# Chapter 10: Unsupervised Learning and Clustering

- Introduction
- Mixture Densities and Identifiability
- ML Estimates
- Application to Normal Mixtures

# I Introduction

- Previously, all our training samples were labeled: these samples were said “supervised”
- We now investigate a number of “unsupervised” procedures which use unlabeled samples
- Collecting and Labeling a large set of sample patterns can be costly
- We can train with large amounts of (less expensive) unlabeled data, and only then use supervision to label the groupings found, this is appropriate for large “data mining” applications where the contents of a large database are not known beforehand

- **This is also appropriate in many applications when the characteristics of the patterns can change slowly with time**

- **Improved performance can be achieved if classifiers running in a unsupervised mode are used**

- **We can use unsupervised methods to identify features that will then be useful for categorization**

- **We gain some insight into the nature (or structure) of the data**

# l Mixture Densities and Identifiability

– **We shall begin with the assumption that the functional forms for the underlying probability densities are known and that the only thing that must be learned is the value of an unknown parameter vector**

– **We make the following assumptions:**

- **The samples come from a known number c of classes**
- **The prior probabilities P($w_j$) for each class are known ($j = 1, \ldots, c$)**
- **P($x \mid w_j, q_j$) ($j = 1, \ldots, c$) are known**
- **The values of the c parameter vectors $q_1, q_2, \ldots, q_c$ are unknown**

- **The category labels are unknown**

component densities

$$P(x \mid \theta) = \sum_{j=1}^{c} P(x \mid \omega_j, \theta_j) \cdot P(\omega_j)$$

mixing parameters

**where** $\theta = (\theta_1, \theta_2, \ldots, \theta_c)^t$

– **This density function is called a mixture density**

– **Our goal will be to use samples drawn from this mixture density to estimate the unknown parameter vector** $\theta$**. Once** $\theta$ **is known, we can decompose the mixture into its components and use a MAP classifier on the derived densities**

## – Definition

- **A density P(x | $\theta$) is said to be identifiable if**

  **$\theta \neq \theta'$ implies that there exists an x such that:**

$$P(x \mid \theta) \neq P(x \mid \theta')$$

**As a simple example, consider the case where x is binary and P(x | $\theta$) is the mixture:**

$$P(x \mid \theta) = \frac{1}{2}\theta_1^x(1-\theta_1)^{1-x} + \frac{1}{2}\theta_2^x(1-\theta_2)^{1-x}$$

$$= \begin{cases} \frac{1}{2}(\theta_1+\theta_2) & \text{if } x = 1 \\ 1-\frac{1}{2}(\theta_1+\theta_2) & \text{if } x = 0 \end{cases}$$

**Assume that:**

$$P(x = 1 \mid \theta) = 0.6 \Rightarrow P(x = 0 \mid \theta) = 0.4$$

**by replacing these probabilities values, we obtain:**

$$\theta_1 + \theta_2 = 1.2$$

- **Thus, we have a case in which the mixture distribution is completely unidentifiable, and therefore unsupervised learning is impossible**

- **In the discrete distributions, if there are too many components in the mixture, there may be more unknowns than independent equations, and identifiability can become a serious problem!**

- **While it can be shown that mixtures of normal densities are usually identifiable, the parameters in the simple mixture density**

$$P(x \mid \theta) = \frac{P(\omega_1)}{\sqrt{2\pi}} \exp\left[-\frac{1}{2}(x - \theta_1)^2\right] + \frac{P(\omega_2)}{\sqrt{2\pi}} \exp\left[-\frac{1}{2}(x - \theta_2)^2\right]$$

**Cannot be uniquely identified if $P(\omega_1) = P(\omega_2)$**

**(we cannot recover a unique $\theta$ even from an infinite amount of data!)**

- **$q = (q_1, q_2)$ and $q = (q_2, q_1)$ are two possible vectors that can be interchanged without affecting P(x | $q$)**

- **Identifiability can be a problem, we always assume that the densities we are dealing with are identifiable!**

# ML Estimates

- Suppose that we have a set $D = \{x_1, \ldots, x_n\}$ of n unlabeled samples drawn independently from the mixture density

$$p(x \mid \theta) = \sum_{j=1}^{c} p(x \mid \omega_j, \theta_j) P(\omega_j)$$

($\theta$ is fixed but unknown!)

$$\hat{\theta} = \arg\max_{\theta} p(D \mid \theta) \text{ with } p(D \mid \theta) = \prod_{k=1}^{n} p(x_k \mid \theta)$$

The gradient of the log-likelihood is:

$$\nabla_{\theta_i} l = \sum_{k=1}^{n} P(\omega_i \mid x_k, \theta) \nabla_{\theta_i} \ln p(x_k \mid \omega_i, \theta_i)$$

- **Use these relations:**

$$l = \sum_{k=1}^{n} \ln p(x_k | q)$$

$$p(w_i \mid x_k, q) = \frac{p(x_k | w_i, q_i)\, p(w_i)}{p(x_k | q)}$$

- **And assume that the elements of $\theta_i$ and $\theta_j$ are functionally independent if $i \neq j$**

Since the gradient must vanish at the value of $\theta_i$ that maximizes $l$ ($l = \sum_{k=1}^{n} \ln p(x_k \mid \theta)$) therefore, the ML estimate $\hat{\theta}_i$ must satisfy the conditions

$$\sum_{k=1}^{n} P(\omega_i \mid x_k, \hat{\theta}) \tilde{\nabla}_{\theta_i} \ln p(x_k \mid \omega_i, \hat{\theta}_i) = 0 \quad (i = 1, \ldots, c)$$

By including the prior probabilities as unknown variables, we finally obtain:

$$\hat{P}(\omega_i) = \frac{1}{n} \sum \hat{P}(\omega_i \mid x_k, \hat{\theta})$$

and $$\sum_{k=1}^{n} \hat{P}(\omega_i \mid x_k, \hat{\theta}) \tilde{\nabla}_{\theta_i} \ln p(x_k \mid \omega_i, \hat{\theta}_i) = 0$$

where : $$\hat{P}(\omega_i \mid x_k, \hat{\theta}) = \frac{p(x_k \mid \omega_i, \hat{\theta}_i)\hat{P}(\omega_i)}{\sum_{j=1}^{c} p(x_k \mid \omega_j, \hat{\theta}_j)\hat{P}(\omega_j)}$$

# Applications to Normal Mixtures

$$p(x \mid \omega_i, \theta_i) \sim N(\mu_i, \Sigma_i)$$

| Case | $\mu_i$ | $\Sigma_i$ | $P(\omega_i)$ | c |
|---|---|---|---|---|
| 1 | ? | x | x | x |
| 2 | ? | ? | ? | x |
| 3 | ? | ? | ? | ? |

**Case 1 = Simplest case**

– **Case 1: Unknown mean vectors**

$$\mu_i = \theta_i \quad \forall\ i = 1, \ldots, c$$

$$\ln p(x \mid \omega_i, \mu_i) = -\ln\left[(2\pi)^{d/2} \left|\Sigma_i\right|^{1/2}\right] - \frac{1}{2}(x - \mu_i)^t \Sigma_i^{-1}(x - \mu_i)$$

**ML estimate of $\mu = (\mu_i)$ is:**

$$\hat{\mu}_i = \frac{\sum_{k=1}^{n} P(\omega_i \mid x_k, \hat{\mu}) x_k}{\sum_{k=1}^{n} P(\omega_i \mid x_k, \hat{\mu})} \qquad (1)$$

$P(\omega_i \mid x_k, \hat{\mu})$ **is the fraction of those samples**

**having value $x_k$ that come from the ith class, and $\hat{\mu}_i$ is the average of the samples coming from the ith class.**

- **Unfortunately, equation (1) does not give $\hat{\mu}_i$ explicitly**

- **However, if we have some way of obtaining good initial estimates $\hat{\mu}_i(0)$ for the unknown means, therefore equation (1) can be seen as an iterative process for improving the estimates**

$$\hat{\mu}_i(j+1) = \frac{\sum_{k=1}^{n} P(\omega_i \mid x_k, \hat{\mu}(j))x_k}{\sum_{k=1}^{n} P(\omega_i \mid x_k, \hat{\mu}(j))}$$

- **This is a gradient ascent for maximizing the log-likelihood function**
- **Example:**

  **Consider the simple two-component one-dimensional normal mixture**

$$p(x \mid \mu_1, \mu_2) = \frac{1}{3\sqrt{2\pi}} \exp\left[-\frac{1}{2}(x-\mu_1)^2\right] + \frac{2}{3\sqrt{2\pi}} \exp\left[-\frac{1}{2}(x-\mu_2)^2\right]$$

  **(2 clusters!)**

  **Let's set $\mu_1 = -2$, $\mu_2 = 2$ and draw 25 samples sequentially from this mixture. The log-likelihood function is:**

$$l(\mu_1, \mu_2) = \sum_{k=1}^{n} \ln p(x_k \mid \mu_1, \mu_2)$$

**The maximum value of l occurs at:**

$$\hat{m}_1 = -2.130 \text{ and } \hat{m}_2 = 1.668$$

**(which are not far from the true values: $m_1$ = -2 and $m_2$ = +2)**

**There is another peak at** $\hat{m}_1 = 2.085$ **and** $\hat{m}_2 = -1.257$ **which has almost the same height as can be seen from the following figure:**

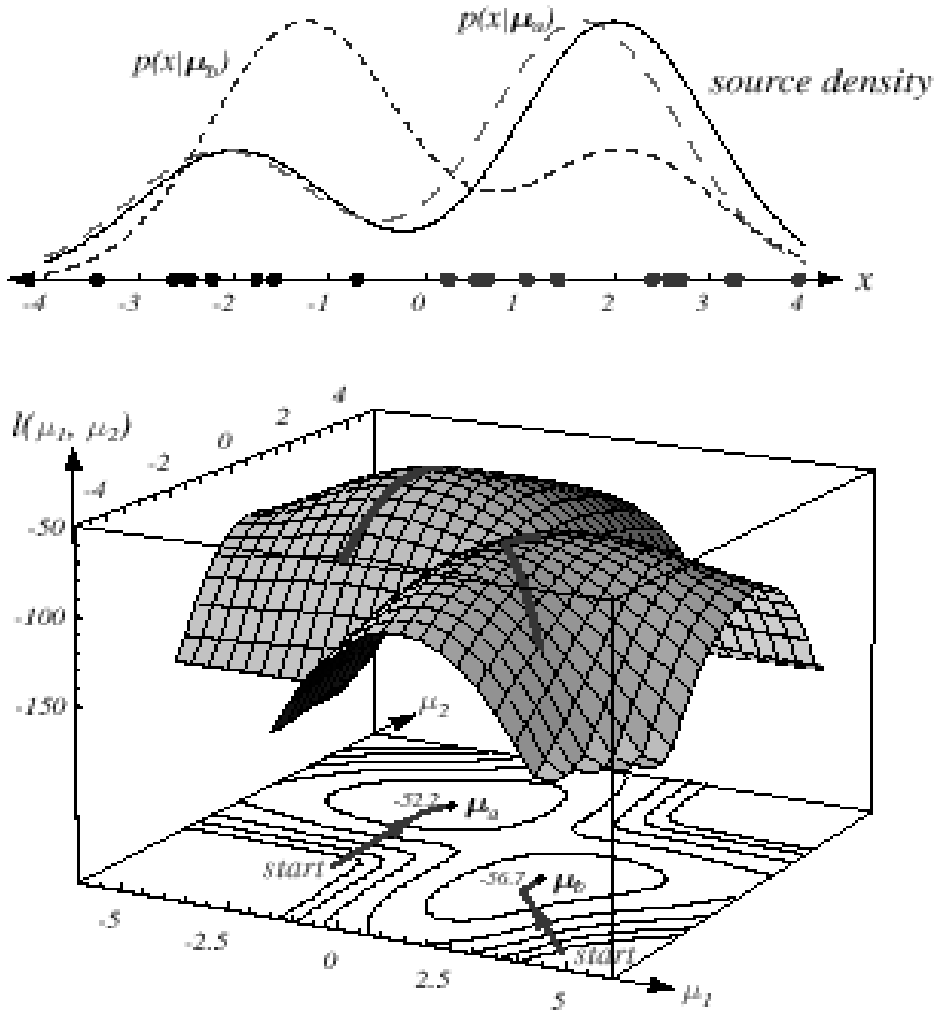



**FIGURE 10.1.** (Above) The source mixture density used to generate sample data, and two maximum-likelihood estimates based on the data in the table. (Bottom) Log-likelihood of a mixture model consisting of two univariate Gaussians as a function of their means, for the data in the table. Trajectories 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.

– **Case 2: All parameters unknown**

- **Adding an assumption**

  **Consider the largest of the finite local maxima of the likelihood function and use the ML estimation.**

  **We obtain the following:**

Iterative scheme

$$\hat{P}(\omega_i) = \frac{1}{n}\sum_{k=1}^{n} \hat{P}(\omega_i \mid x_k, \hat{\theta})$$

$$\hat{\mu}_i = \frac{\sum_{k=1}^{n} \hat{P}(\omega_i \mid x_k, \hat{\theta})\, x_k}{\sum_{k=1}^{n} \hat{P}(\omega_i \mid x_k, \hat{\theta})}$$

$$\hat{\Sigma}_i = \frac{\sum_{k=1}^{n} \hat{P}(\omega_i \mid x_k, \hat{\theta})(x_k - \hat{\mu}_i)(x_k - \hat{\mu}_i)^t}{\sum_{k=1}^{n} \hat{P}(\omega_i \mid x_k, \hat{\theta})}$$

**Where:**

$$\hat{P}(\omega_i \mid x_k, \hat{\theta}) = \frac{|S_i|^{-1/2} \exp\left[-\frac{1}{2}(x_k - \hat{\mu}_i)^t \hat{S}_i^{-1}(x_k - \hat{\mu}_i)\right]\hat{P}(\omega_i)}{\sum_{j=1}^{c} |\hat{S}_j|^{-1/2} \exp\left[-\frac{1}{2}(x_k - \hat{\mu}_j)^t \hat{S}_j^{-1}(x_k - \hat{\mu}_j)\right]\hat{P}(\omega_j)}$$

– K-Means Clustering

- **Goal: find the c mean vectors $\mu_1, \mu_2, \ldots, \mu_c$**
- **Replace the squared Mahalanobis distance**

$$(x_k - \hat{\mu}_i)^t \hat{S}_i^{-1}(x_k - \hat{\mu}_i) \text{ by the squared Euclidean distance } \|x_k - \hat{\mu}_i\|^2$$

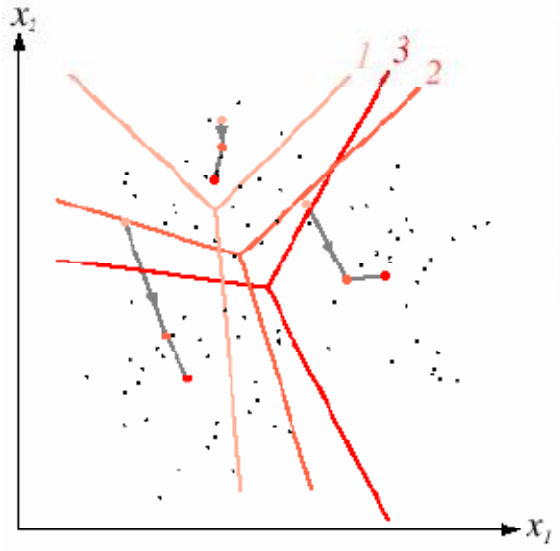
- **Find the mean $\hat{\mu}_m$ nearest to $x_k$ and approximate $\hat{P}(\omega_i \mid x_k, \hat{\theta})$ as:**

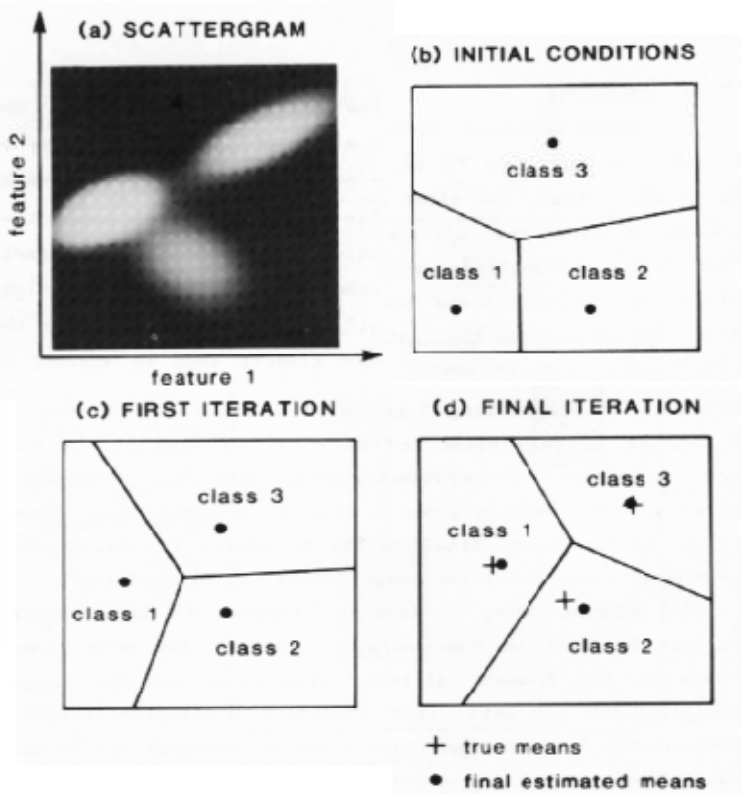
$$\hat{P}(\omega_i \mid x_k, \theta) \approx \begin{cases} 1 & \text{if } i = m \\ 0 & \text{otherwise} \end{cases}$$

- **Use the iterative scheme to find** $\hat{m}_1, \hat{m}_2, \ldots, \hat{m}_c$
- **if n is the known number of patterns and c the desired number of clusters, the k-means algorithm is:**

```
Begin
    initialize n, c, m1, m2, …, mc(randomly
    selected)
          do classify n samples according to
                nearest mi
                recompute mi
          until no change in mI
    return m1, m2, …, mc
End
```

# k-means clustering



# Choice of model

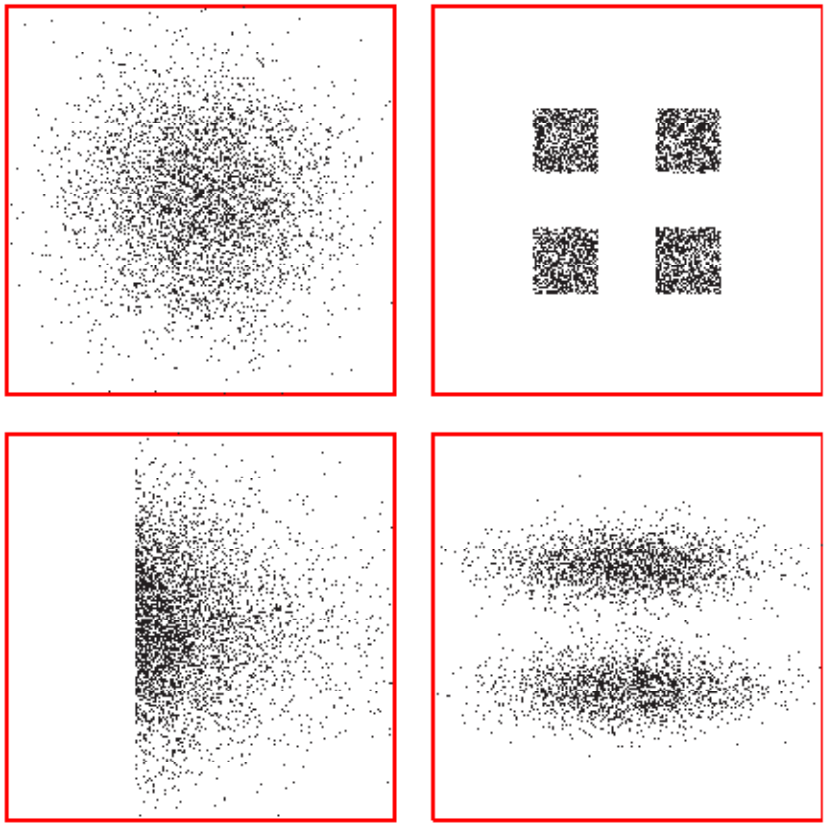


**FIGURE 10.6.** These four data sets have identical statistics up to second-order—that is, the same mean $\boldsymbol{\mu}$ and covariance $\boldsymbol{\Sigma}$. In such cases it is important to include in the model more parameters to represent the structure more completely. From: Richard O. Duda, Peter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

# Similarity measurement (distance)

- **Problem of finding best thresholds**

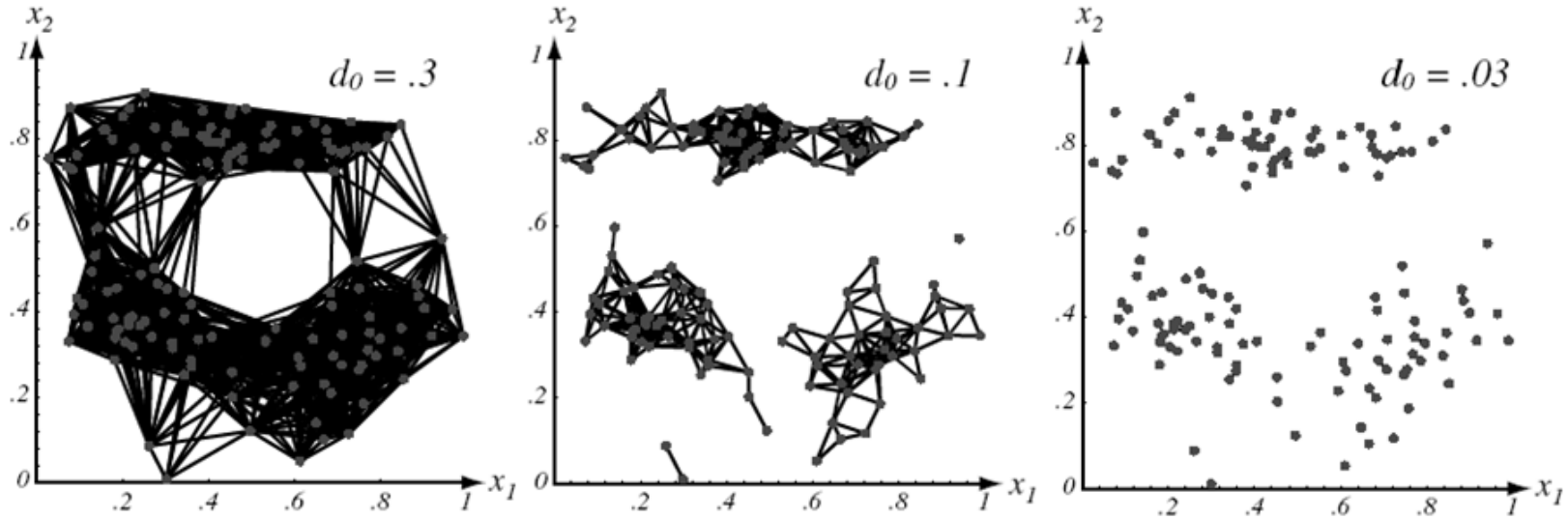



**FIGURE 10.7.** The distance threshold affects the number and size of clusters in similarity based clustering methods. For three different values of distance $d_0$, lines are drawn between points closer than $d_0$—the smaller the value of $d_0$, the smaller and more numerous the clusters. From: Richard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

# Hierarchical method

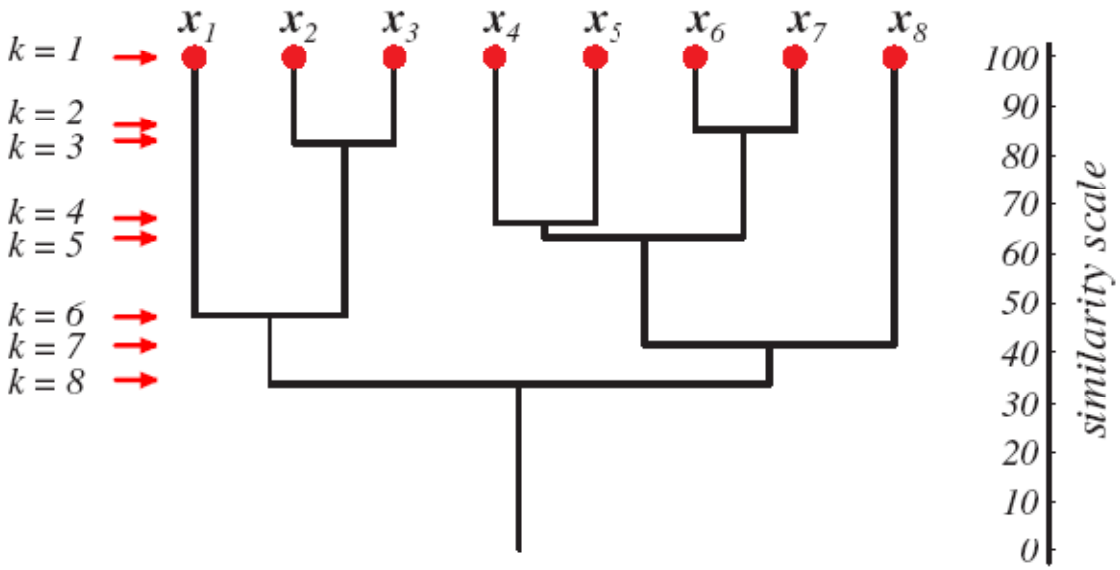



**FIGURE 10.11.** A dendrogram can represent the results of hierarchical clustering algorithms. The vertical axis shows a generalized measure of similarity among clusters. Here, at level 1 all eight points lie in singleton clusters; each point in a cluster is highly similar to itself, of course. Points $\mathbf{x}_6$ 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.

# Independent Component Analysis (ICA)

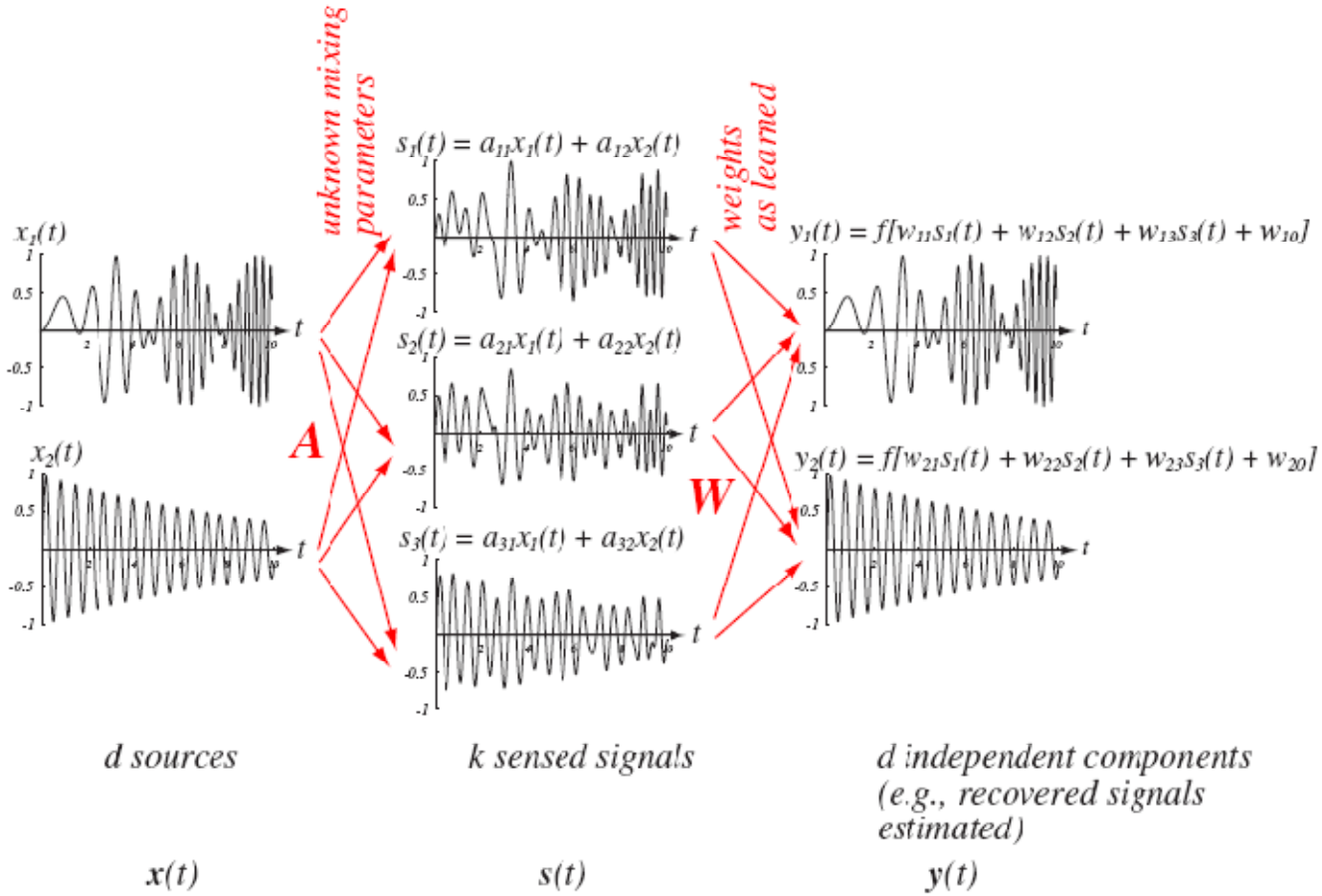



**FIGURE 10.25.** Independent component analysis (ICA) is an unsupervised method that can be applied to the problem of blind source separation. In such problems, two or more source signals (assumed independent) $x_1(t), x_2(t), \cdots, x_d(t)$ are mixed linearly to yield sum signals $s_1(t), s_2(t), \cdots, s_k(t)$, where $k \geq d$. (This figure illustrates the case $d = 2$ and $k = 3$.) Given merely the sensed signals $\mathbf{x}(t)$ and an assumed number of components, $d$, the task of ICA is to find independent components in $\mathbf{s}$. In a blind source separation application, these are merely the source signals. From: Richard O. Duda, Peter E. Hart, and David G. Stork, *Pattern Classification*. Copyright © 2001 by John Wiley & Sons, Inc.

# Tree proceeds

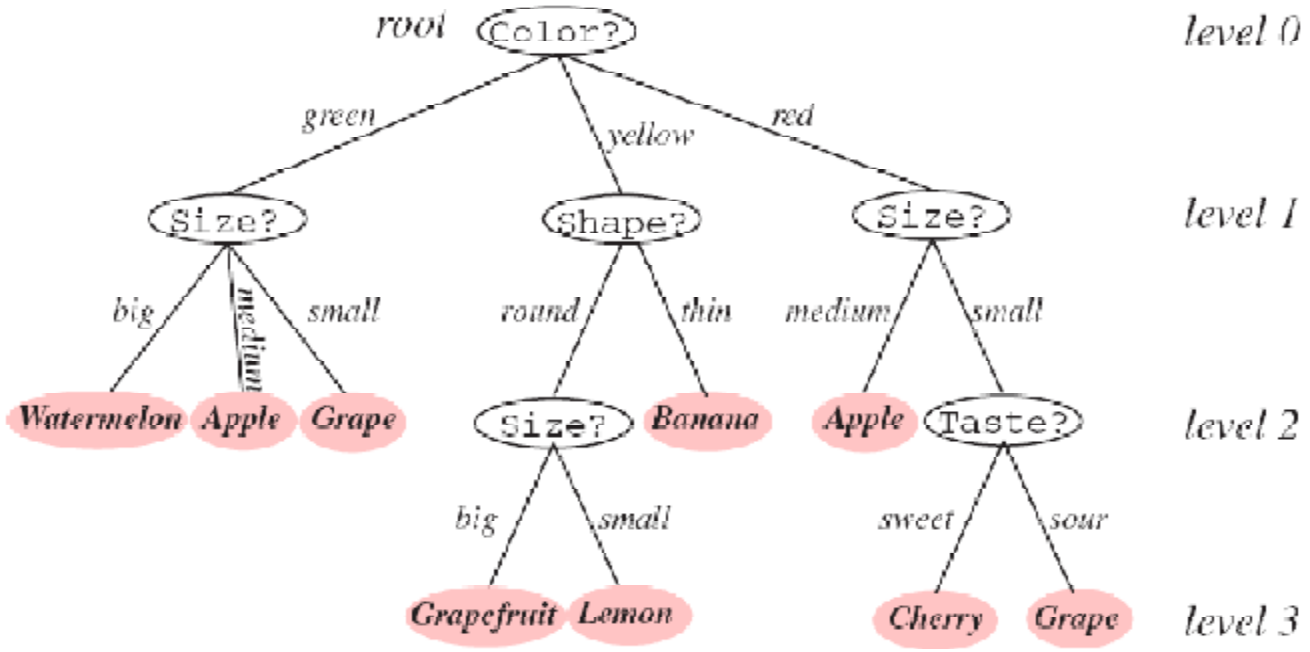



**FIGURE 8.1.** Classification in a basic decision tree proceeds from top to bottom. The questions asked at each node concern a particular property of the pattern, and the downward links correspond to the possible values. Successive nodes are visited until a terminal or leaf node is reached, where the category label is read. Note that the same question, `Size?`, appears in different places in the tree and that different questions can have different numbers of branches. Moreover, different leaf nodes, shown in pink, can be labeled by the same category (e.g., **Apple**). From: Richard O. Duda, Peter E. Hart, and David G. Stork, *Pattern Classification*. Copyright © 2001 by John Wiley & Sons, Inc.

# Tree proceeds

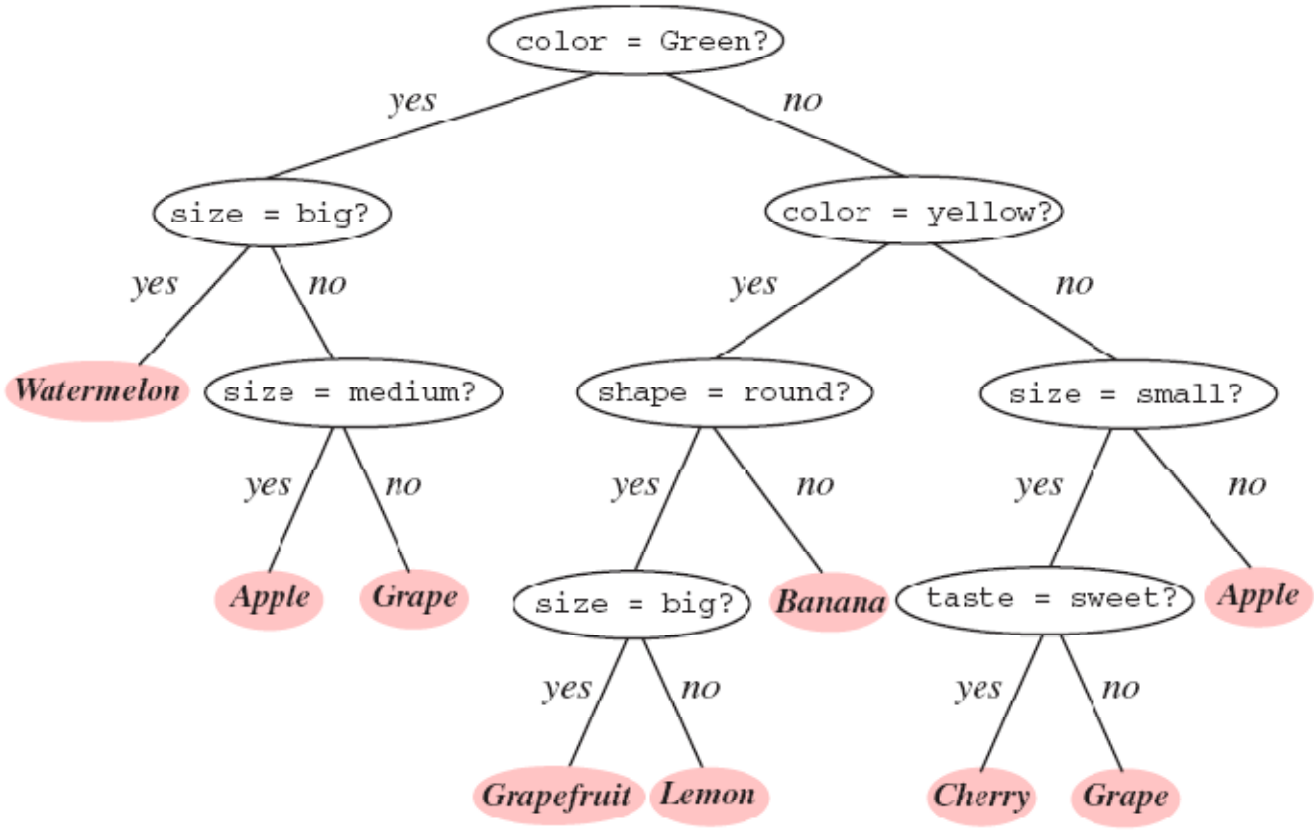



**FIGURE 8.2.** A tree with arbitrary branching factor at different nodes can always be represented by a functionally equivalent binary tree—that is, one having branching factor $B = 2$ throughout, as shown here. By convention the "yes" branch is on the left, the "no" branch on the right. This binary tree contains the same information and implements the same classification as that in Fig. 8.1. From: Richard O. Duda, Peter E. Hart, and David G. Stork, *Pattern Classification*. Copyright © 2001 by John Wiley & Sons, Inc.

# Tree proceeds

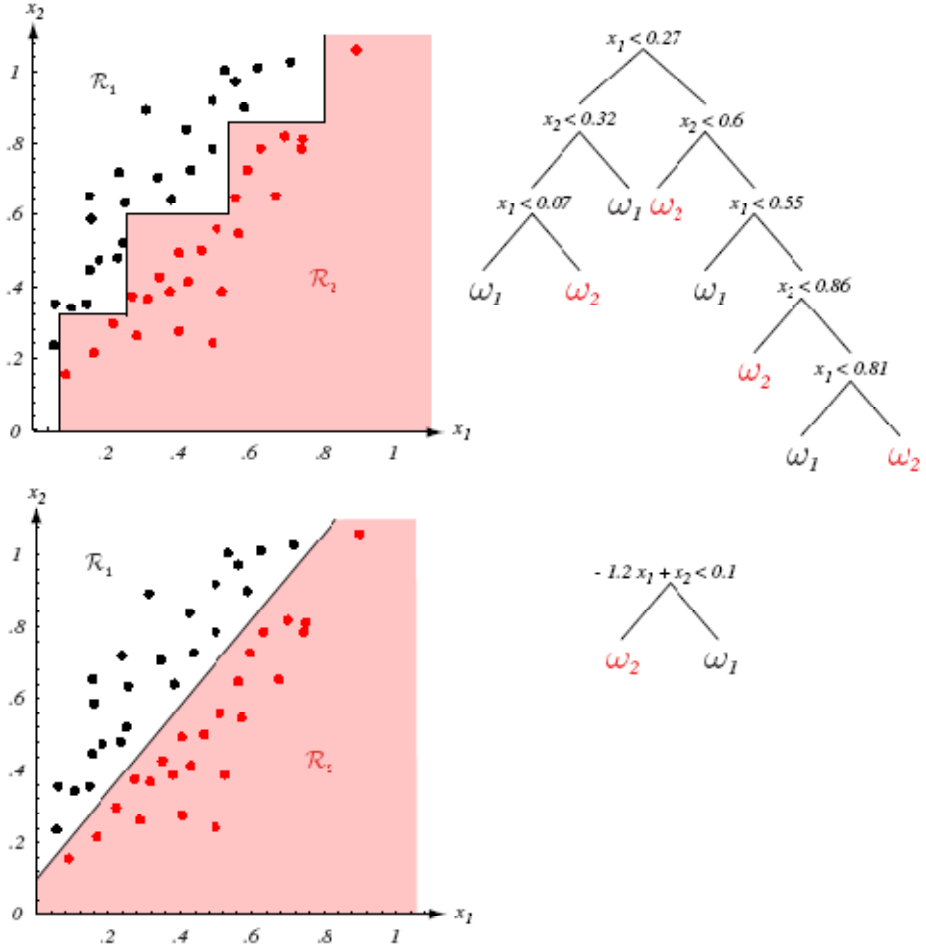



**FIGURE 8.5.** If the class of node decisions does not match the form of the training data, a very complicated decision tree will result, as shown at the top. Here decisions are parallel to the axes while in fact the data is better split by boundaries along another direction. If, however, "proper" decision forms are used (here, linear combinations of the features), the tree can be quite simple, as shown at the bottom. From: Richard O. Duda, Peter E. Hart, and David G. Stork, *Pattern Classification*. Copyright © 2001 by John Wiley & Sons, Inc.

# Tree proceeds

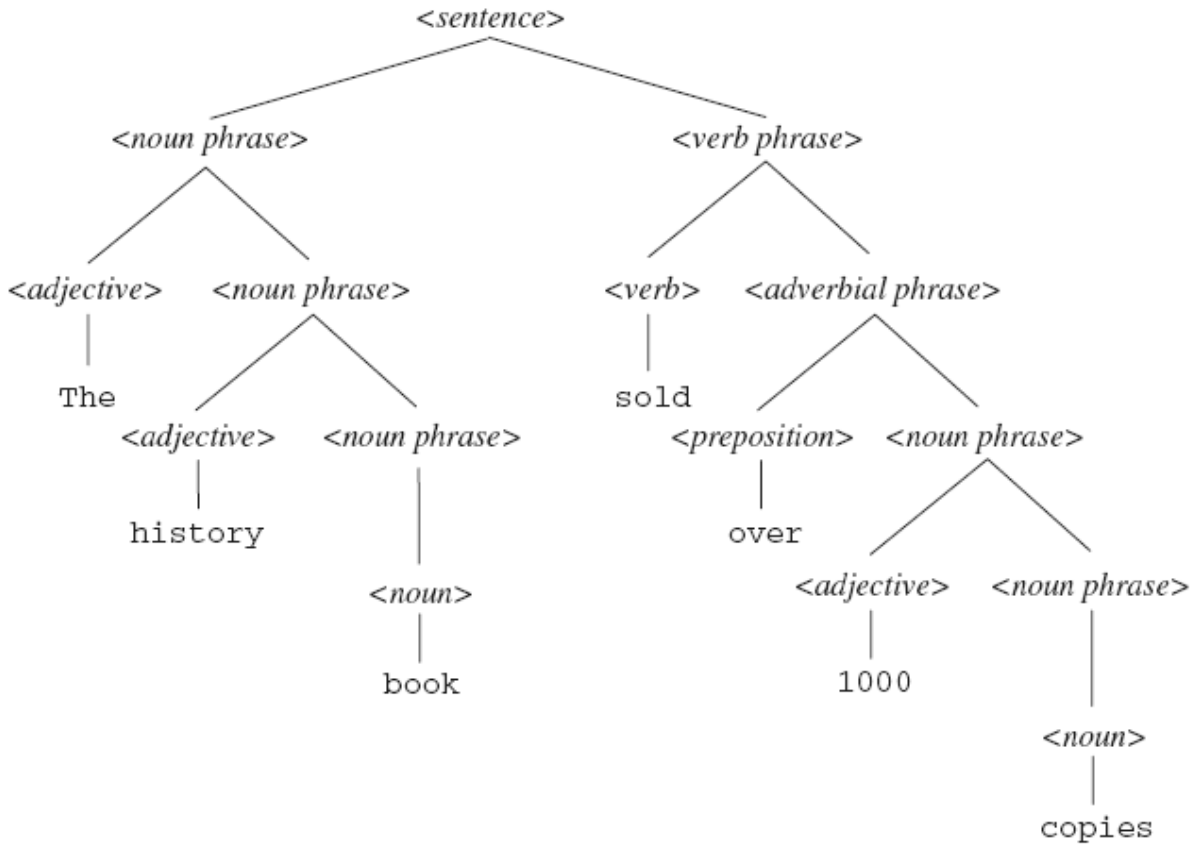



**FIGURE 8.12.** This derivation tree illustrates how a portion of English grammar can transform the root symbol, here ⟨*sentence*⟩, into a particular sentence or string of elements, here English words, which are read from left to right. From: Richard O. Duda, Peter E. Hart, and David G. Stork, *Pattern Classification*. Copyright © 2001 by John Wiley & Sons, Inc.