Chapter 10. Cluster Analysis: Basic Concepts and Methods

- Cluster Analysis: Basic Concepts
- Partitioning Methods
- Hierarchical Methods
- Density-Based Methods
- Grid-Based Methods
- Evaluation of Clustering
What is Cluster Analysis?

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

- Data reduction
  - Summarization: Preprocessing for regression, PCA, classification, and association analysis
  - Compression: Image processing: vector quantization
- Hypothesis generation and testing
- Prediction based on groups
  - Cluster & find characteristics/patterns for each group
- Finding K-nearest Neighbors
  - Localizing search to one or a small number of clusters
- Outlier detection: Outliers are often viewed as those “far away” from any cluster
Clustering: Application Examples

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

- Feature selection
  - Select info concerning the task of interest
  - Minimal information redundancy
- Proximity measure
  - Similarity of two feature vectors
- Clustering criterion
  - Expressed via a cost function or some rules
- Clustering algorithms
  - Choice of algorithms
- Validation of the results
  - Validation test (also, clustering tendency test)
- Interpretation of the results
  - Integration with applications
Quality: What Is Good Clustering?

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

- **Dissimilarity/Similarity metric**
  - Similarity is expressed in terms of a distance function, typically metric: \( d(i, j) \)
  - The definitions of distance functions are usually rather different for interval-scaled, boolean, categorical, ordinal ratio, and vector variables
  - Weights should be associated with different variables based on applications and data semantics

- **Quality of clustering**:
  - There is usually a separate “quality” function that measures the “goodness” of a cluster.
  - It is hard to define “similar enough” or “good enough”
    - The answer is typically highly subjective
Considerations for Cluster Analysis

- Partitioning criteria
  - Single level vs. hierarchical partitioning (often, multi-level hierarchical partitioning is desirable)

- Separation of clusters
  - Exclusive (e.g., one customer belongs to only one region) vs. non-exclusive (e.g., one document may belong to more than one class)

- Similarity measure
  - Distance-based (e.g., Euclidian, road network, vector) vs. connectivity-based (e.g., density or contiguity)

- Clustering space
  - Full space (often when low dimensional) vs. subspaces (often in high-dimensional clustering)
Requirements and Challenges

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

- **Partitioning approach:**
  - Construct various partitions and then evaluate them by some criterion, e.g., minimizing the sum of square errors
  - Typical methods: k-means, k-medoids, CLARANS

- **Hierarchical approach:**
  - Create a hierarchical decomposition of the set of data (or objects) using some criterion
  - Typical methods: Diana, Agnes, BIRCH, CAMELEON

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

- **Grid-based approach:**
  - based on a multiple-level granularity structure
  - Typical methods: STING, WaveCluster, CLIQUE
Major Clustering Approaches (II)

- **Model-based:**
  - A model is hypothesized for each of the clusters and tries to find the best fit of that model to each other
  - Typical methods: EM, SOM, COBWEB

- **Frequent pattern-based:**
  - Based on the analysis of frequent patterns
  - Typical methods: p-Cluster

- **User-guided or constraint-based:**
  - Clustering by considering user-specified or application-specific constraints
  - Typical methods: COD (obstacles), constrained clustering

- **Link-based clustering:**
  - Objects are often linked together in various ways
  - Massive links can be used to cluster objects: SimRank, LinkClus
Chapter 10. Cluster Analysis: Basic Concepts and Methods

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

- **Partitioning method**: Partitioning a database \( D \) of \( n \) objects into a set of \( k \) clusters, such that the sum of squared distances is minimized (where \( c_i \) is the centroid or medoid of cluster \( C_i \))

\[
E = \sum_{i=1}^{k} \sum_{p \in C_i} (d(p, c_i))^2
\]

- Given \( k \), find a partition of \( k \) clusters that optimizes the chosen partitioning criterion
  - **Global optimal**: exhaustively enumerate all partitions
  - **Heuristic methods**: \textit{k-means} and \textit{k-medoids} algorithms
    - **k-means** (MacQueen’67, Lloyd’57/’82): Each cluster is represented by the center of the cluster
    - **k-medoids** or PAM (Partition around medoids) (Kaufman & Rousseeuw’87): Each cluster is represented by one of the objects in the cluster
Given $k$, the $k$-means algorithm is implemented in four steps:

- Partition objects into $k$ nonempty subsets
- Compute seed points as the centroids of the clusters of the current partitioning (the centroid is the center, i.e., *mean point*, of the cluster)
- Assign each object to the cluster with the nearest seed point
- Go back to Step 2, stop when the assignment does not change
An Example of \textit{K-Means} Clustering

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

The initial data set

K=2

Arbitrarily partition objects into \( k \) groups

Loop if needed

Update the cluster centroids

Reassign objects

Update the cluster centroids
Comments on the *K-Means Method*

- **Strength:** Efficient: $O(tkn)$, where $n$ is # objects, $k$ is # clusters, and $t$ is # iterations. Normally, $k, t << n$.
  - Comparing: PAM: $O(k(n-k)^2)$, CLARA: $O(ks^2 + k(n-k))$
- **Comment:** Often terminates at a *local optimal*
- **Weakness**
  - Applicable only to objects in a continuous n-dimensional space
    - Using the k-modes method for categorical data
    - In comparison, k-medoids can be applied to a wide range of data
  - Need to specify $k$, the *number* of clusters, in advance (there are ways to automatically determine the best $k$ (see Hastie et al., 2009)
  - Sensitive to noisy data and *outliers*
  - Not suitable to discover clusters with *non-convex shapes*
Variations of the *K-Means* Method

- Most of the variants of the *k-means* which differ in:
  - Selection of the initial *k* means
  - Dissimilarity calculations
  - Strategies to calculate cluster means
- Handling categorical data: *k-modes*
  - Replacing means of clusters with *modes*
  - Using new dissimilarity measures to deal with categorical objects
  - Using a *frequency*-based method to update modes of clusters
  - A mixture of categorical and numerical data: *k-prototype* method
What Is the Problem of the K-Means Method?

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

- **K-Medoids Clustering**: Find *representative* objects (medoids) in clusters
  - **PAM** (Partitioning Around Medoids, Kaufmann & Rousseeuw 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 (due to the computational complexity)
- Efficiency improvement on PAM
  - **CLARA** (Kaufmann & Rousseeuw, 1990): PAM on samples
  - **CLARANS** (Ng & Han, 1994): Randomized re-sampling
Chapter 10. Cluster Analysis: Basic Concepts and Methods

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

![Hierarchical Clustering Diagram]

- agglomerative (AGNES)
- divisive (DIANA)
AGNES (Agglomerative Nesting)

- Introduced in Kaufmann and Rousseeuw (1990)
- Implemented in statistical packages, e.g., Splus
- Use the **single-link** method and the dissimilarity matrix
- Merge nodes that have the least dissimilarity
- Go on in a non-descending fashion
- Eventually all nodes belong to the same cluster
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
Distance between Clusters

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

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

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

- **Centroid**: distance between the centroids of two clusters, i.e., \( \text{dist}(K_i, K_j) = \text{dist}(C_i, C_j) \)

- **Medoid**: distance between the medoids of two clusters, i.e., \( \text{dist}(K_i, K_j) = \text{dist}(M_i, M_j) \)
  - Medoid: a chosen, centrally located object in the cluster
Centroid, Radius and Diameter of a Cluster (for numerical data sets)

- Centroid: the “middle” of a cluster
- Radius: maximum distance between all the points and the centroid
  - Alternatively (text): square root of average distance from any point of the cluster to its centroid
- Diameter: maximum distance between any two points of the cluster
  - Alternatively (text): square root of average mean squared distance between all pairs of points in the cluster
- Note: radius and diameter of a cluster are not related directly as they’re in a circle. But have a tendency to be proportional
Extensions to Hierarchical Clustering

- Major weakness of agglomerative clustering methods
  - Can never undo what was done previously
  - Do not scale well: time complexity of at least $O(n^2)$, where $n$ is the number of total objects
- Integration of hierarchical & distance-based clustering
  - BIRCH (1996): uses CF-tree and incrementally adjusts the quality of sub-clusters
  - CHAMELEON (1999): hierarchical clustering using dynamic modeling
**BI RCH (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 (CF): \( CF = (N, LS, SS) \)

- \( N \): Number of data points

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

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

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

\( (3,4) \)

\( (2,6) \)

\( (4,5) \)

\( (4,7) \)

\( (3,8) \)
CF-Tree in BIRCH

- Clustering feature:
  - Summary of the statistics for a given subcluster: the 0-th, 1st, and 2nd moments of the subcluster from the statistical point of view
  - Registers crucial measurements for computing cluster and utilizes storage efficiently

- A CF tree is a height-balanced tree that stores the clustering features for a hierarchical clustering
  - A nonleaf node in a tree has descendants or “children”
  - The nonleaf nodes store sums of the CFs of their children

- A CF tree has two parameters
  - Branching factor: max # of children
  - Threshold: max diameter of sub-clusters stored at the leaf nodes
The CF Tree Structure

<table>
<thead>
<tr>
<th>CF1</th>
<th>CF2</th>
<th>CF3</th>
<th>......</th>
<th>CF6</th>
</tr>
</thead>
<tbody>
<tr>
<td>child1</td>
<td>child2</td>
<td>child3</td>
<td>......</td>
<td>child6</td>
</tr>
</tbody>
</table>

B = 7
L = 6

Root

Non-leaf node

Leaf node

Leaf node
The Birch Algorithm

- Cluster Diameter
  \[ \sqrt{\frac{1}{n(n-1)} \sum (x_i - x_j)^2} \]

- For each point in the input
  - Find closest leaf entry
  - Add point to leaf entry and update CF
  - If entry diameter > \text{max\_diameter}, then split leaf, and possibly parents

- Algorithm is \(O(n)\)

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

- CHAMELEON: G. Karypis, E. H. Han, and V. Kumar, 1999
- Measures the similarity based on a dynamic model
  - Two clusters are merged only if the interconnectivity and closeness (proximity) between two clusters are high relative to the internal interconnectivity of the clusters and closeness of items within the clusters
- Graph-based, and 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
KNN Graphs & Interconnectivity

- **k-nearest graphs from an original data in 2D:**
  
  (a) Original Data in 2D  
  (b) 1-nearest neighbor graph  
  (c) 2-nearest neighbor graph  
  (d) 3-nearest neighbor graph

- **$EC_{C_i, C_j}$:** The absolute inter-connectivity between $C_i$ and $C_j$: the sum of the weight of the edges that connect vertices in $C_i$ to vertices in $C_j$.

- **Internal inter-connectivity of a cluster $C_i$: the size of its min-cut bisector $EC_{C_i}$** (i.e., the weighted sum of edges that partition the graph into two roughly equal parts).

- **Relative Inter-connectivity (RI):**

  $$RI(C_i, C_j) = \frac{|EC_{C_i, C_j}|}{|EC_{C_i}| + |EC_{C_j}|}$$
Relative Closeness & Merge of Sub-Clusters

- **Relative closeness** between a pair of clusters $C_i$ and $C_j$:
  the absolute closeness between $C_i$ and $C_j$ normalized w.r.t. the internal closeness of the two clusters $C_i$ and $C_j$

\[
RC(C_i, C_j) = \frac{\overline{SEC}_{(C_i,C_j)}}{|C_i| + |C_j|} \overline{SEC}_{C_i} + \frac{|C_j|}{|C_i| + |C_j|} \overline{SEC}_{C_j}
\]

- $\overline{SEC}_{C_i}$ and $\overline{SEC}_{C_j}$ are the average weights of the edges that belong in the min-cut bisector of clusters $C_i$ and $C_j$, respectively, and $\overline{SEC}_{(C_i,C_j)}$ is the average weight of the edges that connect vertices in $C_i$ to vertices in $C_j$

- **Merge Sub-Clusters:**
  - Merges only those pairs of clusters whose RI and RC are both above some user-specified thresholds
  - Merge those maximizing the function that combines RI and RC
Overall Framework of CHAMELEON

Construct (K-NN)

Sparse Graph

Data Set

K-NN Graph

P and q are connected if q is among the top k closest neighbors of p

Partition the Graph

Final Clusters

Merge Partition

Relative interconnectivity:
connectivity of $c_1$ and $c_2$ over internal connectivity

Relative closeness:
closeness of $c_1$ and $c_2$ over internal closeness
CHAMELEON (Clustering Complex Objects)
Chapter 10. Cluster Analysis: Basic Concepts and Methods

- Cluster Analysis: Basic Concepts
- Partitioning Methods
- Hierarchical Methods
- Density-Based Methods
- Grid-Based Methods
- Evaluation of Clustering
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)
  - **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 belongs to D | dist(p,q) $\leq$ Eps\}

- **Directly density-reachable**: A point $p$ is directly density-reachable from a point $q$ w.r.t. Eps, MinPts if
  - $p$ belongs to $N_{Eps}(q)$
  - core point condition:
    $$|N_{Eps}(q)| \geq MinPts$$

MinPts = 5
Eps = 1 cm
Density-Reachable and Density-Connected

- **Density-reaching:**
  - A point \( p \) is density-reachable from a point \( q \) w.r.t. \( \epsilon, \text{MinPts} \) if there is a chain of points \( p_1, ..., p_n, p_1 = q, p_n = p \) 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. \( \epsilon, \text{MinPts} \) if there is a point \( o \) such that both, \( p \) and \( q \) are density-reachable from \( o \) w.r.t. \( \epsilon \) and \( \text{MinPts} \).
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.

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
- *If a spatial index is used, the computational complexity of DBSCAN is \( O(n \log n) \), where \( n \) is the number of database objects. Otherwise, the complexity is \( O(n^2) \)*
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.

DBSCAN online Demo:

- OPTICS: Ordering Points To Identify the Clustering Structure
  - Ankerst, Breunig, Kriegel, and Sander (SIGMOD’99)
  - Produces a special order of the database wrt its density-based clustering structure
  - This cluster-ordering contains info equiv to the density-based clusterings corresponding to a broad range of parameter settings
  - Good for both automatic and interactive cluster analysis, including finding intrinsic clustering structure
  - Can be represented graphically or using visualization techniques
OPTICS: Some Extension from DBSCAN

- Index-based: $k =$ # of dimensions, $N =$ # of points
  - Complexity: $O(N \cdot \log N)$

- Core Distance of an object $p$: the smallest value $\varepsilon$ such that the $\varepsilon$-neighborhood of $p$ has at least $\text{MinPts}$ objects

  Let $N_{\varepsilon}(p)$: $\varepsilon$-neighborhood of $p$, $\varepsilon$ is a distance value

  Core-distance$_{\varepsilon, \text{MinPts}}(p) =$ Undefined if $\text{card}(N_{\varepsilon}(p)) < \text{MinPts}$

  MinPts-distance$(p)$, otherwise

- Reachability Distance of object $p$ from core object $q$ is the min radius value that makes $p$ density-reachable from $q$

  Reachability-distance$_{\varepsilon, \text{MinPts}}(p, q) =$

  Undefined if $q$ is not a core object

  $\max(\text{core-distance}(q), \text{distance}(q, p))$, otherwise
Core Distance & Reachability Distance

Figure 10.16: OPTICS terminology. Based on [ABKS99].
Reachability-distance

undefined

Cluster-order of the objects

\( \varepsilon \)
Density-Based Clustering: OPTICS & Applications

demo: http://www.dbs.informatik.uni-muenchen.de/Forschung/KDD/Clustering/OPTICS/Demo
Chapter 10. Cluster Analysis: Basic Concepts and Methods

- Cluster Analysis: Basic Concepts
- Partitioning Methods
- Hierarchical Methods
- Density-Based Methods
- Grid-Based Methods
- Evaluation of Clustering
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)
  - **CLIQUE**: Agrawal, et al. (SIGMOD’98)
    - Both grid-based and subspace clustering
  - **WaveCluster** by Sheikholeslami, Chatterjee, and Zhang (VLDB’98)
    - A multi-resolution clustering approach using wavelet method
STING: A Statistical Information Grid Approach

- Wang, Yang and Muntz (VLDB’ 97)
- The spatial 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
STING Algorithm and Its Analysis

- 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
Chapter 10. Cluster Analysis: Basic Concepts and Methods

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

- 3 kinds of measures: External, internal and relative
- External: supervised, employ criteria not inherent to the dataset
  - Compare a clustering against prior or expert-specified knowledge (i.e., the ground truth) using certain clustering quality measure
- Internal: unsupervised, criteria derived from data itself
  - Evaluate the goodness of a clustering by considering how well the clusters are separated, and how compact the clusters are, e.g., Silhouette coefficient
- Relative: directly compare different clusterings, usually those obtained via different parameter settings for the same algorithm
Measuring Clustering Quality: External Methods

- Clustering quality measure: $Q(C, T)$, for a clustering $C$ given the ground truth $T$
- $Q$ is good if it satisfies the following 4 essential criteria
  - Cluster homogeneity: the purer, the better
  - Cluster completeness: should assign objects belong to the same category in the ground truth to the same cluster
  - Rag bag: putting a heterogeneous object into a pure cluster should be penalized more than putting it into a *rag bag* (i.e., “miscellaneous” or “other” category)
  - Small cluster preservation: splitting a small category into pieces is more harmful than splitting a large category into pieces
Chapter 11. Cluster Analysis: Advanced Methods

- Probability Model-Based Clustering
- Clustering High-Dimensional Data
- Clustering Graphs and Network Data
- Clustering with Constraints
Fuzzy Set and Fuzzy Cluster

- Clustering methods discussed so far
  - Every data object is assigned to exactly one cluster
- Some applications may need for fuzzy or soft cluster assignment
  - Ex. An e-game could belong to both entertainment and software
- Methods: fuzzy clusters and probabilistic model-based clusters
- Fuzzy cluster: A fuzzy set \( S : F_S : X \rightarrow [0, 1] \) (value between 0 and 1)
- Example: Popularity of cameras is defined as a fuzzy mapping

<table>
<thead>
<tr>
<th>Camera</th>
<th>Sales (units)</th>
</tr>
</thead>
<tbody>
<tr>
<td>A</td>
<td>50</td>
</tr>
<tr>
<td>B</td>
<td>1320</td>
</tr>
<tr>
<td>C</td>
<td>860</td>
</tr>
<tr>
<td>D</td>
<td>270</td>
</tr>
</tbody>
</table>

\[
\text{Pop}(o) = \begin{cases} 
1 & \text{if } 1,000 \text{ or more units of } o \text{ are sold} \\
\frac{i}{1000} & \text{if } i \ (i < 1000) \text{ units of } o \text{ are sold}
\end{cases}
\]

Then, \( A(0.05), B(1), C(0.86), D(0.27) \)
Fuzzy (Soft) Clustering

- Example: Let cluster features be
  - \( C_1 \) : “digital camera” and “lens”
  - \( C_2 \) : “computer”

- Fuzzy clustering
  - \( k \) fuzzy clusters \( C_1, \ldots, C_k \), represented as a partition matrix \( M = [w_{ij}] \)
  - P1: for each object \( o_i \) and cluster \( C_j \), \( 0 \leq w_{ij} \leq 1 \) (fuzzy set)
  - P2: for each object \( o_i \), \( \sum_{j=1}^{k} w_{ij} = 1 \), equal participation in the clustering
  - P3: for each cluster \( C_j \), \( 0 < \sum_{i=1}^{n} w_{ij} < n \) ensures there is no empty cluster

- Let \( c_1, \ldots, c_k \) as the center of the \( k \) clusters

- For an object \( o_i \), sum of the squared error (SSE), \( p \) is a parameter:

- For a cluster \( C_j \), SSE:
  \[
  \text{SSE}(C_j) = \sum_{i=1}^{n} w_{ij}^p \cdot \text{dist}(o_i, c_j)^2
  \]

- Measure how well a clustering fits the data:
  \[
  \text{SSE}(C) = \sum_{i=1}^{n} \sum_{j=1}^{k} w_{ij}^p \cdot \text{dist}(o_i, c_j)^2
  \]
Cluster analysis is to find hidden categories.

A hidden category (i.e., *probabilistic cluster*) is a distribution over the data space, which can be mathematically represented using a **probability density function** (or distribution function).

Ex. 2 categories for digital cameras sold
- consumer line vs. professional line
- density functions $f_1, f_2$ for $C_1, C_2$
- obtained by probabilistic clustering

A **mixture model** assumes that a set of observed objects is a mixture of instances from multiple probabilistic clusters, and conceptually each observed object is generated independently.

**Out task**: infer a set of $k$ probabilistic clusters that is mostly likely to generate $D$ using the above data generation process.
Model-Based Clustering

- A set $C$ of $k$ probabilistic clusters $C_1, \ldots, C_k$ with probability density functions $f_1, \ldots, f_k$, respectively, and their probabilities $\omega_1, \ldots, \omega_k$.

- Probability of an object $o$ generated by cluster $C_j$ is $P(o|C_j) = \omega_j f_j(o)$

- Probability of $o$ generated by the set of cluster $C$ is $P(o|C) = \sum_{j=1}^{k} \omega_j f_j(o)$

- Since objects are assumed to be generated independently, for a data set $D = \{o_1, \ldots, o_n\}$, we have,

$$P(D|C) = \prod_{i=1}^{n} P(o_i|C) = \prod_{i=1}^{n} \sum_{j=1}^{k} \omega_j f_j(o_i)$$

- Task: Find a set $C$ of $k$ probabilistic clusters s.t. $P(D|C)$ is maximized

- However, maximizing $P(D|C)$ is often intractable since the probability density function of a cluster can take an arbitrarily complicated form

- To make it computationally feasible (as a compromise), assume the probability density functions being some parameterized distributions
The EM (Expectation Maximization) Algorithm

- EM: A popular framework to approach maximum likelihood or maximum a posteriori estimates of parameters in statistical models.
- EM clustering: an extension to k-means. Starts with an initial estimates of the parameter vector, then iterates:
  - **E-step (expectation step)** assigns objects to clusters according to the current fuzzy clustering or parameters of probabilistic clusters
  - **M-step (maximization step)** finds the new clustering or parameters that maximize the sum of squared error (SSE) or the expected likelihood
- k-means also has two steps at each iteration:
  - **E-step**: Given the current cluster centers, each object is assigned to the cluster whose center is closest to the object: An object is *expected to belong to the closest cluster*
  - **M-step**: Given the cluster assignment, for each cluster, the algorithm *adjusts the center* so that *the sum of distance* from the objects assigned to this cluster and the new center is minimized
- Algorithm converges fast but may not be in global optima
EM Clustering

- Initially, randomly assign k cluster centers
- Iteratively refine the clusters based on two steps
  - **Expectation step:** assign each data point $X_i$ to cluster $C_k$ 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
    \[
    m_k = \frac{1}{N} \sum_{i=1}^{N} \frac{X_i P(X_i \in C_k)}{\sum_j P(X_i \in C_j)}.
    \]
EM Clustering Example

- Initially, let \( c_1 = a \) and \( c_2 = b \)
- 1st E-step: assign \( o \) to \( c_1 \), w. wt =
  - \( w_{c_1, o} = \frac{41}{45 + 41} = 0.48 \)
- 1st M-step: recalculate the centroids according to the partition matrix, minimizing the sum of squared error (SSE)
  \[
  c_1 = \left( \frac{1^2 + 0^2 + 0^2 \times 4 + 0.48^2 \times 9 + 0.42^2 \times 14 + 0.41^2 \times 18 + 0.47^2 \times 21}{1^2 + 0^2 + 0.48^2 + 0.42^2 + 0.41^2 + 0.47^2} \right) = \left( \frac{1^2 + 0^2 + 0.48^2 \times 10 + 0.42^2 \times 6 + 0.41^2 \times 8 + 0.47^2 \times 11 + 0.47^2 \times 7}{1^2 + 0^2 + 0.48^2 + 0.42^2 + 0.41^2 + 0.47^2} \right) = (8.47, 5.12)
  \]
- Iteratively calculate this until the cluster centers converge or the change is small enough
Univariate Gaussian Mixture Model

- $O = \{o_1, \ldots, o_n\}$ (n observed objects), $\Theta = \{\theta_1, \ldots, \theta_k\}$ (parameters of the k distributions), and $P_j(o_i|\theta_j)$ is the probability that $o_i$ is generated from the j-th distribution using parameter $\theta_j$, we have

$$P(o_i|\Theta) = \sum_{j=1}^{k} \omega_j P_j(o_i|\Theta_j)$$

$$P(O|\Theta) = \prod_{i=1}^{n} \sum_{j=1}^{k} \omega_j P_j(o_i|\Theta_j)$$

- Univariate Gaussian mixture model
  - Assume the probability density function of each cluster follows a 1-d Gaussian distribution. Suppose that there are k clusters.
  - The probability density function of each cluster are centered at $\mu_j$ with standard deviation $\sigma_j$, $\theta_j = (\mu_j, \sigma_j)$, we have

$$P(o_i|\Theta_j) = \frac{1}{\sqrt{2\pi}\sigma_j} e^{-\frac{(o_i-\mu_j)^2}{2\sigma^2}}$$

$$P(o_i|\Theta) = \sum_{j=1}^{k} \frac{1}{\sqrt{2\pi}\sigma_j} e^{-\frac{(o_i-\mu_j)^2}{2\sigma^2}}$$

$$P(O|\Theta) = \prod_{i=1}^{n} \sum_{j=1}^{k} \frac{1}{\sqrt{2\pi}\sigma_j} e^{-\frac{(o_i-\mu_j)^2}{2\sigma^2}}$$
Computing Mixture Models with EM

- Given \(n\) objects \(O = \{o_1, \ldots, o_n\}\), we want to mine a set of parameters \(\Theta = \{\theta_1, \ldots, \theta_k\}\) s.t., \(P(O|\Theta)\) is maximized, where \(\theta_j = (\mu_j, \sigma_j)\) are the mean and standard deviation of the \(j\)-th univariate Gaussian distribution.

- We initially assign random values to parameters \(\theta_j\), then iteratively conduct the E- and M-steps until converge or sufficiently small change.

- At the E-step, for each object \(o_i\), calculate the probability that \(o_i\) belongs to each distribution,

\[
P(\Theta_j|o_i, \Theta) = \frac{P(o_i|\Theta_j)}{\sum_{l=1}^{k} P(o_i|\Theta_l)}
\]

- At the M-step, adjust the parameters \(\theta_j = (\mu_j, \sigma_j)\) so that the expected likelihood \(P(O|\Theta)\) is maximized.

\[
\mu_j = \sum_{i=1}^{n} o_i \frac{P(\Theta_j|o_i, \Theta)}{\sum_{l=1}^{n} P(\Theta_j|o_l, \Theta)} = \frac{\sum_{i=1}^{n} o_i P(\Theta_j|o_i, \Theta)}{\sum_{i=1}^{n} P(\Theta_j|o_i, \Theta)}
\]

\[
\sigma_j = \sqrt{\frac{\sum_{i=1}^{n} P(\Theta_j|o_i, \Theta)(o_i - \mu_j)^2}{\sum_{i=1}^{n} P(\Theta_j|o_i, \Theta)}}
\]
Strengths and Weaknesses of Mixture Models

- Strength
  - Mixture models are more general than partitioning and fuzzy clustering
  - Clusters can be characterized by a small number of parameters
  - The results may satisfy the statistical assumptions of the generative models

- Weakness
  - Converge to local optimal (overcome: run multi-times w. random initialization)
  - Computationally expensive if the number of distributions is large, or the data set contains very few observed data points
  - Need large data sets
  - Hard to estimate the number of clusters
Chapter 11. Cluster Analysis: Advanced Methods

- Probability Model-Based Clustering
- Clustering High-Dimensional Data
- Clustering Graphs and Network Data
- Clustering with Constraints
Clustering High-Dimensional Data

- Clustering high-dimensional data (How high is high-D in clustering?)
  - 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
  - Subspace-clustering: Search for clusters existing in subspaces of the given high dimensional data space
    - CLIQUE, ProClus, and bi-clustering approaches
  - Dimensionality reduction approaches: Construct a much lower dimensional space and search for clusters there (may construct new dimensions by combining some dimensions in the original data)
    - Dimensionality reduction methods and spectral clustering
Traditional Distance Measures May Not Be Effective on High-D Data

- Traditional distance measure could be dominated by noises in many dimensions
- Ex. Which pairs of customers are more similar?

<table>
<thead>
<tr>
<th>Customer</th>
<th>( P_1 )</th>
<th>( P_2 )</th>
<th>( P_3 )</th>
<th>( P_4 )</th>
<th>( P_5 )</th>
<th>( P_6 )</th>
<th>( P_7 )</th>
<th>( P_8 )</th>
<th>( P_9 )</th>
<th>( P_{10} )</th>
</tr>
</thead>
<tbody>
<tr>
<td>Ada</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>Bob</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>1</td>
</tr>
<tr>
<td>Cathy</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>1</td>
</tr>
</tbody>
</table>

- By Euclidean distance, we get,
\[ \text{dist}(\text{Ada, Bob}) = \text{dist}(\text{Bob, Cathy}) = \text{dist}(\text{Ada, Cathy}) = \sqrt{2} \]
  - despite Ada and Cathy look more similar

- Clustering should not only consider dimensions but also attributes (features)
  - Feature transformation: effective if most dimensions are relevant (PCA & SVD useful when features are highly correlated/redundant)
  - Feature selection: useful to find a subspace where the data have nice clusters
The Curse of Dimensionality

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

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
Subspace Clustering Methods

- Subspace search methods: Search various subspaces to find clusters
  - Bottom-up approaches
  - Top-down approaches
- Correlation-based clustering methods
  - E.g., PCA based approaches
- Bi-clustering methods
  - Optimization-based methods
  - Enumeration methods
Subspace Clustering Method (I): Subspace Search Methods

- Search various subspaces to find clusters
- **Bottom-up approaches**
  - Start from low-D subspaces and search higher-D subspaces only when there may be clusters in such subspaces
  - Various pruning techniques to reduce the number of higher-D subspaces to be searched
  - Ex. CLIQUE (Agrawal et al. 1998)
- **Top-down approaches**
  - Start from full space and search smaller subspaces recursively
  - Effective only if the *locality assumption* holds: restricts that the subspace of a cluster can be determined by the local neighborhood
  - Ex. PROCLUS (Aggarwal et al. 1999): a *k*-medoid-like method
CLIQUE (SubSpace Clustering with Aprori Pruning)

- 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
\( \tau = 3 \)
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.
Bi-Cluster Methods

- Real-world data is noisy: Try to find approximate bi-clusters
- Methods: Optimization-based methods vs. enumeration methods
- Optimization-based methods
  - Try to find a submatrix at a time that achieves the best significance as a bi-cluster
  - Due to the cost in computation, greedy search is employed to find local optimal bi-clusters
  - Ex. $\delta$-Cluster Algorithm (Cheng and Church, ISMB’ 2000)
- Enumeration methods
  - Use a tolerance threshold to specify the degree of noise allowed in the bi-clusters to be mined
  - Then try to enumerate all submatrices as bi-clusters that satisfy the requirements
  - Ex. $\delta$-pCluster Algorithm (H. Wang et al.’ SIGMOD’ 2002, MaPle: Pei et al., ICDM’ 2003)
Bi-Clustering for Micro-Array Data Analysis

- Left figure: Micro-array “raw” data shows 3 genes and their values in a multi-D space: Difficult to find their patterns
- Right two: Some subsets of dimensions form nice shift and scaling patterns
- No globally defined similarity/distance measure
- Clusters may not be exclusive
  - An object can appear in multiple clusters
Chapter 11. Cluster Analysis: Advanced Methods

- Probability Model-Based Clustering
- Clustering High-Dimensional Data
- Clustering Graphs and Network Data
- Clustering with Constraints
Clustering Graphs and Network Data

Applications
- Bi-partite graphs, e.g., customers and products, authors and conferences
- Web search engines, e.g., click through graphs and Web graphs
- Social networks, friendship/coauthor graphs

Similarity measures
- Geodesic distances
- Distance based on random walk (SimRank)

Graph clustering methods
- Density-based clustering: SCAN (Xu et al., KDD’ 2007)
Similarity Measure (I): Geodesic Distance

- Geodesic distance \((A, B)\): length (i.e., \# of edges) of the shortest path between \(A\) and \(B\) (if not connected, defined as infinite)

- **Eccentricity** of \(v\), \(\text{eccen}(v)\): The largest geodesic distance between \(v\) and any other vertex \(u \in V - \{v\}\).
  - E.g., \(\text{eccen}(a) = \text{eccen}(b) = 2; \text{eccen}(c) = \text{eccen}(d) = \text{eccen}(e) = 3\)

- **Radius** of graph \(G\): The minimum eccentricity of all vertices, i.e., the distance between the “most central point” and the “farthest border”
  - \(r = \min_{v \in V} \text{eccen}(v)\)
  - E.g., radius \((g) = 2\)

- **Diameter** of graph \(G\): The maximum eccentricity of all vertices, i.e., the largest distance between any pair of vertices in \(G\)
  - \(d = \max_{v \in V} \text{eccen}(v)\)
  - E.g., diameter \((g) = 3\)

- **A peripheral vertex** is a vertex that achieves the diameter.
  - E.g., Vertices \(c\), \(d\), and \(e\) are peripheral vertices
SimRank: Similarity Based on Random Walk and Structural Context

- SimRank: structural-context similarity, i.e., based on the similarity of its neighbors
- In a directed graph $G = (V,E)$,
  - individual in-neighborhood of $v$: $I(v) = \{u \mid (u, v) \in E\}$
  - individual out-neighborhood of $v$: $O(v) = \{w \mid (v, w) \in E\}$
- Similarity in SimRank:
  - Initialization: $s_0(u, v) = \begin{cases} 0 & \text{if } u \neq v \\ 1 & \text{if } u = v \end{cases}$
  - Then we can compute $s_{i+1}$ from $s_i$ based on the definition
- Similarity based on random walk: in a strongly connected component
  - Expected distance: $d(u, v) = \sum_{t:u \leadsto v} P[t]l(t)$
  - Expected meeting distance: $m(u, v) = \sum_{t:u,v \leadsto (x,x)} P[t]l(t)$
  - Expected meeting probability: $p(u, v) = \sum_{t:u,v \leadsto (x,x)} P[t]C^d(t)$

$P[t]$ is the probability of the tour.
Graph Clustering: Sparsest Cut

- $G = (V,E)$. The cut set of a cut is the set of edges \( \{(u, v) \in E \mid u \in S, v \in T\} \) and $S$ and $T$ are in two partitions

- Size of the cut: # of edges in the cut set

- Min-cut (e.g., $C_1$) is not a good partition

- A better measure: **Sparsity**: $\Phi = \frac{\text{the size of the cut}}{\min\{|S|, |T|\}}$

- A cut is **sparsest** if its sparsity is not greater than that of any other cut

- Ex. Cut $C_2 = (\{a, b, c, d, e, f, l\}, \{g, h, i, j, k\})$ is the sparsest cut

- For $k$ clusters, the **modularity** of a clustering assesses the quality of the clustering:
  \[
  Q = \sum_{i=1}^{k} \left( \frac{l_i}{|E|} - \left( \frac{d_i}{2|E|} \right)^2 \right)
  \]
  $l_i$: # edges between vertices in the $i$-th cluster $d_i$: the sum of the degrees of the vertices in the $i$-th cluster

- The modularity of a clustering of a graph is the difference between the fraction of all edges that fall into individual clusters and the fraction that would do so if the graph vertices were randomly connected

- The optimal clustering of graphs maximizes the modularity
Graph Clustering: Challenges of Finding Good Cuts

- High computational cost
  - Many graph cut problems are computationally expensive
  - The sparsest cut problem is NP-hard
  - Need to tradeoff between efficiency/scalability and quality
- Sophisticated graphs
  - May involve weights and/or cycles.
- High dimensionality
  - A graph can have many vertices. In a similarity matrix, a vertex is represented as a vector (a row in the matrix) whose dimensionality is the number of vertices in the graph
- Sparsity
  - A large graph is often sparse, meaning each vertex on average connects to only a small number of other vertices
  - A similarity matrix from a large sparse graph can also be sparse
Two Approaches for Graph Clustering

- Two approaches for clustering graph data
  - Use generic clustering methods for high-dimensional data
  - Designed specifically for clustering graphs
- Using clustering methods for high-dimensional data
  - Extract a similarity matrix from a graph using a similarity measure
  - A generic clustering method can then be applied on the similarity matrix to discover clusters
  - Ex. Spectral clustering: approximate optimal graph cut solutions
- Methods specific to graphs
  - Search the graph to find well-connected components as clusters
  - Ex. SCAN (Structural Clustering Algorithm for Networks)
Chapter 11. Cluster Analysis: Advanced Methods

- Probability Model-Based Clustering
- Clustering High-Dimensional Data
- Clustering Graphs and Network Data
- Clustering with Constraints
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
Categorization of Constraints

- Constraints on instances: specifies how a pair or a set of instances should be grouped in the cluster analysis
  - Must-link vs. cannot link constraints
    - must-link(x, y): x and y should be grouped into one cluster
  - Constraints can be defined using variables, e.g.,
    - cannot-link(x, y) if dist(x, y) > d
- Constraints on clusters: specifies a requirement on the clusters
  - E.g., specify the min # of objects in a cluster, the max diameter of a cluster, the shape of a cluster (e.g., a convex), # of clusters (e.g., k)
- Constraints on similarity measurements: specifies a requirement that the similarity calculation must respect
  - E.g., driving on roads, obstacles (e.g., rivers, lakes)
- Issues: Hard vs. soft constraints; conflicting or redundant constraints
Constraint-Based Clustering Methods (I): Handling Hard Constraints

- Handling hard constraints: Strictly respect the constraints in cluster assignments
- Example: The COP-k-means algorithm
  - Generate super-instances for must-link constraints
    - Compute the transitive closure of the must-link constraints
    - To represent such a subset, replace all those objects in the subset by the mean.
    - The super-instance also carries a weight, which is the number of objects it represents
  - Conduct modified k-means clustering to respect cannot-link constraints
    - Modify the center-assignment process in k-means to a nearest feasible center assignment
    - An object is assigned to the nearest center so that the assignment respects all cannot-link constraints
Constraint-Based Clustering Methods (II): Handling Soft Constraints

- Treated as an optimization problem: When a clustering violates a soft constraint, a penalty is imposed on the clustering.
- Overall objective: Optimizing the clustering quality, and minimizing the constraint violation penalty.
- Ex. CVQE (Constrained Vector Quantization Error) algorithm: Conduct k-means clustering while enforcing constraint violation penalties.
- Objective function: Sum of distance used in k-means, adjusted by the constraint violation penalties.
  - Penalty of a *must-link* violation:
    - If objects x and y must-be-linked but they are assigned to two different centers, $c_1$ and $c_2$, $\text{dist}(c_1, c_2)$ is added to the objective function as the penalty.
  - Penalty of a cannot-link violation:
    - If objects x and y cannot-be-linked but they are assigned to a common center $c$, $\text{dist}(c, c')$, between $c$ and $c'$ is added to the objective function as the penalty, where $c'$ is the closest cluster to $c$ that can accommodate x or y.
An Example: Clustering With Obstacle Objects

Not Taking obstacles into account  Taking obstacles into account
What Are Outliers? (ch11)

- **Outlier**: A data object that deviates significantly from the normal objects as if it were generated by a different mechanism
  - Unusual credit card purchase, Michael Jordan in sports ...
- Outliers are different from the noise data
  - Noise is random error or variance in a measured variable
  - Noise should be removed before outlier detection
- Outliers are interesting: It violates the mechanism that generates the normal data
- Outlier detection vs. *novelty detection*: early stage, outlier; but later merged into the model
- Applications:
  - Credit card fraud detection
  - Telecom fraud detection
  - Customer segmentation
  - Medical analysis