IBL and clustering

- Distance based methods
- IBL and kNN
- Clustering
  - Distance based and hierarchical
  - Probability-based
  - Expectation Maximization (EM)

Relationship of IBL with CBR

- + uses previously processed cases to do problem solving on new cases
- - CBR modifies cases and uses parts of cases in problem solving
- - CBR focuses on indexing and retrieval

IBL – training and testing

Training algorithm:

- For each training example $\langle x, f(x) \rangle$, add the example to the list *training_examples*

Classification algorithm:

- Given a query instance x_q to be classified,
  - Let $x_1 \dots x_k$ denote the k instances from *training_examples* that are nearest to x_q
  - Return

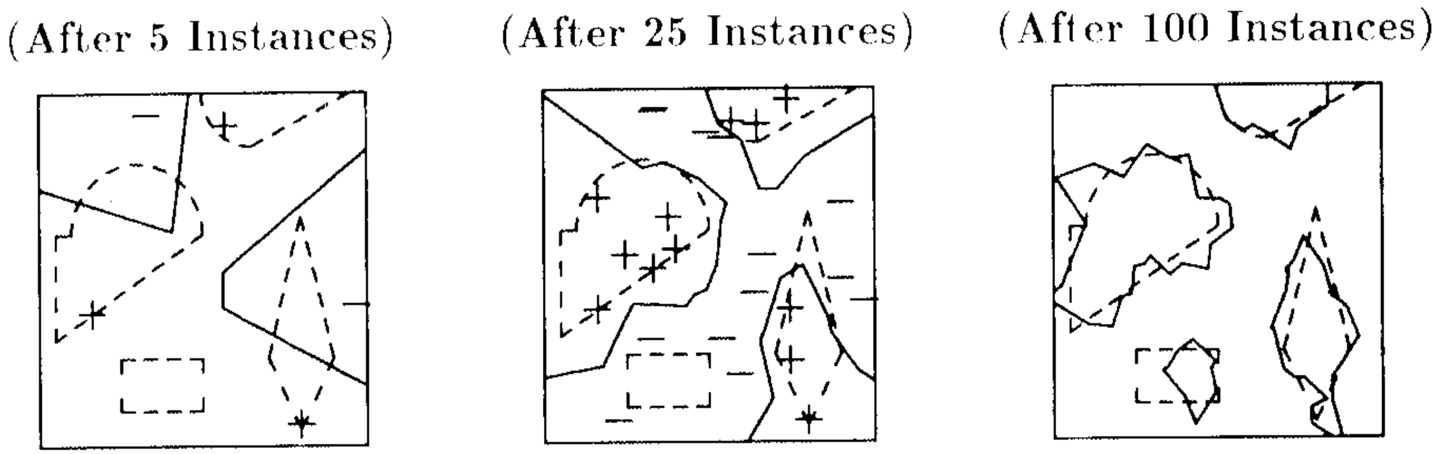
$$\hat{f}(x_q) \leftarrow \underset{v \in V}{\operatorname{argmax}} \sum_{i=1}^{k} \delta(v, f(x_i))$$

where $\delta(a, b) = 1$ if $a = b$ and where $\delta(a, b) = 0$ otherwise.

TABLE 8.1

The k-NEAREST NEIGHBOR algorithm for approximating a discrete-valued function $f : \Re^n \rightarrow V$.

- uniform distribution
- ea predicted concept boundary is halfway between a pair of adjacent pos and neg



ure 1. The extension of IBI's concept description, denoted by the solid lines, improves with training. Dashed es delineate the four disjuncts of the target concept. Positive (+) and negative ( ) instances are shown where sible.

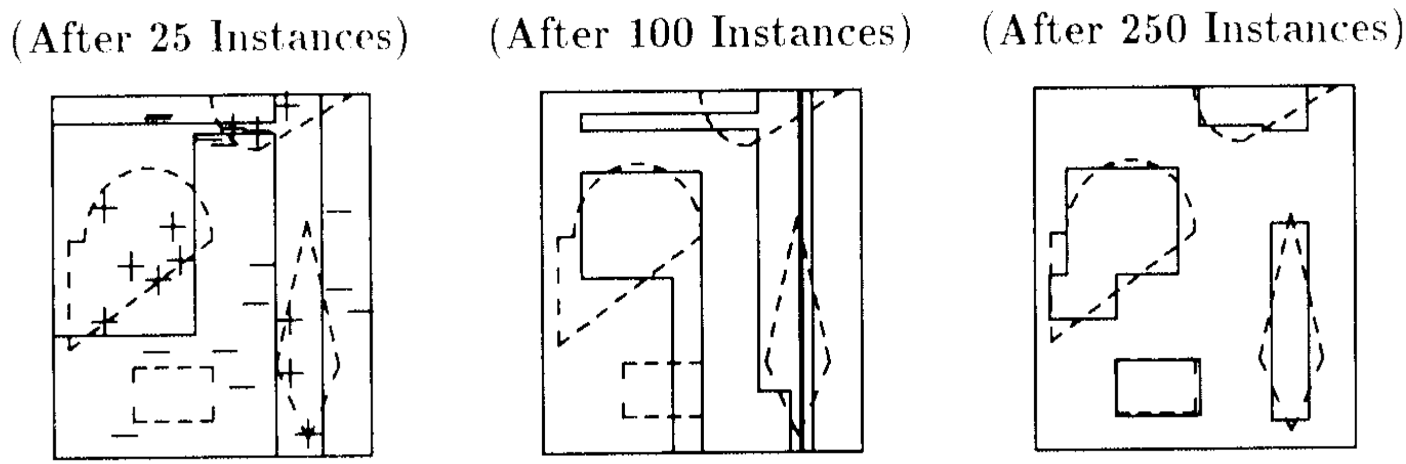


Figure 2. C4's extension eventually converges to an approximation similar to IBI's on this same training set, although C4's significance testing slows its learning rate.

similarity(x, y) =sqrt(Σf(x_i, y_i)),

f(x_i, y_i) = (x_i - y_i)**2 for numeric attrs, ($x_\iota \neq y_i$) for symbolic attrs.

Witten discusses extensions dealing with memory savings (only keep for testing instances that are misclassified during “training”), noise robustness.

Running time is a major problem with IB methods: the simple approach requires computing O(*N*) distances to classify an instance, *N* = size of the training set

A much more efficient approach involves a smart data structure known as *k*D-trees and ball trees (Witten 4.7). With *k*D trees, finding nearest neighbours costs O(*log*N)

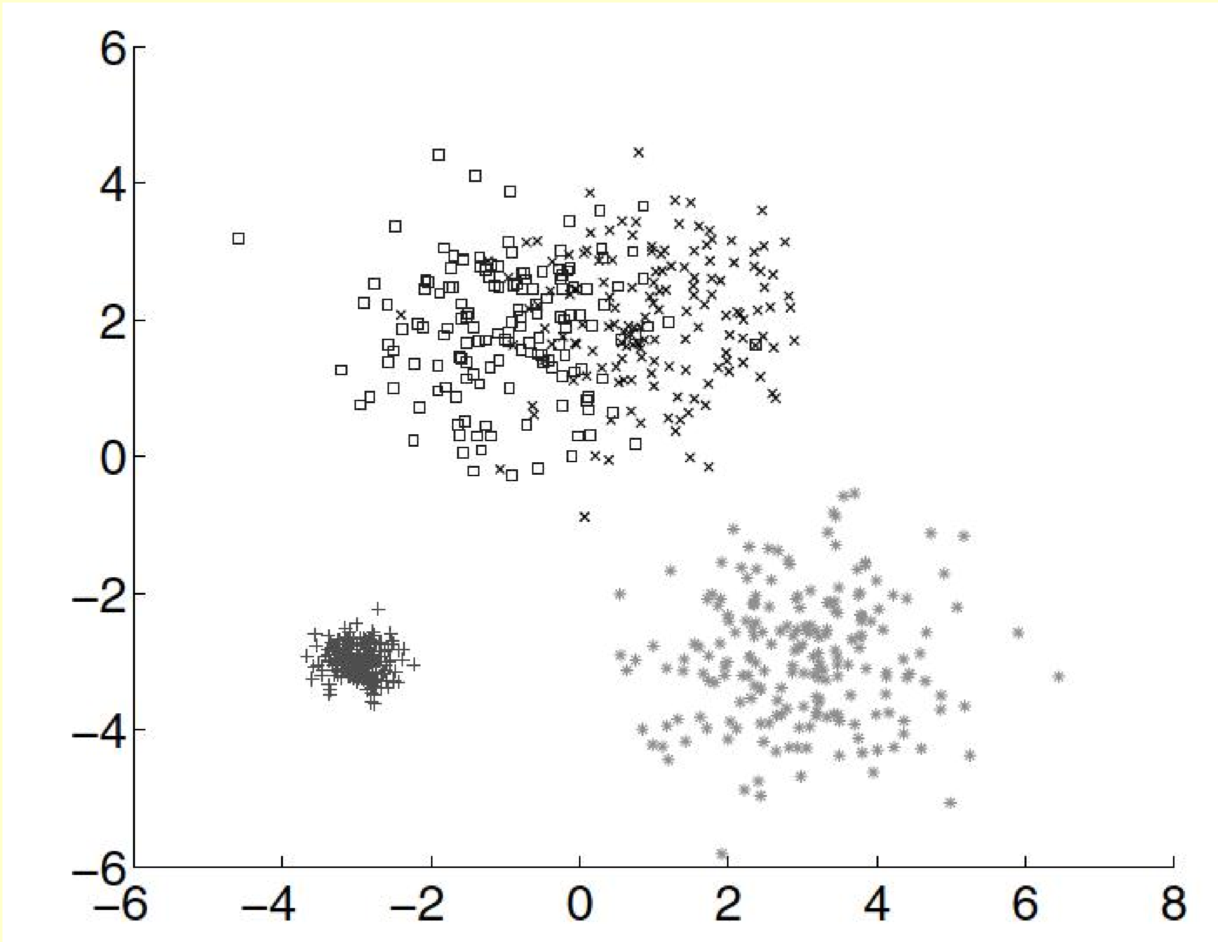
Clustering

- Unsupervised learning task; data has **no labels**
- The task is to find “natural groupings” in data
- Practically important, often the first step in exploratory data analysis
- Comes in different variants:
  - “Exclusive” clusters
  - “Shared” clusters
  - Probabilistic cluster membership

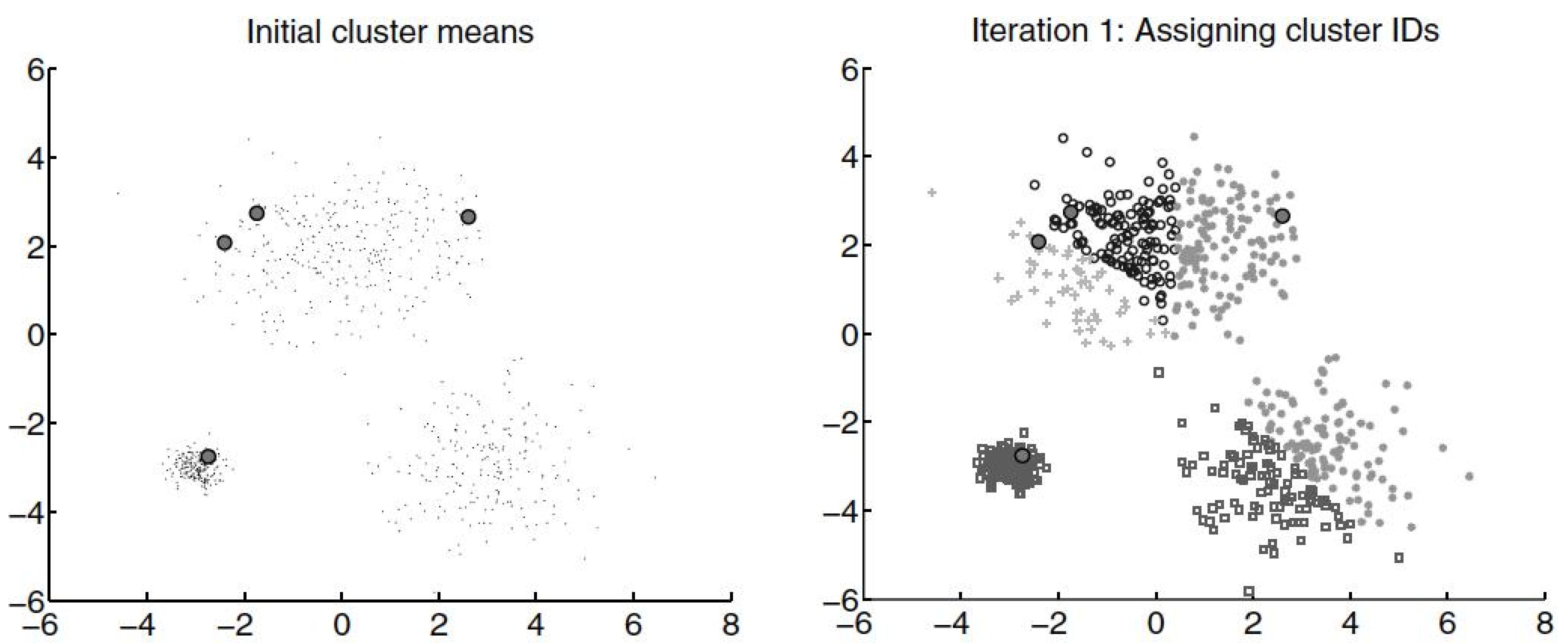
Clustering – k means

1. Define *k*- the number of clusters
2. Choose *k* points randomly as cluster centres
3. For any instance, assign it to the cluster whose centre is the closest
4. If no cluster gets modified, STOP
5. Make centroids (“instances” created by taking means of all instances in the cluster) new clusters
6. go to 3

iterative relocation

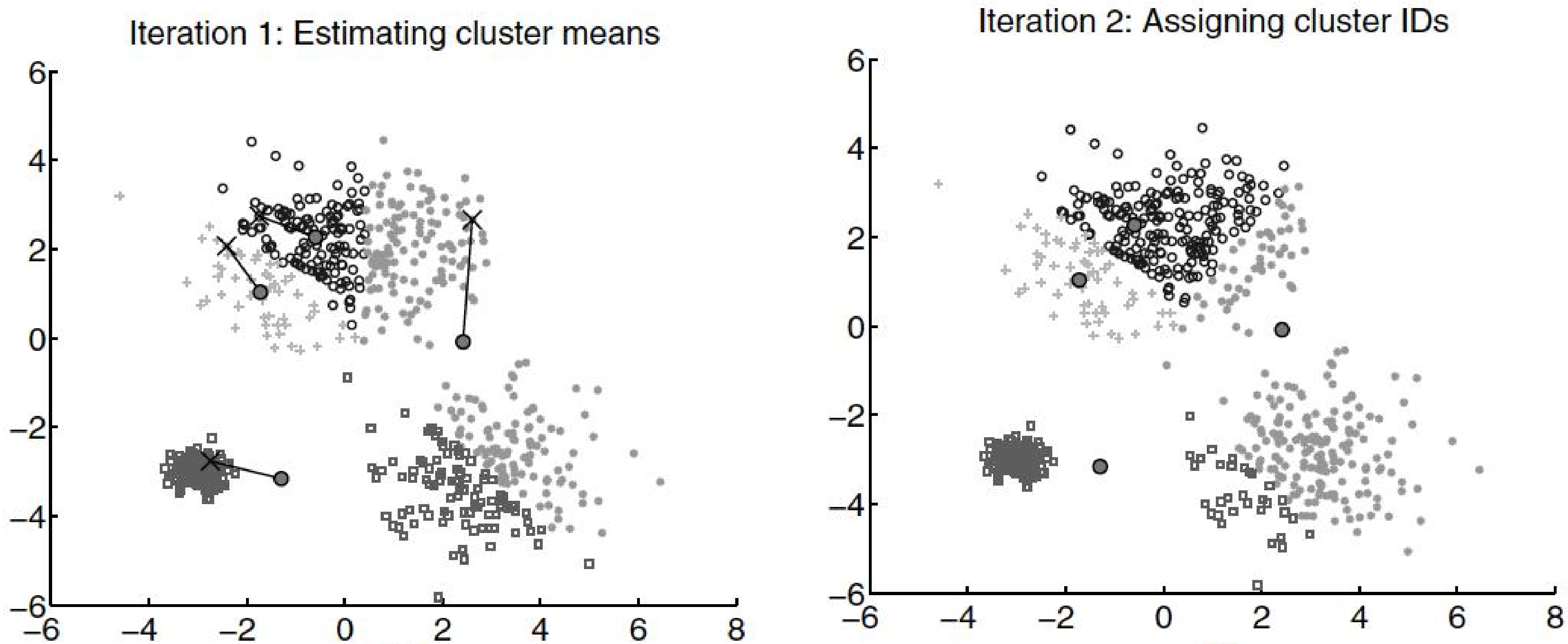


6
4
2
0
−2
−4
−6
−6
−4
−2
0
2
4
6
8

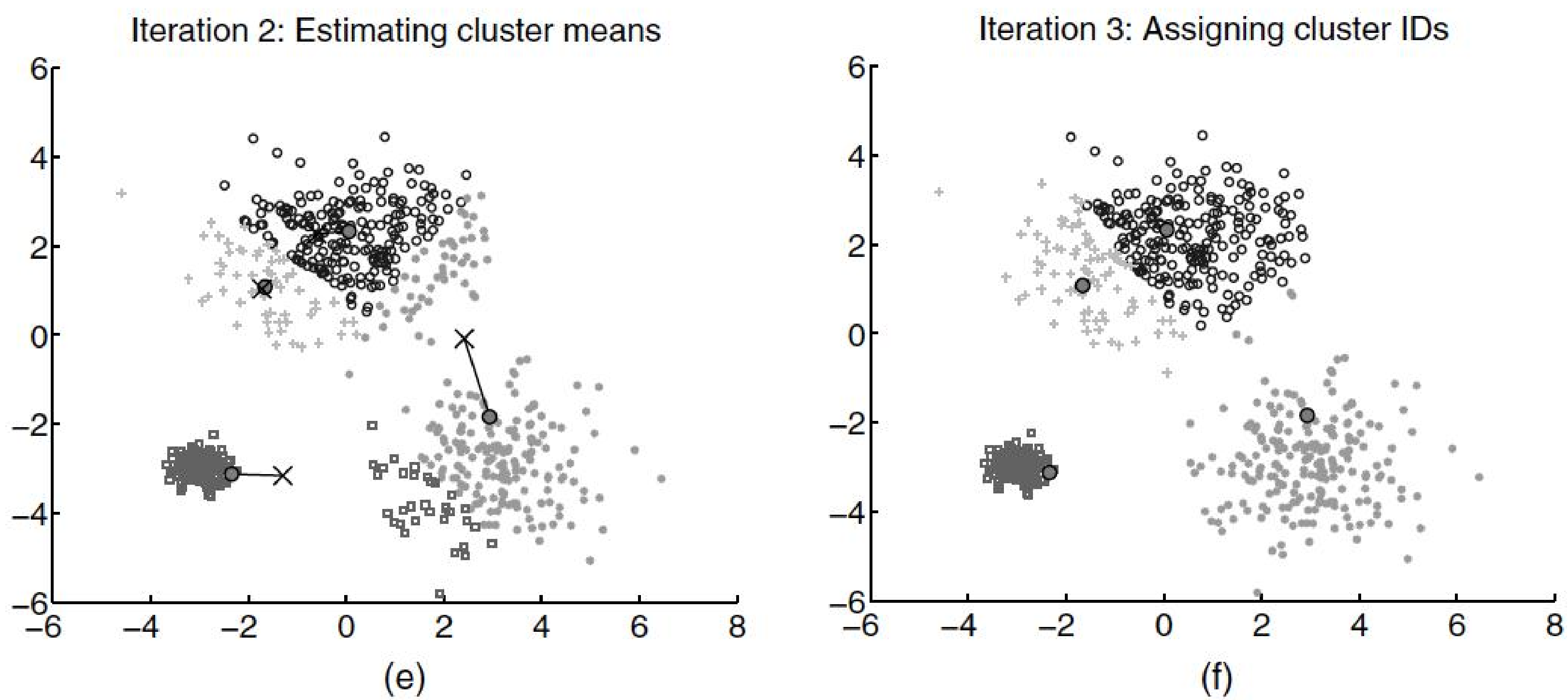

Initial cluster means
Iteration 1: Assigning cluster IDs
6
4
2
0
−2
−4
−6
−6
−4
−2
0
2
4
6
8

(a)

(b)


Iteration 1: Estimating cluster means
Iteration 2: Assigning cluster IDs
6
4
2
0
−2
−4
−6
−6
−4
−2
0
2
4
6
8
(c)

(d)


Iteration 2: Estimating cluster means
Iteration 3: Assigning cluster IDs
6
4
2
0
−2
−4
−6
−6
−4
−2
0
2
4
6
8
(e)
(f)

- When *k*-means terminates, the sum of all distances of points to their cluster centres is minimal
- This is only local, i.e. depends on the initial choice of *k*
- Efficiency problem - #iterations**k***N*
- *k*D trees can be used to improve efficiency
- *k*-medoids *vs* *k*-means

Sensitivity to outliers

- Example: {1, 2, 3, 8, 9, 10, 25}
- Clustering {1, 2, 3}, {8, 9, 10, 25} *vs* clustering {1, 2, 3, 8}, {9, 10, 25}

- How to choose *k*?
- x-val on the minimum distance: expensive
- Iterative on *k*; create 2 clusters, split recursively. "freeze" the initial 2-clustering
- When to stop splitting? Pitfall of a non-solution with 1-instance clusters; remedy – MDL-based splitting crietrion:
  - if (info. required to represent 2 new cluster centres and instances wrt these centres) > (info required to represent 1 original cluster centre and instances wrt that centre) then don't split else split

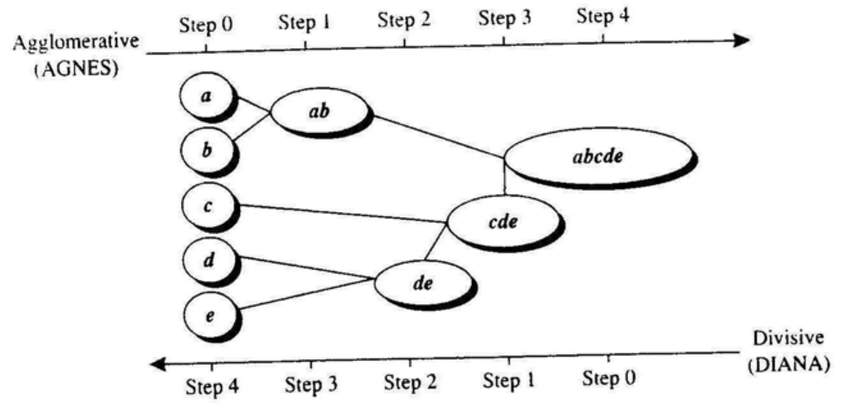
k-medoids clustering

- Instead of the mean as the cluster centre, use an instance
- More robust and less sensitive to outliers

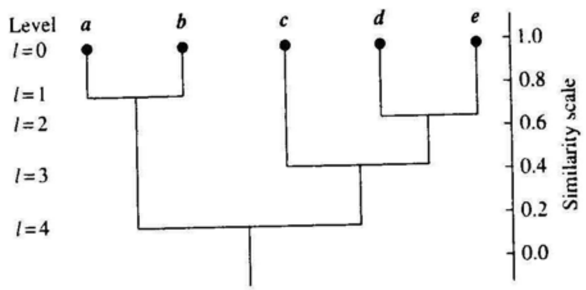
Hierarchical clustering

- Grouping instances into a hierarchy (itself not given)
- Agglomerative clustering (bottom-up) and divisive clustering (top-down)

Hierarchical clustering – example



Agglomerative and divisive hierarchical clustering on data objects $\{a, b, c, d, e\}$.



Dendrogram representation for hierarchical clustering of data objects $\{a, b, c, d, e\}$.

Evaluation of clustering

- Difficult task
- Intrinsic measures exist
- Often done on classification datasets, which is a bit of a miss
- Human comprehensibility of clusters a valuable part of evaluation

Probabilistic clustering

- Finite mixture model
- Set of *k* probability distributions represents *k* clusters: each distribution determines the probability that an instance *x* would have a certain set of attribute values ***if it was known that*** *x* ***belongs to this cluster***
- There is also a probability distribution that reflects the relative population sizes of each cluster

Finite mixture problem

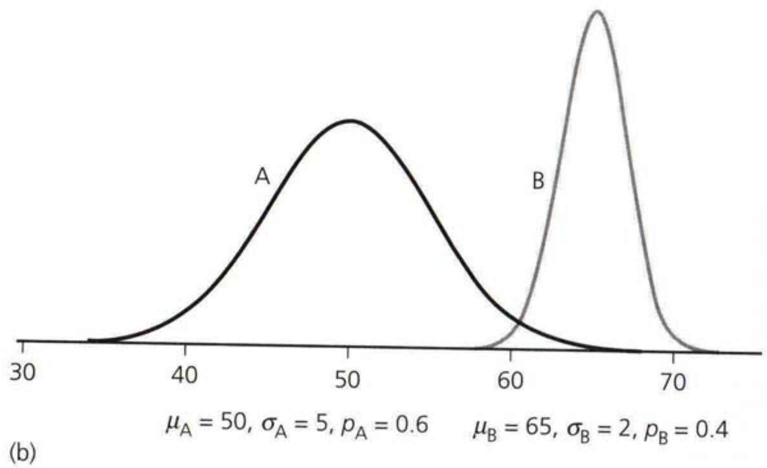


Figure 6.19 A two-class mixture model.

- Given set of instances without knowing which gaussian generated which imnstance, determine μ_A, σ_A, p_A, μ_B, σ_B ($p_B = 1 - p_A$)

Mixed model cont'd

- Had we known from which distribution (*A* or *B*) a instance comes from, we could easily compute the two μ, σ, and *p*
- If we knew the five parametrs, we would assign a new *x* to cluster *A* if

$$\frac{\Pr[A \mid x]}{\Pr[B \mid x]} = \frac{f(x, \mu_A, \sigma_A)}{f(x, \mu_B, \sigma_B)} > 1$$

- where

$$f(x, \mu, \sigma) = \frac{1}{\sqrt{2\pi\sigma}} e^{\frac{(x-\mu)^2}{2\sigma^2}}$$

The EM algorithm

- Since we do **not** know any of the five parameters, we estimate and maximize:
  - Start with a random assignment of the 5
  - Compute cluster probabilities for each instance ("expected" cluster assignments)
  - Use these cluster assignments to compute the 5 parameters ("maximize" the likelihood of the distribution given the data)
- Note that the same algorithm, with label assignment instead of cluster assignment, can be used to assign labels to unlabeled data generated by a mixture model!

EM cont'd

- But when to stop?
- Essentially, when the learning curve flattens. Specifically, when the overall probability that the data comes from this model

  $$\prod_i (p_A \Pr[x_i \mid A] + p_B \Pr[x_B \mid B])$$

  (where the cluster probabilities are given by the *f(x,μ,σ)*) starts to yield very small differences in a number of consecutive iterations
- in practice EM works with log-likelihoods to avoid multiplications

EM cont'd

- The framework is extended to mixtures of *k* gaussians (two-class to *k*-class, but *k* must be known)
- The framework is further easily extended to multiple attributes, under the assumption of independence of attributes…
- …and further extended with dropping the independence assumption and replacing the standard deviation by the covariance matrix

EM cont'd

- Parameters: for n independent attributes, $2n$ parameters; for covariant attributes, $n+n(n+1)/2$ parameters: n means and the symmetric nxn covariance matrix
- For (independent) nominal attributes, EM is like Naïve Bayes: instead of normal distribution, kv parameters per attribute are estimated, where v is the number of values of the attribute:
  – Expectation: determine the cluster (like the class in NB)
  – Maximization: like estimating NB priors (attribute-value probabilities) from data