R·I·T

Clustering

DHS 10.6-10.7, 10.9-10.10, 10.4.3-10.4.4

Clustering

Definition

A form of unsupervised learning, where we identify groups in feature space for an unlabeled sample set

- *Define* class regions in feature space using unlabeled data
- *Note:* the classes identified are abstract, in the sense that we obtain 'cluster 0' ... 'cluster n' as our classes (e.g. clustering MNIST digits, we may not get 10 clusters)

Applications

Clustering Applications Include:

- Data reduction: represent samples by their associated cluster
- Hypothesis generation
  - Discover possible patterns in the data: validate on *other* data sets
- Hypothesis testing
  - Test assumed patterns in data
- Prediction based on groups
  - e.g. selecting medication for a patient using clusters of previous patients and their reactions to medication for a given disease

Kuncheva:
Supervised vs. Unsupervised Classification

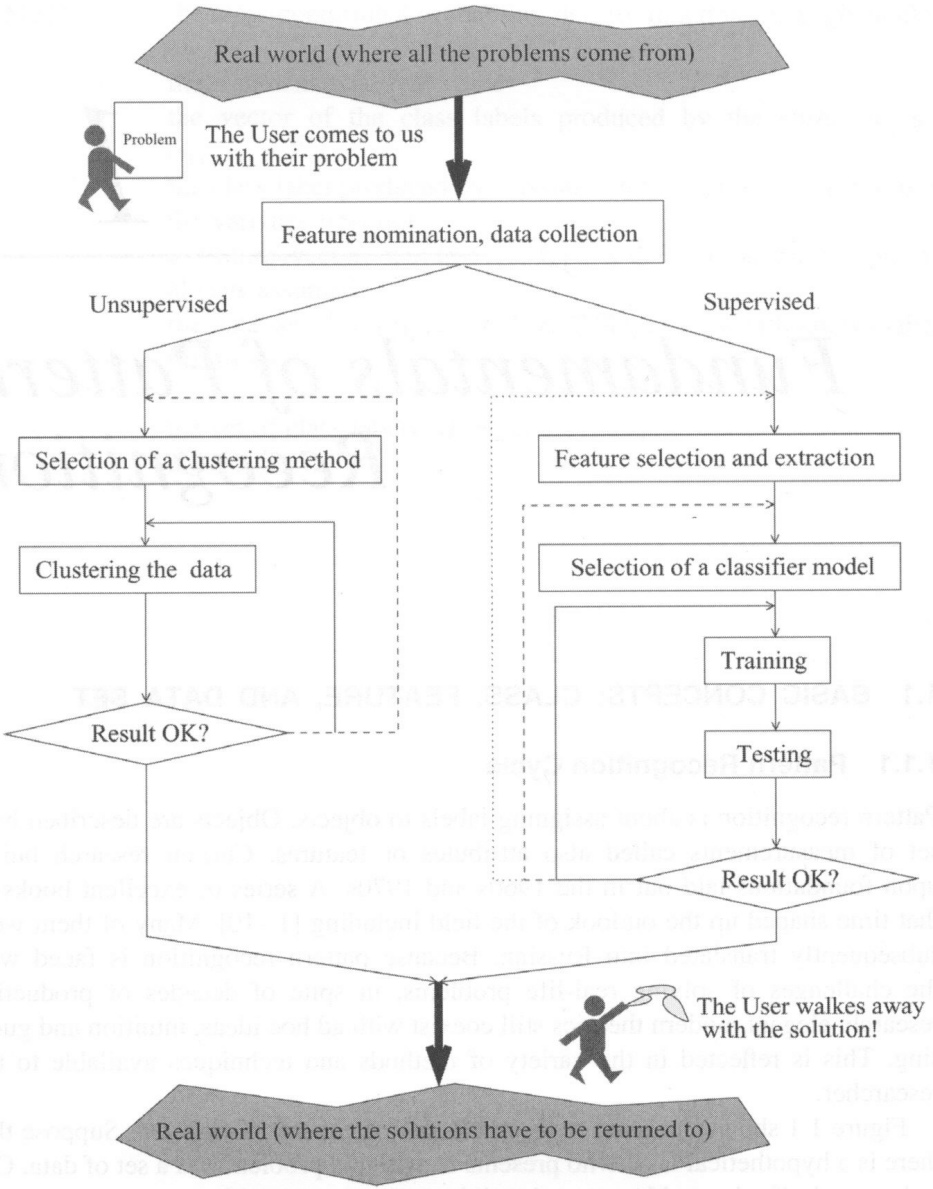


Fig. 1.1 *The pattern recognition cycle.*

A Simple Example

Assume Class Distributions Known to be Normal

Can define clusters by mean and covariance matrix

However...

We may need more information to cluster well

- Many different distributions can share a mean and covariance matrix
- ....*number* of clusters?

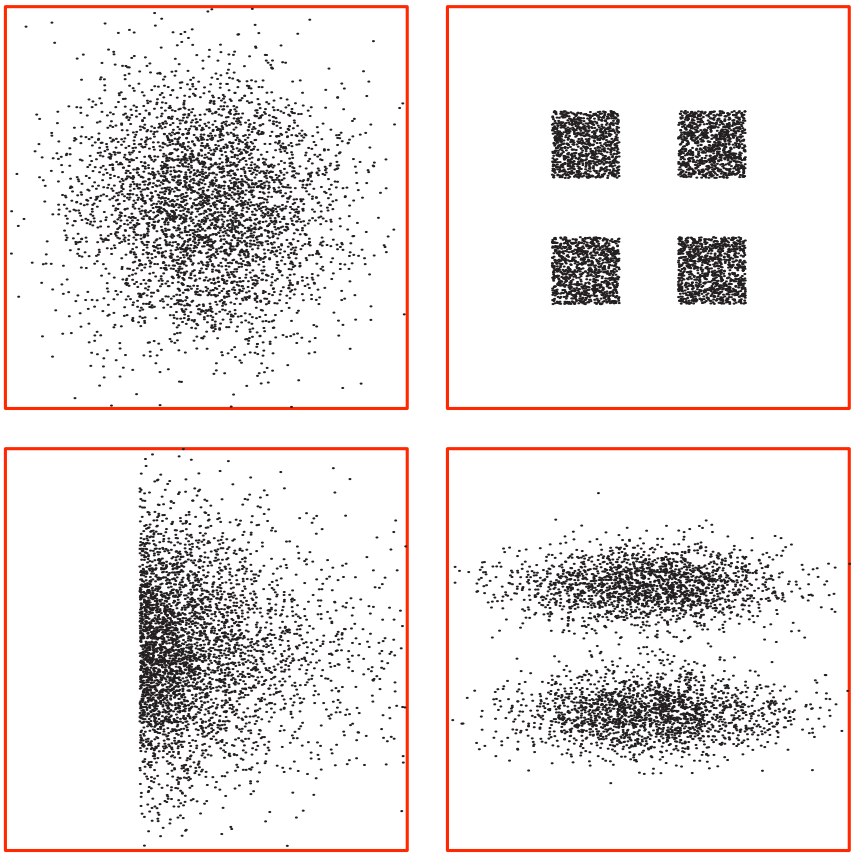


FIGURE 10.6. These four data sets have identical statistics up to second-order—that is, the same mean $\boldsymbol{\mu}$ and covariance $\boldsymbol{\Sigma}$. In such cases it is important to include in the model more parameters to represent the structure more completely. From: Richard O. Duda, Peter E. Hart, and David G. Stork, *Pattern Classification*. Copyright © 2001 by John Wiley & Sons, Inc.

Steps for Clustering

1. Feature Selection

- Ideal: small number of features with little redundancy

2. Similarity (or Proximity) Measure

- Measure of similarity or *dissimilarity*

Red: defining 'cluster space'

3. Clustering Criterion

- Determine how distance patterns determine cluster likelihood (e.g. preferring circular to elongated clusters)

4. Clustering Algorithm

- Search method used with the clustering criterion to identify clusters

5. Validation of Results

- Using appropriate tests (e.g. statistical)

6. Interpretation of Results

- Domain expert interprets clusters (clusters are *subjective*)

Choosing a Similarity Measure

Most Common: Euclidean Distance

Roughly speaking, want distance between samples in a cluster to be smaller than the distance between samples in different clusters

- Example (next slide): define clusters by a maximum distance d_0 between a point and a point in a cluster
- Rescaling features can be useful (transform the space)
  - Unfortunately, normalizing data (e.g. by setting features to zero mean, unit variance) may eliminate subclasses
  - One might also choose to rotate axes so they coincide with eigenvectors of the covariance matrix (i.e. apply PCA)

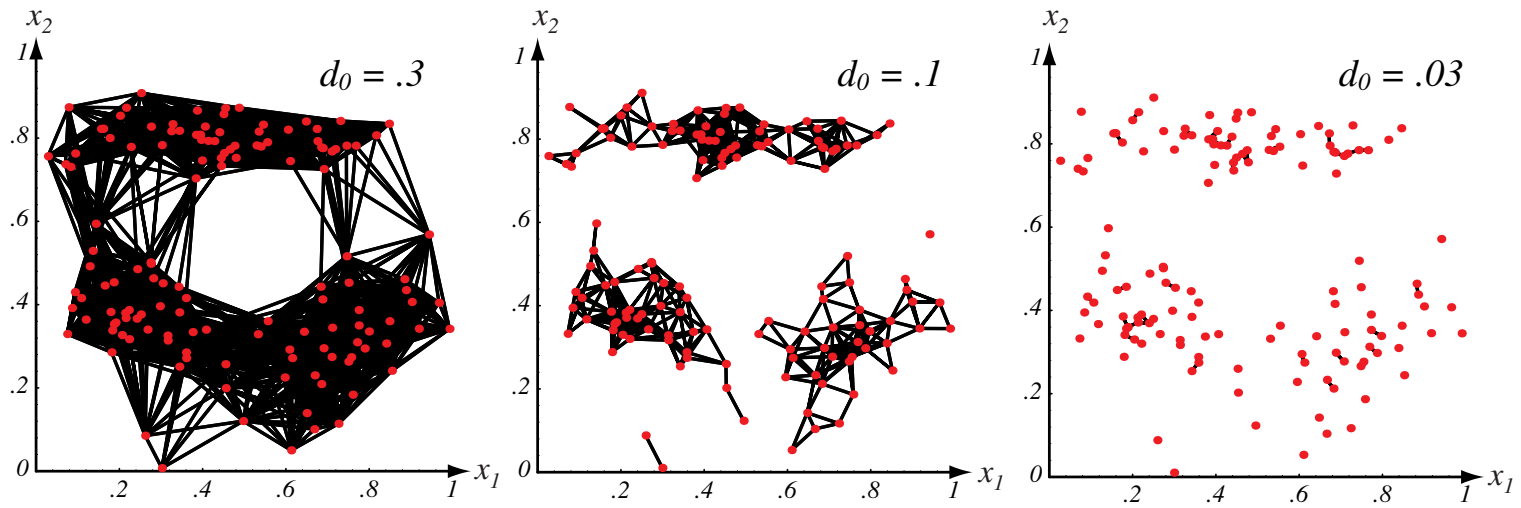


FIGURE 10.7. The distance threshold affects the number and size of clusters in similarity based clustering methods. For three different values of distance d_0, lines are drawn between points closer than d_0—the smaller the value of d_0, the smaller and more numerous the clusters. From: Richard O. Duda, Peter E. Hart, and David G. Stork, *Pattern Classification*. Copyright © 2001 by John Wiley & Sons, Inc.

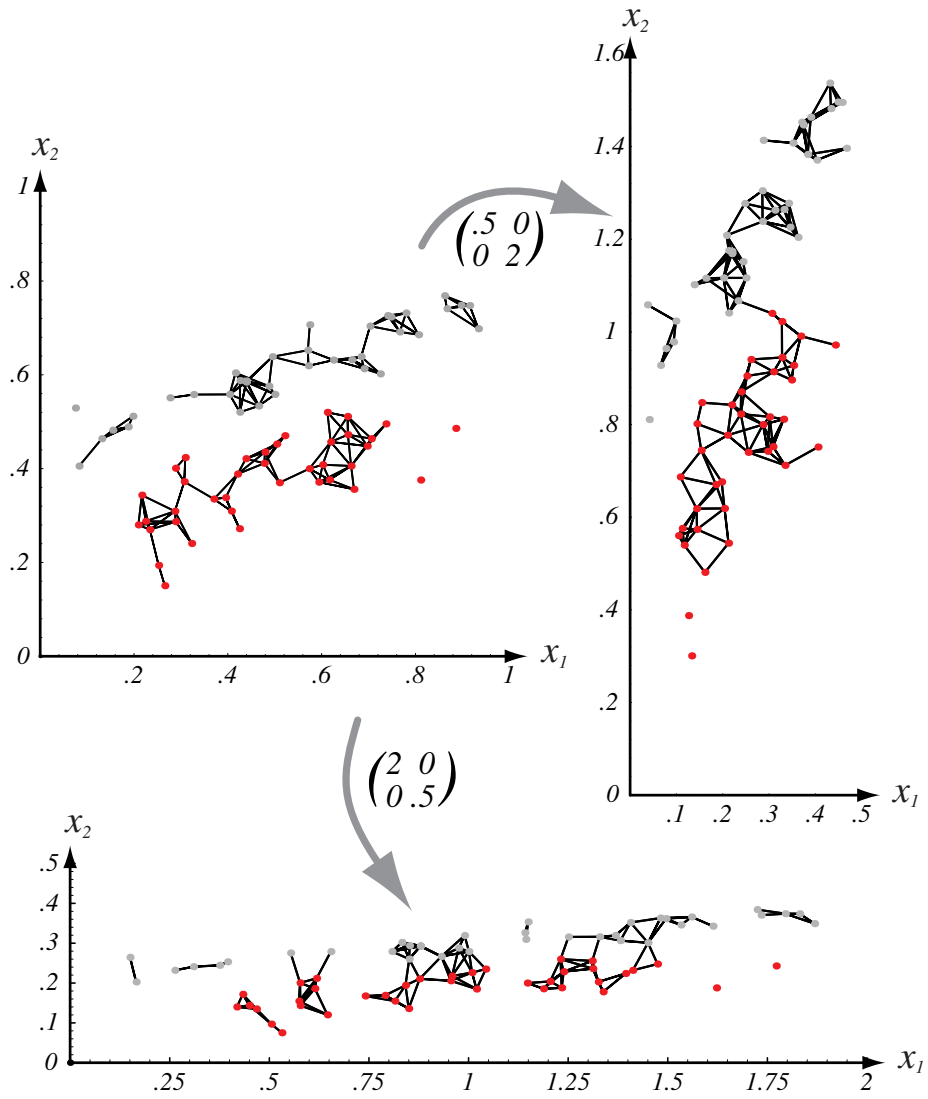


FIGURE 10.8. Scaling axes affects the clusters in a minimum distance cluster method. The original data and minimum-distance clusters are shown in the upper left; points in one cluster are shown in red, while the others are shown in gray. When the vertical axis is expanded by a factor of 2.0 and the horizontal axis shrunk by a factor of 0.5, the clustering is altered (as shown at the right). Alternatively, if the vertical axis is shrunk by

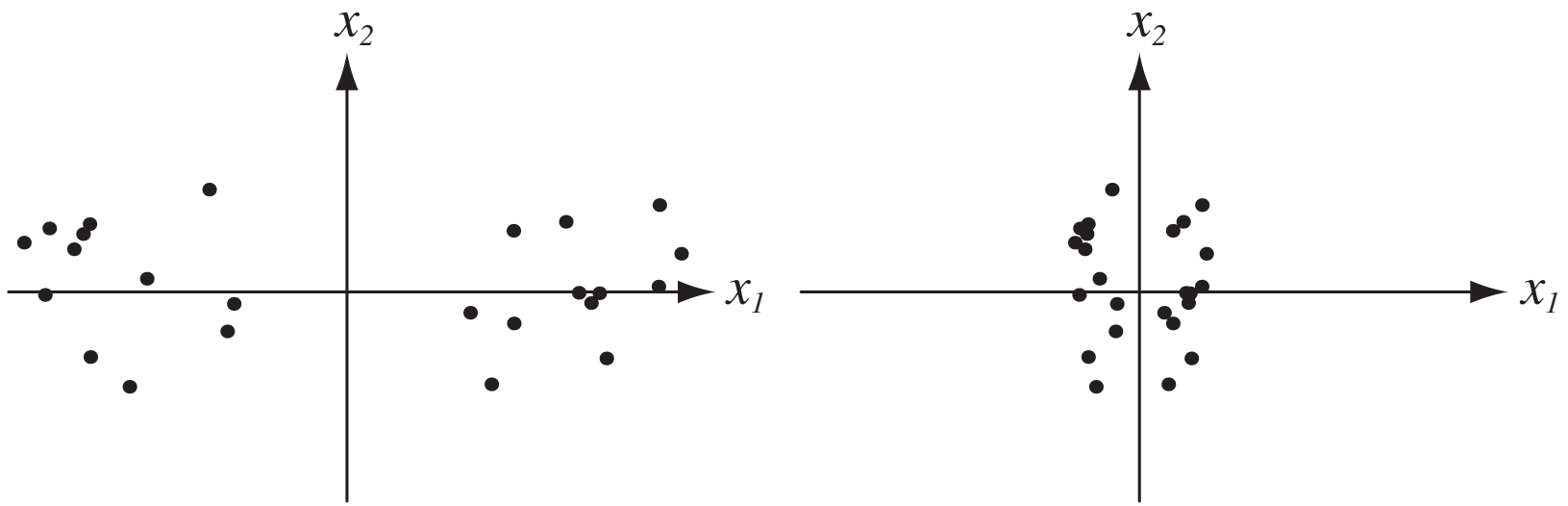


FIGURE 10.9. If the data fall into well-separated clusters (left), normalization by scaling for unit variance for the full data may reduce the separation, and hence be undesirable (right). Such a normalization may in fact be appropriate if the full data set arises from a single fundamental process (with noise), but inappropriate if there are several different processes, as shown here. From: Richard O. Duda, Peter E. Hart, and David G. Stork, *Pattern Classification*. Copyright © 2001 by John Wiley & Sons, Inc.

Other Similarity Measures

Minkowski Metric (*Dissimilarity*)

Change the exponent q:

$$d(\mathbf{x}, \mathbf{x}') = \left(\sum_{k=1}^{d} |x_k - x'_k|^q \right)^{1/q}$$

- q = 1: Manhattan (city-block) distance
- q = 2: Euclidean distance (only form invariant to translation and rotation in feature space)

Cosine *Similarity*

$$s(\mathbf{x}, \mathbf{x}') = \frac{\mathbf{x}^{\mathbf{T}} \mathbf{x}'}{||\mathbf{x}||\ ||\mathbf{x}'||}$$

Characterizes similarity by the cosine of the angle between two feature vectors (in [0,1])

- Ratio of inner product to vector magnitude product
- Invariant to rotations and dilation (*not* translation)

More on Cosine Similarity

If features binary-valued: $s(\mathbf{x}, \mathbf{x}') = \frac{\mathbf{x}^{\mathbf{T}}\mathbf{x}'}{||\mathbf{x}||\ ||\mathbf{x}'||}$

- Inner product is sum of shared feature values
- Product of magnitudes is geometric mean of number of attributes in the two vectors

Variations

Frequently used for Information Retrieval

- Ratio of shared attributes (identical lengths): $s(\mathbf{x}, \mathbf{x}') = \frac{\mathbf{x}^{\mathbf{T}}\mathbf{x}'}{d}$
- Tanimoto distance: ratio of shared attributes to attributes in x or x'

$$s(\mathbf{x}, \mathbf{x}') = \frac{\mathbf{x}^{\mathbf{T}}\mathbf{x}'}{\mathbf{x}^{\mathbf{T}}\mathbf{x} + \mathbf{x}'^{\mathbf{T}}\mathbf{x}' - \mathbf{x}^{\mathbf{T}}\mathbf{x}'}$$

Cosine Similarity: Tag Sets for YouTube Videos (Example by K. Kluever)

Let A and B be binary vectors of the same length (represent all tags in A&B)

Tag Set	Occ. Vector	dog	puppy	funny	cat
A_t	A	1	1	1	0
R_t	B	1	1	0	1

$$\text{SIM}(A, B) = \cos\theta = \frac{A \cdot B}{\|A\|\|B\|} \qquad \cos\theta = \frac{|A_t \cap R_t|}{\sqrt{|A_t|}\sqrt{|R_t|}}$$

Here SIM(A, B) is 2/3.

Additional Similarity Metrics

Theodoridis Text

Defines a *large* number of alternative distance metrics, including:

- Hamming distance: number of locations where two vectors (usually bit vectors) disagree
- Correlation coefficient
- Weighted distances...

Criterion Functions for Clustering

Criterion Function

Quantifies 'quality' of a set of clusters

- Clustering task: partition data set *D* into *c* disjoint sets D_1 ... D_c
- Choose partition maximizing the criterion function

Criterion: Sum of Squared Error

$$J_e = \sum_{i=1}^{c} \sum_{\mathbf{x} \in D_i} ||\mathbf{x} - \mu_{D_i}||^2$$

Measures total squared 'error' incurred by choice of cluster centers (cluster means)

'Optimal' Clustering

Minimizes this quantity

Issues

- Well suited when clusters compact and well-separated
- Different # points in each cluster can lead to large clusters being split 'unnaturally' (next slide)
- Sensitive to outliers

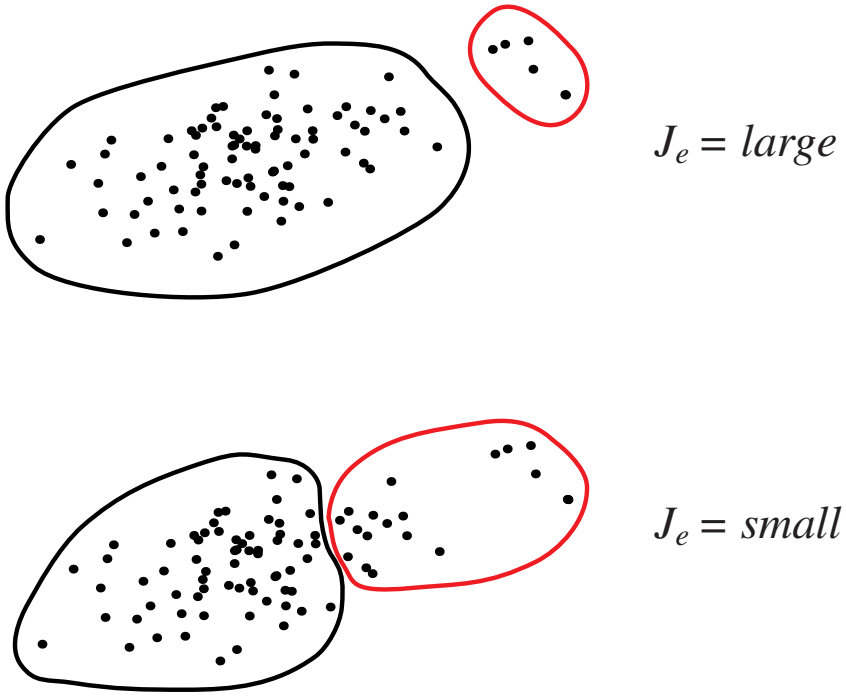


FIGURE 10.10. When two natural groupings have very different numbers of points, the clusters minimizing a sum-squared-error criterion J_e of Eq. 54 may not reveal the true underlying structure. Here the criterion is smaller for the two clusters at the bottom than for the more natural clustering at the top. From: Richard O. Duda, Peter E. Hart, and David G. Stork, *Pattern Classification*. Copyright © 2001 by John Wiley & Sons, Inc.

Related Criteria: Min Variance

$$J_e = \frac{1}{2}\sum_{i=1}^{c} n_i \bar{s}_i \qquad \bar{s}_i = \frac{1}{n_i^2}\sum_{\mathbf{x}\in D_i}\sum_{\mathbf{x}'\in D_i} ||\mathbf{x}-\mathbf{x}'||^2$$

An Equivalent Formulation for SSE

$\bar{s}_i$: mean squared distance between points in cluster i (variance)

- Alternative Criterions: use median, maximum, other descriptive statistic on distance for $\bar{s}_i$

Variation: Using Similarity (e.g. Tanimoto)

s may be any similarity function (in this case, *maximize)*

$$\bar{s}_i = \frac{1}{n_i^2}\sum_{\mathbf{x}\in D_i}\sum_{\mathbf{x}'\in D_i} s(\mathbf{x},\mathbf{x}') \qquad \bar{s}_i = \min_{\mathbf{x},\mathbf{x}'\in D_i} s(x,x')$$

Criterion: Scatter Matrix-Based

$$trace[S_w] = \sum_{i=1}^{c} \sum_{\mathbf{x} \in D_i} ||\mathbf{x} - \mu_\mathbf{i}||^\mathbf{2} = \mathbf{J_e}$$

$$S_w = \sum_{i=1}^{c} \sum_{\mathbf{x} \in D_i} (\mathbf{x} - \mu_\mathbf{i})(\mathbf{x} - \mu_\mathbf{i})^T$$

Minimize Trace of S_w (within-class)

Equivalent to SSE!

Recall that total scatter is the sum of within and between-class scatter (Sm = Sw + Sb). This means that by minimizing the trace of Sw, we also maximize Sb (*as Sm is fixed*):

$$trace[S_b] = \sum_{i=1}^{c} n_i ||\mu_i - \mu_0||^2$$

Scatter-Based Criterions, Cont'd

$$J_d = |S_w| = \left| \sum_{i=1}^{c} \sum_{\mathbf{x} \in D_i} (\mathbf{x} - \mu_\mathbf{i})(\mathbf{x} - \mu_\mathbf{i})^T \right|$$

Determinant Criterion

Roughly measures square of the scattering volume; proportional to product of variances in principal axes (*minimize!*)

- Minimum error partition will not change with axis scaling, unlike SSE

Scatter-Based: Invariant Criteria

Invariant Criteria (Eigenvalue-based)

Eigenvalues: measure ratio of between to within-cluster scatter in direction of eigenvectors (*maximize!*)

- Trace of a matrix is sum of eigenvalues (here *d* is length of feature vector)
- Eigenvalues are *invariant* under non-singular linear transformations (rotations, translations, scaling, etc.)

$$trace[S_w^{-1} S_b] = \sum_{i=1}^{d} \lambda_i$$

$$J_f = trace[S_m^{-1} S_w] = \sum_{i=1}^{d} \frac{1}{1+\lambda_i}$$

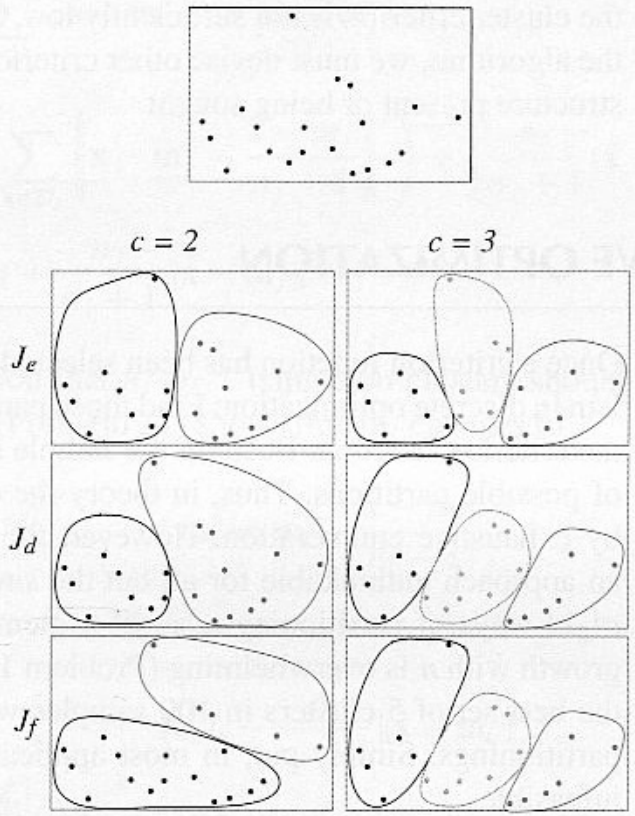
Clustering with a Criterion

Choosing Criterion

Creates a well-defined problem

- Define clusters so as to maximize the criterion function
- A search problem
  - Brute force solution: enumerate partitions of the training set, select the partition with maximum criterion value

Comparison: Scatter-Based Criteria



The raw data shown at the top does not exhibit any obvious clusters. The clusters found by minimizing a criterion depends upon the criterion function as well as the assumed number of clusters. The sum-of-squared-error criterion J_e (Eq. 54), the determinant criterion J_d (Eq. 68) and the more subtle trace criterion J_f (Eq. 70) were applied to the 20 points in the table with the assumption of $c = 2$ and $c = 3$ clusters. (Each point in the table is shown, with bounding boxes defined by $-1.8 < x_1 < 2.5$ and $-0.6 < x_2 < 1.9$.)

R·I·T

Hierarchical Clustering

Motivation

Capture similarity/distance relationships between sub-groups and samples *within* the chosen clusters

- Common in scientific taxonomies (e.g. biology)
- Can operate bottom up (individual samples to clusters, or agglomerative clustering) or top-down (single cluster to individual samples, or divisive clustering)

Agglomerative Hierarchical Clustering

Problem: Given n samples, we want c clusters

One solution: Create a *sequence* of partitions (clusterings)

- First partition, k = 1: n clusters (one cluster per sample)
- Second partition, k = 2: n-1 clusters
  - Continue reducing the number of clusters by one: merge 2 closest clusters (a cluster may be a single sample) at each step *k* until...
- Goal partition: k = n - c + 1: c clusters
  - Done; but if we're curious, we can continue on until the...
- ....Final partition, k = n: one cluster

Result

All samples and sub-clusters organized into a tree (a *dendrogram)*

- Often show cluster similarity for a dendrogram diagram (Y-axis)

If as stated above whenever two samples share a cluster they remain in a cluster at higher levels, we have a hierarchical clustering

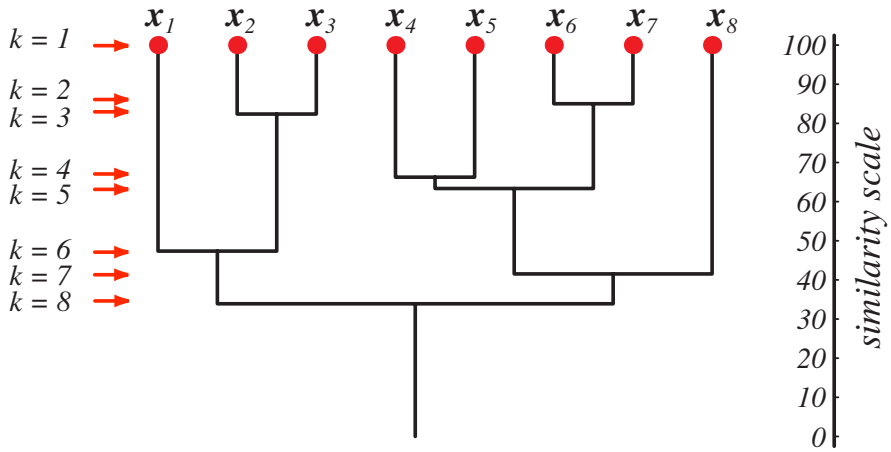


FIGURE 10.11. A dendrogram can represent the results of hierarchical clustering algorithms. The vertical axis shows a generalized measure of similarity among clusters. Here, at level 1 all eight points lie in singleton clusters; each point in a cluster is highly similar to itself, of course. Points $\mathbf{x}_6$ and $\mathbf{x}_7$ happen to be the most similar, and are merged at level 2, and so forth. From: Richard O. Duda, Peter E. Hart, and David G. Stork, *Pattern Classification*. Copyright © 2001 by John Wiley & Sons, Inc.

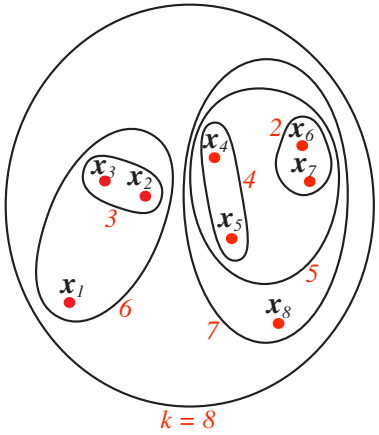


FIGURE 10.12. A set or Venn diagram representation of two-dimensional data (which was used in the dendrogram of Fig. 10.11) reveals the hierarchical structure but not the quantitative distances between clusters. The levels are numbered by k, in red. From: Richard O. Duda, Peter E. Hart, and David G. Stork, *Pattern Classification*. Copyright © 2001 by John Wiley & Sons, Inc.

Distance Measures

$$d_{min}(D_i, D_j) = \min_{x \in D_i, x \in D_j} ||x - x'||$$

$$d_{max}(D_i, D_j) = \max_{x \in D_i, x \in D_j} ||x - x'||$$

$$d_{avg}(D_i, D_j) = \frac{1}{n_i n_j} \sum_{x \in D_i} \sum_{x' \in D_j} ||x - x'||$$

$$d_{mean}(D_i, D_j) = ||m_i - m_j||$$

Listed Above:

Minimum, maximum and average inter-sample distance (samples for clusters i,j: D_i , D_j)

Difference in cluster means (m_i, m_j)

Nearest-Neighbor Algorithm

Also Known as "Single-Linkage" Algorithm

$$d_{min}(D_i, D_j) = \min_{x \in D_i, x \in D_j} ||x - x'||$$

Agglomerative hierarchical clustering using d_{min}

- Two nearest neighbors in *separate* clusters determine clusters merged at each step
- If we continue until k = n (c = 1), produce a *minimum spanning tree* (similar to Kruskal's alg.)
    - MST: Path exists between all node (sample) pairs, sum of edge costs minimum for all spanning trees

Issues

Sensitive to noise and slight changes in position of data points (*chaining effect)*

- Example: next slide

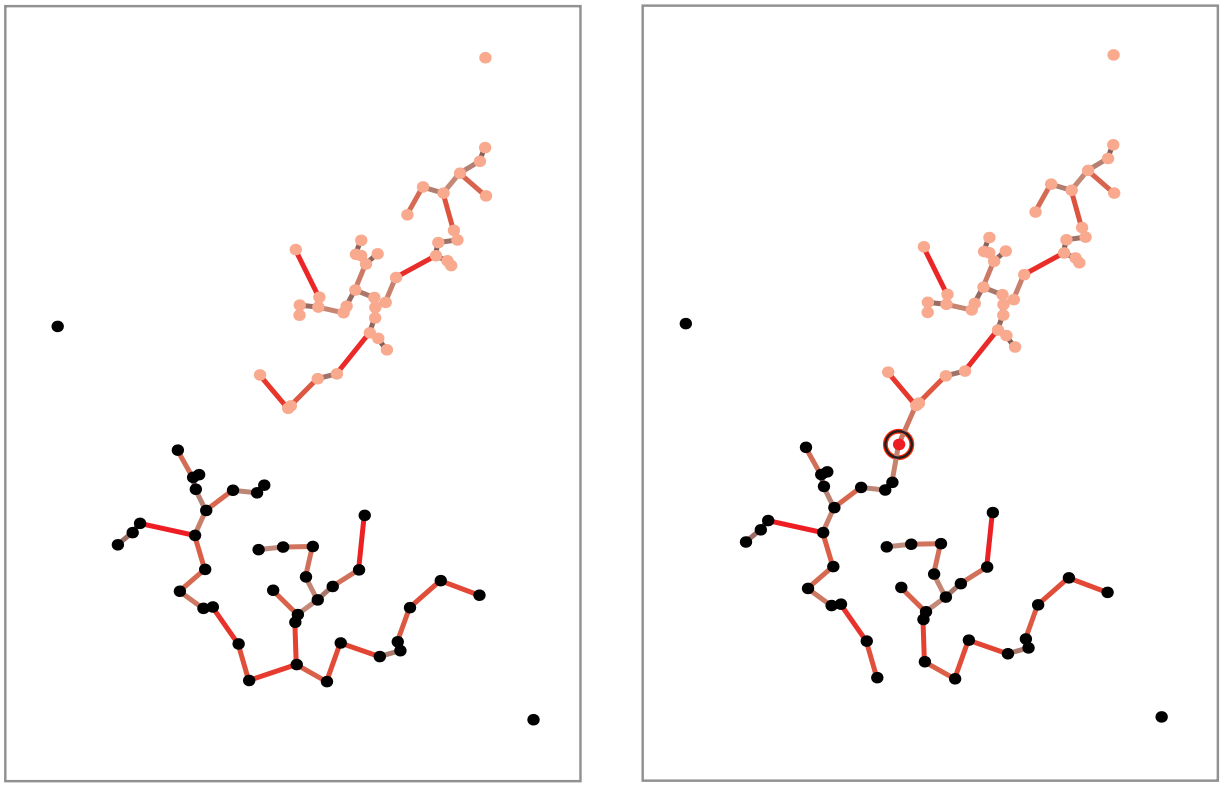


FIGURE 10.13. Two Gaussians were used to generate two-dimensional samples, shown in pink and black. The nearest-neighbor clustering algorithm gives two clusters that well approximate the generating Gaussians (left). If, however, another particular sample is generated (circled red point at the right) and the procedure is restarted, the clusters do not well approximate the Gaussians. This illustrates how the algorithm is sensitive to the details of the samples. From: Richard O. Duda, Peter E. Hart, and David G. Stork, *Pattern Classification*. Copyright © 2001 by John Wiley & Sons, Inc.

Farthest-Neighbor Algorithm

$$d_{max}(D_i, D_j) = \max_{x \in D_i, x \in D_j} ||x - x'||$$

Agglomerative hierarchical clustering using d_{max}

- Clusters with the smallest maximum distance between two points are merged at each step
  - Goal: minimal increase to largest cluster diameter at each iteration (discourages elongated clusters)
- Known as 'Complete-Linkage Algorithm' if terminated when distance between nearest clusters exceeds a given threshold distance

Issues

Works well for compact and roughly equal in size; with elongated clusters, result can be meaningless

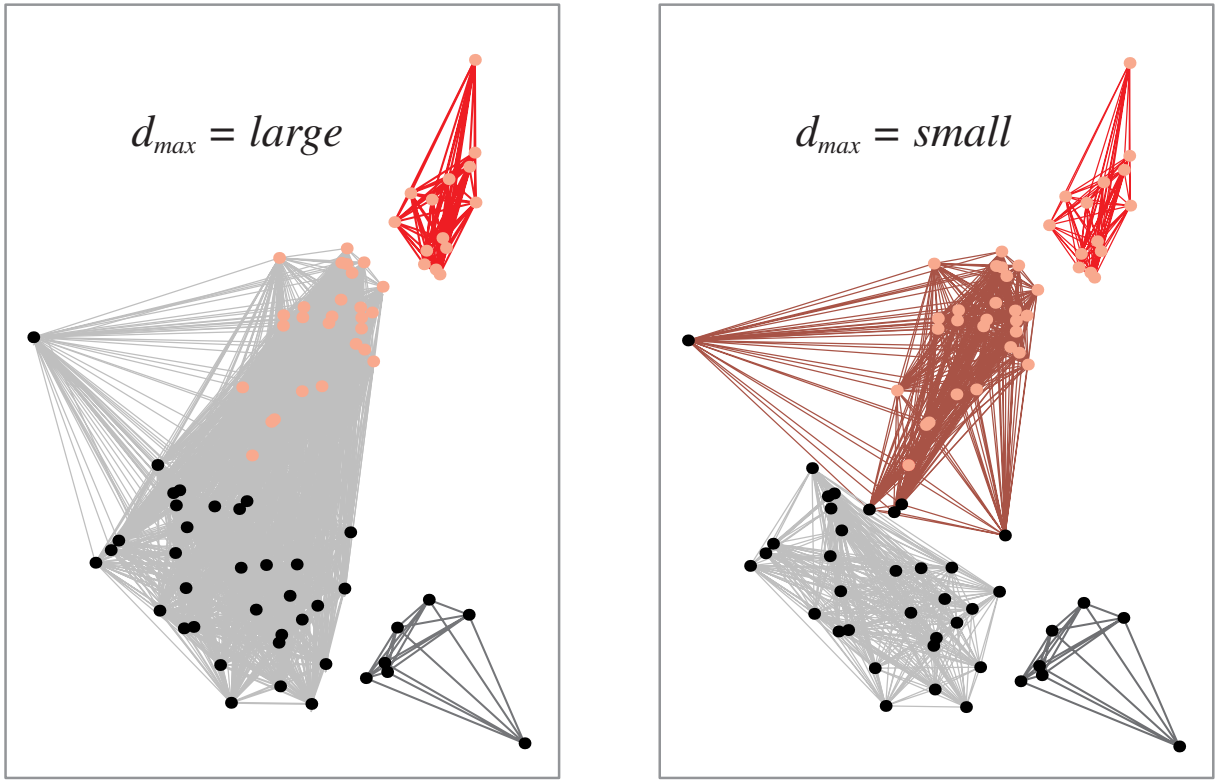


FIGURE 10.14. The farthest-neighbor clustering algorithm uses the separation between the most distant points as a criterion for cluster membership. If this distance is set very large, then all points lie in the same cluster. In the case shown at the left, a fairly large d_{max} leads to three clusters; a smaller d_{max} gives four clusters, as shown at the right. From: Richard O. Duda, Peter E. Hart, and David G. Stork, *Pattern Classification*. Copyright © 2001 by John Wiley & Sons, Inc.

Using Mean, Avg Distances

$$d_{avg}(D_i, D_j) = \frac{1}{n_i n_j} \sum_{x \in D_i} \sum_{x' \in D_j} ||x - x'||$$

$$d_{mean}(D_i, D_j) = ||m_i - m_j||$$

Reduces Sensitivity to Outliers

Mean less expensive to compute than avg, min, max (each require n_i * n_j distances)

Stepwise Optimal Hierarchical Clustering

Problem

None of the agglomerative methods discussed so far directly minimize a specific criterion function

Modified Agglomerative Algorithm:

For k = 1 to (n - c + 1)

- Find clusters whose merger changes criterion least, D_i and D_j
- Merge D_i and D_j

Example: Minimal increase in SSE (J_e)

$$d_e(D_i, D_j) = \sqrt{\frac{n_i n_j}{n_i + n_j}} ||m_i - m_j||$$

d_e defines the cluster pair that increases J_e as little as possible. May not minimize SSE, but often good starting point

- prefers merging single elements or small with large clusters vs. merging medium-size clusters

k-Means Clustering

k-Means Algorithm

For a number of clusters k:

1. Choose k data points at random
2. Assign all data points to closest of the k cluster centers
3. Re-compute k cluster centers as the mean vector of each cluster
   - If cluster centers do not change, stop
   - Else, goto 2

Complexity

O(ndcT) - T iterations, d features, n points, c clusters, in practice usually T << n (much fewer than n iterations)

Note: means tend to move minimizing squared error criterion

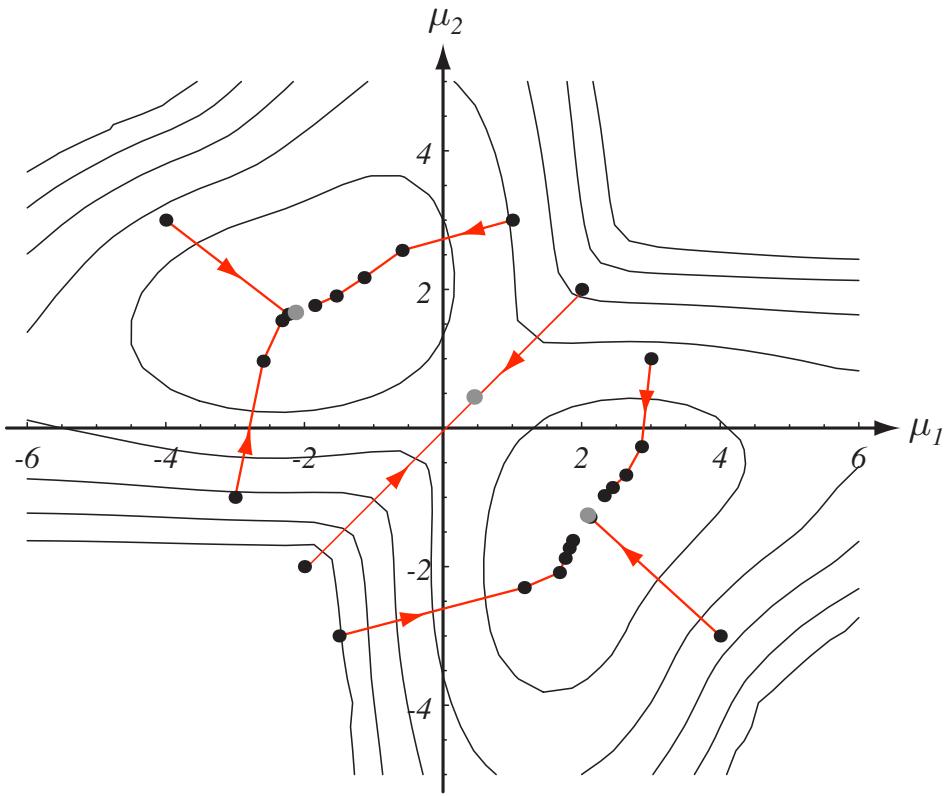


FIGURE 10.2. The *k*-means clustering procedure is a form of stochastic hill climbing in the log-likelihood function. The contours represent equal log-likelihood values for the one-dimensional data in Fig. 10.1. The dots indicate parameter values after different iterations of the *k*-means algorithm. Six of the starting points shown lead to local maxima, whereas two (i.e., $\mu_1(0) = \mu_2(0)$) lead to a saddle point near $\boldsymbol{\mu} = \mathbf{0}$. From: Richard O. Duda, Peter E. Hart, and David G. Stork, *Pattern Classification*. Copyright © 2001 by John Wiley & Sons, Inc.

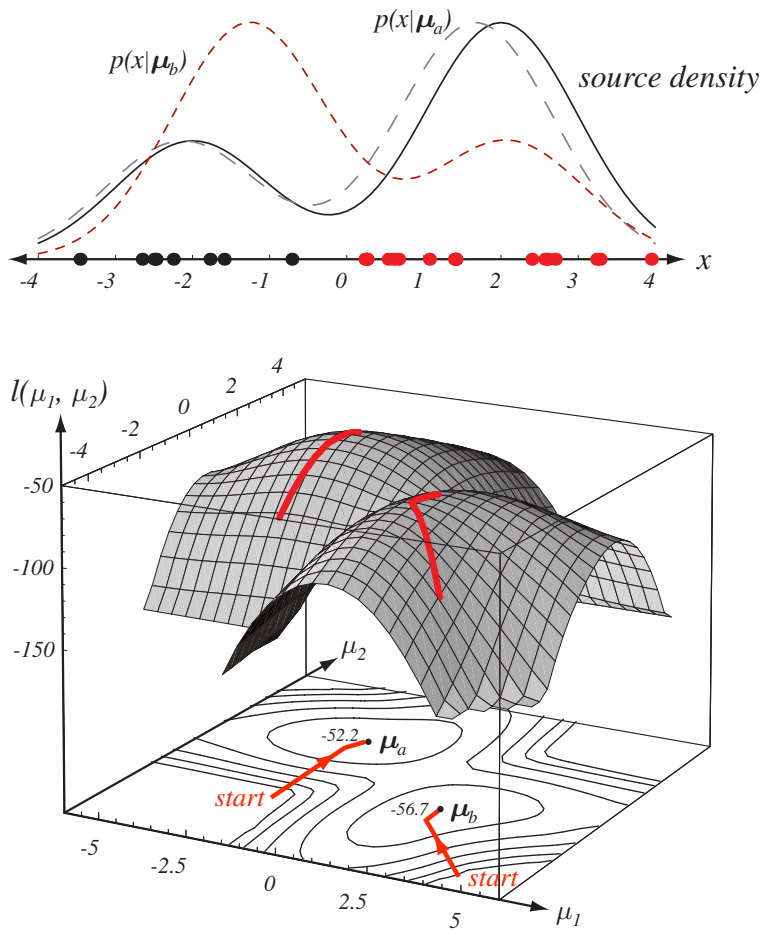


FIGURE 10.1. (Above) The source mixture density used to generate sample data, and two maximum-likelihood estimates based on the data in the table. (Bottom) Log-likelihood of a mixture model consisting of two univariate Gaussians as a function of their means, for the data in the table. Trajectories for the iterative maximum-likelihood estimation of the means of a two-Gaussian mixture model based on the data are shown as red lines. Two local optima (with log-likelihoods −52.2 and −56.7) correspond to the two density estimates shown above. From: Richard O. Duda, Peter E. Hart, and David G. Stork, *Pattern Classification*. Copyright © 2001 by John Wiley & Sons, Inc.

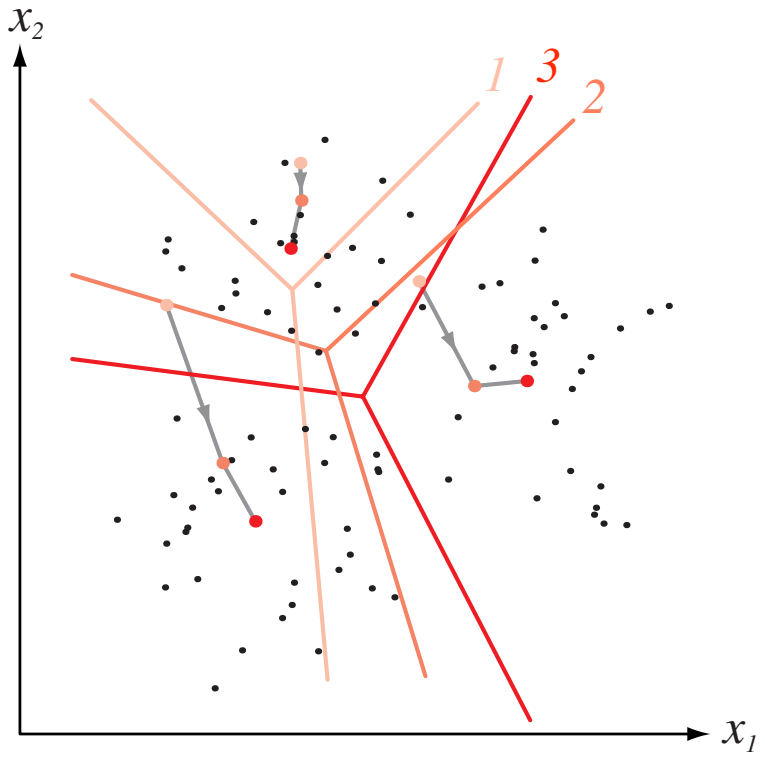


FIGURE 10.3. Trajectories for the means of the *k*-means clustering procedure applied to two-dimensional data. The final Voronoi tesselation (for classification) is also shown—the means correspond to the "centers" of the Voronoi cells. In this case, convergence is obtained in three iterations. From: Richard O. Duda, Peter E. Hart, and David G. Stork, *Pattern Classification*. Copyright © 2001 by John Wiley & Sons, Inc.

Fuzzy k-means

Basic Idea

Allow every point to have a probability of membership in every cluster. The criterion (cost function) minimized is:

$$J_{fuz} = \sum_{i=1}^{c} \sum_{j=1}^{n} [\hat{P}(\omega_i | x_j, \hat{\Theta})]^b ||x_j - \mu_i||^2$$

Theta is the membership function parameter set. *b* ('blending') is a free parameter:

- b = 0: Sum of squared error criterion (one cluster per data point)
- b > 1: each pattern may belong to multiple clusters

Fuzzy k-Mean Clustering Algorithm

$$\mu_j = \frac{\sum_{j=1}^{n}[\hat{P}(\omega_i|x_j)]^b x_j}{\sum_{j=1}^{n}[\hat{P}(\omega_i|x_j)]^b}$$

$$\hat{P}(\omega_i|x_j) = \frac{(1/d_{ij})^{1/(b-1)}}{\sum_{r=1}^{c}(1/d_{rj})^{1/(b-1)}}$$

$$d_{ij} = ||x_j - \mu_i||^2$$

Algorithm

1. Compute probability of each class for every point in the training set (uniform probability: equal likelihood in each cluster)
2. Recompute means using expression at top-left
3. Recompute probability of each class for each point using expression at top right
    - If change in means and membership probabilities for points is small, stop
    - Else goto 2

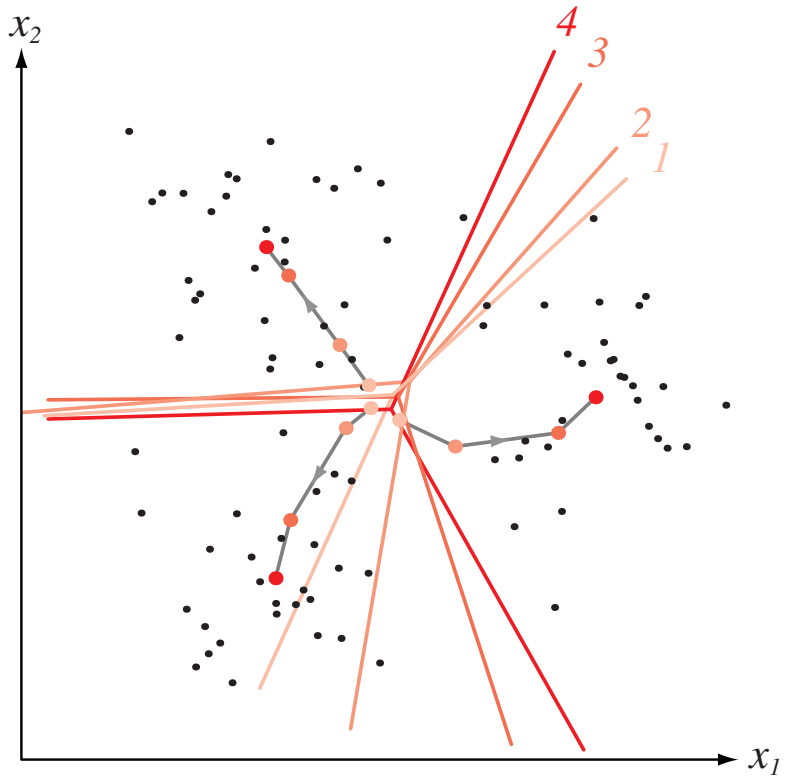


FIGURE 10.4. At each iteration of the fuzzy *k*-means clustering algorithm, the probability of category memberships for each point are adjusted according to Eqs. 32 and 33 (here $b = 2$). While most points have nonnegligible memberships in two or three clusters, we nevertheless draw the boundary of a Voronoi tesselation to illustrate the progress of the algorithm. After four iterations, the algorithm has converged to the red cluster centers and associated Voronoi tesselation. From: Richard O. Duda, Peter E. Hart, and David G. Stork, *Pattern Classification*. Copyright © 2001 by John Wiley & Sons, Inc.

Fuzzy k-means, Cont'd

Convergence Properties

Sometimes fuzzy k-means improves convergence over classical k-means

However, probability of cluster membership depends on the number of clusters; can lead to problems if poor choice of k is made

Cluster Validity

So far...

We've assumed that we know the number of clusters

When number of clusters isn't known

We can try a clustering procedure using c=1, c=2, etc., and making note of sudden decreases in the error criterion (e.g. SSE)

More formal: statistical tests, however problem of testing cluster validity is unsolved

- DHS: Section 10.10 presents a statistical test centered around testing the null hypothesis of having c clusters, by comparing with c+1 clusters