Clustering

COMP 135 Intro to Machine Learning
Liping Liu

slides modified from Roni Khardon’s with permission

Unsupervised Learning

Clustering is often a form of data exploration allowing us to identify groupings that are otherwise not apparent

<table>
<thead>
<tr>
<th>Domain/problem</th>
<th>Groups</th>
</tr>
</thead>
<tbody>
<tr>
<td>Gene-array data</td>
<td>Similar activity patterns</td>
</tr>
<tr>
<td>Text</td>
<td>Word Classes</td>
</tr>
<tr>
<td>Customer Activity</td>
<td>Customer “types” (phone; web; movies; etc)</td>
</tr>
</tbody>
</table>

Clustering

• Here we assume data is in $\mathbb{R}^d$
  - $X = \{x_i; i = 1, ..., N\}, x_i \in \mathbb{R}^d$
  - No labels any more
• Task: partition data $X$ into groups, or clusters, in some sensible way,
  - each cluster containing instances similar to each other
  - “Similar”: short distance
  - No order of clusters
(Some methods can work with distance directly without assuming $\mathbb{R}^d$ space)

K-Means Clustering

• Formal definition
  - Partition the dataset into clusters $C_1, ..., C_K$
  - Cluster means
    \[
    \mu_k = \frac{1}{|C_k|} \sum_{i \in C_k} x_i
    \]
  - Objective: minimize within-cluster similarity
    \[
    L = \frac{1}{|C_k|} \sum_{k} \sum_{i \in C_k} ||x_i - \mu_k||^2
    \]

Toy examples

Linearly separable
Not linearly separable
Even more complex problem, not linearly separable

k-Means Clustering

• Pick k cluster centers (talk later)
• Repeat:
  - Associate examples with centers pick nearest center
  - Re-calculate means as average of examples in cluster
• Until convergence
k-Means Clustering

Properties
- Always converge (why?)
- Converge fast in practice (though slow in theory / worst cases)
- Always form linearly separable clusters

Result sensitive to initialization
- Initial cluster centers should be far apart and representative

Methods:
- Repeat k-Means with random initializations
- k-Means++: iteratively choose cluster centers far from other cluster centers

Calculation of mean is sensitive to outliers
- k-Medoids Clustering

Sensitivity to feature scaling and transformations

Visualization from Carla Brodley's slides
How to Choose k?

- **Solution 1:**
  - Run algorithm with \( k = 2, 3, \ldots \)
  - Evaluate criterion (e.g. CS) for each run
  - Hope to see big drop in criterion until we get “the right \( k \)” and moderate drop after that

- **Solution 2:** BIC criterion – add penalty for number of clusters
  \[
  \text{BIC}(k) = L(k) + k \cdot \log(D)
  \]
  - Increase \( k \):
    - K-means objective \( L(k) \) goes down, penalty \( k \cdot \log(D) \) goes up
    - For some \( k \), the total starts going up

In which case \( L \) goes to zero?

Clustering Evaluation

- How can we evaluate how good our clustering is?
  - Evaluation by our criterion
  - Evaluation by expert
  - Evaluation by using clustering result for other task.

- Comparing different clustering results (and/or comparing to labels)
  - Evaluation by NMI - defined later on slides

Comparing Clustering Results

- Sometimes it is useful to check if two clustering results are close or not
- For purpose of evaluating new clustering algorithm: we can compare its results to labels on a labeled dataset

- Normalized Mutual Information (NMI)

Mutual Information

- MI for clustering: information about the second clustering result, given the first cluster result

Comparing Clustering Results

- Probability of cluster assignments
  Let \( C_1, \ldots, C_l \) be one clustering result, \( C'_1, \ldots, C'_l \) be another clustering result, then
  \[
  p(z^1 = k_1, z^2 = k_2) = \frac{n_{k_1 k_2}}{N}
  \]
  Similarly calculate \( p(z^1 = k_2, z^2 = k_1) \) and \( p(z^2 = k_1 | z^1 = k_2) \)

<table>
<thead>
<tr>
<th>( C' )</th>
<th>( C )</th>
<th>( \ldots )</th>
<th>( C'_l )</th>
</tr>
</thead>
<tbody>
<tr>
<td>( C_1 )</td>
<td>( n_{11} )</td>
<td>( n_{12} )</td>
<td>( \ldots )</td>
</tr>
<tr>
<td>( C_2 )</td>
<td>( n_{21} )</td>
<td>( n_{22} )</td>
<td>( \ldots )</td>
</tr>
<tr>
<td>( \ldots )</td>
<td>( \ldots )</td>
<td>( \ldots )</td>
<td>( \ldots )</td>
</tr>
<tr>
<td>( C_l )</td>
<td>( n_{l1} )</td>
<td>( n_{l2} )</td>
<td>( \ldots )</td>
</tr>
</tbody>
</table>

Table 1: the contingency table, \( n_{i,j} = |C_i \cap C'_j| \)
Comparing Clustering Results

• Mutual information of cluster assignments

\[ I[z^1, z^2] = \sum_{k_1, k_2} \sum_{i, j} p(z^1 = k_1, z^2 = k_2) \log \frac{p(z^1 = k_1, z^2 = k_2)}{p(z^1 = k_1)p(z^2 = k_2)} \]

\[ = H[z^1] - H[z^1|z^2] \]

NMI

• Mutual Information is sensitive to the number of clusters
  - more clusters will artificially have higher mutual information

• Normalized Mutual Information corrects for that.
  - normalize MI by the average entropy

\[ \text{NMI} = \frac{I[z^1, z^2]}{H[z^1] + H[z^2]} \]

Soft k-Means Clustering

• Pick k cluster centers
• Repeat:
  - Associate examples with centers
    \[ p_{i,j} \sim \text{similarity b/w example i and center j} \]
  - Re-calculate means as weighted average of examples in cluster
• Until convergence

Alternatives: k-Medoids Clustering

• Pick k cluster medoids
• Repeat:
  - Associate examples with medoids
    pick nearest medoid
  - Re-calculate medoid
    the example in cluster that has the smallest mean distance to other points in the cluster
• Until convergence

Alternatives: Spectral Clustering

• Can use any distance function
• Or a weighted adjacency matrix of graph induced by examples
• To produce "Laplacian" similarity matrix
• Performs standard clustering on eigen-decomposition of that matrix
• [details beyond scope of course]

Clustering

• Data Exploration
• Evaluation by ...
• Several possible criteria

• Model selection (pick k)
• Comparing different partitions