

Unsupervised Learning

Supervised vs Unsupervised learning

- So far we have assumed that the training samples used to design the classifier were labeled by their class membership (supervised learning)
- We assume now that all one has is a collection of samples without being told their categories (unsupervised learning)

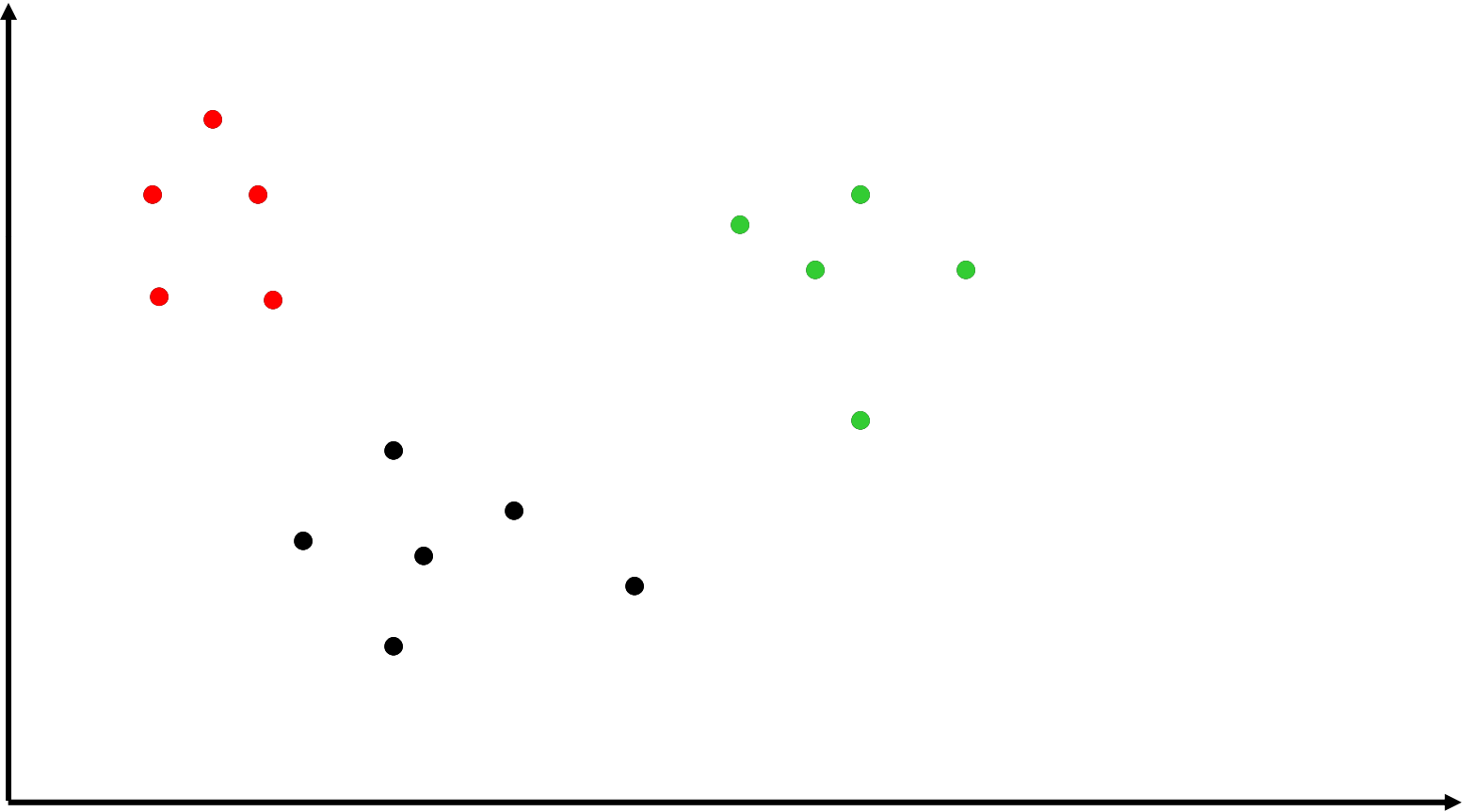
What is Unsupervised Learning

- Learning without Class Labels (or correct outputs)
 - Clustering
 - Partition data into clusters
 - Dimensionality Reduction
 - Discover low-dimensional representation of data

Clustering

- **Goal:** Partition a collection of unlabeled examples into disjoint subsets of *clusters*, such that:
 - Examples within a cluster are very similar to one another
 - Examples in different clusters are very different from one another

Clustering Example



Example Applications

- Information retrieval – cluster retrieved documents to present more organized and understandable results
- Explorative data analysis – find underlying group of examples that correspond to important concepts
 - Discovery of gene functional groups
- Automated creation of taxonomies
 - e.g., Yahoo-style
- Compression

Similarity/Distance measures

- An appropriate similarity/distance measure is fundamental to many clustering algorithms
- What is similarity?



Similarity is hard to define, but...
"We know it when we see it"

The real meaning of similarity is a philosophical question. We will take a more pragmatic approach.

Slide by E. Keogh

Similarity (Distance) Measures Based on Features

- Euclidean Distance (L_2 norm):

$$L_2(\mathbf{x}, \mathbf{x}') = \sum_{i=1}^d (x_i - x'_i)^2$$

- L_1 norm: $L_1(\mathbf{x}, \mathbf{x}') = \sum_{i=1}^d |x_i - x'_i|$

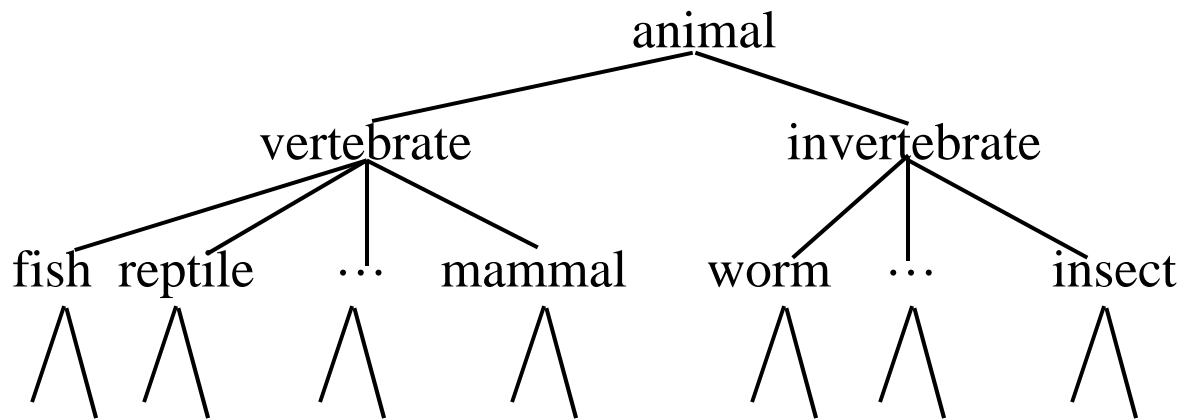
- Cosine similarity:

$$\cos(\mathbf{x}, \mathbf{x}') = \frac{\langle \mathbf{x}, \mathbf{x}' \rangle}{|\mathbf{x}| \cdot |\mathbf{x}'|}$$

- Kernels

Flat and Hierarchical Clustering

- A flat (non-hierarchical) clustering produces a single partition of the unlabeled data
- Hierarchical clustering builds a tree-based hierarchical taxonomy (*dendrogram*)



- Recursive application of flat clustering can produce a hierarchical clustering.

Hierarchical Agglomerative Clustering (HAC)

- Assumes a *similarity function* for determining the similarity of two instances.
- Starts with each instance in a separate cluster and then repeatedly joins the two clusters that are most similar until there is only one cluster.
- The history of merging forms a binary tree or hierarchy.

HAC Algorithm

Start with all instances in their own cluster.

Until there is only one cluster:

Among the current clusters, determine the two clusters, c_i and c_j , that are most similar.

Replace c_i and c_j with a single cluster $c_i \cup c_j$

Question: our assumed similarity function computes similarity between instances, how to compute the similarity between clusters?

Cluster Similarity

- Assume a similarity function that determines the similarity of two instances: $sim(x,y)$.
 - e.g., Cosine similarity of document vectors.
- Compute similarity of two clusters each containing multiple instances:
 - **Single Link**: Similarity of two most similar members
 - **Complete Link**: Similarity of two least similar members
 - **Average Link**: Average similarity between members

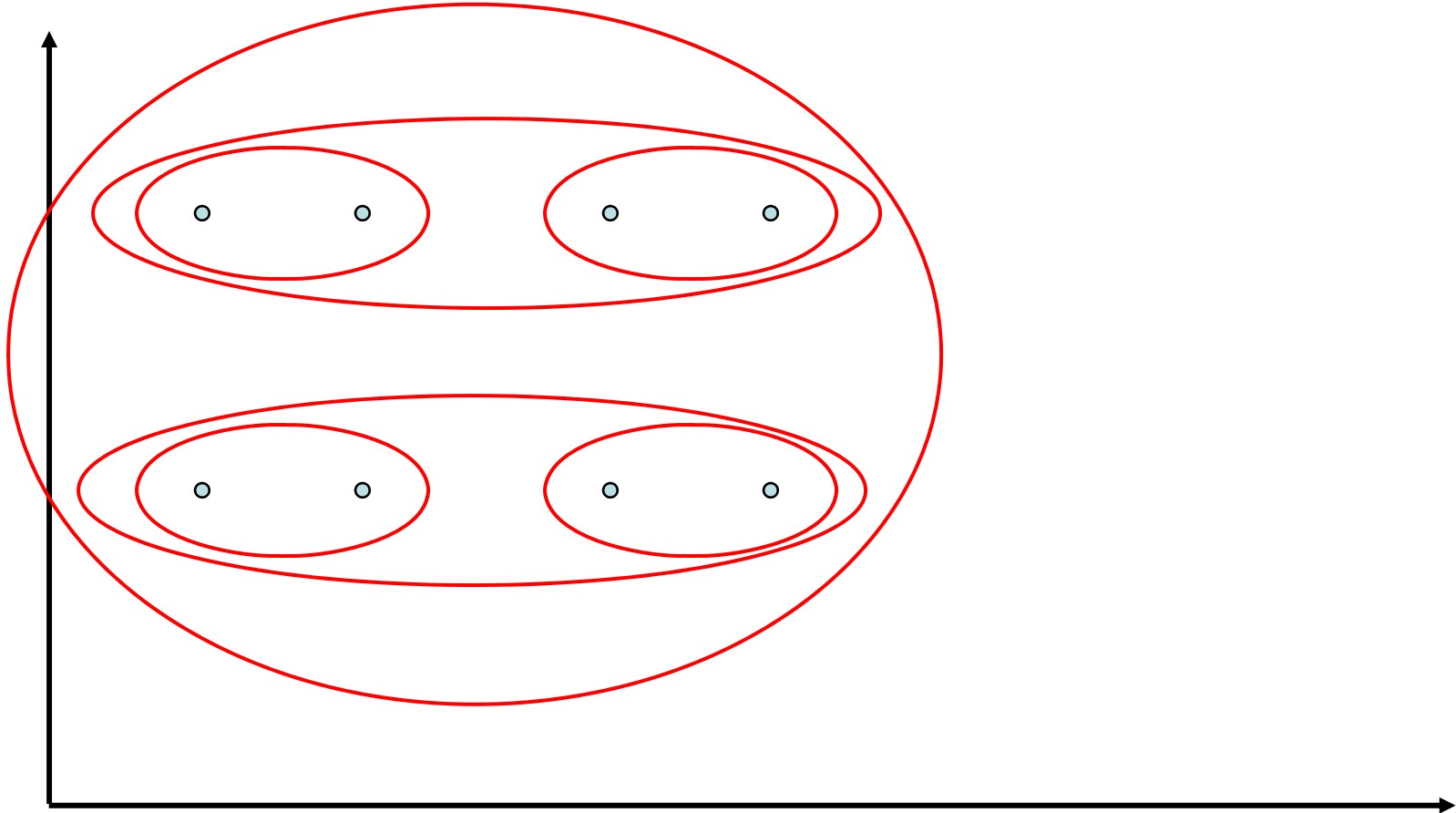
Single Link Agglomerative Clustering

- Use maximum similarity of pairs:

$$sim(c_i, c_j) = \max_{\mathbf{x} \in c_i, \mathbf{x}' \in c_j} sim(\mathbf{x}, \mathbf{x}')$$

- Can result in “straggly” (long and thin) clusters due to *chaining effect*.
 - Appropriate in some domains, such as clustering islands.

Single Link Example



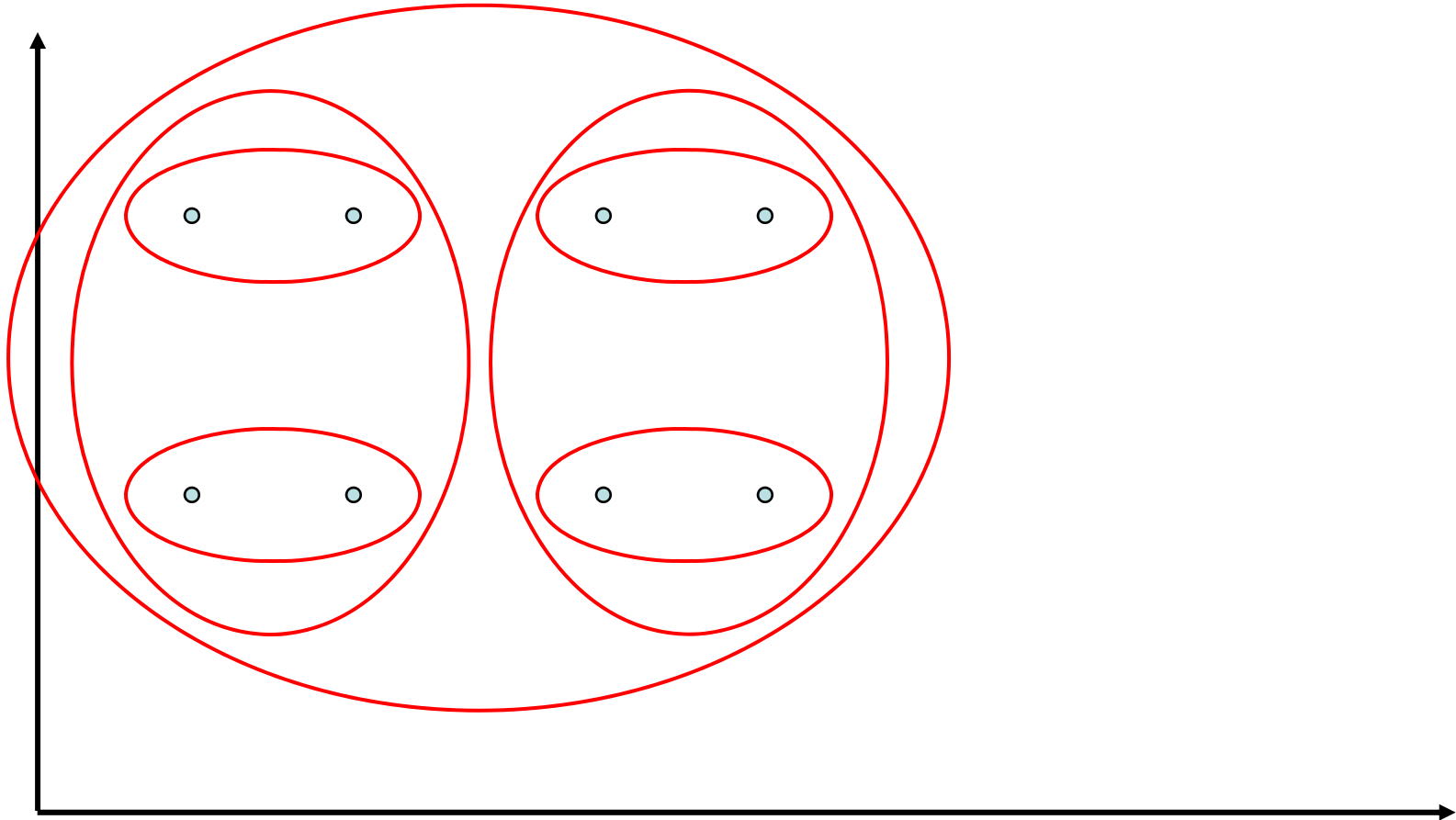
Complete Link Agglomerative Clustering

- Use minimum similarity of pairs:

$$sim(c_i, c_j) = \min_{\mathbf{x} \in c_i, \mathbf{x}' \in c_j} sim(\mathbf{x}, \mathbf{y})$$

- Makes more “tight,” spherical clusters that are typically preferable.

Complete Link Example



Computational Complexity

- In total we have $n-1$ merging iterations.
- In the 1st iteration, all HAC methods need to compute similarity of all pairs of n individual instances - $O(n^2)$.
- In the subsequent iterations, for the i -th iteration, it must compute the distance between the new cluster and all other existing $(n-i)$ clusters.
$$\sum_{i=2}^{n-1} (n - i) = O(n^2)$$
- In order to maintain an overall $O(n^2)$ performance, computing similarity to each other cluster must be done in constant time – can we do it for single/complete link?

Computing Similarity in Constant Time



- After merging c_i and c_j , the similarity of the resulting cluster to any other cluster, c_k , can be computed by:

- Single Link:

$$\text{sim}((c_i \cup c_j), c_k) = \max(\text{sim}(c_i, c_k), \text{sim}(c_j, c_k))$$

- Complete Link:

$$\text{sim}((c_i \cup c_j), c_k) = \min(\text{sim}(c_i, c_k), \text{sim}(c_j, c_k))$$

Average Link Agglomerative Clustering

- Use average similarity across all pairs within the merged cluster to measure the similarity of two clusters.

$$sim(c_i, c_j) = \frac{1}{|c_i \cup c_j|(|c_i \cup c_j| - 1)} \sum_{\mathbf{x} \in (c_i \cup c_j)} \sum_{\mathbf{x}' \in (c_i \cup c_j): \mathbf{x}' \neq \mathbf{x}} sim(\mathbf{x}, \mathbf{x}')$$

- Compromise between single and complete link.
- Two options
 - Averaged across all ordered pairs in the merged cluster
 - Averaged across all pairs between the original two clusters
- No clear difference in performance

Computing Average Link Similarity

- Assume cosine similarity and normalized vectors with unit length.
- Always maintain sum of vectors in each cluster.

$$\mathbf{s}_j = \sum_{\mathbf{x} \in c_j} \mathbf{x}$$

- Compute similarity of clusters in constant time:

$$\text{sim}(c_i, c_j) = \frac{\langle (\mathbf{s}_i + \mathbf{s}_j) \cdot (\mathbf{s}_i + \mathbf{s}_j) \rangle - (|c_i| + |c_j|)}{(|c_i| + |c_j|)(|c_i| + |c_j| - 1)}$$

HAC – Centroid/medoid Based

Start with all instances in their own cluster.

Until there is only one cluster:

Among the current clusters, determine the two clusters, c_i and c_j , that are most similar.

Replace c_i and c_j with a single cluster $c_i \cup c_j$

Choose the pair of clusters whose centroids / medoids are most similar

- Difference between centroid and medoid:
 - mean and median
 - The centroid may not be a true item, the medoid is a true item that's closest to centroid
 - Consider document clustering: centroid likely be a dense vector

Visualization of the hierarchy: Dendrogram

- Height of the merging line = the similarity between the merged clusters
- Often used to identify the right number of clusters
 - A horizontal cutting line will create a unique clustering
 - Moving the cutting line from top to bottom, we get more and more clusters
 - Large gaps between the merging lines indicate a good cutting point

