques such as: *deterministic annealing* and *genetic algorithms*

Comments on the *K-Means* Method

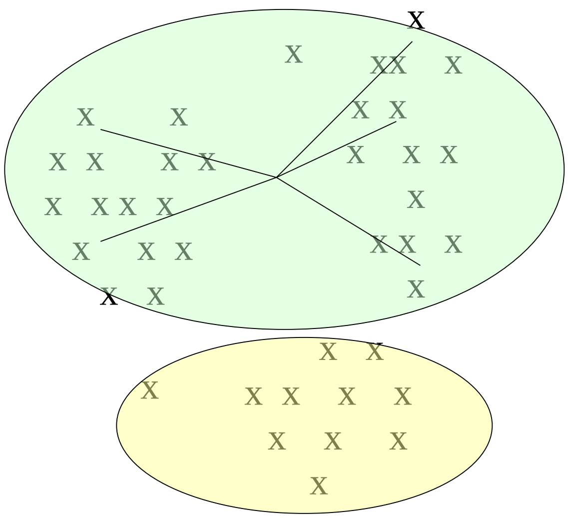
- Strength: *Relatively efficient*: *O*(*tkn*), where *n* is # objects, *k* is # clusters, and *t* is # iterations. Normally, *k*, *t* << *n*.
    - Comparing: PAM: $O(k(n-k)^2)$, CLARA: $O(ks^2 + k(n-k))$
- Comment: Often terminates at a *local optimum*. The *global optimum* may be found using techniques such as: *deterministic annealing* and *genetic algorithms*
- Weakness?

Comments on the *K-Means* Method

- Strength: *Relatively 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 optimum*. The *global optimum* may be found using techniques such as: *deterministic annealing* and *genetic algorithms*
- Weakness
    - Applicable only when *mean* is defined, then what about categorical data?
    - Need to specify *k,* the *number* of clusters, in advance
    - Unable to handle noisy data and *outliers*
    - Not suitable to discover clusters with *non-convex shapes*

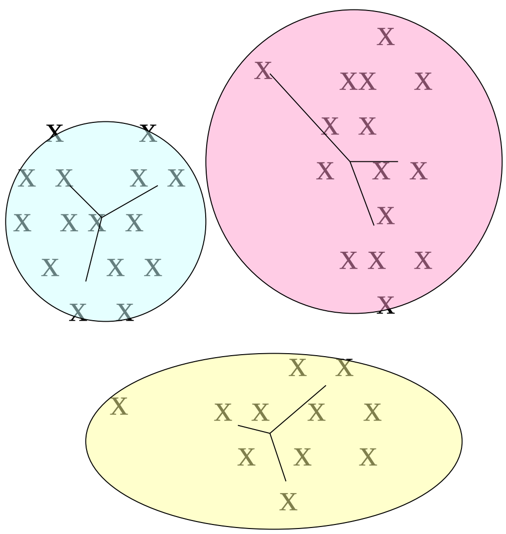
Example: Picking *k*

Too few;
many long
distances
to centroid.



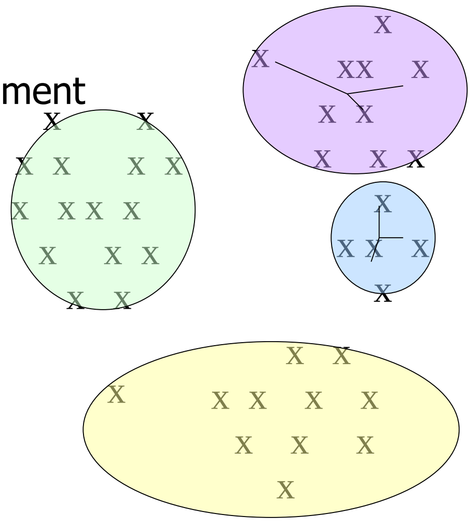
Example: Picking *k*

Just right;
distances
rather short.



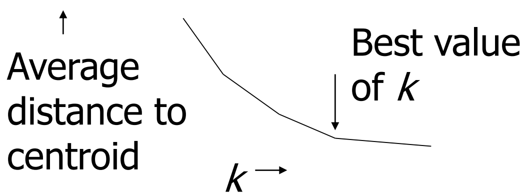
Example: Picking *k*

Too many;
little improvement
in average
distance.



Getting *k* Right

- Try different *k*, looking at the change in the average distance to centroid, as *k* increases.
- Average falls rapidly until right *k*, then changes little.

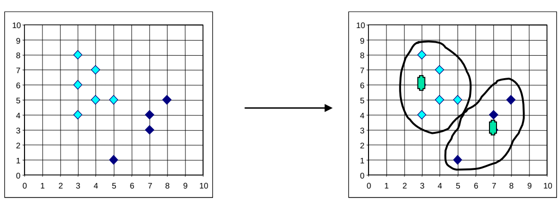


Variations of the *K-Means* Method

- A few 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* (Huang'98)
  - 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?

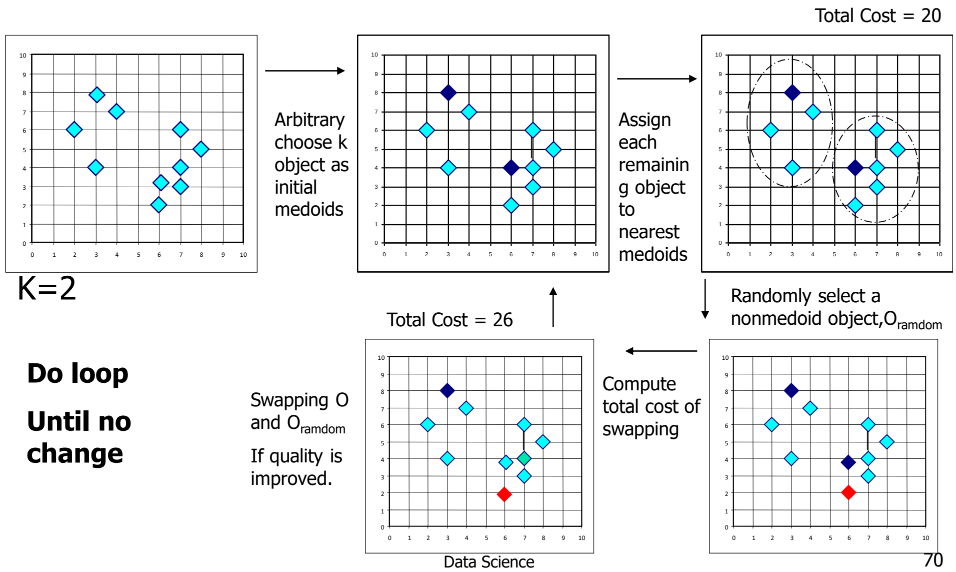
- The k-means algorithm is sensitive to outliers !
  - Since an object with an extremely large value may substantially distort the distribution of the data.
- K-Medoids: Instead of taking the **mean** value of the object in a cluster as a reference point, **medoids** can be used, which is the **most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



The *K-Medoids* Clustering Method

- Find *representative* objects, called medoids, in clusters
- *PAM* (Partitioning Around Medoids, 1987)
  - starts from an initial set of medoids and iteratively replaces one of the medoids by one of the non-medoids if it improves the total distance of the resulting clustering
  - *PAM* works effectively for small data sets, but does not scale well for large data sets
- *CLARA* (Kaufmann & Rousseeuw, 1990)
- *CLARANS* (Ng & Han, 1994): Randomized sampling
- Focusing + spatial data structure (Ester et al., 1995)

A Typical K-Medoids Algorithm (PAM)



PAM (Partitioning Around Medoids) (1987)

- PAM (Kaufman and Rousseeuw, 1987), built in Splus
- Use real object to represent the cluster
  - Select ***k*** representative objects arbitrarily
  - For each pair of non-selected object ***h*** and selected object ***i***, calculate the total swapping cost TC_{ih}
  - For each pair of ***i*** and ***h***,
    - If $TC_{ih} < 0$, ***i*** is replaced by ***h***
    - Then assign each non-selected object to the most similar representative object
  - repeat steps 2-3 until there is no change

What Is the Problem with PAM?

- Pam is more robust than k-means in the presence of noise and outliers because a medoid is less influenced by outliers or other extreme values than a mean
- Pam works efficiently for small data sets but does not **scale well** for large data sets.
  - $O(k(n-k)^2)$ for each iteration

where n is # of data,k is # of clusters

➔Sampling based method,
CLARA(Clustering LARge Applications)

CLARA (Clustering Large Applications) (1990)

- *CLARA* (Kaufmann and Rousseeuw in 1990)
  - Built in statistical analysis packages, such as S+
- It 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) *(1994)*

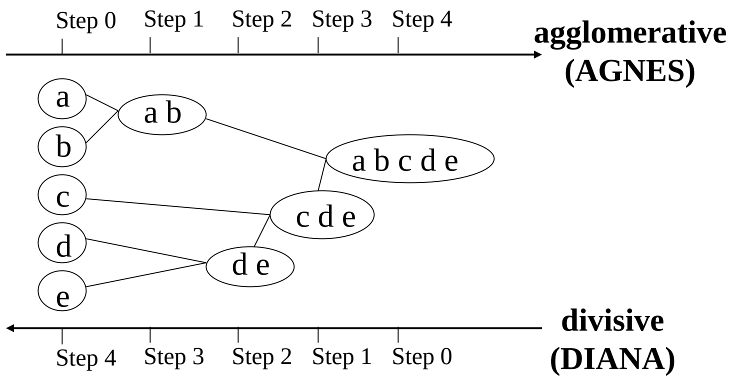
- *CLARANS* (A Clustering Algorithm based on Randomized Search) (Ng and Han'94)
- CLARANS draws sample of neighbors dynamically
- The clustering process can be presented as searching a graph where every node is a potential solution, that is, a set of *k* medoids
- If the local optimum is found, *CLARANS* starts with new randomly selected node in search for a new local optimum
- It is more efficient and scalable than both *PAM* and *CLARA*
- Focusing techniques and spatial access structures may further improve its performance (Ester et al.'95)

Roadmap

1. What is Cluster Analysis?
2. Types of Data in Cluster Analysis
3. A Categorization of Major Clustering Methods
4. Partitioning Methods
5. Hierarchical Methods
6. Density-Based Methods
7. Grid-Based Methods
8. Model-Based Methods
9. Clustering High-Dimensional Data
10. Constraint-Based Clustering
11. Summary

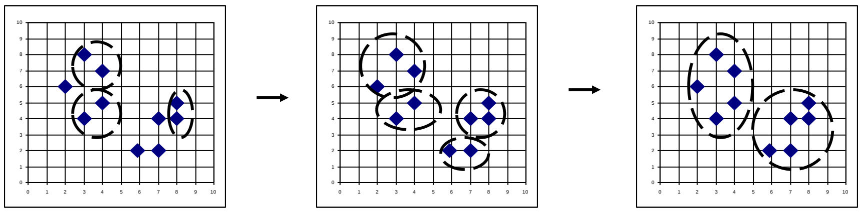
Hierarchical Clustering

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

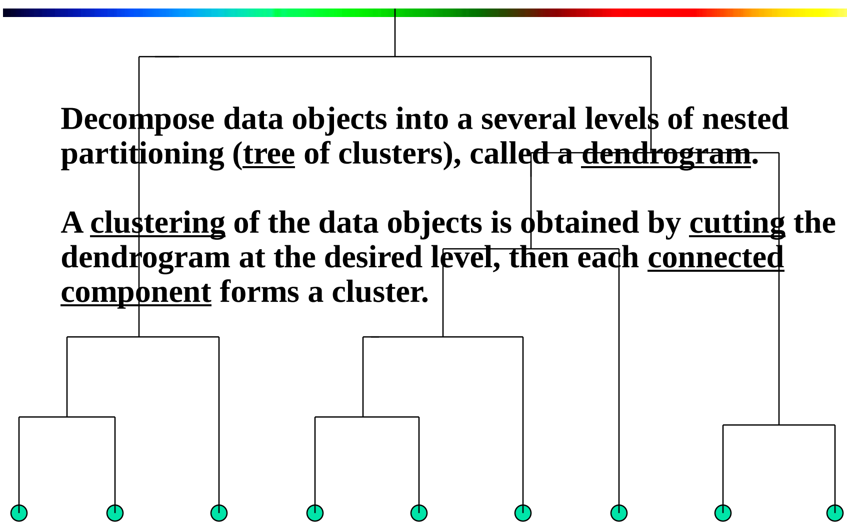


AGNES (Agglomerative Nesting)

- Introduced in Kaufmann and Rousseeuw (1990)
- Implemented in statistical analysis 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 the Clusters are Merged

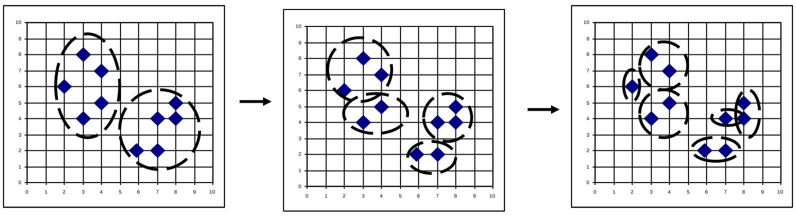


Decompose data objects into a several levels of nested partitioning (<u>tree</u> of clusters), called a <u>dendrogram</u>.

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

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



Recent Hierarchical Clustering Methods

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

BIRCH (1996)

- 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

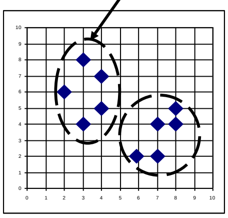
Clustering Feature: $CF = (N, \overrightarrow{LS}, SS)$

N: **Number of data points**

$LS: \Sigma^{N}_{i=1}=\overrightarrow{X_i}$

$SS: \Sigma^{N}_{i=1}=\overrightarrow{X_i}^2$

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

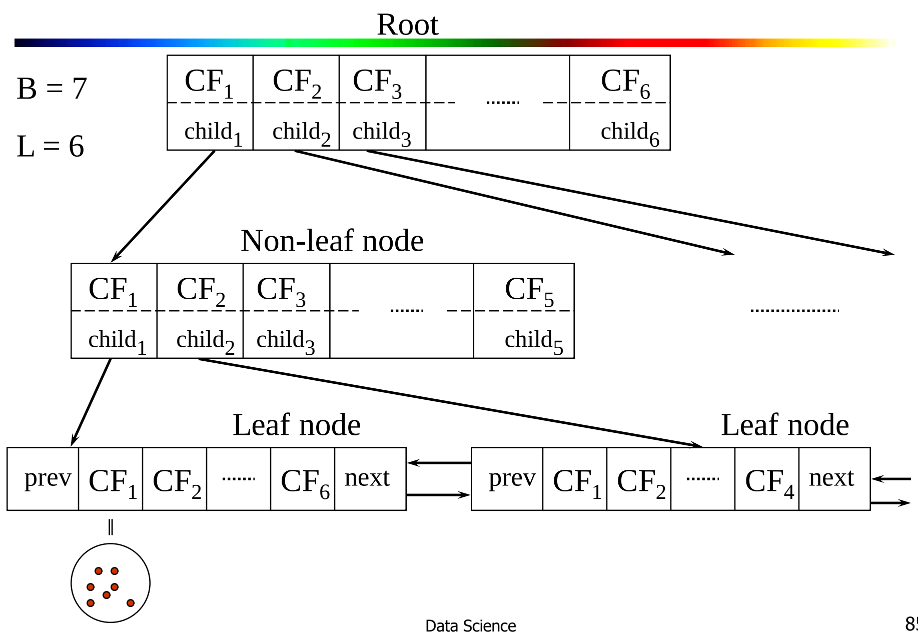


(3,4)
(2,6)
(4,5)
(4,7)
(3,8)

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: specify the maximum number of children.
  - threshold: max diameter of sub-clusters stored at the leaf nodes

The CF Tree Structure



Clustering Categorical Data: The ROCK Algorithm

- ROCK: RObust Clustering using linKs
  - S. Guha, R. Rastogi & K. Shim, ICDE'99
- Major ideas
  - Not distance-based
  - Use links to measure similarity/proximity
  - Measure similarity between points, as well as between their corresponding neighborhoods
    - two points are closer together if they share some of their neighbors
- Algorithm: sampling-based clustering
  - Draw random sample
  - Cluster with links
  - Label data in disk
  - Computational complexity: $O(n^2 + nm_m m_a + n^2 \log n)$

Similarity Measure in ROCK

- Traditional measures for categorical data may not work well, e.g., Jaccard coefficient
- Example: Two groups (clusters) of transactions
  - C_1. <a, b, c, d, e>: {a, b, c}, {a, b, d}, {a, b, e}, {a, c, d}, {a, c, e}, {a, d, e}, {b, c, d}, {b, c, e}, {b, d, e}, {c, d, e}
  - C_2. <a, b, f, g>: {a, b, f}, {a, b, g}, {a, f, g}, {b, f, g}

Similarity Measure in ROCK

- Traditional measures for categorical data may not work well, e.g., Jaccard coefficient
- Example: Two groups (clusters) of transactions
  - C_1. <a, b, c, d, e>: {a, b, c}, {a, b, d}, {a, b, e}, {a, c, d}, {a, c, e}, {a, d, e}, {b, c, d}, {b, c, e}, {b, d, e}, {c, d, e}
  - C_2. <a, b, f, g>: {a, b, f}, {a, b, g}, {a, f, g}, {b, f, g}
- Jaccard co-efficient may lead to wrong clustering result
  - C_1: 0.2 ({a, b, c}, {b, d, e}} to 0.5 ({a, b, c}, {a, b, d})
  - C_1 & C_2: could be as high as 0.5 ({a, b, c}, {a, b, f})
- Jaccard co-efficient-based similarity function: $Sim(T_1, T_2) = \frac{|T_1 \cap T_2|}{|T_1 \cup T_2|}$
  - Ex. Let T_1 = {a, b, c}, T_2 = {c, d, e}

$$Sim(T_1, T_2) = \frac{|\{c\}|}{|\{a, b, c, d, e\}|} = \frac{1}{5} = 0.2$$

Link Measure in ROCK

- Links: # of common neighbors
  - C_1 <a, b, c, d, e>: {a, b, c}, {a, b, d}, {a, b, e}, {a, c, d}, {a, c, e}, {a, d, e}, {b, c, d}, {b, c, e}, {b, d, e}, {c, d, e}
  - C_2 <a, b, f, g>: {a, b, f}, {a, b, g}, {a, f, g}, {b, f, g}

Link Measure in ROCK

- Links: # of common neighbors
  - C_1 <a, b, c, d, e>: {a, b, c}, {a, b, d}, {a, b, e}, {a, c, d}, {a, c, e}, {a, d, e}, {b, c, d}, {b, c, e}, {b, d, e}, {c, d, e}
  - C_2 <a, b, f, g>: {a, b, f}, {a, b, g}, {a, f, g}, {b, f, g}
- Let T_1 = {a, b, c}, T_2 = {c, d, e}, T_3 = {a, b, f}

Link Measure in ROCK

- Links: # of common neighbors
  - C_1 <a, b, c, d, e>: {a, b, c}, {a, b, d}, {a, b, e}, {a, c, d}, {a, c, e}, {a, d, e}, {b, c, d}, {b, c, e}, {b, d, e}, {c, d, e}
  - C_2 <a, b, f, g>: {a, b, f}, {a, b, g}, {a, f, g}, {b, f, g}
- Let T_1 = {a, b, c}, T_2 = {c, d, e}, T_3 = {a, b, f}
  - link(T_1, T_2) = 4, *since they have 4 common neighbors*
    - {a, c, d}, {a, c, e}, {b, c, d}, {b, c, e}

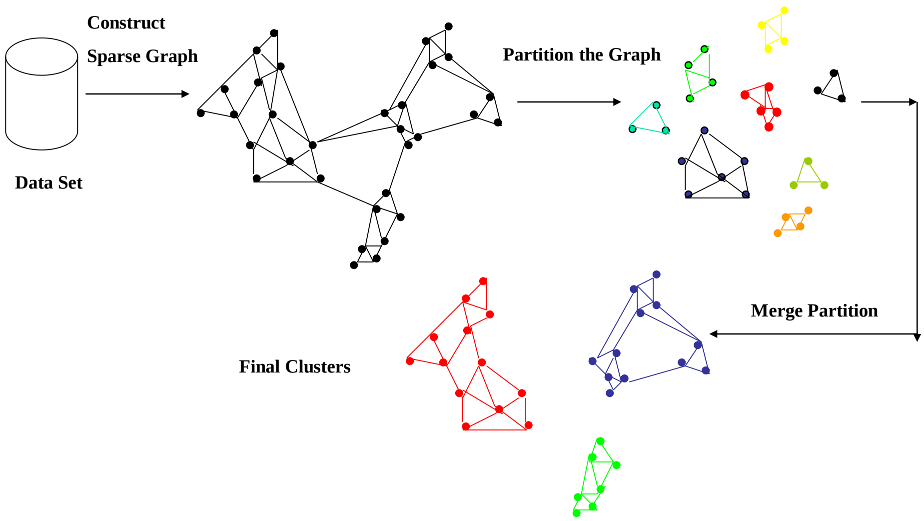
Link Measure in ROCK

- Links: # of common neighbors
  - C_1 <a, b, c, d, e>: {a, b, c}, {a, b, d}, {a, b, e}, {a, c, d}, {a, c, e}, {a, d, e}, {b, c, d}, {b, c, e}, {b, d, e}, {c, d, e}
  - C_2 <a, b, f, g>: {a, b, f}, {a, b, g}, {a, f, g}, {b, f, g}
- Let T_1 = {a, b, c}, T_2 = {c, d, e}, T_3 = {a, b, f}
  - link(T_1, T_2) = 4, *since they have 4 common neighbors*
    - {a, c, d}, {a, c, e}, {b, c, d}, {b, c, e}
  - link(T_1, T_3) = 3, *since they have 3 common neighbors*
    - {a, b, d}, {a, b, e}, {a, b, g}
- Thus, link is a better measure than Jaccard coefficient

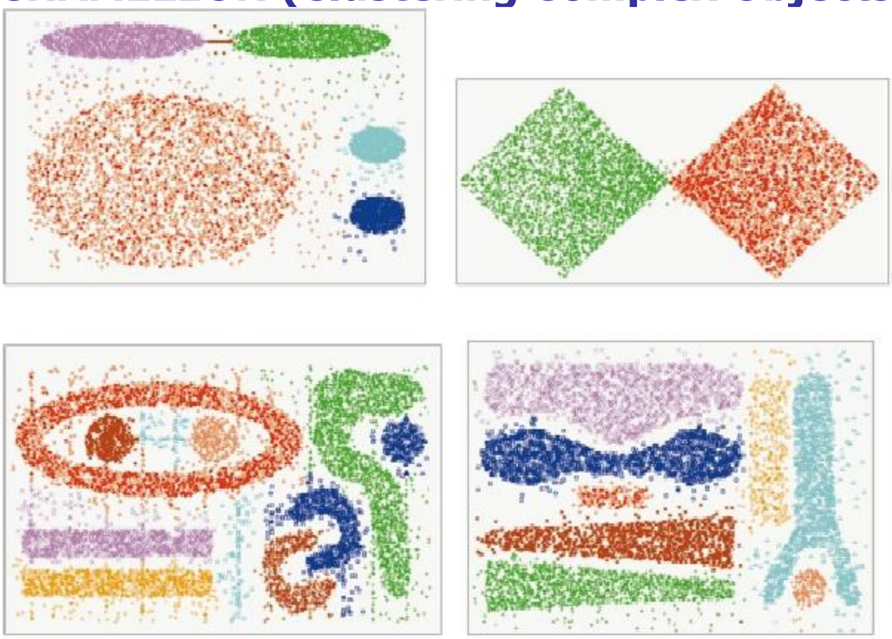
CHAMELEON: Hierarchical Clustering Using Dynamic Modeling (1999)

- CHAMELEON: by G. Karypis, E.H. Han, and V. Kumar'99
- 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 closeness of items within the clusters
  - **Cure** ignores information about **interconnectivity** of the objects, **Rock** ignores information about the **closeness** of two clusters
- A two-phase algorithm
  1. Use a graph partitioning algorithm: cluster objects into a large number of relatively small sub-clusters
  2. Use an agglomerative hierarchical clustering algorithm: find the genuine clusters by repeatedly combining these sub-clusters

Overall Framework of CHAMELEON



CHAMELEON (Clustering Complex Objects)



Roadmap

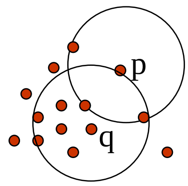
Density-Based Clustering Methods

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

Density-Based Clustering: Basic Concepts

- Two parameters:
  - *Eps*: Maximum radius of the neighbourhood
  - *MinPts*: Minimum number of points in an Eps-neighbourhood of that point
- $N_{Eps}(p)$: $\{q \text{ belongs to } D \mid dist(p,q) <= Eps\}$
- 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:

    $|N_{Eps}(q)| >= MinPts$

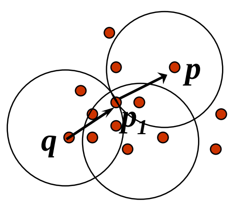


MinPts = 5

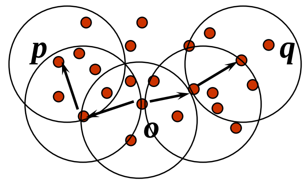
Eps = 1 cm

Density-Reachable and Density-Connected

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

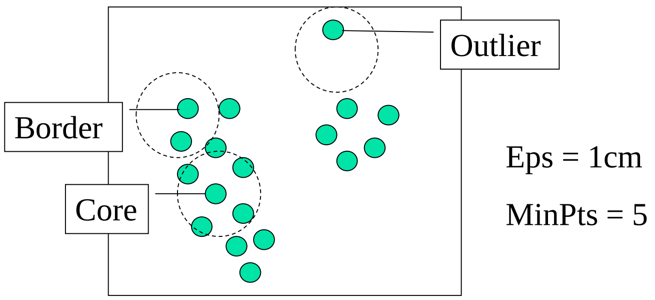


- Density-connected
  - A point *p* is density-connected to a point *q* w.r.t. *Eps*, *MinPts* if there is a point *o* such that both, *p* and *q* are density-reachable from *o* w.r.t. *Eps* and *MinPts*



DBSCAN: Density Based Spatial Clustering of Applications with Noise

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



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.

DBSCAN: Sensitive to Parameters

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

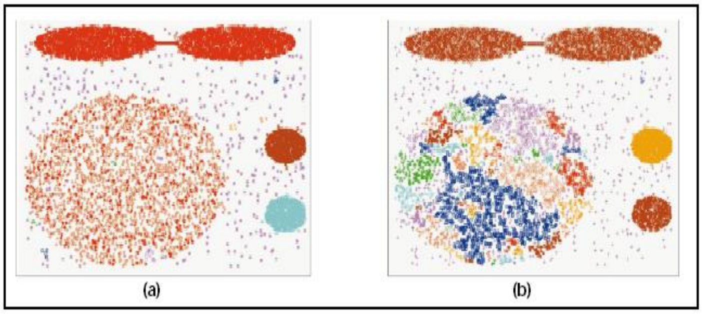
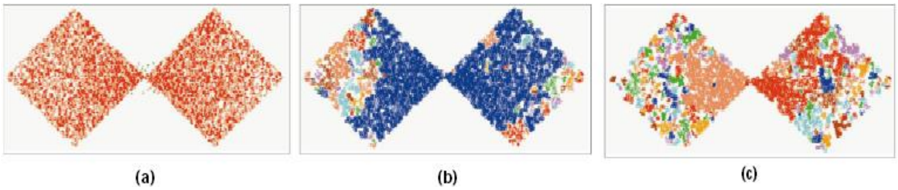


Figure 9. DBScan results for DS2 with MinPts at 4 and Eps at (a) 5.0, (b) 3.5, and (c) 3.0.

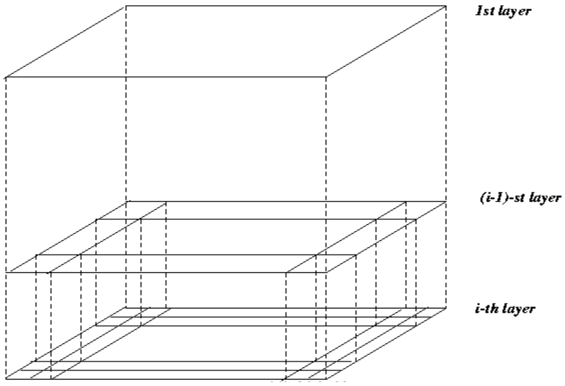


Grid-Based Clustering Method

- Using multi-resolution grid data structure
- Several interesting methods
  - STING (a STatistical INformation Grid approach) by Wang, Yang and Muntz (1997)
  - WaveCluster by Sheikholeslami, Chatterjee, and Zhang (VLDB'98)
    - A multi-resolution clustering approach using wavelet method
  - CLIQUE: Agrawal, et al. (SIGMOD'98)
    - On high-dimensional data (thus put in the section of clustering high-dimensional data

STING: A Statistical Information Grid Approach

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



114

114

The STING Clustering Method

- Each cell at a high level is partitioned into a number of smaller cells in the next lower level
- Statistical info of each cell is calculated and stored beforehand and is used to answer queries
- Parameters of higher level cells can be easily calculated from parameters of lower level cell
  - *count*, *mean*, *s*, *min*, *max*
  - type of distribution—normal, *uniform*, etc.
- Use a top-down approach to answer spatial data queries
- Start from a pre-selected layer—typically with a small number of cells
- For each cell in the current level compute the confidence interval

Comments on STING

- Remove the irrelevant cells from further consideration
- When finish examining the current layer, proceed to the next lower level
- Repeat this process until the bottom layer is reached
- Advantages:
  - Query-independent, easy to parallelize, incremental update
  - *O(K),* where *K* is the number of grid cells at the lowest level
- Disadvantages:
  - All the cluster boundaries are either horizontal or vertical, and no diagonal boundary is detected

Roadmap

1. What is Cluster Analysis?
2. Types of Data in Cluster Analysis
3. A Categorization of Major Clustering Methods
4. Partitioning Methods
5. Hierarchical Methods
6. Density-Based Methods
7. Grid-Based Methods
8. Model-Based Methods
9. Clustering High-Dimensional Data
10. Constraint-Based Clustering
11. Summary

Model-Based Clustering

- What is model-based clustering?
  - Attempt to optimize the fit between the given data and some mathematical model
  - Based on the assumption: Data are generated by a mixture of underlying probability distribution
- Typical methods
  - Statistical approach
    - EM (Expectation maximization), AutoClass
  - Machine learning approach
    - COBWEB, CLASSIT
  - Neural network approach
    - SOM (Self-Organizing Feature Map)

EM — Expectation Maximization

- EM — A popular iterative refinement algorithm
- An extension to k-means
  - Assign each object to a cluster according to a weight (prob. distribution)
  - New means are computed based on weighted measures
- General idea
  - Starts with an initial estimate of the parameter vector
  - Iteratively rescores the patterns against the mixture density produced by the parameter vector
  - The rescored patterns are used to update the parameter updates
  - Patterns belonging to the same cluster, if they are placed by their scores in a particular component
- Algorithm converges fast but may not be in global optima

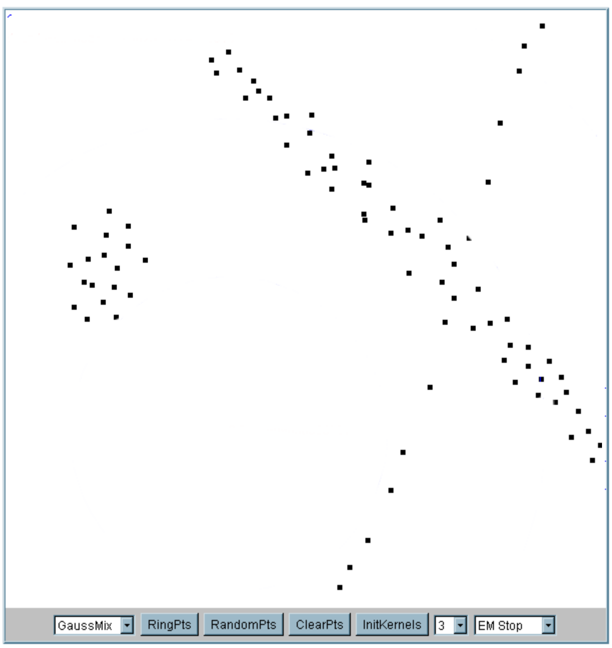
The EM (Expectation Maximization) Algorithm

- Initially, randomly assign k cluster centers
- Iteratively refine the clusters based on two steps
  - Expectation step: assign each data point X_i to cluster C_i with the following probability

$$P(X_i \in C_k) = p(C_k|X_i) = \frac{p(C_k)p(X_i|C_k)}{p(X_i)},$$

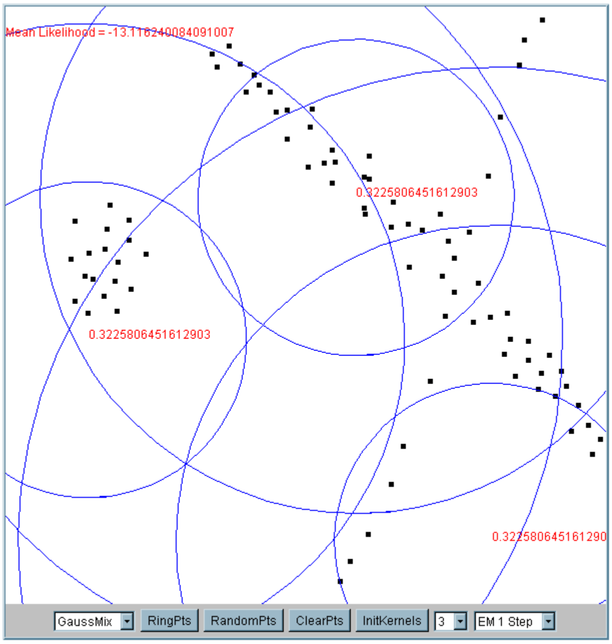
  - Maximization step:
    - Estimation of model parameters

$$m_k = \frac{1}{N}\Sigma_{i=1}^{N}\frac{X_i P(X_i \in C_k)}{\Sigma_j P(X_i \in C_j)}.$$



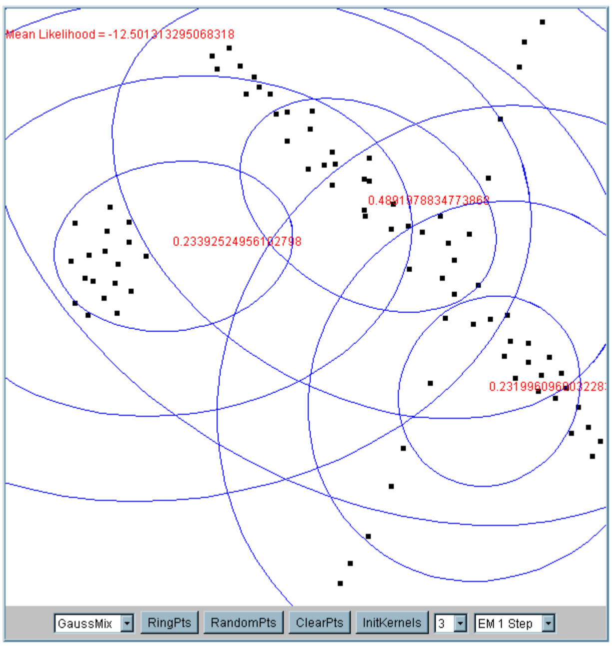
Iteration 1

The cluster means are randomly assigned



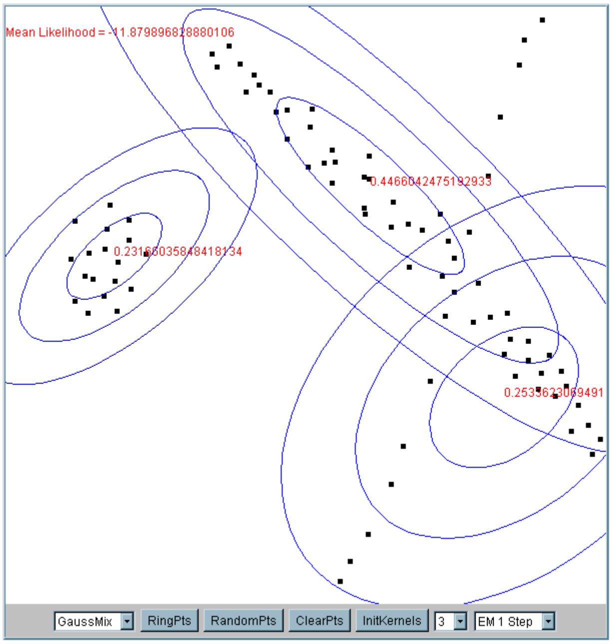
Data Science 126

Iteration 2



Data Science 127

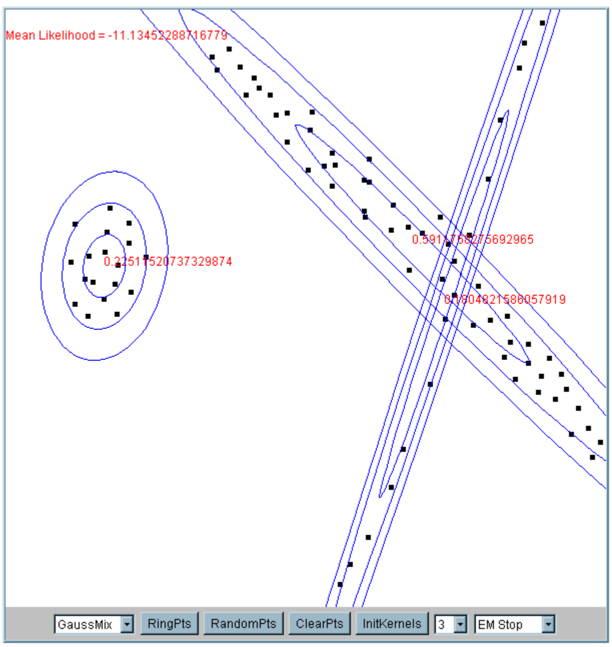
Iteration 5



Data Science 128

128

Iteration 25



Data Science 129

129

Roadmap

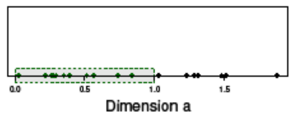
Clustering High-Dimensional Data

- Clustering high-dimensional data
  - Many applications: text documents, DNA micro-array data
  - Major challenges:
    - Many irrelevant dimensions may mask clusters
    - Distance measure becomes meaningless—due to equi-distance
    - Clusters may exist only in some subspaces
- Methods
  - Feature transformation: only effective if most dimensions are relevant
    - PCA & SVD useful only when features are highly correlated/redundant
  - Feature selection: wrapper or filter approaches
    - useful to find a subspace where the data have nice clusters
  - Subspace-clustering: find clusters in all the possible subspaces
    - CLIQUE, ProClus, and frequent pattern-based clustering

The Curse of Dimensionality

(graphs adapted from Parsons et al. KDD Explorations 2004)

- Data in only one dimension is relatively packed

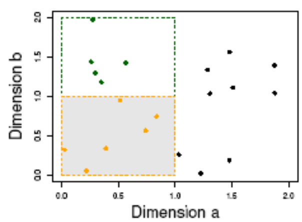


(a) 11 Objects in One Unit Bin

The Curse of Dimensionality

(graphs adapted from Parsons et al. KDD Explorations 2004)

- Data in only one dimension is relatively packed
- Adding a dimension "stretch" the points across that dimension, making them further apart

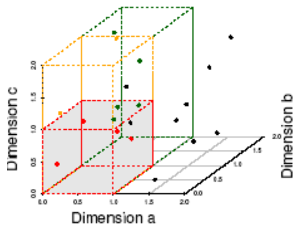


(b) 6 Objects in One Unit Bin

The Curse of Dimensionality

(graphs adapted from Parsons et al. KDD Explorations 2004)

- Data in only one dimension is relatively packed
- Adding a dimension "stretch" the points across that dimension, making them further apart
- Adding more dimensions will make the points further apart—high dimensional data is extremely sparse

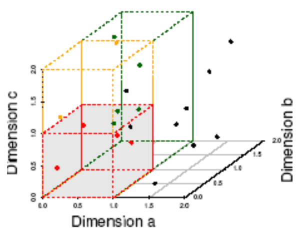


(c) 4 Objects in One Unit Bin

The Curse of Dimensionality

(graphs adapted from Parsons et al. KDD Explorations 2004)

- Data in only one dimension is relatively packed
- Adding a dimension "stretch" the points across that dimension, making them further apart
- Adding more dimensions will make the points further apart—high dimensional data is extremely sparse

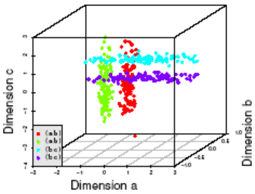


(c) 4 Objects in One Unit Bin

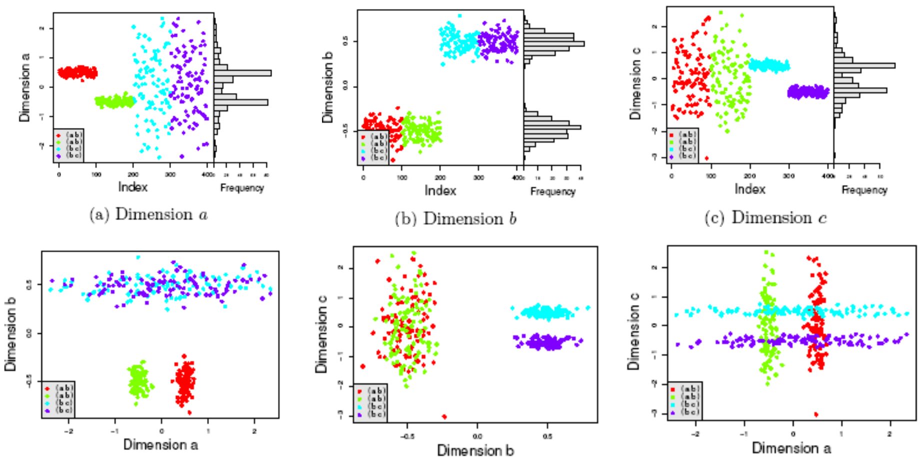
- Distance measure becomes meaningless—due to equi-distance

Why Subspace Clustering?

(adapted from Parsons et al. SIGKDD Explorations 2004)



- Clusters may exist only in some subspaces
- Subspace-clustering: find clusters in all the subspaces



(a) Dimension a (b) Dimension b (c) Dimension c

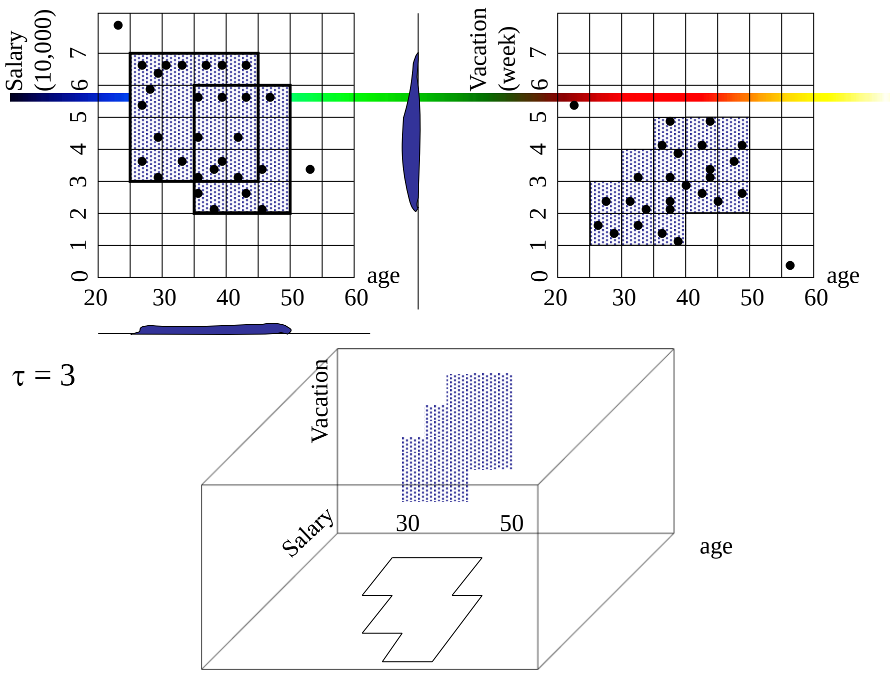
(a) Dims a & b (b) Dims b & c (c) Dims a & c

CLIQUE (Clustering In QUEst)

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

CLIQUE: The Major Steps

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



Strength and Weakness of *CLIQUE*

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

Roadmap

1. What is Cluster Analysis?
2. Types of Data in Cluster Analysis
3. A Categorization of Major Clustering Methods
4. Partitioning Methods
5. Hierarchical Methods
6. Density-Based Methods
7. Grid-Based Methods
8. Model-Based Methods
9. Clustering High-Dimensional Data
10. Constraint-Based Clustering
11. 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
- Outlier detection and analysis are very useful for fraud detection, etc. and can be performed by statistical, distance-based or deviation-based approaches
- There are still lots of research issues on cluster analysis

Problems and Challenges

- Considerable progress has been made in scalable clustering methods
  - Partitioning: k-means, k-medoids, CLARANS
  - Hierarchical: BIRCH, ROCK, CHAMELEON
  - Density-based: DBSCAN, OPTICS, DenClue
  - Grid-based: STING, WaveCluster, CLIQUE
  - Model-based: EM, Cobweb, SOM
  - Frequent pattern-based: pCluster
  - Constraint-based: COD, constrained-clustering
- Current clustering techniques do not address all the requirements adequately, still an active area of research

References (1)

- 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.
- P. Arabie, L. J. Hubert, and G. De Soete. Clustering and Classification. World Scientific, 1996
- Beil F., Ester M., Xu X.: "Frequent Term-Based Text Clustering", KDD'02
- 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.
- D. Gibson, J. Kleinberg, and P. Raghavan. Clustering categorical data: An approach based on dynamic systems. VLDB'98.

References (2)

- V. Ganti, J. Gehrke, 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. In Proc. 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. In *ICDE'99*, pp. 512-521, Sydney, Australia, March 1999.
- A. Hinneburg, D.l 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. Printice Hall, 1988.
- G. Karypis, E.-H. Han, and V. Kumar. CHAMELEON: A Hierarchical Clustering Algorithm Using Dynamic Modeling. *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.
- G. J. McLachlan and K.E. Bkasford. Mixture Models: Inference and Applications to Clustering. John Wiley and Sons, 1988.
- P. Michaud. Clustering techniques. Future Generation Computer systems, 13, 1997.
- R. Ng and J. Han. Efficient and effective clustering method for spatial data mining. VLDB'94.

References (3)

- *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. Proc. 1996 Int. Conf. on Pattern Recognition,.
- G. Sheikholeslami, S. Chatterjee, and A. Zhang. WaveCluster: A multi-resolution clustering approach for very large spatial databases. VLDB'98.
- 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, 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.