Data Mining: Concepts and Techniques

Cluster Analysis

Li Xiong

Slide credits: Jiawei Han and Micheline Kamber
Tan, Steinbach, Kumar
Chapter 7. Cluster Analysis

- Overview
- Partitioning methods
- Hierarchical methods
- Density-based methods
- Other Methods
- Outlier analysis
- Summary
What is Cluster Analysis?

- Finding groups of objects (clusters)
  - Objects similar to one another in the same group
  - Objects different from the objects in other groups
- Unsupervised learning

Intra-cluster distances are minimized

Inter-cluster distances are maximized
Clustering Applications

- Marketing research
- Social network analysis
Clustering Applications

- WWW: Documents and search results clustering

### ApricotKing Orchard
Great prices on tasty apricots! Organic Certified. Buy online. - [www.apricotking.com](http://www.apricotking.com)

### Apricots
Whatever you're looking for you can get it on eBay. - [www.ebay.com](http://www.ebay.com)

1. **Apricot**
   - The Apricot (*Prunus armeniaca*, syn. *Armeniaca vulgaris*) is a fruit-bearing tree native to China. It is related to the Plum, and classified with it in the *subgenus* *Prunus* of the genus *Prunus*.
   - [en.wikipedia.org/wiki/Apricot](http://en.wikipedia.org/wiki/Apricot)

2. **Apricot Computers**
   - Apricot Computers was a British manufacturer of business personal computers, originally founded in 1985 as "Applied Computer Techniques".

3. **APRICOTS - California Fresh Apricot Council**
   - General information, recipes, retail trade and export information about apricots.
   - [www.californiaapricot.com](http://www.californiaapricot.com)

4. **BellyBytes.com: Apricots**
   - All about Apricots and their nutrients with recipe. ... Apricots. There are at least ten varieties of apricots. Some of the most common ones found in ...
   - [www.bellybytes.com/food/apricots.html](http://www.bellybytes.com/food/apricots.html)

5. **APRICOT**
   - Official site of the Asia Pacific Regional Internet Conference on Operational Technologies. ... The dates for future APRICOTS are ...
   - [www.apricot.net](http://www.apricot.net)

6. **Dried fruit, dried cherries, dried apricots, dried peaches, gourmet food ...**
   - dried fruit, organic dried fruit, dried cherries, dried apricots, dried peaches, dried apples, dried figs ... Our Guarantee: We offer only the finest quality products and have done so ...
   - [www.apricot-farm.com](http://www.apricot-farm.com)
Clustering Applications

- Earthquake studies

![Map of World Earthquakes 1960 - 2000](image)
Clustering Applications

- Biology: plants and animals
- Bioinformatics: microarray data, genes and sequences
Requirements of Clustering

- Scalability
- Ability to deal with different types of attributes
- Ability to handle dynamic data
- Ability to deal with noise and outliers
- Ability to deal with high dimensionality
- Minimal requirements for domain knowledge to determine input parameters
- Incorporation of user-specified constraints
- Interpretability and usability
Quality: What Is Good Clustering?

- Agreement with “ground truth”
- A good clustering will produce high quality clusters with
  - Homogeneity - high intra-class similarity
  - Separation - low inter-class similarity

Intra-cluster distances are minimized

Inter-cluster distances are maximized
Bad Clustering vs. Good Clustering
Similarity or Dissimilarity between Data Objects

\[
\begin{bmatrix}
  x_{11} & \ldots & x_{1f} & \ldots & x_{1p} \\
  \vdots & \ddots & \vdots & \ddots & \vdots \\
  x_{i1} & \ldots & x_{if} & \ldots & x_{ip} \\
  \vdots & \ddots & \vdots & \ddots & \vdots \\
  x_{n1} & \ldots & x_{nf} & \ldots & x_{np}
\end{bmatrix}
\]

1. **Euclidean distance**

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

2. **Manhattan distance**

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

3. **Minkowski distance**

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

4. **Weighted**
Other Similarity or Dissimilarity Metrics

\[
\begin{bmatrix}
x_{11} & \ldots & x_{1f} & \ldots & x_{1p} \\
\vdots & \ddots & \vdots & \ddots & \vdots \\
x_{il} & \ldots & x_{if} & \ldots & x_{ip} \\
\vdots & \ddots & \vdots & \ddots & \vdots \\
x_{nl} & \ldots & x_{nf} & \ldots & x_{np}
\end{bmatrix}
\]

- Pearson correlation
  \[
  \frac{\sum (X_i - \bar{X}_i)(X_j - \bar{X}_j)}{(p - 1)\sigma_{X_i}\sigma_{X_j}}
  \]

- Cosine measure
  \[
  \frac{X_i \cdot X_j}{||X_i|| \cdot ||X_j||}
  \]

- KL divergence, Bregman divergence, …
Different Attribute Types

- To compute $|x_{if} - x_{jf}|$
  - $f$ is continuous
    - Normalization if necessary
    - Logarithmic transformation for ratio-scaled values
      $$x_{if} = Ae^{Bt}, \quad y_{if} = \log(x_{if})$$
  - $f$ is ordinal
    - Mapping by rank
      $$Z_{if} = \frac{r_{if} - 1}{M_f - 1}$$
  - $f$ is categorical
    - Mapping function
      $$|x_{if} - x_{jf}| = 0 \text{ if } x_{if} = x_{jf}, \text{ or } 1 \text{ otherwise}$$
    - Hamming distance (edit distance) for strings
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, 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

- Others
Chapter 7. Cluster Analysis

- Overview
- Partitioning methods
- Hierarchical methods
- Density-based methods
- Other Methods
- Outlier analysis
- 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., the sum of squared distance is minimized

$$\sum_{i=1}^{k} \sum_{p \in C_i} (p - m_i)^2$$

- 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
**K-Means Clustering: Lloyd Algorithm**

- Given \( k \), and randomly choose \( k \) initial cluster centers.
- Partition objects into \( k \) nonempty subsets by assigning each object to the cluster with the nearest centroid.
- Update centroid, i.e. *mean point* of the cluster.
- Go back to Step 2, stop when no more new assignment.
The *K-Means* Clustering Method

- **Example**

  Arbitrarily choose *K* object as initial cluster center

  Assign each objects to most similar center

  Update the cluster means

  Reassign

  Update the cluster means

  Reassign

  K=2
K-means Clustering – Details

- Initial centroids are often chosen randomly.
- The centroid is (typically) the mean of the points in the cluster.
- ‘Closeness’ is measured by Euclidean distance, cosine similarity, correlation, etc.
- Most of the convergence happens in the first few iterations.
  - Often the stopping condition is changed to ‘Until relatively few points change clusters’
- Complexity is $O(tkn)$
  
  $n$ is # objects, $k$ is # clusters, and $t$ is # iterations.
Comments on the \textit{K-Means} Method

- **Strength**
  - Simple and works well for “regular” disjoint clusters
  - Relatively efficient and scalable (normally, $k, t \ll n$)

- **Weakness**
  - Need to specify $k$, the \textit{number} of clusters, in advance
  - Depending on initial centroids, may terminate at a \textit{local optimum}
    - Potential solutions
  - Unable to handle noisy data and \textit{outliers}
  - Not suitable for clusters of
    - Different sizes
    - Non-convex shapes
Importance of Choosing Initial Centroids – Case 1
Importance of Choosing Initial Centroids – Case 2
Limitations of K-means: Differing Sizes

Original Points

K-means (3 Clusters)
Limitations of K-means: Non-convex Shapes

Original Points

K-means (2 Clusters)
Overcoming K-means Limitations

Original Points

K-means Clusters
Overcoming K-means Limitations

Original Points

K-means Clusters
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

PAM (Kaufman and Rousseeuw, 1987)

- Arbitrarily select $k$ objects as medoid
- Assign each data object in the given data set to most similar medoid.
- Randomly select non-medoid object $O'$
- Compute total cost, $S$, of swapping a medoid object to $O'$ (cost as total sum of absolute error)
- If $S<0$, then swap initial medoid with the new one
- Repeat until there is no change in the medoid.

$k$-medoids and $(n-k)$ instances pair-wise comparison
A Typical K-Medoids Algorithm (PAM)

K=2

Do loop
Until no change

Swapping $O$ and $O_{\text{random}}$
If quality is improved.

Total Cost = 20

Randomly select a nonmedoid object, $O_{\text{random}}$

Compute total cost of swapping
What Is the Problem with PAM?

- Pam is more robust than k-means in the presence of noise and outliers.
- Pam works efficiently for small data sets but does not scale well for large data sets.
  - Complexity? $O(k(n-k)^2t)$
  - $n$ is # of data, $k$ is # of clusters, $t$ is # of iterations

Sampling based method,
CLARA(Clustering LARge Applications)
**CLARA (Clustering Large Applications) (1990)**

- **CLARA** (Kaufmann and Rousseeuw in 1990)
- 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)

- The clustering process can be presented as searching a graph where every node is a potential solution, that is, a set of $k$ medoids
  - PAM examines neighbors for local minimum
  - CLARA works on subgraphs of samples
  - CLARANS examines neighbors dynamically
    - If local optimum is found, starts with new randomly selected node in search for a new local optimum
Chapter 7. Cluster Analysis

- Overview
- Partitioning methods
- Hierarchical methods and graph-based methods
- Density-based methods
- Other Methods
- Outlier analysis
- Summary
Hierarchical Clustering

- Produces a set of nested clusters organized as a hierarchical tree
- Can be visualized as a dendrogram
  - A tree-like diagram representing a hierarchy of nested clusters
- Clustering obtained by cutting at desired level
Strengths of Hierarchical Clustering

- Do not have to assume any particular number of clusters
- May correspond to meaningful taxonomies
Hierarchical Clustering

- Two main types of hierarchical clustering
  - Agglomerative:
    - Start with the points as individual clusters
    - At each step, merge the closest pair of clusters until only one cluster (or k clusters) left
  - Divisive:
    - Start with one, all-inclusive cluster
    - At each step, split a cluster until each cluster contains a point (or there are k clusters)
Agglomerative Clustering Algorithm

1. Compute the proximity matrix
2. Let each data point be a cluster
3. **Repeat**
4. Merge the two closest clusters
5. Update the proximity matrix
6. **Until** only a single cluster remains
Starting Situation

- Start with clusters of individual points and a proximity matrix.

![Proximity Matrix Diagram]
Intermediate Situation
How to Define Inter-Cluster Similarity

![Diagram showing two clusters with a question mark indicating similarity]

<table>
<thead>
<tr>
<th></th>
<th>p1</th>
<th>p2</th>
<th>p3</th>
<th>p4</th>
<th>p5</th>
<th>…</th>
</tr>
</thead>
<tbody>
<tr>
<td>p1</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>p2</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>p3</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>p4</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>p5</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Proximity Matrix
Distance Between Clusters

- **Single Link**: smallest distance between points
- **Complete Link**: largest distance between points
- **Average Link**: average distance between points
- **Centroid**: distance between centroids
Hierarchical Clustering: MIN

Nested Clusters

Dendrogram
MST (Minimum Spanning Tree)

- Start with a tree that consists of any point.
- In successive steps, look for the closest pair of points \((p, q)\) such that one point \((p)\) is in the current tree but the other \((q)\) is not.
- Add \(q\) to the tree and put an edge between \(p\) and \(q\).
Min vs. Max vs. Group Average

MIN

MAX

Group Average
Strength of MIN

- Can handle non-elliptical shapes
Limitations of MI N

- Sensitive to noise and outliers

Original Points

Two Clusters
Strength of MAX

- Less susceptible to noise and outliers
Limitations of MAX

- Tends to break large clusters
- Biased towards globular clusters

Original Points

Two Clusters
Hierarchical Clustering: Group Average

- Compromise between Single and Complete Link

  - Strengths
    - Less susceptible to noise and outliers

  - Limitations
    - Biased towards globular clusters
Hierarchical Clustering: Major Weaknesses

- Do not scale well (N: number of points)
  - Space complexity: $O(N^2)$
  - Time complexity: $O(N^3)$
  - $O(N^2 \log(N))$ for some cases/approaches

- Cannot undo what was done previously

- Quality varies in terms of distance measures
  - MIN (single link): susceptible to noise/outliers
  - MAX/GROUP AVERAGE: may not work well with non-globular clusters
Recent Hierarchical Clustering Methods

- **BIRCH (1996)**: uses CF-tree and incrementally adjusts the quality of sub-clusters
- **CURE (1998)**: uses representative points for inter-cluster distance
- **ROCK (1999)**: clustering categorical data by neighbor and link analysis
- **CHAMELEON (1999)**: hierarchical clustering using dynamic modeling
Birch

Birch: Balanced Iterative Reducing and Clustering using Hierarchies (Zhang, Ramakrishnan & Livny, SIGMOD’96)

Main ideas:
- Use in-memory clustering feature to summarize data/cluster
  - Minimize database scans and I/O cost
- Use hierarchical clustering for microclustering and other clustering methods (e.g. partitioning) for macroclustering
  - Fix the problems of hierarchical clustering

Features:
- Scales linearly: single scan and improves the quality with a few additional scans
- handles only numeric data, and sensitive to the order of the data record.
Cluster Statistics

Given a cluster of instances \( \{ \bar{X}_i \} \)

**Centroid:** \[ \bar{X}_0 = \frac{\sum_{i=1}^{N} \bar{X}_i}{N} \]

**Radius:** average distance from member points to centroid
\[ R = \left( \frac{\sum_{i=1}^{N} (\bar{X}_i - \bar{X}_0)^2}{N} \right)^{\frac{1}{2}} \]

**Diameter:** average pair-wise distance within a cluster
\[ D = \left( \frac{\sum_{i=1}^{N} \sum_{j=1}^{N} (\bar{X}_i - \bar{X}_j)^2}{N(N - 1)} \right)^{\frac{1}{2}} \]
Intra-Cluster Distance

Given two clusters

**Centroid Euclidean distance:**

\[ D_0 = \left( \left( \bar{X}_{01} - \bar{X}_{02} \right)^2 \right)^{\frac{1}{2}} \]

**Centroid Manhattan distance:**

\[ D_1 = |\bar{X}_{01} - \bar{X}_{02}| = \sum_{i=1}^{d} |X^{(i)}_{01} - X^{(i)}_{02}| \]

**Average distance:**

\[ D_2 = \left( \frac{\sum_{i=1}^{N_1} \sum_{j=N_1+1}^{N_1+N_2} (\bar{X}_i - \bar{X}_j)^2}{N_1 N_2} \right)^{\frac{1}{2}} \]
Clustering Feature (CF)

Given a cluster \( \{X_1, X_2, \ldots, X_N\} \)

\[
\text{CF} = (N, \overrightarrow{LS}, SS)
\]

- \( N \) is the number of data points
- \( \overrightarrow{LS} = \sum_{i=1}^{N} \overrightarrow{X_i} \)
- \( SS = \sum_{i=1}^{N} \overrightarrow{X_i}^2 \)

\[
\text{CF}_1 + \text{CF}_2 = (N_1 + N_2, \overrightarrow{LS}_1 + \overrightarrow{LS}_2, SS_1 + SS_2)
\]

\[
\text{CF} = (5, (16,30),(54,190))
\]

March 6, 2008
Properties of Clustering Feature

- CF entry is more compact
  - Stores significantly less than all of the data points in the sub-cluster
- A CF entry has sufficient information to calculate statistics about the cluster and intra-cluster distances
- Additivity theorem allows us to merge sub-clusters incrementally & consistently
Hierarchical CF-Tree

- 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

Root

Non-leaf node

Leaf node

Leaf node
CF-Tree Insertion

- Traverse down from root, find the appropriate leaf
  - Follow the "closest"-CF path, w.r.t. intra-cluster distance measures
- Modify the leaf
  - If the closest-CF leaf cannot absorb, make a new CF entry.
  - If there is no room for new leaf, split the parent node
- Traverse back & up
  - Updating CFs on the path or splitting nodes
BI RCH Overview

Data

Phase 1: Load into memory by building a CF tree

Initial CF tree

Phase 2 (optional): Condense into desirable range by building a smaller CF tree

smaller CF tree

Phase 3: Global Clustering

Good Clusters

Phase 4: (optional and offline): Cluster Refining

Better Clusters
The Algorithm: BIRCH

- **Phase 1**: Scan database to build an initial in-memory CF-tree
  - Subsequent phases become fast, accurate, less order sensitive
- **Phase 2**: Condense data (optional)
  - Rebuild the CF-tree with a larger T
- **Phase 3**: Global clustering
  - Use existing clustering algorithm on CF entries
  - Helps fix problem where natural clusters span nodes
- **Phase 4**: Cluster refining (optional)
  - Do additional passes over the dataset & reassign data points to the closest centroid from phase 3
CURE

- CURE: An Efficient Clustering Algorithm for Large Databases (1998) Sudipto Guha, Rajeev Rastogi, Kyusck Shim
- Main ideas:
  - Use representative points for inter-cluster distance
  - Random sampling and partitioning
- Features:
  - Handles non-spherical shapes and arbitrary sizes better
CURE: Cluster Points

- Uses a number of points to represent a cluster

- Representative points are found by selecting a constant number of points from a cluster and then “shrinking” them toward the center of the cluster
  - How to shrink?

- Cluster similarity is the similarity of the closest pair of representative points from different clusters
Experimental Results: CURE

a) BIRCH  b) MST METHOD  c) CURE

Picture from CURE, Guha, Rastogi, Shim.
Experimental Results: CURE

a) BIRCH (centroid)
b) MST METHOD (single link)
c) CURE

Picture from CURE, Guha, Rastogi, Shim.
CURE Cannot Handle Differing Densities

Original Points

CURE
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
  - Sampling-based clustering
- Features:
  - More meaningful clusters
  - Emphasizes interconnectivity but ignores proximity
Similarity Measure in ROCK

- Market basket data clustering
- Jaccard co-efficient-based similarity function:
  \[
  Sim(T_1, T_2) = \frac{|T_1 \cap T_2|}{|T_1 \cup T_2|}
  \]

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\}

Let \(T_1 = \{a, b, c\}\), \(T_2 = \{c, d, e\}\), \(T_3 = \{a, b, f\}\)

- Jaccard co-efficient may lead to wrong clustering result
  \[
  Sim(T_1, T_2) = \frac{|\{c\}|}{|\{a, b, c, d, e\}|} = \frac{1}{5} = 0.2
  \]
  \[
  Sim(T_1, T_3) = \frac{|\{c, f\}|}{|\{a, b, c, f\}|} = \frac{2}{4} = 0.5
  \]
Link Measure in ROCK

- Neighbor: \( Sim(P_1, P_3) \geq \theta \)
- Links: \# of common neighbors
- Example:
  - \( \text{link}(T_1, T_2) = 4, \text{ since they have 4 common neighbors} \)
    - \( \{a, c, d\}, \{a, c, e\}, \{b, c, d\}, \{b, c, e\} \)
  - \( \text{link}(T_1, T_3) = 3, \text{ since they have 3 common neighbors} \)
    - \( \{a, b, d\}, \{a, b, e\}, \{a, b, g\} \)
Rock Algorithm

1. Obtain a sample of points from the data set
2. Compute the link value for each set of points, from the original similarities (computed by Jaccard coefficient)
3. Perform an agglomerative hierarchical clustering on the data using the “number of shared neighbors” as similarity measure
4. Assign the remaining points to the clusters that have been found

- CHAMELEON: by G. Karypis, E.H. Han, and V. Kumar’99
- Basic ideas:
  - A graph-based clustering approach
  - A two-phase algorithm:
    - Partitioning: cluster objects into a large number of relatively small sub-clusters
    - Agglomerative hierarchical clustering: repeatedly combine these sub-clusters
  - Measures the similarity based on a dynamic model
    - interconnectivity and closeness (proximity)
- Features:
  - Handles clusters of arbitrary shapes, sizes, and density
  - Scales well
Graph-Based Clustering

- Uses the proximity graph
  - Start with the proximity matrix
  - Consider each point as a node in a graph
  - Each edge between two nodes has a weight which is the proximity between the two points
- Fully connected proximity graph
  - MIN (single-link) and MAX (complete-link)
- Sparsification
  - Clusters are connected components in the graph
  - CHAMELEON
Overall Framework of CHAMELEON

1. Construct Sparse Graph
2. Partition the Graph
3. Merge Partition

Data Set

Final Clusters
Chameleon: Steps

- **Preprocessing Step:**
  Represent the Data by a Graph
  - Given a set of points, construct the k-nearest-neighbor (k-NN) graph to capture the relationship between a point and its k nearest neighbors
  - Concept of neighborhood is captured dynamically (even if region is sparse)

- **Phase 1:** Use a multilevel graph partitioning algorithm on the graph to find a large number of clusters of well-connected vertices
  - Each cluster should contain mostly points from one “true” cluster, i.e., is a sub-cluster of a “real” cluster
Chameleon: Steps …

- **Phase 2**: Use Hierarchical Agglomerative Clustering to merge sub-clusters
  - Two clusters are combined if the *resulting cluster shares certain properties with the constituent clusters*

- Two key properties used to model cluster similarity:
  - **Relative Interconnectivity**: Absolute interconnectivity of two clusters normalized by the internal connectivity of the clusters
  - **Relative Closeness**: Absolute closeness of two clusters normalized by the internal closeness of the clusters
Cluster Merging: Limitations of Current Schemes

- Existing schemes are static in nature
  - MIN or CURE:
    - merge two clusters based on their *closeness* (or minimum distance)
  - GROUP-AVERAGE or ROCK:
    - merge two clusters based on their average *connectivity*
Limitations of Current Merging Schemes

Closeness schemes will merge (a) and (b)

Average connectivity schemes will merge (c) and (d)
Chameleon: Clustering Using Dynamic Modeling

- Adapt to the characteristics of the data set to find the natural clusters
- Use a dynamic model to measure the similarity between clusters
  - Main property is the relative closeness and relative inter-connectivity of the cluster
  - Two clusters are combined if the resulting cluster shares certain properties with the constituent clusters
  - The merging scheme preserves self-similarity
CHAMELEON (Clustering Complex Objects)
Chapter 7. Cluster Analysis

- Overview
- Partitioning methods
- Hierarchical methods
- Density-based methods
- Other methods
- Cluster evaluation
- Outlier analysis
- Summary
Density-Based Clustering Methods

- Clustering based on density
- Major features:
  - Clusters of arbitrary shape
  - Handle noise
  - Need density parameters as termination condition
- Several interesting studies:
  - **DBSCAN**: Ester, et al. (KDD’96)
  - **DENCLUE**: Hinneburg & D. Keim (KDD’98)
  - **CLIQUE**: Agrawal, et al. (SIGMOD’98) (more grid-based)
DBSCAN: Basic Concepts

- Density = number of points within a specified radius
- **core point**: has high density
- **border point**: has less density, but in the neighborhood of a core point
- **noise point**: not a core point or a border point.
DBScan: Definitions

- Two parameters:
  - \( Eps \): radius of the neighbourhood
  - \( MinPts \): Minimum number of points in an Eps-neighbourhood of that point

\[
N_{Eps}(p) = \{ q \text{ belongs to } D \mid \text{dist}(p,q) \leq Eps \}
\]

- core point: \( |N_{Eps}(q)| \geq MinPts \)

MinPts = 5
Eps = 1 cm
DBScan: Definitions

- **Directly density-reachable**: \( p \) belongs to \( N_{Eps}(q) \)

- **Density-reachable**: 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**: if there is a point \( o \) such that both, \( p \) and \( q \) are density-reachable from \( o \) w.r.t. \( Eps \) and \( MinPts \)

MinPts = 5
Eps = 1 cm
DBSCAN: Cluster Definition

- A *cluster* is defined as a maximal set of density-connected points

![Diagram of DBSCAN clusters]

- **Core**: Points that are density-connected to at least MinPts points
- **Border**: Points that are density-connected to a core point
- **Outlier**: Points that are not density-connected to any other points

**Parameters**
- **Eps**: 1cm
- **MinPts**: 5
DBSCAN: The Algorithm

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

- Basic idea:
  - For points in a cluster, their $k^{th}$ nearest neighbors are at roughly the same distance.
  - Noise points have the $k^{th}$ nearest neighbor at farther distance.
  - Plot sorted distance of every point to its $k^{th}$ nearest neighbor.
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.
Chapter 7. Cluster Analysis

- Overview
- Partitioning methods
- Hierarchical methods
- Density-based methods
- Other methods
  - Clustering by mixture models: mixed Gaussian model
  - Conceptual clustering: COBWEB
  - Neural network approach: SOM
- Cluster evaluation
- Outlier analysis
- Summary
Model-Based Clustering

- Attempt to optimize the fit between the given data and some mathematical model
- Typical methods
  - Statistical approach
    - EM (Expectation maximization)
  - Machine learning approach
    - COBWEB
  - Neural network approach
    - SOM (Self-Organizing Feature Map)
Clustering by Mixture Model

- Assume data are generated by a mixture of probabilistic model
  - Each cluster can be represented by a probabilistic model, like a Gaussian (continuous) or a Poisson (discrete) distribution.
Expectation Maximization (EM)

- Starts with an initial estimate of the parameters of the mixture model
- Iteratively refine the parameters using EM method
  - Expectation step: computes expectation of the likelihood of each data point $X_i$ belonging to cluster $C_i$
    \[
    P(X_i \in C_k) = p(C_k | X_i) = \frac{p(C_k)p(X_i | C_k)}{p(X_i)},
    \]
  - Maximization step: computes maximum likelihood estimates of the parameters
    \[
    m_k = \frac{1}{N} \sum_{i=1}^{N} \frac{X_i P(X_i \in C_k)}{\sum_{j} P(X_i \in C_j)}.
    \]
Conceptual Clustering

- Conceptual clustering
  - Generates a concept description for each concept (class)
  - Produces a hierarchical category or classification scheme
  - Related to decision tree learning and mixture model learning

- COBWEB (Fisher’87)
  - A popular and 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
COBWEB Classification Tree

- **animal**
  - $P(C0) = 1.0$
  - $P(scales|C0) = 0.25$
  - ...

- **fish**
  - $P(C1) = 0.25$
  - $P(scales|C1) = 1.0$
  - ...

- **amphibian**
  - $P(C2) = 0.25$
  - $P(moist|C2) = 1.0$
  - ...

- **mammal/bird**
  - $P(C3) = 0.5$
  - $P(hair|C3) = 0.5$
  - ...

  - **mammal**
    - $P(C4) = 0.5$
    - $P(hair|C4) = 1.0$
    - ...

  - **bird**
    - $P(C5) = 0.5$
    - $P(feathers|C5) = 1.0$
    - ...
COBWEB: Learning the Classification Tree

- Incrementally builds the classification tree
- Given a new object
  - Search for the best node at which to incorporate the object or add a new node for the object
  - Update the probabilistic description at each node
- Merging and splitting
- Use a heuristic measure - Category Utility - to guide construction of the tree
COBWEB: Comments

- Limitations

  - The assumption that the attributes are independent of each other is often too strong because correlation may exist

  - Not suitable for clustering large database – skewed tree and expensive probability distributions
Neural Network Approach

- Neural network approach for unsupervised learning
  - Involves a hierarchical architecture of several units (neurons)
- Two modes
  - Training: builds the network using input data
  - Mapping: automatically classifies a new input vector.
- Typical methods
  - SOM (Soft-Organizing feature Map)
  - Competitive learning
Self-Organizing Feature Map (SOM)

- SOMs, also called topological ordered maps, or Kohonen Self-Organizing Feature Map (KSOMs)
- Produce a low-dimensional (typically two) representation of the high-dimensional input data, called a **map**
  - The distance and proximity relationship (i.e., topology) are preserved as much as possible
- Visualization tool for high-dimensional data
- Clustering method for grouping similar objects together
- Competitive learning
  - believed to resemble processing that can occur in the brain
Learning SOM

- Network structure – a set of units associated with a weight vector
- Training – competitive learning
  - The unit whose weight vector is closest to the current object becomes the winning unit
  - The winner and its neighbors learn by having their weights adjusted
- Demo: http://www.sis.pitt.edu/~ssyn/som/demo.html
Web Document Clustering Using SOM

- The result of SOM clustering of 12088 Web articles
- The picture on the right: drilling down on the keyword “mining”
Chapter 7. Cluster Analysis

- Overview
- Partitioning methods
- Hierarchical methods
- Density-based methods
- Other methods
- Cluster evaluation
- Outlier analysis
Cluster Evaluation

- Determine clustering tendency of data, i.e. distinguish whether non-random structure exists
- Determine correct number of clusters
- Evaluate how well the cluster results fit the data without external information
- Evaluate how well the cluster results are compared to externally known results
- Compare different clustering algorithms/results
Clusters found in Random Data

Random Points

K-means

DBSCAN

Complete Link
Measures of Cluster Validity

- **Unsupervised (internal indices):** Used to measure the goodness of a clustering structure *without* respect to external information.
  - Sum of Squared Error (SSE)

- **Supervised (external indices):** Used to measure the extent to which cluster labels match externally supplied class labels.
  - Entropy

- **Relative:** Used to compare two different clustering results
  - Often an external or internal index is used for this function, e.g., SSE or entropy
Cluster Cohesion: how closely related are objects in a cluster

Cluster Separation: how distinct or well-separated a cluster is from other clusters

Example: Squared Error
- Cohesion: within cluster sum of squares (SSE)
  \[ WSS = \sum_i \sum_{x \in C_i} (x - m_i)^2 \]
- Separation: between cluster sum of squares
  \[ BSS = \sum_i \sum_j (m_i - m_j)^2 \]
Internal Measures: SSE

- SSE is good for comparing two clusterings
- Can also be used to estimate the number of clusters
Internal Measures: SSE

- Another example of a more complicated data set

SSE of clusters found using K-means
Statistics framework for cluster validity

- More “atypical” -> likely valid structure in the data
- Use values resulting from random data as baseline

Example

- Clustering: SSE = 0.005
- SSE of three clusters in 500 sets of random data points
External Measures

- Compare cluster results with “ground truth” or manually clustering
- Classification-oriented measures: entropy, purity, precision, recall, F-measures
- Similarity-oriented measures: Jaccard scores
External Measures: Classification-Oriented Measures

- Entropy: the degree to which each cluster consists of objects of a single class
- Precision: the fraction of a cluster that consists of objects of a specified class
- Recall: the extent to which a cluster contains all objects of a specified class
External Measure: Similarity-Oriented Measures

Given a reference clustering T and clustering S

- \( f_{00} \): number of pair of points belonging to different clusters in both T and S
- \( f_{01} \): number of pair of points belonging to different cluster in T but same cluster in S
- \( f_{10} \): number of pair of points belonging to same cluster in T but different cluster in S
- \( f_{11} \): number of pair of points belonging to same clusters in both T and S

\[
\text{Rand} = \frac{f_{00} + f_{11}}{f_{00} + f_{01} + f_{10} + f_{11}}
\]

\[
\text{Jaccard} = \frac{f_{11}}{f_{01} + f_{10} + f_{11}}
\]