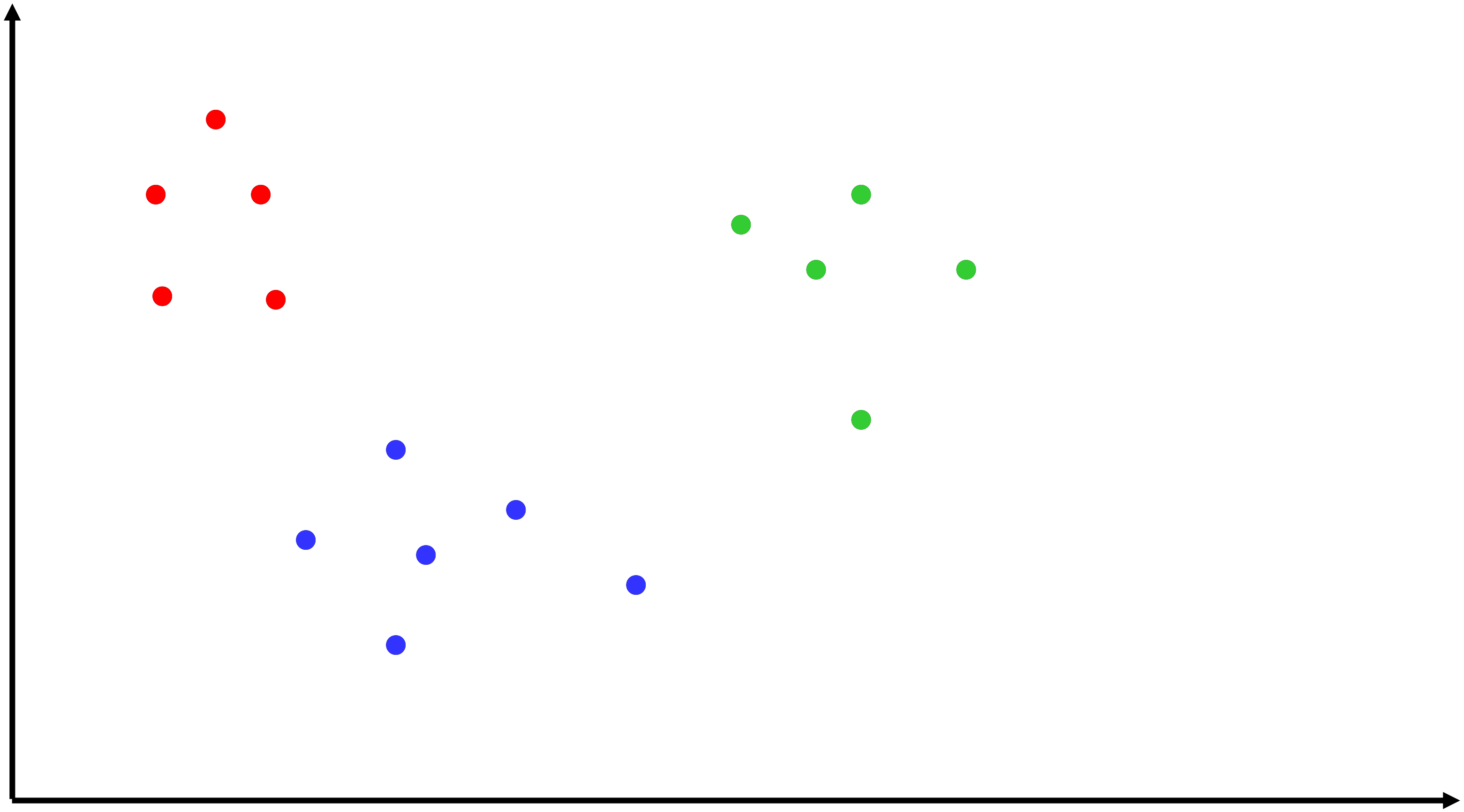
Text Clustering

Clustering

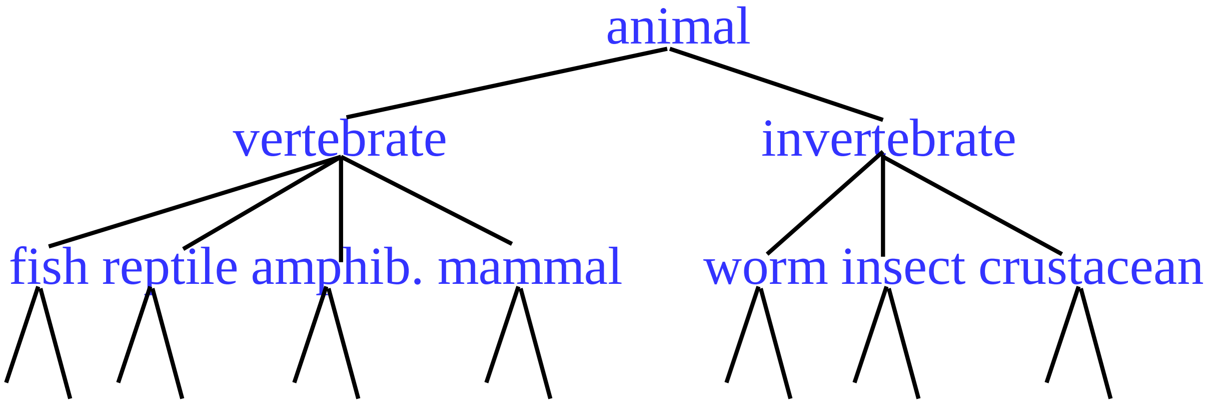
- Partition unlabeled examples into disjoint subsets of *clusters*, such that:
  - Examples within a cluster are very similar
  - Examples in different clusters are very different
- Discover new categories in an *unsupervised* manner (no sample category labels provided).

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 $c_i \cup c_j$

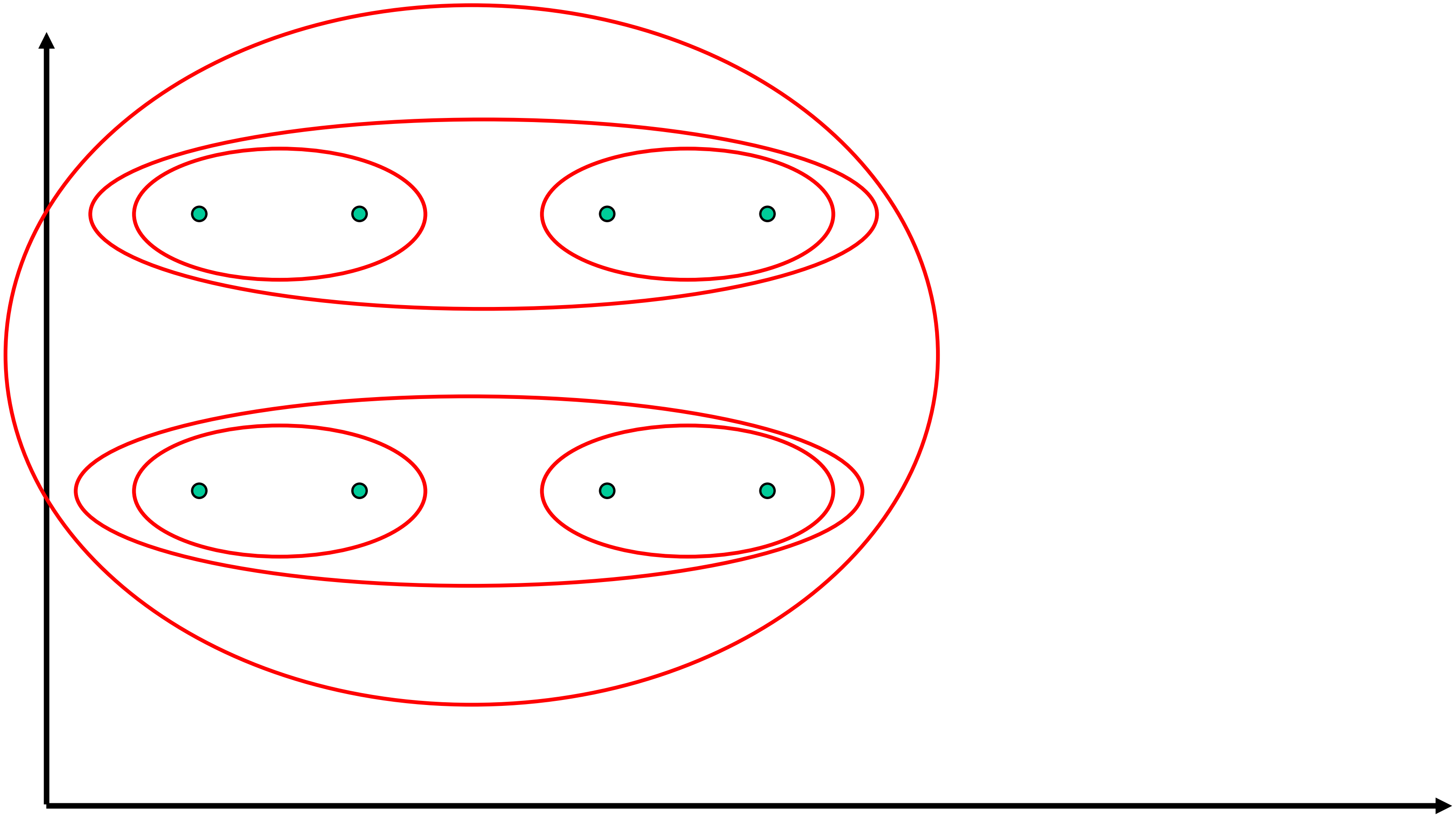
Single Link Agglomerative Clustering

- 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



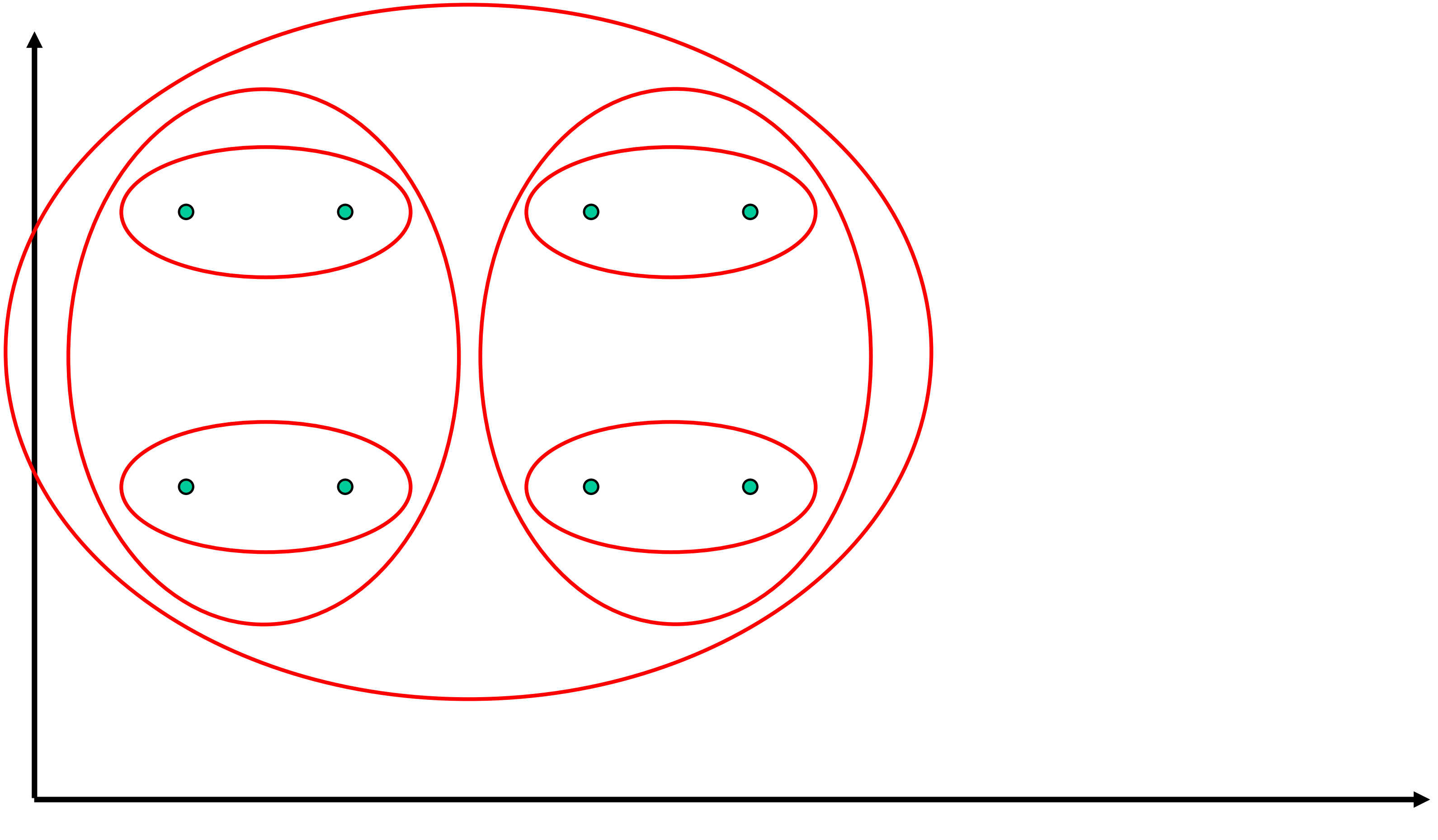
Complete Link Agglomerative Clustering

- 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



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.

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:

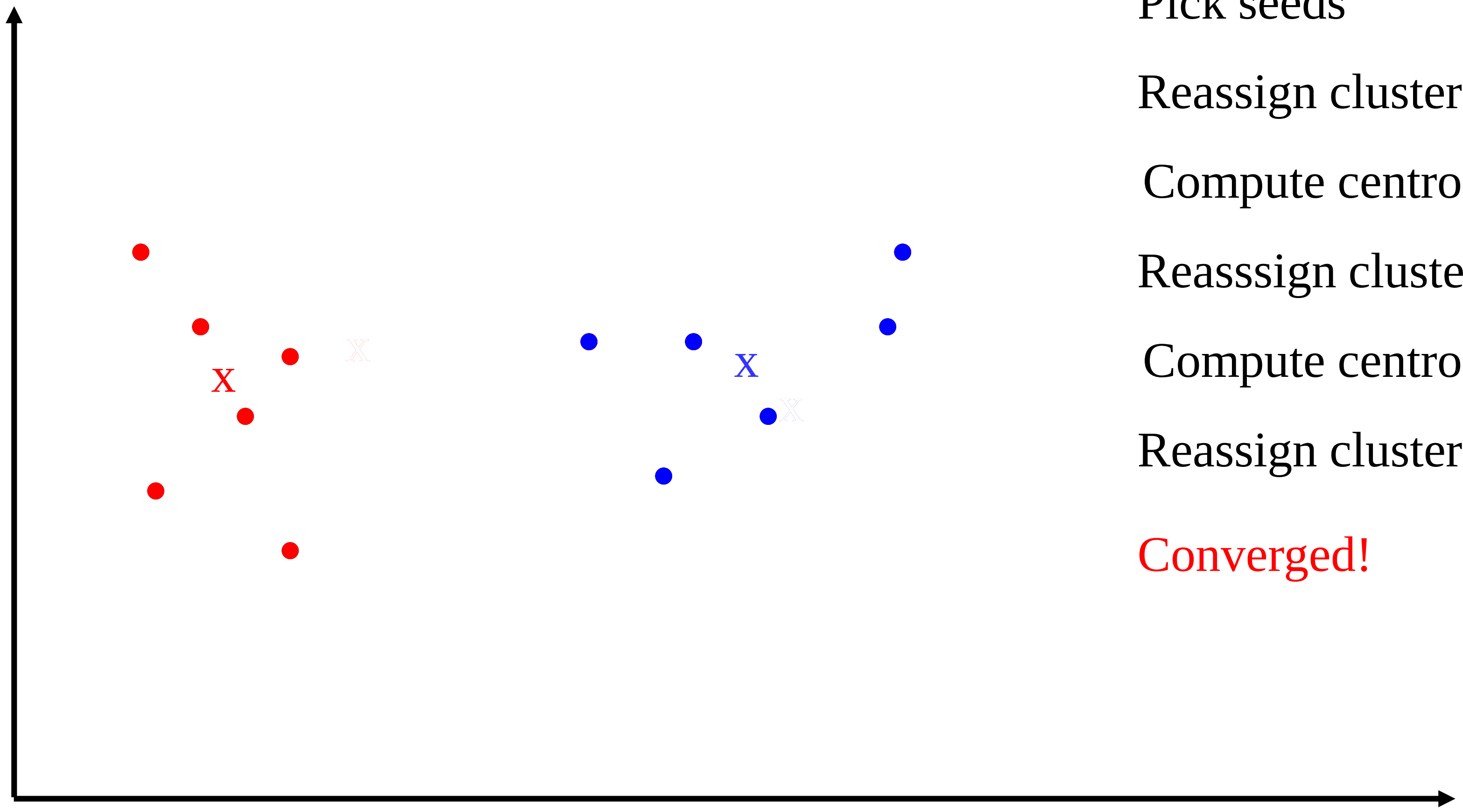
Assign x_i to the cluster c_j such that $d(x_i, s_j)$ is minimal.

(*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!

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.

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

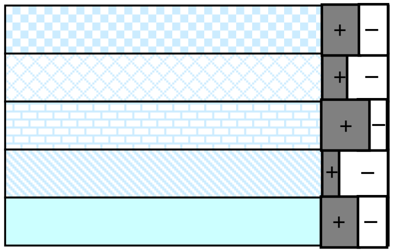
- Iterative method for learning probabilistic categorization model from unsupervised data.
- Initially assume random assignment of examples to categories.
- Learn an initial probabilistic model by estimating model parameters θ from this randomly labeled data.
- Iterate following two steps until convergence:
  - 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

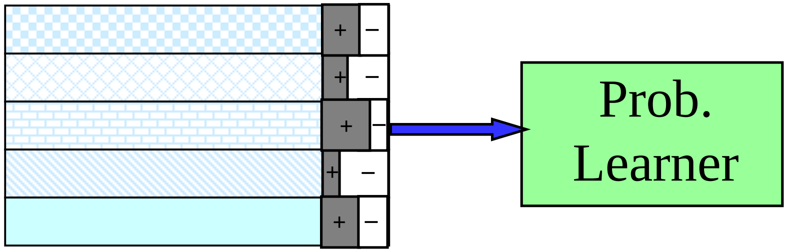
Unlabeled Examples



EM

Initialize:

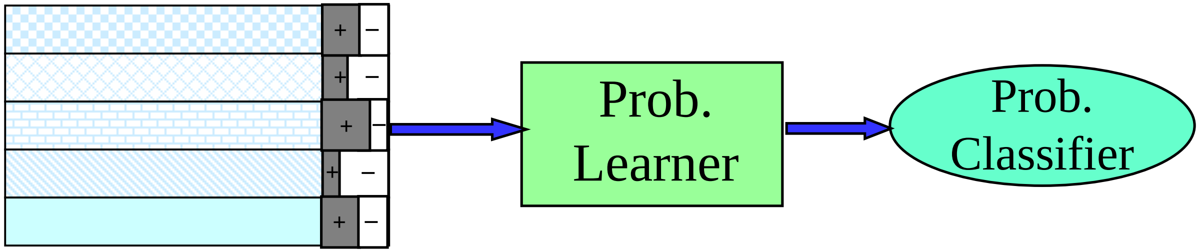
Give soft-labeled training data to a probabilistic learner



EM

Initialize:

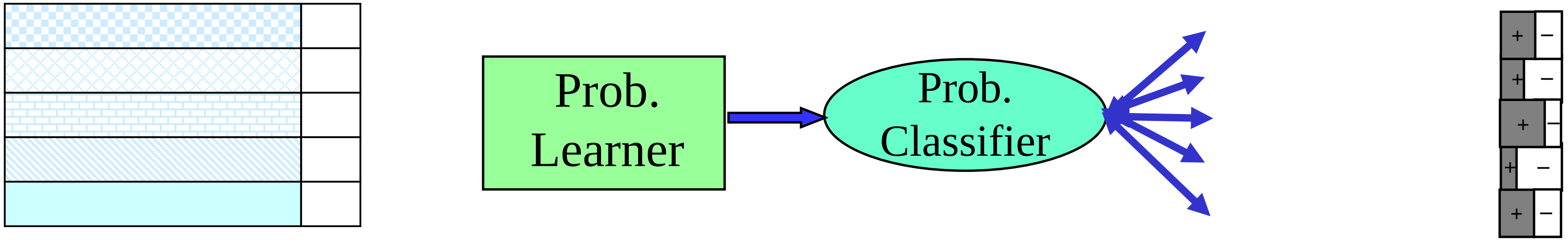
Produce a probabilistic classifier



EM

E Step:

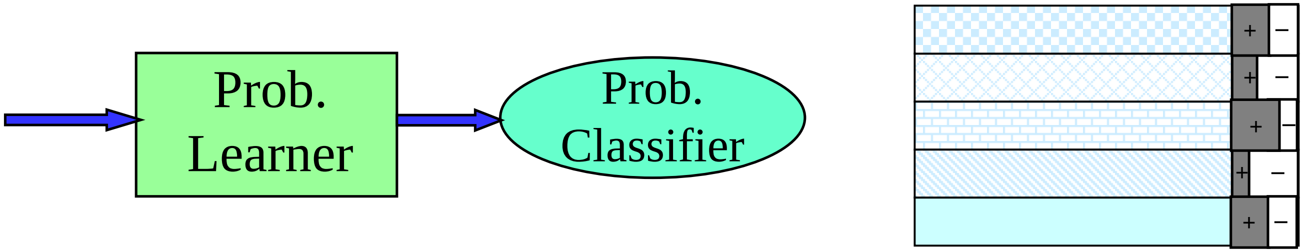
Relabel unlabled data using the trained classifier



EM

M step:

Retrain classifier on relabeled data



Continue EM iterations until probabilistic labels on unlabeled data converge.

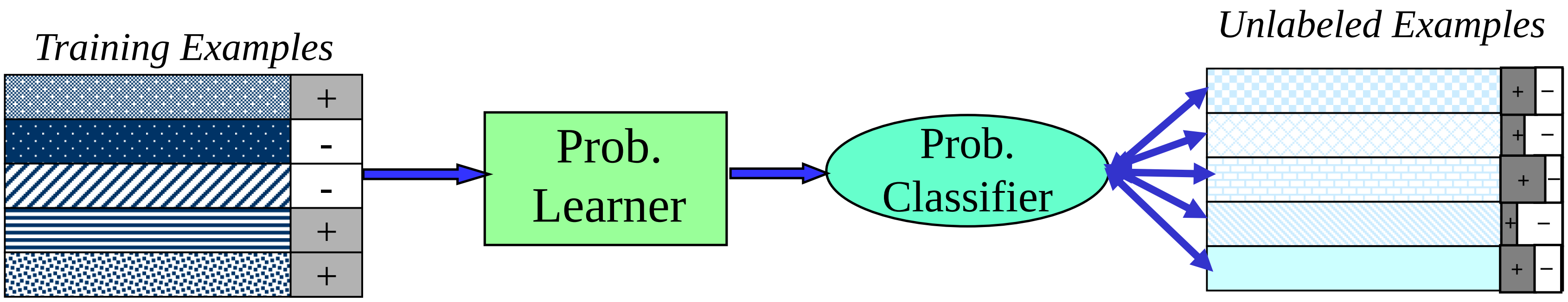
Learning from Probabilistically Labeled Data

- 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 $P(c_1 \mid E) = 0.8$ and $P(c_2 \mid E) = 0.2$, 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}.

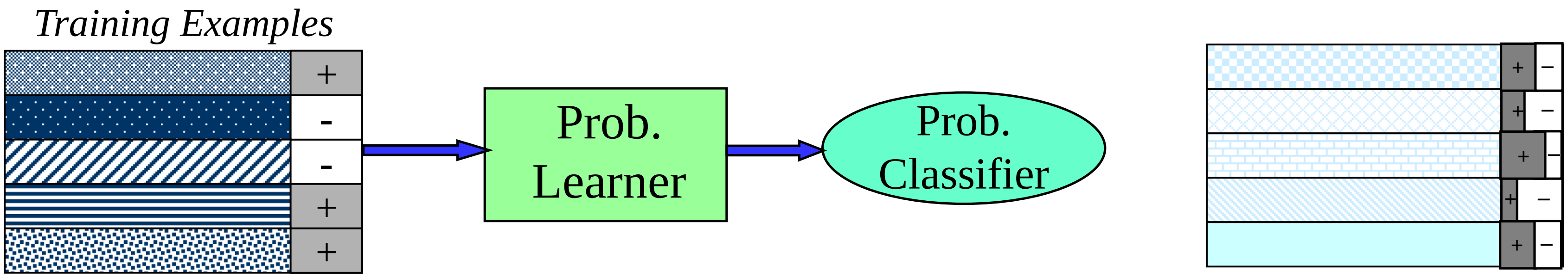
Semi-Supervised Learning

- For supervised categorization, generating labeled training data is expensive.
- Idea: Use unlabeled data to aid supervised categorization.
- Use EM in a *semi-supervised* mode by training EM on both labeled and unlabeled data.
  - Train initial probabilistic model on user-labeled subset of data instead of randomly labeled unsupervised data.
  - Labels of user-labeled examples are “frozen” and never relabeled during EM iterations.
  - Labels of unsupervised data are constantly probabilistically relabeled by EM.

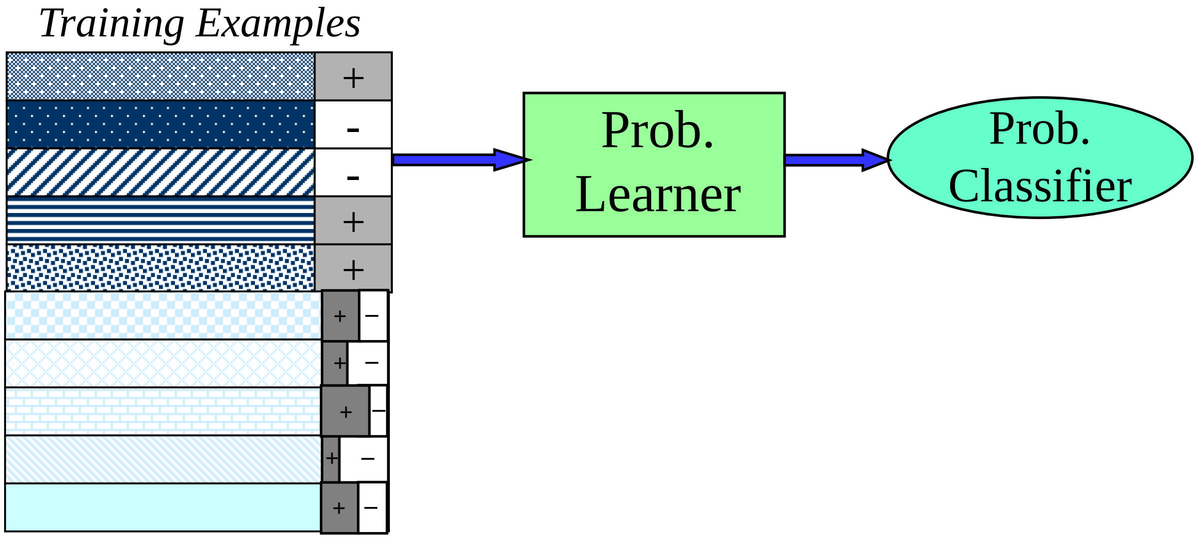
Semi-Supervised EM



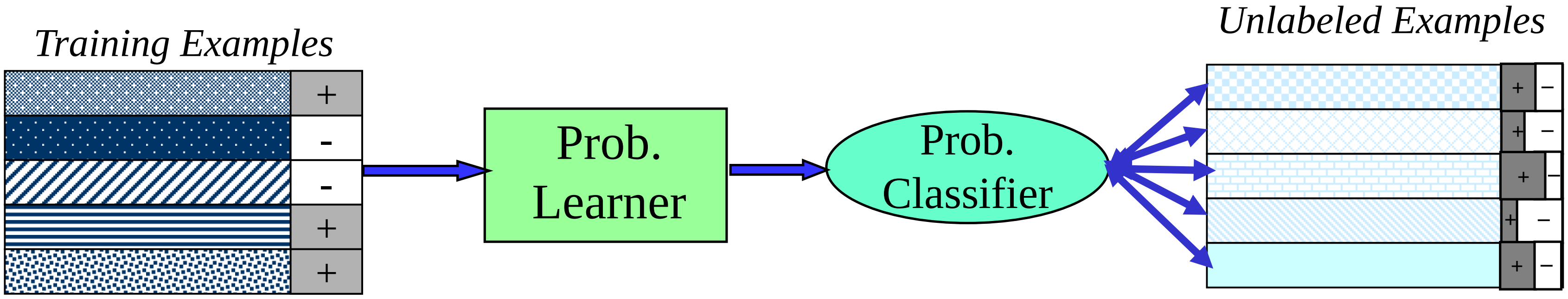
Semi-Supervised EM



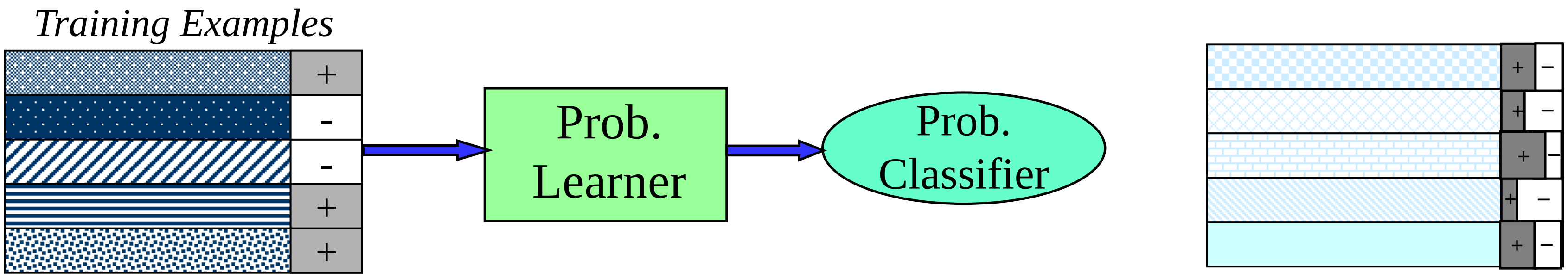
Semi-Supervised EM



Semi-Supervised EM



Semi-Supervised EM



Continue retraining iterations until probabilistic labels on unlabeled data converge.

Semi-Supervised EM Results

- Experiments on assigning messages from 20 Usenet newsgroups their proper newsgroup label.
- With very few labeled examples (2 examples per class), semi-supervised EM significantly improved predictive accuracy:
  - 27% with 40 labeled messages only.
  - 43% with 40 labeled + 10,000 unlabeled messages.
- With more labeled examples, semi-supervision can actually decrease accuracy, but refinements to standard EM can help prevent this.
  - Must weight labeled data appropriately more than unlabeled data.
- For semi-supervised EM to work, the "natural clustering of data" must be consistent with the desired categories
  - Failed when applied to English POS tagging (Merialdo, 1994)

Issues in Clustering

- How to evaluate clustering?
  - Internal:
    - Tightness and separation of clusters (e.g. k-means objective)
    - Fit of probabilistic model to data
  - External
    - Compare to known class labels on benchmark data
- Improving search to converge faster and avoid local minima.
- Overlapping clustering.

Conclusions

- Unsupervised learning induces categories from unlabeled data.
- There are a variety of approaches, including:
  - HAC
  - k-means
  - EM
- Semi-supervised learning uses both labeled and unlabeled data to improve results.