Cluster Analysis

Chengyu Sun
California State University, Los Angeles

Clustering

- Group similar objects together
- Applications
  - Identify users who share similar interests
  - Automatically generate concept hierarchies
  - Reduce algorithmic complexity
  - ...

Types of Clusters

- Well separated
- Prototype based
- Contiguity based
- Density based
- Conceptual clusters

Well-separated Clusters

- Each point is closer to all of the points in its cluster than to any point in another cluster

Prototype-based Clusters

Contiguity-based Clusters

- A cluster can be considered as a connected component in a graph
Density-based Clusters

A cluster is a dense region of objects surrounded by a region of low density.

Conceptual Clusters

A cluster is a set of objects that share some property.

Types of Clustering

- Partitional vs. Hierarchical
- Exclusive vs. Overlapping vs. Fuzzy
- Complete vs. Partial

Similarity Measure

<table>
<thead>
<tr>
<th>TID</th>
<th>Home Owner</th>
<th>Marital Status</th>
<th>Annual Income</th>
<th>Defaulted</th>
<th>Borrower</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>Yes</td>
<td>Single</td>
<td>125K</td>
<td>No</td>
<td></td>
</tr>
<tr>
<td>2</td>
<td>No</td>
<td>Married</td>
<td>100K</td>
<td>No</td>
<td></td>
</tr>
<tr>
<td>3</td>
<td>No</td>
<td>Single</td>
<td>70K</td>
<td>No</td>
<td></td>
</tr>
</tbody>
</table>

Is #1 more similar to #2 or #3?

Interval-Scaled Attributes

- Continuous-valued data measured with a linear scale (vs. exponential or logarithmic scale)

Distance Measures

- \( \mathbf{X} = (x_1, x_2, \ldots, x_n) \) and \( \mathbf{Y} = (y_1, y_2, \ldots, y_n) \)
- E.g. (1, 2) and (3, 5)

Euclidean Distance:

\[
d_{\text{euclid}}(X, Y) = \sqrt{\sum_{i=1}^{n} (x_i - y_i)^2}
\]

Manhattan Distance:

\[
d_{\text{manhattan}}(X, Y) = \sum_{i=1}^{n} |x_i - y_i|
\]
Minkowski Distance

\[ \text{dist}(X,Y) = \sqrt[p]{\sum_{i=1}^{n} |x_i - y_i|^p} \]

- \( p=1 \) (Manhattan Distance)
  - a.k.a. \( L_1 \) norm or \( L_1 \) distance
- \( p=2 \) (Euclidean Distance)
  - a.k.a. \( L_2 \) norm or \( L_2 \) distance

Requirements of Distance Functions

- \( \text{dist}(X,Y) \geq 0 \)
- \( \text{dist}(X,X) = 0 \)
- \( \text{dist}(X,Y) = \text{dist}(Y,X) \)
- \( \text{dist}(X,Y) \leq \text{dist}(X,Z) + \text{dist}(Z,Y) \) - Triangular Inequality

Problem of Units

- \((10\text{m},2\text{km})\) and \((5\text{m},2.1\text{km})\)?
- \((10\text{m},200\text{lb})\) and \((5\text{m}, 210\text{lb})\)?

Standardize Interval-Scaled Attributes

- Given attribute \( A \) with values \( a_1, a_2, \ldots, a_n \)
  - Mean:
    \[ \bar{a} = \frac{1}{n} \sum_{i=1}^{n} a_i \]
  - Mean absolute deviation:
    \[ s = \frac{1}{n} \sum_{i=1}^{n} |a_i - \bar{a}| \]
  - Standardized measurement (z-score):
    \[ z_i = \frac{a_i - \bar{a}}{s} \]

Binary Attributes

- Symmetric
  - E.g. gender
- Asymmetric
  - E.g. HIV test result

Contingency Table for Binary Attributes

<table>
<thead>
<tr>
<th>Record X</th>
<th>Record Y</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>0</td>
</tr>
<tr>
<td>1</td>
<td>q</td>
</tr>
<tr>
<td>q</td>
<td>r</td>
</tr>
<tr>
<td>0</td>
<td>s</td>
</tr>
<tr>
<td>s</td>
<td>t</td>
</tr>
</tbody>
</table>

Example

- \( X=(1,1,0,1,0,0,0) \), \( Y=(0,1,0,1,0,1,0) \)
## Distance Measure for Symmetric Binary Attributes

- **Similarity:** \( \text{sim}(X, Y) = \frac{q + r}{q + r + s + t} \)
- **Dissimilarity:** \( \text{dissim}(X, Y) = \frac{r + s}{q + r + s + t} \)
- **Distance:** ??

## Distance Measure for Asymmetric Binary Attributes

- **Similarity (Jaccard Coefficient):** \( \text{sim}(X, Y) = \frac{q}{q + r + s} \)
- **Dissimilarity:** \( \text{dissim}(X, Y) = \frac{r + s}{q + r + s} \)
- **Distance:** ??

### Binary Attribute Example

<table>
<thead>
<tr>
<th>TID</th>
<th>Gender</th>
<th>Fever</th>
<th>Cough</th>
<th>Test-1</th>
<th>Test-2</th>
<th>Test-3</th>
<th>Test-4</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>M</td>
<td>Y</td>
<td>N</td>
<td>P</td>
<td>N</td>
<td>N</td>
<td>N</td>
</tr>
<tr>
<td>2</td>
<td>F</td>
<td>Y</td>
<td>N</td>
<td>P</td>
<td>P</td>
<td>N</td>
<td>N</td>
</tr>
<tr>
<td>3</td>
<td>M</td>
<td>Y</td>
<td>Y</td>
<td>N</td>
<td>N</td>
<td>N</td>
<td>N</td>
</tr>
</tbody>
</table>

\( \text{dist}(1,2) \text{?? dist}(2,3) \text{?? dist}(3,1) ?? \)

### Categorical Attributes

- **Example**
  - Marital status: single, married, divorced
  - \( \text{dist}(X, Y) = \frac{(p-m)}{p} \)
    - \( m \): number of attribute matches
    - \( p \): total number of attributes
- **Or, encode each state with a binary attribute**

### Ordinal Attributes

- **Example**
  - Grade: F, D, C, B, A
  - Given an attribute with \( M \) possible values \( \{1, 2, \ldots, M\} \), map value \( a \) to the range of \( [0.0, 1.0] \)
    \[ z = \frac{a - 1}{M - 1} \]

- **Example**
  - Grade: F, D, C, B, A
  - Given an attribute with \( M \) possible values \( \{1, 2, \ldots, M\} \), map value \( a \) to the range of \( [0.0, 1.0] \)
    \[ z = \frac{a - 1}{M - 1} \]

### Records with Mixed Types of Attributes...

\[ d_{\text{dist}}(X, Y) = \frac{\sum_{i=1}^{n} \delta_i \times \text{dist}(x_i, y_i)}{\sum_{i=1}^{n} \delta_i} \]

\( \delta_i \) is the weight of the \( i \)-th attribute \( a_i \)'s contribution toward the overall distance
- 0 if \( x_i \) or \( y_i \) is missing, or \( a_i \) is asymmetric binary and \( x_i = y_i = 0 \)
- 1 otherwise
... Records with Mixed Types of Attributes

- \(dist(x_i, y_i)\)
  - Interval-based: \(|x_i - y_i| / (\max(a_i) - \min(a_i))\)
  - Binary or categorical: 0 if \(x_i = y_i\); 1 otherwise
  - Ordinal: treat as interval-based using \(z_i\)

Other Distance Measures

- Cosine distance
- Tanimoto distance
- ...
- Weighted distance

K-Means

- Input: dataset \(D\) and number of clusters \(k\)
- Algorithm
  1. Randomly choose \(k\) objects as cluster centers
  2. Assign each object to the closest cluster center
  3. Update each cluster center
  4. Repeat 2 until there is no reassignment occurs

K-Means Example

Key Issues in K-Means

- Distance measure?
  - Euclidean, Manhattan, Cosine ...
- Cluster center?
  - Mean, median

Need for Objective Function

- The best clustering is the one that minimize the "errors" defined by an objective function
Notations

<table>
<thead>
<tr>
<th>Symbol</th>
<th>Definition</th>
</tr>
</thead>
<tbody>
<tr>
<td>D</td>
<td>Dataset</td>
</tr>
<tr>
<td>k</td>
<td>The number of clusters</td>
</tr>
<tr>
<td>C_i</td>
<td>i-th cluster</td>
</tr>
<tr>
<td>c_i</td>
<td>The center of the i-th cluster</td>
</tr>
<tr>
<td>x</td>
<td>An object</td>
</tr>
</tbody>
</table>

Objective Functions

Sum of the Squared Error (SSE):

\[ SSE = \sum_{i=1}^{k} \sum_{x \in C_i} dist_{i}(x, c_i)^2 \]

Sum of the Absolute Error (SAE):

\[ SAE = \sum_{i=1}^{k} \sum_{x \in C_i} dist_{i}(x, c_i) \]

Distances, Centroids, and Objective Functions

<table>
<thead>
<tr>
<th>Distance Function</th>
<th>Centroid</th>
<th>Objective Function</th>
</tr>
</thead>
<tbody>
<tr>
<td>Manhattan (L_1)</td>
<td>Median</td>
<td>Sum of L_1 distance</td>
</tr>
<tr>
<td>Squared Euclidean (L_2)</td>
<td>Mean</td>
<td>Sum of squared L_2 distance</td>
</tr>
<tr>
<td>Cosine</td>
<td>Mean</td>
<td>Sum of cosine distance</td>
</tr>
<tr>
<td>Bregman Divergence</td>
<td>Mean</td>
<td>Sum of Bregman divergence</td>
</tr>
</tbody>
</table>

Minimize an Object Function

Example:
- One dimensional data
- One cluster
- SSE

\[ SSE(c) = \sum_{x} (c - x)^2 \quad \Rightarrow \quad \frac{\partial}{\partial c} SSE(c) = 0 \]

Another K-Means Example ...

... Another K-Means Example
Dealing with the Problem of Initial Centroid Selection

- Perform several runs of K-Means and select the clustering with the smallest SSE
  - Not as effective as you would think, especially with large \( k \) (why??)
- Use a hierarchical clustering algorithm on a sample to get \( k \) initial clusters
- Select centroid one by one, and each one is the farthest away from previously selected ones

Postprocessing

- Escape local SSE minima by performing alternate clustering *splitting* and *merging*

Postprocessing – Splitting

- Splitting the cluster with the largest SSE on the attribute with the largest variance
- Introduce another centroid
  - The point that is farthest from current centroids
  - Randomly chosen

Postprocessing – Merging

- Disperse a cluster and reassign its objects
- Merge two clusters that are close to each other

Bisecting K-Means

1. Initial a list of clusters with one cluster containing all the objects
2. Choose one cluster from the list
3. Split the cluster into two using basic K-Means, and add them back to the list
4. Repeat Step 2 until \( k \) clusters are reached
5. Perform one more basic K-Means using the centroids of the \( k \) clusters as initial centroids

About Bisecting K-Means

- Step 2
  - Choose the largest cluster
  - Choose the cluster with the largest SSE
- Step 3
  - Perform basic K-Means several times and choose the clustering with the smallest SSE
  - Less susceptible to initialization problems
  - Why??
Handling Empty Clusters

- Choose a replacement centroid
  - The point that’s farthest away from any current centroid
  - A point from the cluster with the highest SSE

Limitations of K-Means

- Problem with clusters of different sizes
- Densities
- Non-globular shapes
- Problem with outliers
- Requires the notion of centroid

Limitations of K-Means – Differing Sizes

Limitations of K-Means – Differing Densities

Limitations of K-Means – Non-globular Shapes

K-Medoids

- Instead of using mean/centroid, use medoid, i.e. representative object
- Objective function: sum of the distances of the objects to their medoid
- Differs from K-Means in how the medoids are updated
PAM (Partition Around Medoids)

1. Randomly choose $k$ objects as initial medoids
2. For each non-medoid object $x$:
   For each medoid $c_i$:
   calculate the reduction of the total distance if $c_i$ is replaced by $x$
3. Replace the $c_i$ with $x$ that results in maximum total distance reduction
4. Repeat Step 2 until the total distance cannot be reduced
5. Assign each object to its closest mediod

K-Means vs. K-Medoids

- Requires the notion of mean/centroid
- More susceptible to outliers
- $O(kn)$ per iteration

- Works for all distance measures
- Less susceptible to outliers
- ?? per iteration

Hierarchical Clustering

- Agglomerative
  - Start with each object as a cluster
  - Recursively pick two clusters to merge
- Divisive
  - Start with all objects as a single cluster
  - Recursively pick one cluster to split

Agglomerative Hierarchical Clustering

1. Compute a distance matrix
2. Merge the two closest clusters
3. Update the distance matrix
4. Repeat Step 2 until only one cluster remains

Distance Between Clusters

- Min distance
  - Distance between two closest objects
  - Min $<$ threshold: Single-link Clustering
- Max distance
  - Distance between two farthest objects
  - Max $<$ threshold: Complete-link Clustering
- Average distance
  - Average of all pairs of objects from the two clusters
Centroid-based Distance

- Mean distance
- Increased SSE (Ward’s Method)

Min Distance Clustering Example ...

... Min Distance Clustering Example

Max Distance Clustering Example

Average Distance Clustering Example

Ward’s Clustering Example
About Hierarchical Clustering
- Produces a hierarchy of clusters
- Lack of a global objective function
- Merging decisions are final
- Expensive
- Often used with other clustering algorithms

BIRCH
- Balanced Iterative Reducing and Clustering using Hierarchies

Clustering Feature (CF)
- CF = <N, LS, SS>
  - N: number of objects
  - LS (Linear Sum): \( LS = \sum_{i=1}^{N} x_i \)
  - SS (Square Sum): \( SS = \sum_{i=1}^{N} x_i^2 = \sum_{i=1}^{N} x_i \cdot x_i \)

CF Tree

CF Tree Construction – Input
- Dataset
- Threshold Condition
  - Diameter D of a cluster < d

Centroid: \( x_c = \frac{\sum_{i=1}^{N} x_i}{N} \)
Radius: \( R = \sqrt{\frac{\sum_{i=1}^{N} (x_i - x_c)^2}{N}} \)
Diameter: \( D = \sqrt{\frac{\sum_{i=1}^{N} (x_i - x_c)^2}{N(N-1)}} \)

CF Tree Construction – Insert
- Insert an object into its closest cluster in a leaf node
  - The object is inserted if the resulting cluster does not violate the threshold condition
  - Otherwise the object is inserted as a cluster of by itself
- When a node is full, split it and rebalance the tree (similar to B+ Tree Insertion)
CF Tree Howto’s
- Find closest cluster
  - Object-to-cluster distance
- Insert object into a cluster
  - Update CF
  - Check threshold condition
  - Calculate diameter
- Split node and rebalance tree
  - Merge clusters that are close to one another
    - Cluster-to-cluster distance; calculate CF of the merged cluster

Diameter Calculation
- Calculate diameter using CF
  \[ D = \sqrt{\frac{2N \times SS - 2LS^2}{N(N-1)}} \]

Diameter Calculation Example
- A cluster with three 1-D objects
  - \( x_1 \) = \( x_1 \)
  - \( x_2 \) = \( x_2 \)
  - \( x_3 \) = \( x_3 \)

Cluster-to-Cluster Distances
- Cluster-to-cluster distances that can be calculated using CF
  - \( D_0 \): centroid Euclidean distance
  - \( D_1 \): centroid Manhattan distance
  - \( D_2 \): average inter-cluster distance
  - \( D_3 \): average intra-cluster distance
  - \( D_4 \): variance increase distance

About BIRCH
- Single scan of data
  - CF tree is kept in memory
  - Size of the CF tree can be adjusted using the threshold value
- Cluster the leaf node clusters
  - More natural clusters
  - Sparse clusters detected as outliers
- Require the notion of centroid

DBSCAN
- Density-Based Spatial Clustering of Applications with Noise
- A density-based clustering algorithm
Classification of Points

- Given a radius $\varepsilon$ and the minimum number of points $\text{MinPts}$ within a radius of $\varepsilon$ ($\varepsilon$-neighborhood)
  - Core point
    - Has more points in its $\varepsilon$-neighborhood than $\text{MinPts}$
  - Border points
    - Within the $\varepsilon$-neighborhood of a core point
  - Noise points

Point Examples

- Figure 8.30. Density
- Figure 8.31. Core, border, and noise points.

The DBSCAN Algorithm

- Label all points as core, border, or noise
- Remove all noise points
- Put an edge between all core points that are within $\varepsilon$ of each other
- Make each connected group of core points a cluster
- Assign border points to one of the clusters of their associated core points

DBSCAN Example

Select DBSCAN Parameters

- $k$-dist: distance to the $k$th nearest neighbor
- $k=4$ is usually reasonable for most 2-D datasets

More DBSCAN Examples

Figure 8.32. $k$-dist for sample data.
About DBSCAN

- Handle clusters with arbitrary shapes and sizes
- Limitations
  - Clusters with varying densities
  - High dimensional data
- Could be expensive because of nearest neighbor computer
  - Use a spatial index structure like R tree or k-d tree

Other Clustering Algorithms

- More efficient
  - Speed
  - Scalability
- High dimensional data
- Constraint-based

Cluster Evaluation

- a.k.a. Cluster Validation
- Unsupervised
  - Using no external information other than the data itself
- Supervised
  - With external information such as given class labels

Reasons Not To Evaluate

- Clustering is often used as part of exploratory data analysis
- Clustering is often used as part of other algorithms
- Clustering algorithms, in some sense, define their own types of clusters

Reasons To Evaluate ...

... Reasons To Evaluate
Quality (Validity) of Clusters

- Cohesion
  - Compactness of a cluster
- Separation

Validity of Graph-based Clusters

\[
\text{cohesion}(C_i) = \sum_{y \in C_i} \text{dist}(x, y)
\]
\[
\text{separation}(C_i, C_j) = \sum_{y \in C_i \setminus \{i\}} \text{dist}(y, x)
\]

Validity of Prototype-based Clusters

\[
\text{cohesion}(C_i) = \sum_{x \in C_i} \text{dist}(x, c_i)
\]
\[
\text{separation}(C_i, C_j) = \text{dist}(c_i, c_j)
\]
\[
\text{separation}(C_i) = \text{dist}(c_i, c)
\]

Validity of A Clustering

\[
\text{validity}(C) = \sum_{i \in I} w_i \times \text{validity}(C_i)
\]

Cluster Weights

<table>
<thead>
<tr>
<th>Validity Measures</th>
<th>Weights</th>
</tr>
</thead>
<tbody>
<tr>
<td>(\sum_{x \in C_i} \text{dist}(x, y))</td>
<td>(1/</td>
</tr>
<tr>
<td>(\sum_{x \in C_i} \text{dist}(x, c_i))</td>
<td>(1)</td>
</tr>
<tr>
<td>(\text{dist}(c_i, c))</td>
<td>(</td>
</tr>
</tbody>
</table>

Silhouette Coefficient

- For the \(i\)th object in a cluster
  - \(a_i\): average distance to all other objects in the cluster
  - \(b_i\): max of the average distance to the objects in a cluster that does not contain this object

\[
s_i = (b_i - a_i) / \max(a_i, b_i)
\]
About Silhouette Coefficient

◆ Range of \( s_i \)??
◆ What is a "good" value of \( s_i \)??
◆ Quality of a cluster: average \( s_i \)
◆ Quality of a clustering: average \( s_i \)

Similarity Matrix

<table>
<thead>
<tr>
<th>1</th>
<th>2</th>
<th>3</th>
<th>4</th>
<th>5</th>
<th>6</th>
<th>7</th>
<th>8</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>2</td>
<td>3</td>
<td>4</td>
<td>5</td>
<td>6</td>
<td>7</td>
<td>8</td>
</tr>
</tbody>
</table>

◆ 8 objects, 3 clusters: \{3,7,8\}, \{1,2,5\}, \{4,6\}

Similarity Matrix Example

Clustering Tendency

◆ Do clusters exist in the first place?
◆ Determine clustering tendency
  - Cluster first, then evaluate the quality of the clustering
    - Need to try several different types of clustering algorithms
  - Statistical tests for spatial randomness

... Determine The Correct Number of Clusters

Determine The Correct Number of Clusters...
Hopkins Statistic

- Generate \( p \) random points in the data space
  - \( u_i \): distance of a randomly generated point to its nearest neighbor in the original dataset
- Select \( p \) random points from the original dataset
  - \( w_i \): distance of a randomly selected point to its nearest neighbor in the original dataset

**Interpretation of Hopkins Statistic??**

\[
H = \frac{\sum_{i=1}^{p} w_i}{\sum_{i=1}^{p} u_i + \sum_{i=1}^{p} w_i}
\]

Supervised Measures of Cluster Validity

- **Classification-oriented measures**
  - Evaluate the extent to which a cluster contains the objects of a single class
- **Similarity-oriented measures**
  - Evaluate the extent to which two objects of the same class (or cluster) belong to the same cluster (or class)

Classification-oriented Measures

- Entropy
- Purity
- Precision, recall, F-measure

Similarity-oriented Measures

<table>
<thead>
<tr>
<th></th>
<th>Same class</th>
<th>Different class</th>
</tr>
</thead>
<tbody>
<tr>
<td>Same class</td>
<td>( f_{11} )</td>
<td>( f_{10} )</td>
</tr>
<tr>
<td>Different class</td>
<td>( f_{01} )</td>
<td>( f_{00} )</td>
</tr>
</tbody>
</table>

... Similarity-oriented Measures

- Rand Statistic:
  \[
  R = \frac{f_{11} + f_{10}}{f_{11} + f_{10} + f_{01}}
  \]

- Jaccard Coefficient:
  \[
  J = \frac{f_{11}}{f_{11} + f_{01} + f_{10}}
  \]

Summary

- Types of clusters
- Types of clustering
- Similarity measures
- Clustering algorithms
  - Partitional: K-Means, K-Mediods
  - Hierarchical: Agglomerative, BIRCH
  - Density-based: DBSCAN
- Clustering evaluation
  - Unsupervised and supervised