UNIT V CLUSTERING, APPLICATIONS AND TRENDS IN DATA MINING

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

- Grouping a set of data objects into clusters

Clustering is unsupervised classification: no predefined classes

Typical applications

- As a stand-alone tool to get insight into data distribution
- As a preprocessing step for other algorithms

General Applications of Clustering:

Pattern Recognition

Spatial Data Analysis

- create thematic maps in GIS by clustering feature spaces
- detect spatial clusters and explain them in spatial data mining

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

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.

Requirements of Clustering in Data Mining:

- ✓ Scalability
- ✓ Ability to deal with different types of attributes
- ✓ 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

What is Cluster Analysis?

The process of grouping a set of physical or abstract objects into classes of similar objects is called clustering. A cluster is a collection of data objects that are similar to one another within the same cluster and are dissimilar to the objects in other clusters. A cluster of data objects can be treated collectively as one group and so may be considered as a form of data compression. Although classification is an effective means for distinguishing groups or classes of objects, it requires the often costly collection and labeling of a large set of training tuples or patterns, which the classifier uses to model each group. It is often more desirable to proceed in the reverse direction: First partition the set of data into groups based on data similarity (e.g., using clustering), and then assign labels to the relatively small number of groups.

Additional advantages of such a clustering-based process are that it is adaptable to changes and helps single out useful features that distinguish different groups.

Clustering is a challenging field of research in which its potential applications pose their own special requirements. The following are typical requirements of clustering in data mining: Scalability: Many clustering algorithms work well on small data sets containing fewer than several hundred data objects; however, a large database may contain millions of objects. Clustering on a sample of a given large data set may lead to biased results. Highly scalable clustering algorithms are needed.

Ability to deal with different types of attributes: Many algorithms are designed to cluster interval-based (numerical) data. However, applications may require clustering other types of data, such as binary, categorical (nominal), and ordinal data, or mixtures of these data types.

Discovery of clusters with arbitrary shape: Many clustering algorithms determine clusters based on Euclidean or Manhattan distance measures. Algorithms based on such distance measures tend to find spherical clusters with similar size and density. However, a cluster could be of any shape. It is important to develop algorithms that can detect clusters of arbitrary shape. Minimal requirements for domain knowledge to determine input parameters: Many clustering algorithms require users to input certain parameters in cluster analysis (such as the number of desired clusters). The clustering results can be quite sensitive to input parameters. Parameters are often difficult to determine, especially for data sets containing high-dimensional objects. This not only burdens users, but it also makes the quality of clustering difficult to control.

Ability to deal with noisy data: Most real-world databases contain outliers or missing, unknown, or erroneous data. Some clustering algorithms are sensitive to such data and may lead to clusters of poor quality. Incremental clustering and insensitivity to the order of input records: Some clustering algorithms cannot incorporate newly inserted data (i.e., database updates) into existing clustering structures and, instead, must determine a new clustering from scratch. Some clustering algorithms are sensitive to the order of input data. That is, given a set of data objects, such an algorithm may return dramatically different clustering's depending on the order of presentation of the input objects. It is important to develop incremental clustering algorithms and algorithms that are insensitive to the order of input.

High dimensionality: A database or a data warehouse can contain several dimensions or attributes. Many clustering algorithms are good at handling low-dimensional data, involving only two to three dimensions.

Human eyes are good at judging the quality of clustering for up to three dimensions. Finding clusters of data objects in high dimensional space is challenging, especially considering that such data can be sparse and highly skewed. Constraint-based clustering: Real-world applications may need to perform clustering under various kinds of constraints. Suppose that your job is to choose the locations for a given number of new automatic banking machines (ATMs) in a city. To decide upon this, you may cluster households while considering constraints such as the city's rivers and highway networks, and the type and number of customers per cluster. A challenging task is to find groups of data with good clustering behavior that satisfy specified constraints.

Interpretability and usability: Users expect clustering results to be interpretable, comprehensible, and usable. That is, clustering may need to be tied to specific semantic interpretations and applications. It is important to study how an application goal may influence the selection of clustering features and methods.

Type of data in clustering analysis:

Data Structures:

- **Data matrix**
  - **(two modes)**

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

- **Dissimilarity matrix**
  - **(one mode)**

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

Measure the Quality of Clustering:

- ✓ Dissimilarity/Similarity metric: Similarity is expressed in terms of a distance function, which is 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 and ratio 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.

Type of data in clustering analysis:

- Interval-scaled variables:
- Binary variables:
- Nominal, ordinal, and ratio variables:
- Variables of mixed types:

Interval-valued variables:

- Standardize data
  - Calculate the mean absolute deviation:

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

Where

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

  - 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*:

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

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

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

$$d(i,j) = |x_{i1} - x_{j1}| + |x_{i2} - x_{j2}| + \ldots + |x_{ip} - x_{jp}|$$

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

$$d(i,j) = \sqrt{(|x_{i1} - x_{j1}|^2 + |x_{i2} - x_{j2}|^2 + \ldots + |x_{ip} - x_{jp}|^2)}$$

- Properties
  - $d(i,j) \geq 0$
  - $d(i,i) = 0$
  - $d(i,j) = d(j,i)$
  - $d(i,j) \leq d(i,k) + d(k,j)$
- Also one can use weighted distance, parametric Pearson product moment correlation, or other dissimilarity measures.

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

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
  - replacing 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

Major Clustering Approaches:

Partitioning algorithms: Construct various partitions and then evaluate them by some criterion

Hierarchy algorithms: Create a hierarchical decomposition of the set of data (or objects) using some criterion

Density-based: based on connectivity and density functions

Grid-based: based on a multiple-level granularity structure

Model-based: A model is hypothesized for each of the clusters and the idea is to find the best fit of that model to each other.

A Categorization of Major Clustering Methods:

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, 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

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

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(Ki,Kj) = min(tip, tjq)

✓ Complete link: largest distance between an element in one cluster and an element in the other, i.e., dis(Ki, Kj) = max(tip, tjq)

✓ Average: avg distance between an element in one cluster and an element in the other, i.e., dis(Ki, Kj) = avg(tip, tjq)

✓ Centroid: distance between the centroids of two clusters, i.e., dis(Ki, Kj) = dis(Ci, Cj)

✓ Medoid: distance between the medoids of two clusters, i.e., dis(Ki, Kj) = dis(Mi, Mj)

✓ Medoid: one chosen, centrally located object in the cluster

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

✓ Centroid: the "middle" of a cluster

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

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

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

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

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

Partitioning Methods:

- Partitioning method: Construct a partition of a database D of n objects into a set of k clusters, s.t., some objective is minimized. E.g., min sum of squared distance in k-means

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

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:

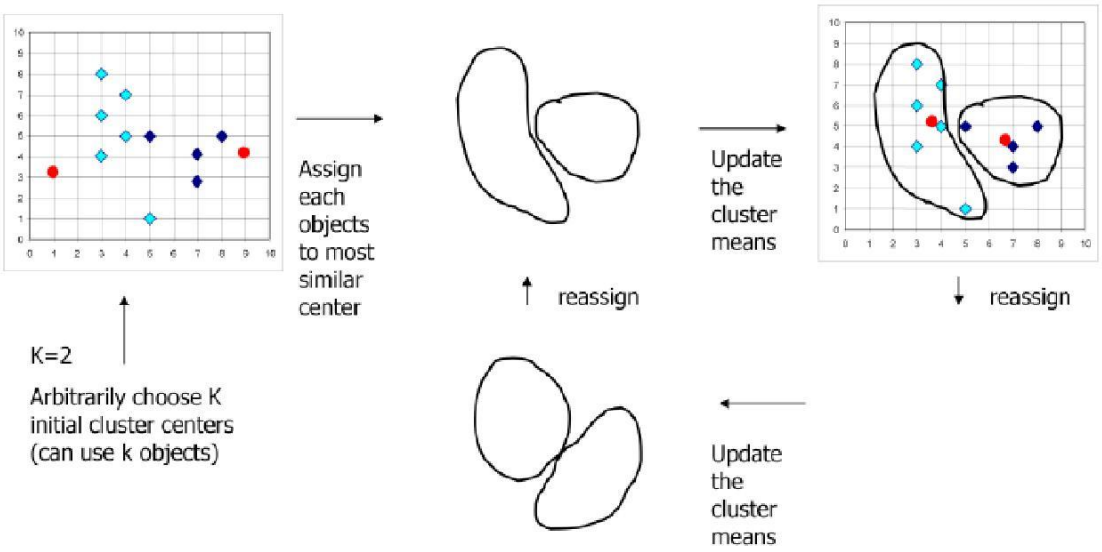
Given *k*, the *k-means* algorithm is implemented in four steps: Partition objects into *k* nonempty subsets

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

Assign each object to the cluster with the nearest seed point

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

The *K-Means* Clustering Method:



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(ks2 + 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*

Variations of the *K-Means* Method:

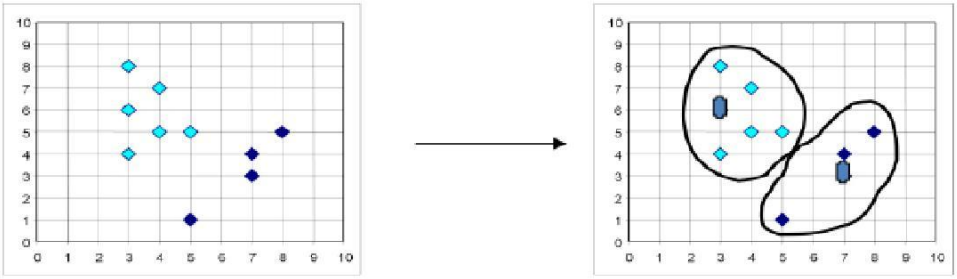
- ✓ A few variants of the *k-means* which differ in
- ✓ Selection of the initial *k* means
- ✓ Dissimilarity calculations
- ✓ Strategies to calculate cluster means

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?:

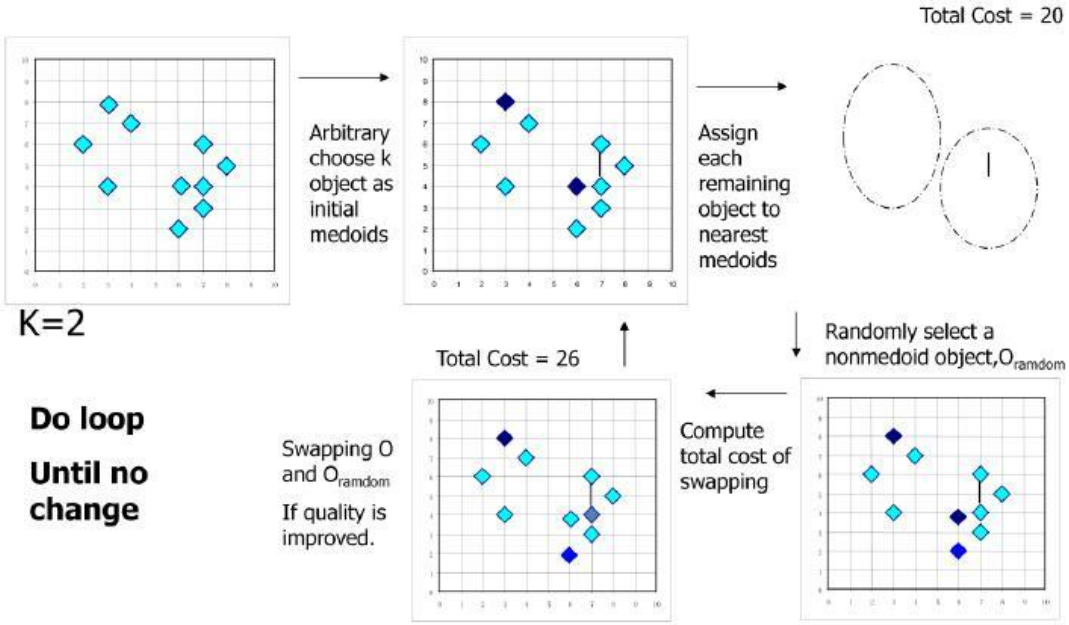
- ✓ The k-means algorithm is sensitive to outliers !
- ✓ Since an object with an extremely large value may substantially distort the distribution of the data.
- ✓ K-Medoids: Instead of taking the mean value of the object in a cluster as a reference point, medoids can be used, which is the most centrally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



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:

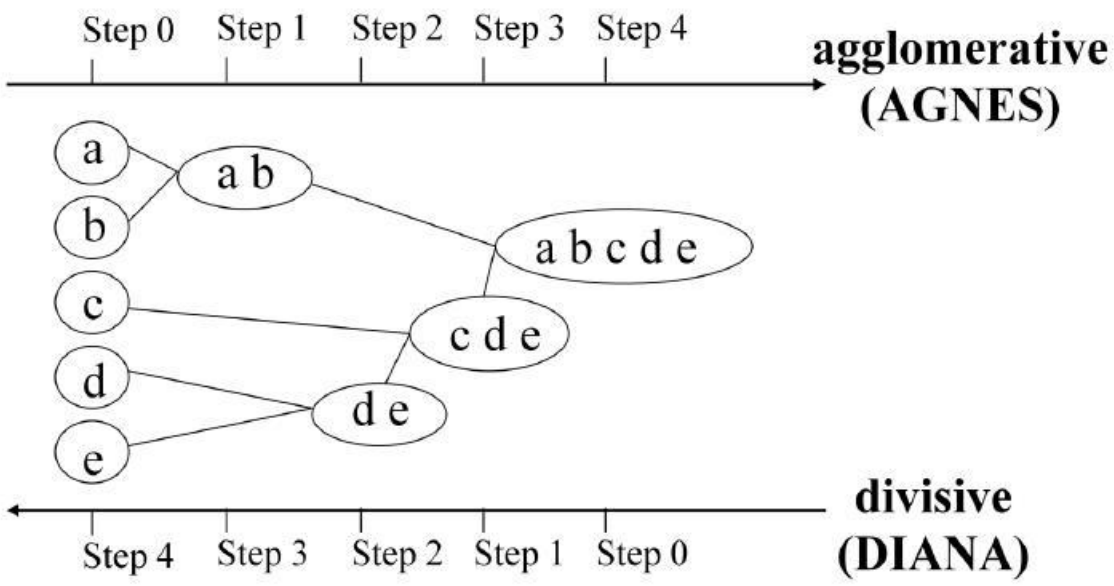
- Efficiency depends on the sample size
- A good clustering based on samples will not necessarily represent a good clustering of the whole data set if the sample is biased

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)

Hierarchical Clustering:

Use distance matrix. 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 (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 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*(*n2*), 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.

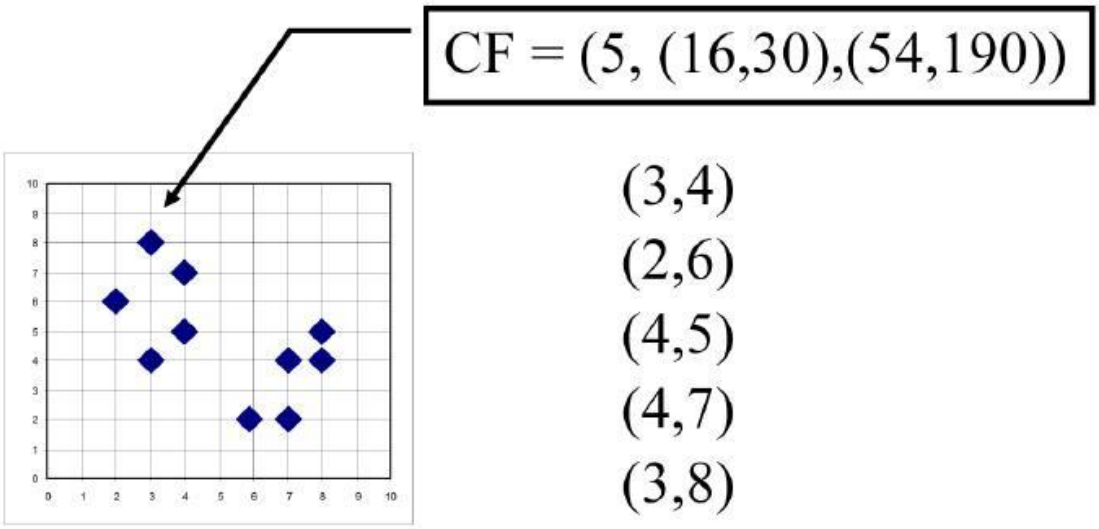
Clustering Feature Vector in BIRCH :

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

N: **Number of data points**

LS (linear sum): $\sum_{i=1}^{N} X_i$

SS (square sum): $\sum_{i=1}^{N} X_i^2$



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

- If memory not enough, rebuild the tree from leaf node by adjusting threshold

Clustering Categorical Data: The ROCK Algorithm:

ROCK: RObust Clustering using linKs

S. Guha, R. Rastogi & K. Shim, ICDE'99

Major ideas

Use links to measure similarity/proximity

Not distance-based

Computational complexity:

Algorithm: sampling-based clustering Draw random sample

Cluster with links

Label data in disk

Experiments

Congressional voting, mushroom data

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})**

$$Sim(T_1, T_2) = \frac{|T_1 \cap T_2|}{|T_1 \cup T_2|}$$

- **Jaccard co-efficient-based similarity function:**
  - **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}
- 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

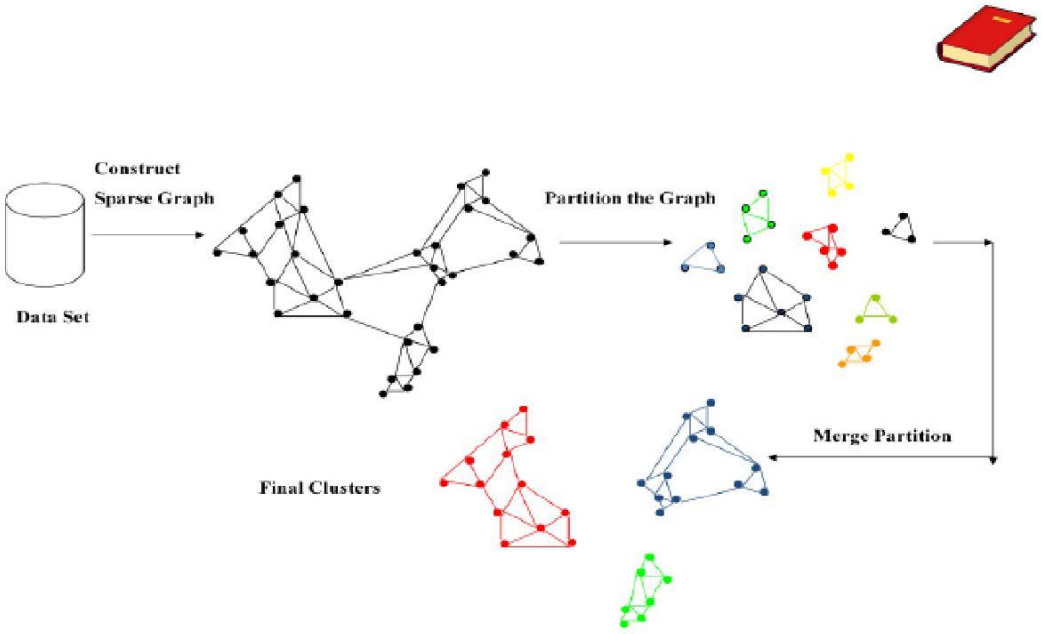
Measures the similarity based on a dynamic model

- 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

- 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:



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)

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.

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

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

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)

Conceptual Clustering:

Conceptual clustering

A form of clustering in machine learning

Produces a classification scheme for a set of unlabeled objects
Finds characteristic description for each concept (class)

COBWEB (Fisher'87)

A popular a simple method of incremental conceptual learning

Creates a hierarchical clustering in the form of a classification tree

Each node refers to a concept and contains a probabilistic description of that concept

Neural Network Approach:

Neural network approaches

- Represent each cluster as an exemplar, acting as a "prototype" of the cluster
- New objects are distributed to the cluster whose exemplar is the most similar according to some distance measure

Typical methods

- SOM (Soft-Organizing feature Map)

- Competitive learning
  - ✓ Involves a hierarchical architecture of several units (neurons)
  - ✓ Neurons compete in a "winner-takes-all" fashion for the object currently being presented

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

Frequent Pattern-Based Approach:

Clustering high-dimensional space (e.g., clustering text documents, microarray data)

- Projected subspace-clustering: which dimensions to be projected on?

  CLIQUE, ProClus

Feature extraction: costly and may not be effective?
Using frequent patterns as "features"

- "Frequent" are inherent features
- Mining freq. patterns may not be so expensive

Typical methods

- Frequent-term-based document clustering
- Clustering by pattern similarity in micro-array data (pClustering)

Why Constraint-Based Cluster Analysis?:

Need user feedback: Users know their applications the best

Less parameters but more user-desired constraints, e.g., an ATM allocation problem: obstacle & desired clusters

A Classification of Constraints in Cluster Analysis:

Clustering in applications: desirable to have user-guided (i.e., constrained) cluster analysis

Different constraints in cluster analysis:

- Constraints on individual objects (do selection first)

  Cluster on houses worth over $300K

- Constraints on distance or similarity functions

  Weighted functions, obstacles (e.g., rivers, lakes)

- Constraints on the selection of clustering parameters

  # of clusters, MinPts, etc.

- User-specified constraints
- Contain at least 500 valued customers and 5000 ordinary ones
- Semi-supervised: giving small training sets as "constraints" or hints

Clustering with User-Specified Constraints:

Example: Locating k delivery centers, each serving at least m valued customers and n ordinary ones

Proposed approach

- Find an initial "solution" by partitioning the data set into k groups and satisfying userconstraints
- Iteratively refine the solution by micro-clustering relocation (e.g., moving δ μ-clusters from cluster Ci to Cj) and "deadlock" handling (break the microclusters when necessary)
- Efficiency is improved by micro-clustering
- How to handle more complicated constraints? E.g., having approximately same number of valued customers in each cluster?! — Can you solve it?

What Is Outlier Discovery?:

What are outliers?

The set of objects are considerably dissimilar from the remainder of the data. Example: Sports: Michael Jordon, Wayne Gretzky, ...

Problem: Define and find outliers in large data sets

Applications:

Credit card fraud detection
Telecom fraud detection
Customer segmentation
Medical analysis

Outlier Discovery: Statistical Approaches

Assume a model underlying distribution that generates data set (e.g. normal distribution)

Use discordancy tests depending on

- **data distribution**
- **distribution parameter (e.g., mean, variance)**
- **number of expected outliers**

Drawbacks

- **most tests are for single attribute**
- **In many cases, data distribution may not be known**

Outlier Discovery: Distance-Based Approach:

Introduced to counter the main limitations imposed by statistical methods

We need multi-dimensional analysis without knowing data distribution

Distance-based outlier: A DB(p, D)-outlier is an object O in a dataset T such that at least a fraction p of the objects in T lies at a distance greater than D from O

Algorithms for mining distance-based outliers

- Index-based algorithm
- Nested-loop algorithm
- Cell-based algorithm

Density-Based Local Outlier Detection:

- Distance-based outlier detection is based on global distance distribution
- It encounters difficulties to identify outliers if data is not uniformly distributed
- Ex. C1 contains 400 loosely distributed points, C2 has 100 tightly condensed points, 2 outlier points o1, o2
- Distance-based method cannot identify o2 as an outlier
- Need the concept of local outlier

Outlier Discovery: Deviation-Based Approach:

- Identifies outliers by examining the main characteristics of objects in a group
- Objects that "deviate" from this description are considered outliers
- Sequential exception technique - simulates the way in which humans can distinguish unusual objects from among a series of supposedly like objects

- OLAP data cube technique - uses data cubes to identify regions of anomalies in large multidimensional data

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