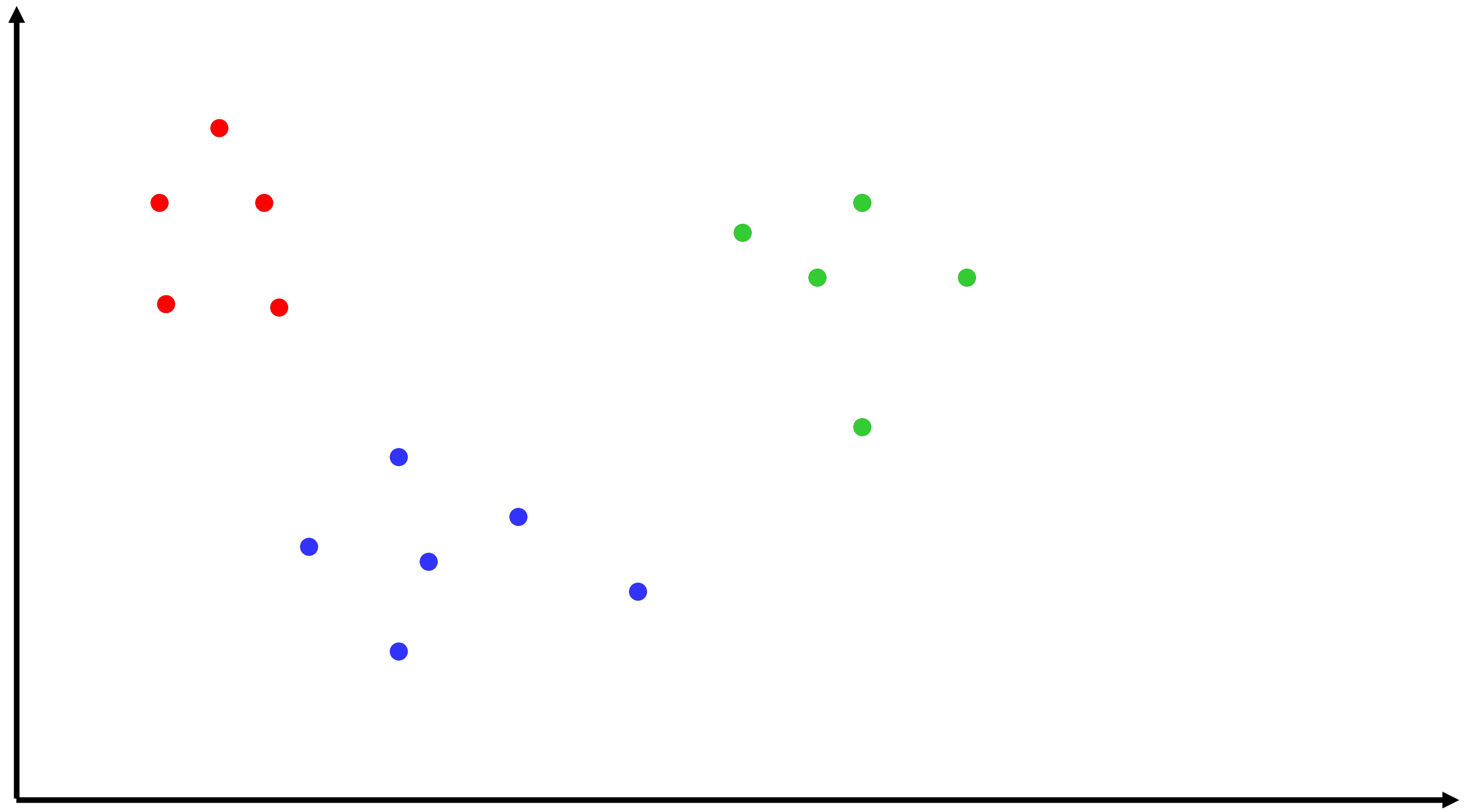
Unsupervised learning. Clustering

Clustering

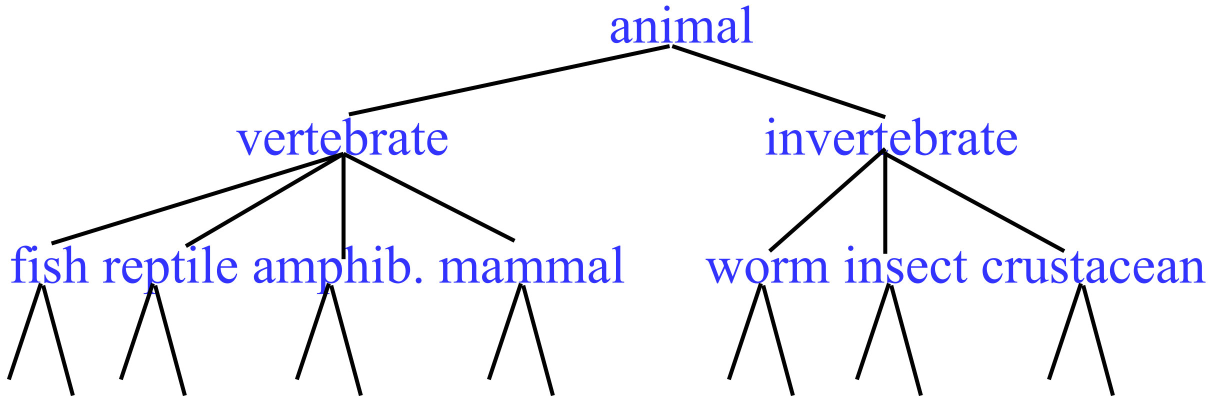
- Partition unlabeled examples into disjoint subsets of *clusters*, such that:
  - Examples within a cluster are very similar
  - Examples in different clusters are very different
- Discover new categories in an *unsupervised* manner (no sample category labels provided).

Clustering Example



Hierarchical Clustering

- Build a tree-based hierarchical taxonomy (*dendrogram*) from a set of unlabeled examples.



- Recursive application of a standard clustering algorithm can produce a hierarchical clustering.

Aglommerative vs. Divisive Clustering

- *Aglommerative* (*bottom-up*) methods start with each example in its own cluster and iteratively combine them to form larger and larger clusters.
- *Divisive* (*partitional, top-down*) separate all examples immediately into clusters.

Direct Clustering Method

- *Direct clustering* methods require a specification of the number of clusters, *k*, desired.
- A *clustering evaluation function* assigns a real-value quality measure to a clustering.
- The number of clusters can be determined automatically by explicitly generating clusterings for multiple values of *k* and choosing the best result according to a clustering evaluation function.

Hierarchical Agglomerative Clustering (HAC)

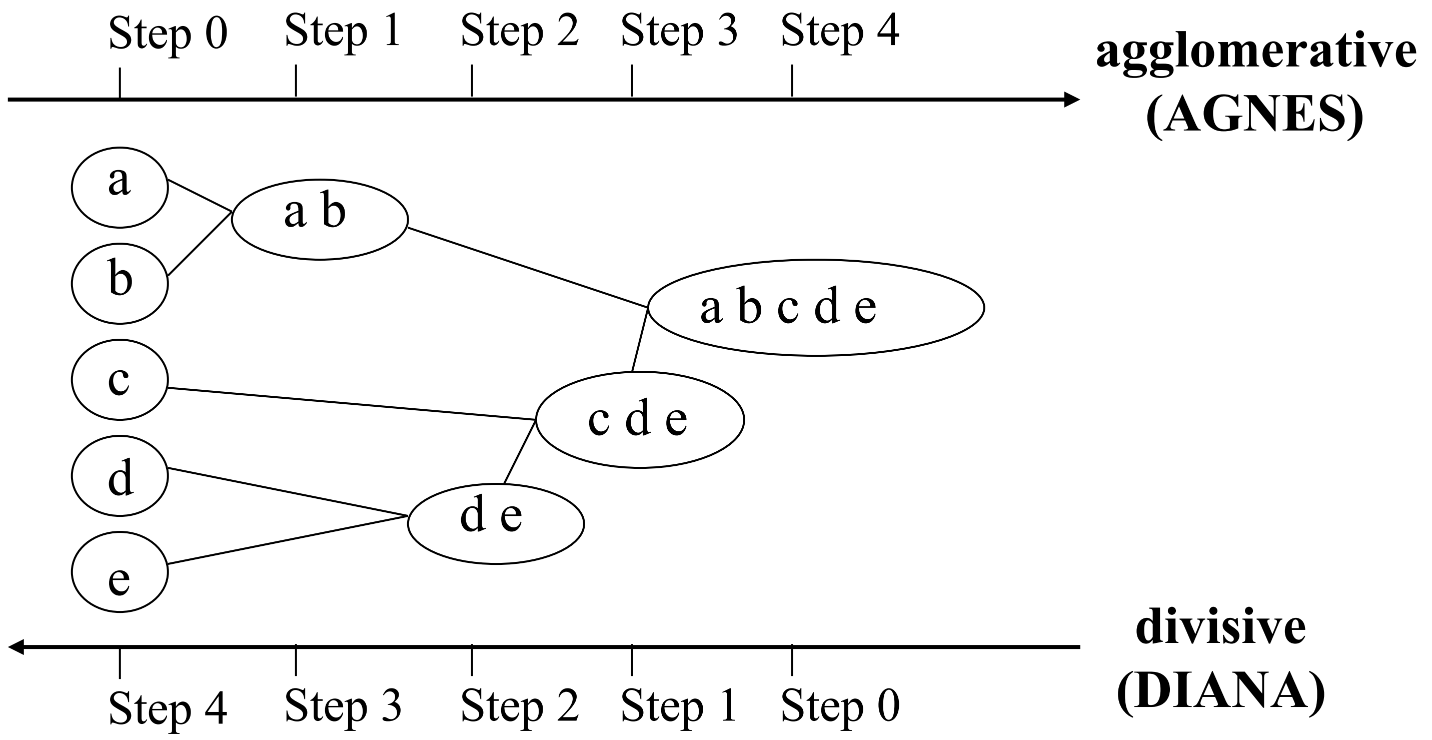
- Assumes a *similarity function* for determining the similarity of two instances.
- Starts with all instances in a separate cluster and then repeatedly joins the two clusters that are most similar until there is only one cluster.
- The history of merging forms a binary tree or hierarchy.

HAC Algorithm

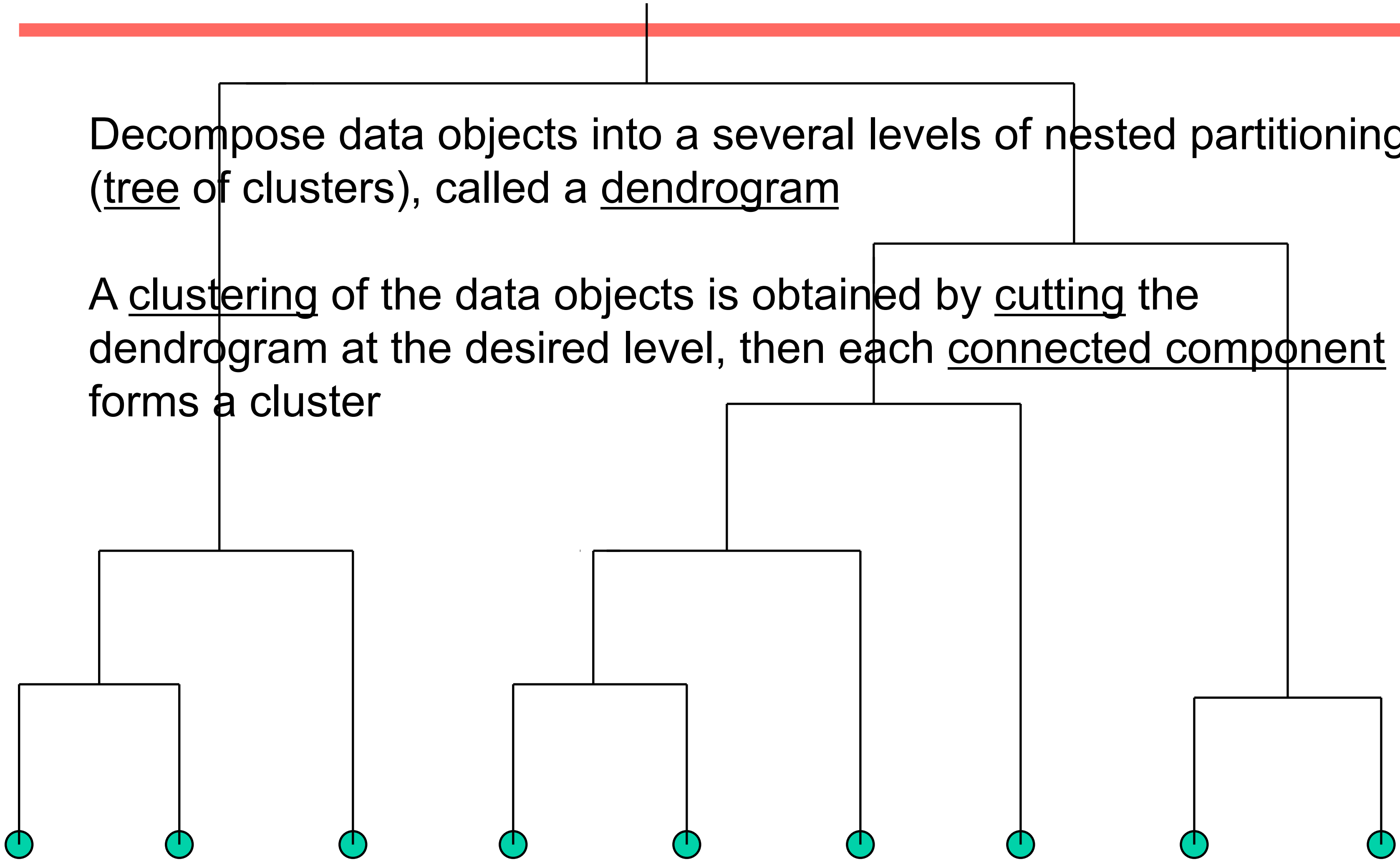
```
Start with all instances in their own cluster.
Until there is only one cluster:
    Among the current clusters, determine the two
        clusters, $c_i$ and $c_j$, that are most similar.
    Replace $c_i$ and $c_j$ with a single cluster $c_i \cup c_j$
```

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



Dendrogram. Shows How Clusters are Merged



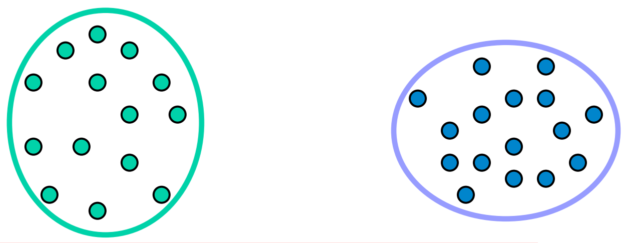
Decompose data objects into a several levels of nested partitioning (tree of clusters), called a dendrogram

A clustering of the data objects is obtained by cutting the dendrogram at the desired level, then each connected component forms a cluster

Cluster Similarity

- Assume a similarity function that determines the similarity of two instances: *sim*(*x*,*y*).
  - Euclidean /Mahalanobis, Hamming, Cosine similarity, Pearson r etc.
- How to compute similarity of two clusters each possibly containing multiple instances?
  - Single Link: Similarity of two most similar members.
  - Complete Link: Similarity of two least similar members.
  - Group Average: Average similarity between members.

Distance between Clusters



- Single link: smallest distance between an element in one cluster and an element in the other, i.e., $dist(K_i, K_j) = min(t_{ip}, t_{jq})$
- Complete link: largest distance between an element in one cluster and an element in the other, i.e., $dist(K_i, K_j) = max(t_{ip}, t_{jq})$
- Average: avg distance between an element in one cluster and an element in the other, i.e., $dist(K_i, K_j) = avg(t_{ip}, t_{jq})$
- Centroid: distance between the centroids of two clusters, i.e., $dist(K_i, K_j) = dist(C_i, C_j)$
- Medoid: distance between the medoids of two clusters, i.e., $dist(K_i, K_j) = dist(M_i, M_j)$
  - Medoid: a chosen, centrally located object in the cluster

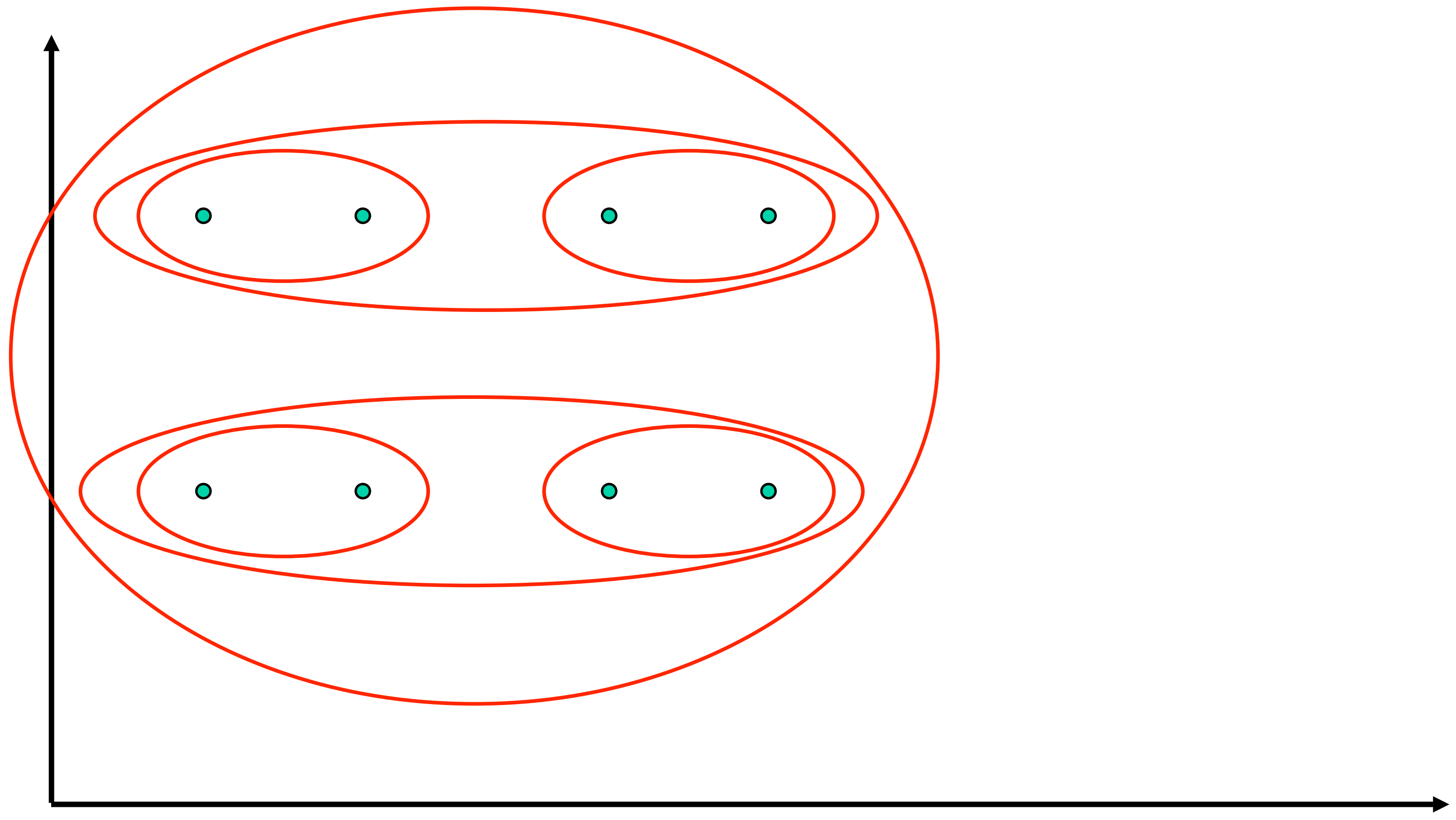
Single Link Agglomerative Clustering

- Use maximum similarity of pairs:

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

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

Single Link Example



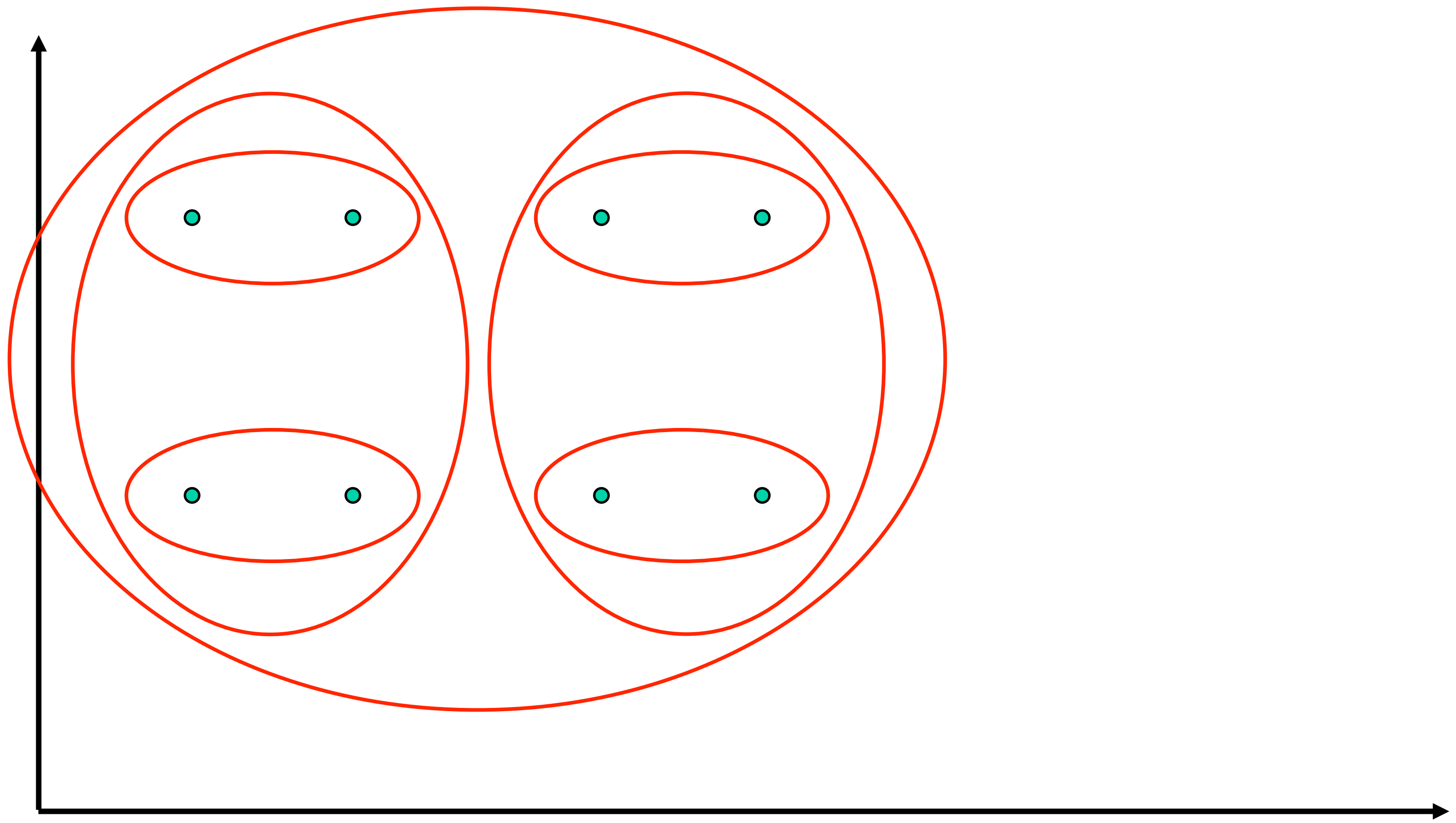
Complete Link Agglomerative Clustering

- Use minimum similarity of pairs:

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

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

Complete Link Example



Computational Complexity

- In the first iteration, all HAC methods need to compute similarity of all pairs of n individual instances which is $O(n^2)$.
- In each of the subsequent n-2 merging iterations, it must compute the distance between the most recently created cluster and all other existing clusters.
- In order to maintain an overall $O(n^2)$ performance, computing similarity to each other cluster must be done in constant time.

Computing Cluster Similarity

- 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:

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

  - Complete Link:

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

Non-Hierarchical Clustering

Non-Hierarchical Clustering

- Typically must provide the number of desired clusters, *k*.
- Randomly choose *k* instances as *seeds*, one per cluster.
- Form initial clusters based on these seeds.
- Iterate, repeatedly reallocating instances to different clusters to improve the overall clustering.
- Stop when clustering converges or after a fixed number of iterations.

K-Means

- Assumes instances are real-valued vectors.
- Clusters based on *centroids*, *center of gravity*, or mean of points in a cluster, c:

$$\vec{\mu}(c) = \frac{1}{|c|} \sum_{\vec{x} \in c} \vec{x}$$

- Reassignment of instances to clusters is based on distance to the current cluster centroids.

Distance Metrics

- Euclidian distance (L_2 norm):

$$L_2(\vec{x}, \vec{y}) = \sum_{i=1}^{m} (x_i - y_i)^2$$

- L_1 norm:

$$L_1(\vec{x}, \vec{y}) = \sum_{i=1}^{m} \left| x_i - y_i \right|$$

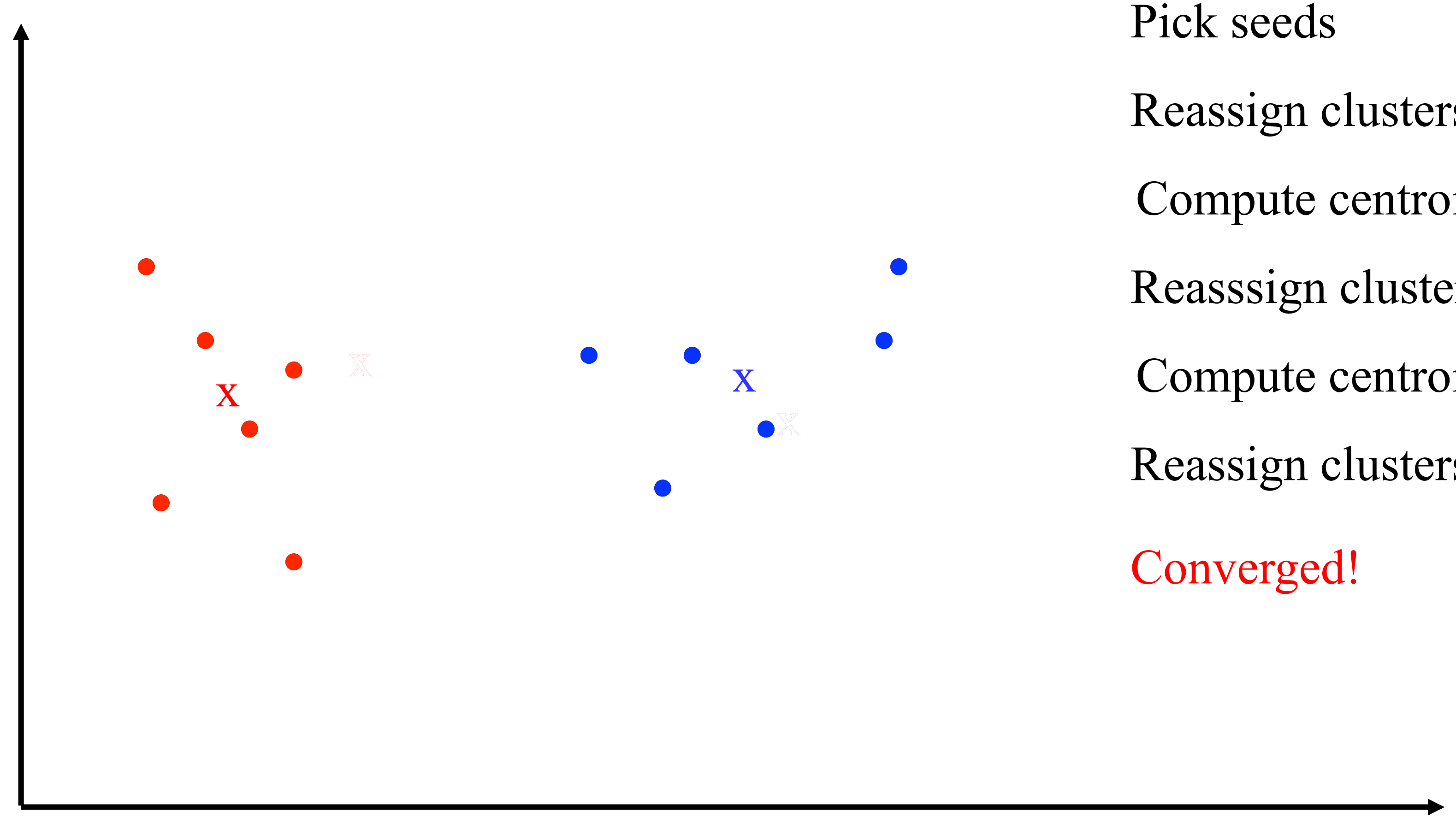
- Cosine Similarity (transform to a distance by subtracting from 1):

$$1 - \frac{\vec{x} \cdot \vec{y}}{|\vec{x}| \cdot |\vec{y}|}$$

K-Means Algorithm

```
Let $d$ be the distance measure between instances.
Select $k$ random instances $\{s_1, s_2, \ldots s_k\}$ as seeds.
Until clustering converges or other stopping criterion:
    For each instance $x_i$:
        Assign $x_i$ to the cluster $c_j$ such that $d(x_i, s_j)$ is minimal.
    (Update the seeds to the centroid of each cluster)
    For each cluster $c_j$
        $s_j = \mu(c_j)$
```

K Means Example
(K=2)



Pick seeds

Reassign clusters

Compute centroids

Reasssign clusters

Compute centroids

Reassign clusters

Converged!

Time Complexity

- Assume computing distance between two instances is O(m) where m is the dimensionality of the vectors.
- Reassigning clusters: O(kn) distance computations, or O(knm).
- Computing centroids: Each instance vector gets added once to some centroid: O(nm).
- Assume these two steps are each done once for I iterations: O($Iknm$).
- Linear in all relevant factors, assuming a fixed number of iterations, more efficient than O(n^2) HAC.

K-Means Objective

- The objective of k-means is to minimize the total sum of the squared distance of every point to its corresponding cluster centroid.

$$\sum_{l=1}^{K} \sum_{x_i \in X_l} || x_i - \mu_l ||^2$$

- Finding the global optimum is NP-hard.
- The k-means algorithm is guaranteed to converge a local optimum.

Seed Choice

- Results can vary based on random seed selection.
- Some seeds can result in poor convergence rate, or convergence to sub-optimal clusterings.
- Select good seeds using a heuristic or the results of another method.

Buckshot Algorithm

- Combines HAC and K-Means clustering.
- First randomly take a sample of instances of size $\sqrt{n}$
- Run group-average HAC on this sample, which takes only O(n) time.
- Use the results of HAC as initial seeds for K-means.
- Overall algorithm is O(n) and avoids problems of bad seed selection.

Soft Clustering

- Clustering typically assumes that each instance is given a "hard" assignment to exactly one cluster.
- Does not allow uncertainty in class membership or for an instance to belong to more than one cluster.
- *Soft clustering* gives probabilities that an instance belongs to each of a set of clusters.
- Each instance is assigned a probability distribution across a set of discovered categories (probabilities of all categories must sum to 1).

Expectation Maximumization (EM)

- Probabilistic method for soft clustering.
- Direct method that assumes k clusters: $\{c_1, c_2, \ldots\, c_k\}$
- Soft version of k-means.
- Assumes a probabilistic model of categories that allows computing $P(c_i \mid E)$ for each category, c_i, for a given example, E.
- For text, typically assume a naïve-Bayes category model.
  – Parameters $\theta = \{P(c_i), P(w_j \mid c_i): i \in \{1, \ldots k\}, j \in \{1, \ldots, |V|\}\}$

EM Algorithm

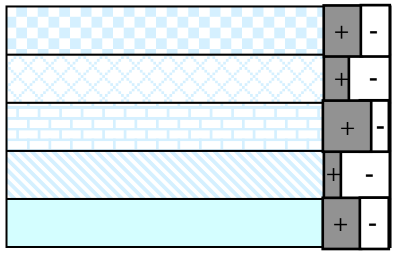
- Iterative method for learning probabilistic categorization model from unsupervised data.
- Initially assume random assignment of examples to categories.
- Learn an initial probabilistic model by estimating model parameters θ from this randomly labeled data.
- Iterate following two steps until convergence:
  - Expectation (E-step): Compute $P(c_i \mid E)$ for each example given the current model, and probabilistically re-label the examples based on these posterior probability estimates.
  - Maximization (M-step): Re-estimate the model parameters, θ, from the probabilistically re-labeled data.

EM

Initialize:

Assign random probabilistic labels to unlabeled data

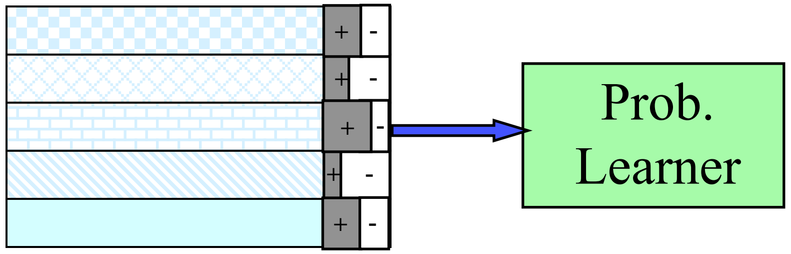
Unlabeled Examples



EM

Initialize:

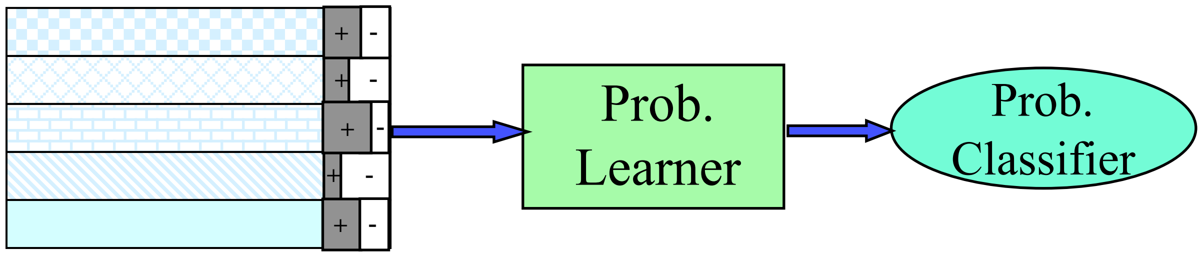
Give soft-labeled training data to a probabilistic learner



EM

Initialize:

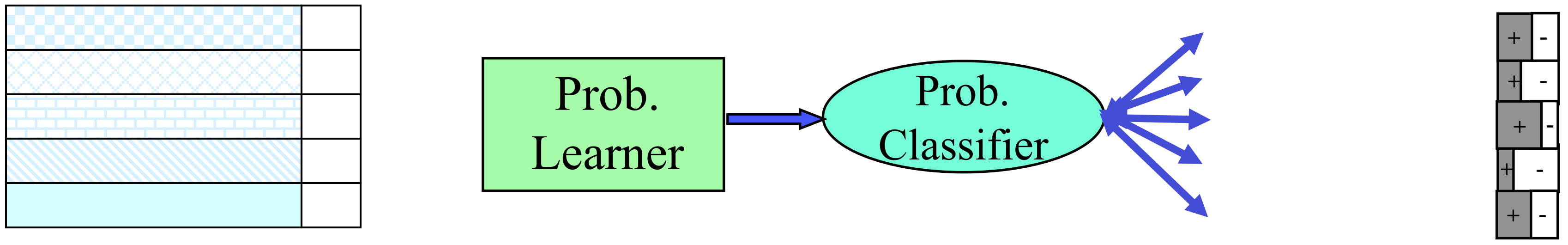
Produce a probabilistic classifier



EM

E Step:

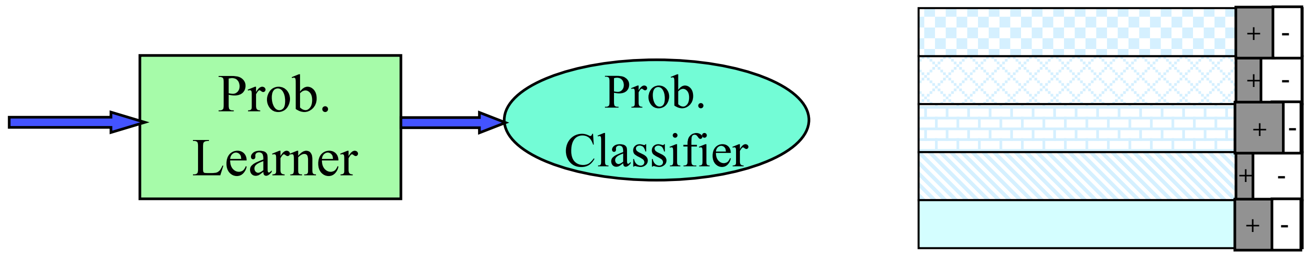
Relabel unlabled data using the trained classifier



EM

M step:

Retrain classifier on relabeled data



Continue EM iterations until probabilistic labels on unlabeled data converge.

Learning from Probabilistically Labeled Data

- Instead of training data labeled with “hard” category labels, training data is labeled with “soft” probabilistic category labels.
- When estimating model parameters θ from training data, weight counts by the corresponding probability of the given category label.
- For example, if $\mathrm{P}(c_1 \mid E) = 0.8$ and $\mathrm{P}(c_2 \mid E) = 0.2$, each word w_j in E contributes only 0.8 towards the counts n_1 and n_{1j}, and 0.2 towards the counts n_2 and n_{2j}.

Naïve Bayes EM

Randomly assign examples probabilistic category labels.
Use standard naïve-Bayes training to learn a probabilistic model
 with parameters θ from the labeled data.
Until convergence or until maximum number of iterations reached:
 E-Step: Use the naïve Bayes model θ to compute $P(c_i \mid E)$ for
  each category and example, and re-label each example
  using these probability values as soft category labels.
 M-Step: Use standard naïve-Bayes training to re-estimate the
  parameters θ using these new probabilistic category labels.

Assessing Clustering Tendency

- Assess if non-random structure exists in the data by measuring the probability that the data is generated by a uniform data distribution
- Test spatial randomness by statistic test: Hopkins Static
  - Given a dataset D regarded as a sample of a random variable o, determine how far away o is from being uniformly distributed in the data space
  - Sample *n* points, $p_1, \ldots, p_n$, uniformly from D. For each p_i, find its nearest neighbor in D: $x_i = \min\{dist(p_i, v)\}$ where v in D
  - Sample *n* points, $q_1, \ldots, q_n$, uniformly from D. For each q_i, find its nearest neighbor in D – $\{q_i\}$: $y_i = \min\{dist(q_i, v)\}$ where v in D and $v \neq q_i$
  - Calculate the Hopkins Statistic: $H = \frac{\sum_{i=1}^{n} y_i}{\sum_{i=1}^{n} x_i + \sum_{i=1}^{n} y_i}$
  - If D is uniformly distributed, $\sum x_i$ and $\sum y_i$ will be close to each other and H is close to 0.5. If D is highly skewed, H is close to 0

Measuring Clustering Quality

- Two methods: extrinsic vs. intrinsic
- Extrinsic: supervised, i.e., the ground truth is available
  - Compare a clustering against the ground truth using certain clustering quality measure
- Intrinsic: unsupervised, i.e., the ground truth is unavailable
  - Evaluate the goodness of a clustering by considering how well the clusters are separated, and how compact the clusters are
  - Ex. Silhouette coefficient

Measuring Clustering Quality: Extrinsic Methods

- Clustering quality measure: $Q(C, C_g)$, for a clustering C given the ground truth C_g.
- 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

Measuring Clustering Quality: Extrinsic Methods

- Clustering quality measure: $Q(C, C_g)$, for a clustering C given the ground truth C_g.
- 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

Silhouette Coefficient

- considering both the intra- and inter-cluster distances.
- For a point x_i, the average of the distances to all points in the same cluster is calculated. This value is set to a_i.
- Then for each cluster that does not contain x_i, the average distance of x_i to all the data points in each cluster is computed. This value is set to b_i.
- Using a_i and b_i the silhouette coefficient of a point is estimated. The average of all the silhouettes in the dataset is called the average silhouettes width for all the points in the dataset.

Silhouette Coefficient

- considering both the intra- and inter-cluster distances.
- For a point x_i, the average of the distances to all points in the same cluster is calculated. This value is set to a_i.
- Then for each cluster that does not contain x_i, the average distance of x_i to all the data points in each cluster is computed. This value is set to b_i.
- Using a_i and b_i the silhouette coefficient of a point is estimated. The average of all the silhouettes in the dataset is called the average silhouettes width for all the points in the dataset.

Silhouette Coefficient

- To evaluate the quality of a clustering one can compute the average silhouette coefficient of all points.

$$S = \frac{\sum_{i=1}^{N} \frac{b_i - a_i}{max(a_i, b_i)}}{N}$$

Conclusions

- Unsupervised learning induces categories from unlabeled data.
- Agglomerative vs. Divisive. Hard vs. soft
- There are a variety of approaches, including:
  - HAC
  - k-means
  - EM