Contents

Chapter 3

Maximum likelihood and Bayesian parameter estimation

3.1 Introduction

In Chap. ?? we saw how we could design an optimal classifier if we knew the prior probabilities $P(\omega_i)$ and the class-conditional densities $p(\mathbf{x}|\omega_i)$. Unfortunately, in pattern recognition applications we rarely if ever have this kind of complete knowledge about the probabilistic structure of the problem. In a typical case we merely have some vague, general knowledge about the situation, together with a number of *design samples* or *training data* — particular representatives of the patterns we want to classify. The problem, then, is to find some way to use this information to design or train the classifier.

TRAINING DATA

One approach to this problem is to use the samples to estimate the unknown probabilities and probability densities, and to use the resulting estimates as if they were the true values. In typical supervised pattern classification problems, the estimation of the prior probabilities presents no serious difficulties (Problem 3). However, estimation of the class-conditional densities is quite another matter. The number of available samples always seems too small, and serious problems arise when the dimensionality of the feature vector $\mathbf{x}$ is large. If we know the number of parameters in advance and our general knowledge about the problem permits us to parameterize the conditional densities, then the severity of these problems can be reduced significantly. Suppose, for example, that we can reasonably assume that $p(\mathbf{x}|\omega_i)$ is a normal density with mean $\boldsymbol{\mu}_i$ and covariance matrix $\boldsymbol{\Sigma}_i$, although we do not know the exact values of these quantities. This knowledge simplifies the problem from one of estimating an unknown *function* $p(\mathbf{x}|\omega_i)$ to one of estimating the *parameters* $\boldsymbol{\mu}_i$ and $\boldsymbol{\Sigma}_i$.

The problem of parameter estimation is a classical one in statistics, and it can be approached in several ways. We shall consider two common and reasonable procedures, *maximum likelihood* estimation and *Bayesian* estimation. Although the results obtained with these two procedures are frequently nearly identical, the approaches

MAXIMUM LIKELIHOOD

BAYESIAN ESTIMATION

are conceptually quite different. Maximum likelihood and several other methods view the parameters as quantities whose values are fixed but unknown. The best estimate of their value is defined to be the one that maximizes the probability of obtaining the samples actually observed. In contrast, Bayesian methods view the parameters as random variables having some known a priori distribution. Observation of the samples converts this to a posterior density, thereby revising our opinion about the true values of the parameters. In the Bayesian case, we shall see that a typical effect of observing additional samples is to sharpen the a posteriori density function, causing it to peak near the true values of the parameters. This phenomenon is known as *Bayesian learning*. In either case, we use the posterior densities for our classification rule, as we have seen before.

BAYESIAN LEARNING

It is important to distinguish between supervised learning and unsupervised learning. In both cases, samples $\mathbf{x}$ are assumed to be obtained by selecting a state of nature ω_i with probability $P(\omega_i)$, and then independently selecting $\mathbf{x}$ according to the probability law $p(\mathbf{x}|\omega_i)$. The distinction is that with supervised learning we know the state of nature (class label) for each sample, whereas with unsupervised learning we do not. As one would expect, the problem of unsupervised learning is the more difficult one. In this chapter we shall consider only the supervised case, deferring consideration of unsupervised learning to Chap. **??**.

3.2 Maximum Likelihood Estimation

Maximum likelihood estimation methods have a number of attractive attributes. First, they nearly always have good convergence properties as the number of training samples increases. Further, maximum likelihood estimation often can be simpler than alternate methods, such as Bayesian techniques or other methods presented in subsequent chapters.

3.2.1 The General Principle

Suppose that we separate a collection of samples according to class, so that we have c sets, $\mathcal{D}_1, ..., \mathcal{D}_c$, with the samples in $\mathcal{D}_j$ having been drawn independently according to the probability law $p(\mathbf{x}|\omega_j)$. We say such samples are *i.i.d.* — independent identically distributed random variables. We assume that $p(\mathbf{x}|\omega_j)$ has a known parametric form, and is therefore determined uniquely by the value of a parameter vector $\boldsymbol{\theta}_j$. For example, we might have $p(\mathbf{x}|\omega_j) \sim N(\boldsymbol{\mu}_j, \boldsymbol{\Sigma}_j)$, where $\boldsymbol{\theta}_j$ consists of the components of $\boldsymbol{\mu}_j$ and $\boldsymbol{\Sigma}_j$. To show the dependence of $p(\mathbf{x}|\omega_j)$ on $\boldsymbol{\theta}_j$ explicitly, we write $p(\mathbf{x}|\omega_j)$ as $p(\mathbf{x}|\omega_j, \boldsymbol{\theta}_j)$. Our problem is to use the information provided by the training samples to obtain good estimates for the unknown parameter vectors $\boldsymbol{\theta}_1, ..., \boldsymbol{\theta}_c$ associated with each category.

I.I.D.

To simplify treatment of this problem, we shall assume that samples in $\mathcal{D}_i$ give no information about $\boldsymbol{\theta}_j$ if $i \neq j$ — that is, we shall assume that the parameters for the different classes are functionally independent. This permits us to work with each class separately, and to simplify our notation by deleting indications of class distinctions. With this assumption we thus have c separate problems of the following form: Use a set $\mathcal{D}$ of training samples drawn independently from the probability density $p(\mathbf{x}|\boldsymbol{\theta})$ to estimate the unknown parameter vector $\boldsymbol{\theta}$.

Suppose that $\mathcal{D}$ contains n samples, $\mathbf{x}_1, ..., \mathbf{x}_n$. Then, since the samples were drawn independently, we have

$$p(\mathcal{D}|\boldsymbol{\theta}) = \prod_{k=1}^{n} p(\mathbf{x}_k|\boldsymbol{\theta}). \tag{1}$$

Recall from Chap. **??** that, viewed as a function of $\boldsymbol{\theta}$, $p(\mathcal{D}|\boldsymbol{\theta})$ is called the *likelihood* of $\boldsymbol{\theta}$ with respect to the set of samples. The *maximum likelihood estimate* of $\boldsymbol{\theta}$ is, by definition, the value $\hat{\boldsymbol{\theta}}$ that maximizes $p(\mathcal{D}|\boldsymbol{\theta})$. Intuitively, this estimate corresponds to the value of $\boldsymbol{\theta}$ that in some sense best agrees with or supports the actually observed training samples (Fig. 3.1).

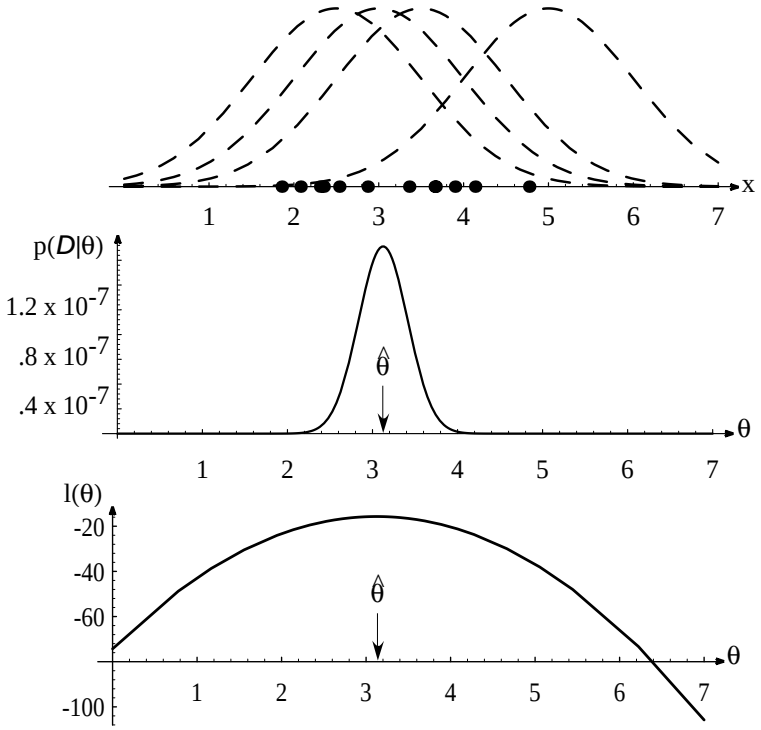


Figure 3.1: The top graph shows several training points in one dimension, known or assumed to be drawn from a Gaussian of a particular variance, but unknown mean. Four of the infinite number of candidate source distributions are shown in dashed lines. The middle figures shows the likelihood $p(\mathcal{D}|\theta)$ as a function of the mean. If we had a very large number of training points, this likelihood would be very narrow. The value that maximizes the likelihood is marked $\hat{\theta}$; it also maximizes the logarithm of the likelihood — i.e., the log-likelihood $l(\theta)$, shown at the bottom. Note especially that the likelihood lies in a different space from $p(x|\hat{\theta})$, and the two can have different functional forms.

For analytical purposes, it is usually easier to work with the logarithm of the likelihood than with the likelihood itself. Since the logarithm is monotonically increasing, the $\hat{\boldsymbol{\theta}}$ that maximizes the log-likelihood also maximizes the likelihood. If $p(\mathcal{D}|\boldsymbol{\theta})$ is a well behaved, differentiable function of $\boldsymbol{\theta}$, $\hat{\boldsymbol{\theta}}$ can be found by the standard methods of differential calculus. If the number of parameters to be set is p, then we let $\boldsymbol{\theta}$ denote

the p-component vector $\boldsymbol{\theta} = (\theta_1, ..., \theta_p)^t$, and $\nabla_{\boldsymbol{\theta}}$ be the gradient operator

$$\nabla_{\boldsymbol{\theta}} \equiv \begin{bmatrix} \frac{\partial}{\partial \theta_1} \\ \vdots \\ \frac{\partial}{\partial \theta_p} \end{bmatrix}. \tag{2}$$

LOG-LIKELIHOOD

We define $l(\boldsymbol{\theta})$ as the *log-likelihood* function*

$$l(\boldsymbol{\theta}) \equiv \ln\, p(\mathcal{D}|\boldsymbol{\theta}). \tag{3}$$

We can then write our solution formally as the argument $\boldsymbol{\theta}$ that maximizes the log-likelihood, i.e.,

$$\hat{\boldsymbol{\theta}} = \arg\max_{\boldsymbol{\theta}} l(\boldsymbol{\theta}), \tag{4}$$

where the dependence on the data set $\mathcal{D}$ is implicit. Thus we have from Eq. 1

$$l(\boldsymbol{\theta}) = \sum_{k=1}^{n} \ln\, p(\mathbf{x}_k|\boldsymbol{\theta}) \tag{5}$$

and

$$\nabla_{\boldsymbol{\theta}} l = \sum_{k=1}^{n} \nabla_{\boldsymbol{\theta}}\, \ln\, p(\mathbf{x}_k|\boldsymbol{\theta}). \tag{6}$$

Thus, a set of necessary conditions for the maximum likelihood estimate for $\boldsymbol{\theta}$ can be obtained from the set of p equations

$$\boxed{\nabla_{\boldsymbol{\theta}} l = \mathbf{0}.} \tag{7}$$

A solution $\hat{\boldsymbol{\theta}}$ to Eq. 7 could represent a true global maximum, a *local* maximum or minimum, or (rarely) an inflection point of $l(\boldsymbol{\theta})$. One must be careful, too, to check if the extremum occurs at a boundary of the parameter space, which might not be apparent from the solution to Eq. 7. If all solutions are found, we are guaranteed that one represents the true maximum, though we might have to check each solution individually (or calculate second derivatives) to identify which is the global optimum. Of course, we must bear in mind that $\hat{\boldsymbol{\theta}}$ is an estimate; it is only in the limit of an infinitely large number of training points that we can expect that our estimate will equal to the true value of the generating function (Sec. 3.5.1).

MAXIMUM A POSTERIORI

MODE

We note in passing that a related class of estimators — *maximum a posteriori* or MAP estimators — find the value of $\boldsymbol{\theta}$ that maximizes $l(\boldsymbol{\theta})p(\boldsymbol{\theta})$. Thus a maximum likelihood estimator is a MAP estimator for the uniform or "flat" prior. As such, a MAP estimator finds the peak, or *mode* of a posterior density. The drawback of MAP estimators is that if we choose some arbitrary nonlinear transformation of the parameter space (e.g., an overall rotation), the density will change, and our MAP solution need no longer be appropriate (Sec. 3.5.2).

* Of course, the base of the logarithm can be chosen for convenience, and in most analytic problems base e is most natural. For that reason we will generally use ln rather than log or $\log_2$.

3.2.2 The Gaussian Case: Unknown $\boldsymbol{\mu}$

To see how maximum likelihood methods results apply to a specific case, suppose that the samples are drawn from a multivariate normal population with mean $\boldsymbol{\mu}$ and covariance matrix $\boldsymbol{\Sigma}$. For simplicity, consider first the case where only the mean is unknown. Under this condition, we consider a sample point $\mathbf{x}_k$ and find

$$\ln p(\mathbf{x}_k|\boldsymbol{\mu}) = -\frac{1}{2}\ln\left[(2\pi)^d|\boldsymbol{\Sigma}|\right] - \frac{1}{2}(\mathbf{x}_k - \boldsymbol{\mu})^t\boldsymbol{\Sigma}^{-1}(\mathbf{x}_k - \boldsymbol{\mu}) \tag{8}$$

and

$$\nabla_{\boldsymbol{\theta}} \ln p(\mathbf{x}_k|\boldsymbol{\mu}) = \boldsymbol{\Sigma}^{-1}(\mathbf{x}_k - \boldsymbol{\mu}). \tag{9}$$

Identifying $\boldsymbol{\theta}$ with $\boldsymbol{\mu}$, we see from Eq. 9 that the maximum likelihood estimate for $\boldsymbol{\mu}$ must satisfy

$$\sum_{k=1}^{n} \boldsymbol{\Sigma}^{-1}(\mathbf{x}_k - \hat{\boldsymbol{\mu}}) = \mathbf{0}, \tag{10}$$

that is, each of the d components of $\hat{\boldsymbol{\mu}}$ must vanish. Multiplying by $\boldsymbol{\Sigma}$ and rearranging, we obtain

$$\hat{\boldsymbol{\mu}} = \frac{1}{n}\sum_{k=1}^{n} \mathbf{x}_k. \tag{11}$$

This is a very satisfying result. It says that the maximum likelihood estimate for the unknown population mean is just the arithmetic average of the training samples — the *sample mean*, sometimes written $\hat{\boldsymbol{\mu}}_n$ to clarify its dependence on the number of samples. Geometrically, if we think of the n samples as a cloud of points, the sample mean is the centroid of the cloud. The sample mean has a number of desirable statistical properties as well, and one would be inclined to use this rather obvious estimate even without knowing that it is the maximum likelihood solution.

SAMPLE MEAN

3.2.3 The Gaussian Case: Unknown $\boldsymbol{\mu}$ and $\boldsymbol{\Sigma}$

In the more general (and more typical) multivariate normal case, neither the mean $\boldsymbol{\mu}$ nor the covariance matrix $\boldsymbol{\Sigma}$ is known. Thus, these unknown parameters constitute the components of the parameter vector $\boldsymbol{\theta}$. Consider first the univariate case with $\theta_1 = \mu$ and $\theta_2 = \sigma^2$. Here the log-likelihood of a single point is

$$\ln p(x_k|\boldsymbol{\theta}) = -\frac{1}{2}\ln 2\pi\theta_2 - \frac{1}{2\theta_2}(x_k - \theta_1)^2 \tag{12}$$

and its derivative is

$$\nabla_{\boldsymbol{\theta}} l = \nabla_{\boldsymbol{\theta}} \ln p(x_k|\boldsymbol{\theta}) = \begin{bmatrix} \frac{1}{\theta_2}(x_k - \theta_1) \\ -\frac{1}{2\theta_2} + \frac{(x_k-\theta_1)^2}{2\theta_2^2} \end{bmatrix}. \tag{13}$$

Applying Eq. 7 to the full log-likelihood leads to the conditions

$$\sum_{k=1}^{n} \frac{1}{\hat{\theta}_2}(x_k - \hat{\theta}_1) = 0 \tag{14}$$

and

$$-\sum_{k=1}^{n}\frac{1}{\hat{\theta}_2}+\sum_{k=1}^{n}\frac{(x_k-\hat{\theta}_1)^2}{\hat{\theta}_2^2}=0, \tag{15}$$

where $\hat{\theta}_1$ and $\hat{\theta}_2$ are the maximum likelihood estimates for θ_1 and θ_2, respectively. By substituting $\hat{\mu}=\hat{\theta}_1$, $\hat{\sigma}^2=\hat{\theta}_2$ and doing a little rearranging, we obtain the following maximum likelihood estimates for μ and σ^2:

$$\hat{\mu}=\frac{1}{n}\sum_{k=1}^{n}x_k \tag{16}$$

and

$$\hat{\sigma}^2=\frac{1}{n}\sum_{k=1}^{n}(x_k-\hat{\mu})^2. \tag{17}$$

While the analysis of the multivariate case is basically very similar, considerably more manipulations are involved (Problem 6). Just as we would predict, though, the result is that the maximum likelihood estimates for $\boldsymbol{\mu}$ and $\boldsymbol{\Sigma}$ are given by

$$\hat{\boldsymbol{\mu}}=\frac{1}{n}\sum_{k=1}^{n}\mathbf{x}_k \tag{18}$$

and

$$\widehat{\boldsymbol{\Sigma}}=\frac{1}{n}\sum_{k=1}^{n}(\mathbf{x}_k-\hat{\boldsymbol{\mu}})(\mathbf{x}_k-\hat{\boldsymbol{\mu}})^t. \tag{19}$$

Thus, once again we find that the maximum likelihood estimate for the mean vector is the sample mean. The maximum likelihood estimate for the covariance matrix is the arithmetic average of the n matrices $(\mathbf{x}_k-\hat{\boldsymbol{\mu}})(\mathbf{x}_k-\hat{\boldsymbol{\mu}})^t$. Since the true covariance matrix is the expected value of the matrix $(\mathbf{x}-\hat{\boldsymbol{\mu}})\ (\mathbf{x}-\hat{\boldsymbol{\mu}})^t$, this is also a very satisfying result.

3.2.4 Bias

BIAS

The maximum likelihood estimate for the variance σ^2 is *biased*; that is, the expected value over all data sets of size n of the sample variance is not equal to the true variance:*

$$\mathcal{E}\left[\frac{1}{n}\sum_{i=1}^{n}(x_i-\bar{x})^2\right]=\frac{n-1}{n}\sigma^2\neq\sigma^2. \tag{20}$$

We shall return to a more general consideration of bias in Chap. **??**, but for the moment we can verify Eq. 20 for an underlying distribution with non-zero variance, σ^2, in the extreme case of $n=1$, in which the expectation value $\mathcal{E}[\cdot]=0\neq\sigma^2$. The maximum likelihood estimate of the covariance matrix is similarly biased.

Elementary *unbiased* estimators for σ^2 and $\boldsymbol{\Sigma}$ are given by

* There should be no confusion over this use of the *statistical* term bias, and that for an offset in neural networks and many other places.

$$\mathcal{E}\left[\frac{1}{n-1}\sum_{i=1}^{n}(x_i-\bar{x})^2\right]=\sigma^2 \qquad \text{and} \tag{21}$$

$$\mathbf{C}=\frac{1}{n-1}\sum_{k=1}^{n}(\mathbf{x}_k-\hat{\boldsymbol{\mu}})(\mathbf{x}_k-\hat{\boldsymbol{\mu}})^t, \tag{22}$$

where $\mathbf{C}$ is the so-called *sample covariance matrix*, as explored in Problem 33. If an estimator is unbiased for *all* distributions, as for example the variance estimator in Eq. 21, then it is called *absolutely unbiased*. If the estimator tends to become unbiased as the number of samples becomes very large, as for instance Eq. 20, then the estimator is *asymptotically unbiased*. In many pattern recognition problems with large training data sets, asymptotically unbiased estimators are acceptable.

SAMPLE COVARIANCE

ABSOLUTELY UNBIASED

ASYMPTOTICALLY UNBIASED

Clearly, $\widehat{\boldsymbol{\Sigma}}=[(n-1)/n]\mathbf{C}$, and $\widehat{\boldsymbol{\Sigma}}$ is asymptotically unbiased — these two estimates are essentially identical when n is large. However, the existence of two similar but nevertheless distinct estimates for the covariance matrix may be disconcerting, and it is natural to ask which one is "correct." Of course, for $n>1$ the answer is that these estimates are neither right nor wrong — they are just different. What the existence of two actually shows is that no single estimate possesses all of the properties we might desire. For our purposes, the most desirable property is rather complex — we want the estimate that leads to the best classification performance. While it is usually both reasonable and sound to design a classifier by substituting the maximum likelihood estimates for the unknown parameters, we might well wonder if other estimates might not lead to better performance. Below we address this question from a Bayesian viewpoint.

If we have a reliable model for the underlying distributions and their dependence upon the parameter vector $\boldsymbol{\theta}$, the maximum likelihood classifier will give excellent results. But what if our model is wrong — do we nevertheless get the best classifier in our assumed set of models? For instance, what if we assume that a distribution comes from $N(\mu,1)$ but instead it actually comes from $N(\mu,10)$? Will the value we find for $\theta=\mu$ by maximum likelihood yield the best of all classifiers of the form derived from $N(\mu,1)$? Unfortunately, the answer is "no," and an illustrative counterexample is given in Problem 7 where the so-called *model error* is large indeed. This points out the need for reliable information concerning the models — if the assumed model is very poor, we cannot be assured that the classifier we derive is the best, even among our model set. We shall return to the problem of choosing among candidate models in Chap. **??**.

3.3 Bayesian estimation

We now consider the Bayesian estimation or Bayesian learning approach to pattern classification problems. Although the answers we get by this method will generally be nearly identical to those obtained by maximum likelihood, there is a conceptual difference: whereas in maximum likelihood methods we view the true parameter vector we seek, $\boldsymbol{\theta}$, to be fixed, in Bayesian learning we consider $\boldsymbol{\theta}$ to be a random variable, and training data allows us to convert a distribution on this variable into a posterior probability density.

3.3.1 The Class-Conditional Densities

The computation of the posterior probabilities $P(\omega_i|\mathbf{x})$ lies at the heart of Bayesian classification. Bayes' formula allows us to compute these probabilities from the prior probabilities $P(\omega_i)$ and the class-conditional densities $p(\mathbf{x}|\omega_i)$, but how can we proceed when these quantities are unknown? The general answer to this question is that the best we can do is to compute $P(\omega_i|\mathbf{x})$ using all of the information at our disposal. Part of this information might be prior knowledge, such as knowledge of the functional forms for unknown densities and ranges for the values of unknown parameters. Part of this information might reside in a set of training samples. If we again let $\mathcal{D}$ denote the set of samples, then we can emphasize the role of the samples by saying that our goal is to compute the posterior probabilities $P(\omega_i|\mathbf{x},\mathcal{D})$. From these probabilities we can obtain the Bayes classifier.

Given the sample $\mathcal{D}$, Bayes' formula then becomes

$$\boxed{P(\omega_i|\mathbf{x},\mathcal{D}) = \frac{p(\mathbf{x}|\omega_i,\mathcal{D})P(\omega_i|\mathcal{D})}{\sum_{j=1}^{c} p(\mathbf{x}|\omega_j,\mathcal{D})P(\omega_j|\mathcal{D})}.} \tag{23}$$

As this equation suggests, we can use the information provided by the training samples to help determine both the class-conditional densities and the a priori probabilities.

Although we could maintain this generality, we shall henceforth assume that the true values of the a priori probabilities are known or obtainable from a trivial calculation; thus we substitute $P(\omega_i) = P(\omega_i|\mathcal{D})$. Furthermore, since we are treating the supervised case, we can separate the training samples by class into c subsets $\mathcal{D}_1, ..., \mathcal{D}_c$, with the samples in $\mathcal{D}_i$ belonging to ω_i. As we mentioned when addressing maximum likelihood methods, in most cases of interest (and in all of the cases we shall consider), the samples in $\mathcal{D}_i$ have no influence on $p(\mathbf{x}|\omega_j,\mathcal{D})$ if $i \neq j$. This has two simplifying consequences. First, it allows us to work with each class separately, using only the samples in $\mathcal{D}_i$ to determine $p(\mathbf{x}|\omega_i,\mathcal{D})$. Used in conjunction with our assumption that the prior probabilities are known, this allows us to write Eq. 23 as

$$P(\omega_i|\mathbf{x},\mathcal{D}) = \frac{p(\mathbf{x}|\omega_i,\mathcal{D}_i)P(\omega_i)}{\sum_{j=1}^{c} p(\mathbf{x}|\omega_j,\mathcal{D}_j)P(\omega_j)}. \tag{24}$$

Second, because each class can be treated independently, we can dispense with needless class distinctions and simplify our notation. In essence, we have c separate problems of the following form: use a set $\mathcal{D}$ of samples drawn independently according to the fixed but unknown probability distribution $p(\mathbf{x})$ to determine $p(\mathbf{x}|\mathcal{D})$. This is the central problem of Bayesian learning.

3.3.2 The Parameter Distribution

Although the desired probability density $p(\mathbf{x})$ is unknown, we assume that it has a known parametric form. The only thing assumed unknown is the value of a parameter vector $\boldsymbol{\theta}$. We shall express the fact that $p(\mathbf{x})$ is unknown but has known parametric form by saying that the function $p(\mathbf{x}|\boldsymbol{\theta})$ is completely known. Any information we might have about $\boldsymbol{\theta}$ prior to observing the samples is assumed to be contained in a *known* prior density $p(\boldsymbol{\theta})$. Observation of the samples converts this to a posterior density $p(\boldsymbol{\theta}|\mathcal{D})$, which, we hope, is sharply peaked about the true value of $\boldsymbol{\theta}$.

Note that we are changing our supervised learning problem into an unsupervised density estimation problem. To this end, our basic goal is to compute $p(\mathbf{x}|\mathcal{D})$, which is as close as we can come to obtaining the unknown $p(\mathbf{x})$. We do this by integrating the joint density $p(\mathbf{x}, \boldsymbol{\theta}|\mathcal{D})$ over $\boldsymbol{\theta}$. That is,

$$p(\mathbf{x}|\mathcal{D}) = \int p(\mathbf{x}, \boldsymbol{\theta}|\mathcal{D})\ d\boldsymbol{\theta}, \tag{25}$$

where the integration extends over the entire parameter space. Now as discussed in Problem 12 we can write $p(\mathbf{x}, \boldsymbol{\theta}|\mathcal{D})$ as the product $p(\mathbf{x}|\boldsymbol{\theta}, \mathcal{D})p(\boldsymbol{\theta}|\mathcal{D})$. Since the selection of $\mathbf{x}$ and that of the training samples in $\mathcal{D}$ is done independently, the first factor is merely $p(\mathbf{x}|\boldsymbol{\theta})$. That is, the distribution of $\mathbf{x}$ is known completely once we know the value of the parameter vector. Thus, Eq. 25 can be rewritten as

$$\boxed{p(\mathbf{x}|\mathcal{D}) = \int p(\mathbf{x}|\boldsymbol{\theta})p(\boldsymbol{\theta}|\mathcal{D})\ d\boldsymbol{\theta}.} \tag{26}$$

This key equation links the desired class-conditional density $p(\mathbf{x}|\mathcal{D})$ to the posterior density $p(\boldsymbol{\theta}|\mathcal{D})$ for the unknown parameter vector. If $p(\boldsymbol{\theta}|\mathcal{D})$ peaks very sharply about some value $\hat{\boldsymbol{\theta}}$, we obtain $p(\mathbf{x}|\mathcal{D}) \simeq p(\mathbf{x}|\hat{\boldsymbol{\theta}})$, i.e., the result we would obtain by substituting the estimate $\hat{\boldsymbol{\theta}}$ for the true parameter vector. This result rests on the assumption that $p(\mathbf{x}|\boldsymbol{\theta})$ is smooth, and that the tails of the integral are not important. These conditions are typically but not invariably the case, as we shall see in Sect. **??**. In general, if we are less certain about the exact value of $\boldsymbol{\theta}$, this equation directs us to average $p(\mathbf{x}|\boldsymbol{\theta})$ over the possible values of $\boldsymbol{\theta}$. Thus, when the unknown densities have a known parametric form, the samples exert their influence on $p(\mathbf{x}|\mathcal{D})$ through the posterior density $p(\boldsymbol{\theta}|\mathcal{D})$. We should also point out that in practice, the integration in Eq. 26 is often performed numerically, for instance by Monte-Carlo simulation.

3.4 Bayesian Parameter Estimation: Gaussian Case

In this section we use Bayesian estimation techniques to calculate the a posteriori density $p(\boldsymbol{\theta}|\mathcal{D})$ and the desired probability density $p(\mathbf{x}|\mathcal{D})$ for the case where $p(\mathbf{x}|\boldsymbol{\mu}) \sim N(\boldsymbol{\mu}, \boldsymbol{\Sigma})$.

3.4.1 The Univariate Case: $p(\mu|\mathcal{D})$

Consider the case where $\boldsymbol{\mu}$ is the only unknown parameter. For simplicity we treat first the univariate case, i.e.,

$$p(x|\mu) \sim N(\mu, \sigma^2), \tag{27}$$

where the only unknown quantity is the mean μ. We assume that whatever prior knowledge we might have about μ can be expressed by a *known* prior density $p(\mu)$. Later we shall make the further assumption that

$$p(\mu) \sim N(\mu_0, \sigma_0^2), \tag{28}$$

where both μ_0 and σ_0^2 are known. Roughly speaking, μ_0 represents our best a priori guess for μ, and σ_0^2 measures our uncertainty about this guess. The assumption that the prior distribution for μ is normal will simplify the subsequent mathematics.

However, the crucial assumption is not so much that the prior distribution for μ is normal, but that it is known.

Having selected the a priori density for μ, we can view the situation as follows. Imagine that a value is drawn for μ from a population governed by the probability law $p(\mu)$. Once this value is drawn, it becomes the true value of μ and completely determines the density for x. Suppose now that n samples $x_1, ..., x_n$ are independently drawn from the resulting population. Letting $\mathcal{D} = \{x_1, ..., x_n\}$, we use Bayes' formula to obtain

$$\begin{aligned} p(\mu|\mathcal{D}) &= \frac{p(\mathcal{D}|\mu)p(\mu)}{\int p(\mathcal{D}|\mu)p(\mu)\ d\mu} \\ &= \alpha \prod_{k=1}^{n} p(x_k|\mu)p(\mu), \end{aligned} \tag{29}$$

where α is a normalization factor that depends on $\mathcal{D}$ but is independent of μ. This equation shows how the observation of a set of training samples affects our ideas about the true value of μ; it relates the prior density $p(\mu)$ to an a posteriori density $p(\mu|\mathcal{D})$. Since $p(x_k|\mu) \sim N(\mu, \sigma^2)$ and $p(\mu) \sim N(\mu_0, \sigma_0^2)$, we have

$$\begin{aligned} p(\mu|\mathcal{D}) &= \alpha \prod_{k=1}^{n} \overbrace{\frac{1}{\sqrt{2\pi}\sigma}\ \exp\left[-\frac{1}{2}\Big(\frac{x_k-\mu}{\sigma}\Big)^2\right]}^{p(x_k|\mu)} \overbrace{\frac{1}{\sqrt{2\pi}\sigma_0}\ \exp\left[-\frac{1}{2}\Big(\frac{\mu-\mu_0}{\sigma_0}\Big)^2\right]}^{p(\mu)} \\ &= \alpha' \exp\left[-\frac{1}{2}\left(\sum_{k=1}^{n}\left(\frac{\mu-x_k}{\sigma}\right)^2 + \left(\frac{\mu-\mu_0}{\sigma_0}\right)^2\right)\right] \\ &= \alpha'' \exp\left[-\frac{1}{2}\left[\Big(\frac{n}{\sigma^2}+\frac{1}{\sigma_0^2}\Big)\mu^2 - 2\left(\frac{1}{\sigma^2}\sum_{k=1}^{n}x_k + \frac{\mu_0}{\sigma_0^2}\right)\mu\right]\right], \end{aligned} \tag{30}$$

where factors that do not depend on μ have been absorbed into the constants α, α', and α''. Thus, $p(\mu|\mathcal{D})$ is an exponential function of a quadratic function of μ, i.e., is again a normal density. Since this is true for any number of training samples, $p(\mu|\mathcal{D})$ remains normal as the number n of samples is increased, and $p(\mu|\mathcal{D})$ is said to be a *reproducing density* and $p(\mu)$ is said to be a *conjugate prior*. If we write $p(\mu|\mathcal{D}) \sim N(\mu_n, \sigma_n^2)$, then μ_n and σ_n^2 can be found by equating coefficients in Eq. 30 with corresponding coefficients in the generic Gaussian of the form

REPRODUCING DENSITY

$$p(\mu|\mathcal{D}) = \frac{1}{\sqrt{2\pi}\sigma_n}\ \exp\left[-\frac{1}{2}\left(\frac{\mu-\mu_n}{\sigma_n}\right)^2\right]. \tag{31}$$

Identifying coefficients in this way yields

$$\frac{1}{\sigma_n^2} = \frac{n}{\sigma^2} + \frac{1}{\sigma_0^2} \tag{32}$$

and

$$\frac{\mu_n}{\sigma_n^2} = \frac{n}{\sigma_2}\ \bar{x}_n + \frac{\mu_0}{\sigma_0^2}, \tag{33}$$

where $\bar{x}_n$ is the sample mean

$$\bar{x}_n = \frac{1}{n}\sum_{k=1}^{n} x_k. \tag{34}$$

We solve explicitly for μ_n and σ_n^2 and obtain

$$\mu_n = \left(\frac{n\sigma_0^2}{n\sigma_0^2 + \sigma^2}\right)\bar{x}_n + \frac{\sigma^2}{n\sigma_0^2 + \sigma^2}\mu_0 \tag{35}$$

and

$$\sigma_n^2 = \frac{\sigma_0^2\sigma^2}{n\sigma_0^2 + \sigma^2}. \tag{36}$$

These equations show how the prior information is combined with the empirical information in the samples to obtain the a posteriori density $p(\mu|\mathcal{D})$. Roughly speaking, μ_n represents our best guess for μ after observing n samples, and σ_n^2 measures our uncertainty about this guess. Since σ_n^2 decreases monotonically with n — approaching σ^2/n as n approaches infinity — each additional observation decreases our uncertainty about the true value of μ. As n increases, $p(\mu|\mathcal{D})$ becomes more and more sharply peaked, approaching a Dirac delta function as n approaches infinity. This behavior is commonly known as *Bayesian learning* (Fig. 3.2).

BAYESIAN LEARNING

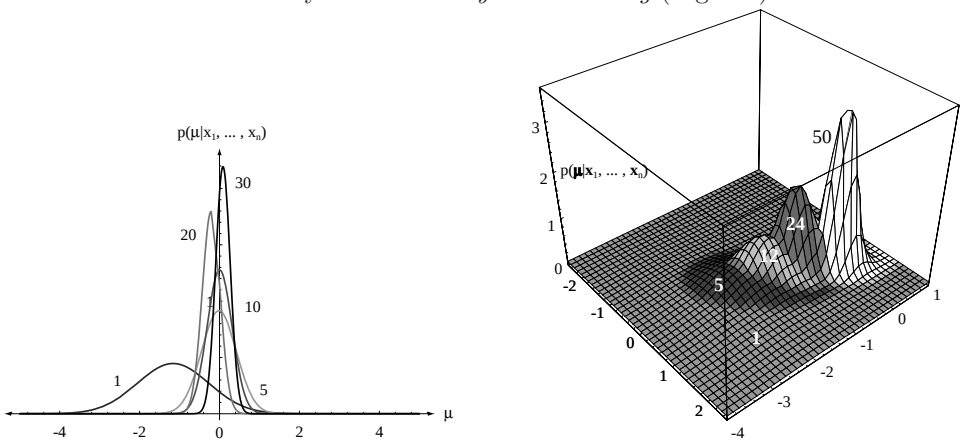


Figure 3.2: Bayesian learning of the mean of normal distributions in one and two dimensions. The posterior distribution estimates are labelled by the number of training samples used in the estimation.

In general, μ_n is a linear combination of $\bar{x}_n$ and μ_0, with coefficients that are non-negative and sum to one. Thus μ_n always lies somewhere between $\bar{x}_n$ and μ_0. If $\sigma \neq 0$, μ_n approaches the sample mean as n approaches infinity. If $\sigma_0 = 0$, we have a degenerate case in which our a priori certainty that $\mu = \mu_0$ is so strong that no number of observations can change our opinion. At the other extreme, if $\sigma_0 \gg \sigma$, we are so uncertain about our a priori guess that we take $\mu_n = \bar{x}_n$, using only the samples to estimate μ. In general, the relative balance between prior knowledge and empirical data is set by the ratio of σ^2 to σ_0^2, which is sometimes called the *dogmatism*. If the dogmatism is not infinite, after enough samples are taken the exact values assumed for μ_0 and σ_0^2 will be unimportant, and μ_n will converge to the sample mean.

DOGMATISM

3.4.2 The Univariate Case: $p(x|\mathcal{D})$

Having obtained the a posteriori density for the mean, $p(\mu|\mathcal{D})$, all that remains is to obtain the "class-conditional" density for $p(x|\mathcal{D})$.* From Eqs. 26, 27 & 31 we have

$$\begin{aligned} p(x|\mathcal{D}) &= \int p(x|\mu)p(\mu|\mathcal{D})\ d\mu \\ &= \int \frac{1}{\sqrt{2\pi}\sigma}\ \exp\Big[-\frac{1}{2}\Big(\frac{x-\mu}{\sigma}\Big)^2\Big]\ \frac{1}{\sqrt{2\pi}\sigma_n}\exp\Big[-\frac{1}{2}\Big(\frac{\mu-\mu_n}{\sigma_n}\Big)^2\Big]\ d\mu \\ &= \frac{1}{2\pi\sigma\sigma_n}\ \exp\Big[-\frac{1}{2}\frac{(x-\mu_n)^2}{\sigma^2+\sigma_n^2}\Big]\ f(\sigma,\sigma_n), \end{aligned} \tag{37}$$

where

$$f(\sigma,\sigma_n) = \int \exp\Big[-\frac{1}{2}\frac{\sigma^2+\sigma_n^2}{\sigma^2\sigma_n^2}\Big(\mu-\frac{\sigma_n^2 x+\sigma^2\mu_n}{\sigma^2+\sigma_n^2}\Big)^2\Big]\ d\mu.$$

That is, as a function of x, $p(x|\mathcal{D})$ is proportional to $\exp[-(1/2)(x-\mu_n)^2/(\sigma^2+\sigma_n^2)]$, and hence $p(x|\mathcal{D})$ is normally distributed with mean μ_n and variance $\sigma^2+\sigma_n^2$:

$$p(x|\mathcal{D}) \sim N(\mu_n, \sigma^2+\sigma_n^2). \tag{38}$$

In other words, to obtain the class-conditional density $p(x|\mathcal{D})$, whose parametric form is known to be $p(x|\mu) \sim N(\mu,\sigma^2)$, we merely replace μ by μ_n and σ^2 by $\sigma^2+\sigma_n^2$. In effect, the conditional mean μ_n is treated as if it were the true mean, and the known variance is increased to account for the additional uncertainty in x resulting from our lack of exact knowledge of the mean μ. This, then, is our final result: the density $p(x|\mathcal{D})$ is the desired class-conditional density $p(x|\omega_j,\mathcal{D}_j)$, and together with the prior probabilities $P(\omega_j)$ it gives us the probabilistic information needed to design the classifier. This is in contrast to maximum likelihood methods that only make points estimates for $\hat{\mu}$ and $\hat{\sigma^2}$, rather that estimate a *distribution* for $p(x|\mathcal{D})$.

3.4.3 The Multivariate Case

The treatment of the multivariate case in which $\boldsymbol{\Sigma}$ is known but $\boldsymbol{\mu}$ is not, is a direct generalization of the univariate case. For this reason we shall only sketch the derivation. As before, we assume that

$$p(\mathbf{x}|\boldsymbol{\mu}) \sim N(\boldsymbol{\mu},\boldsymbol{\Sigma}) \quad \text{and} \quad p(\boldsymbol{\mu}) \sim N(\boldsymbol{\mu}_0,\boldsymbol{\Sigma}_0), \tag{39}$$

where $\boldsymbol{\Sigma}$, $\boldsymbol{\Sigma}_0$, and $\boldsymbol{\mu}_0$ are assumed to be known. After observing a set $\mathcal{D}$ of n independent samples $\mathbf{x}_1,...,\mathbf{x}_n$, we use Bayes' formula to obtain

$$\begin{aligned} p(\boldsymbol{\mu}|\mathcal{D}) &= \alpha\prod_{k=1}^{n} p(\mathbf{x}_k|\boldsymbol{\mu})p(\boldsymbol{\mu}) \qquad (40) \\ &= \alpha' \exp\ \left[-\frac{1}{2}\left(\boldsymbol{\mu}^t(n\boldsymbol{\Sigma}^{-1}+\boldsymbol{\Sigma}_0^{-1})\boldsymbol{\mu} - 2\boldsymbol{\mu}^t\left(\boldsymbol{\Sigma}^{-1}\sum_{k=1}^{n}\mathbf{x}_k + \boldsymbol{\Sigma}_0^{-1}\boldsymbol{\mu}_0\right)\right)\right], \end{aligned}$$

* Recall that for simplicity we dropped class distinctions, but that all samples here come from the same class, say ω_i, and hence $p(\mathbf{x}|\mathcal{D})$ is really $p(\mathbf{x}|\omega_i,\mathcal{D}_i)$.

which has the form

$$p(\boldsymbol{\mu}|\mathcal{D}) = \alpha'' \exp\left[-\frac{1}{2}(\boldsymbol{\mu}-\boldsymbol{\mu}_n)^t \boldsymbol{\Sigma}_n^{-1}(\boldsymbol{\mu}-\boldsymbol{\mu}_n)\right]. \tag{41}$$

Thus, $p(\boldsymbol{\mu}|\mathcal{D}) \sim N(\boldsymbol{\mu}_n, \boldsymbol{\Sigma}_n)$, and once again we have a reproducing density. Equating coefficients, we obtain the analogs of Eqs. 35 & 36,

$$\boldsymbol{\Sigma}_n^{-1} = n\boldsymbol{\Sigma}^{-1} + \boldsymbol{\Sigma}_0^{-1} \tag{42}$$

and

$$\boldsymbol{\Sigma}_n^{-1}\boldsymbol{\mu}_n = n\boldsymbol{\Sigma}^{-1}\hat{\boldsymbol{\mu}}_n + \boldsymbol{\Sigma}_0^{-1}\boldsymbol{\mu}_0, \tag{43}$$

where $\hat{\boldsymbol{\mu}}_n$ is the sample mean

$$\hat{\boldsymbol{\mu}}_n = \frac{1}{n}\sum_{k=1}^{n}\mathbf{x}_k. \tag{44}$$

The solution of these equations for $\boldsymbol{\mu}$ and $\boldsymbol{\Sigma}_n$ is simplified by knowledge of the matrix identity

$$(\mathbf{A}^{-1}+\mathbf{B}^{-1})^{-1} = \mathbf{A}(\mathbf{A}+\mathbf{B})^{-1}\mathbf{B} = \mathbf{B}(\mathbf{A}+\mathbf{B})^{-1}\mathbf{A}, \tag{45}$$

which is valid for any pair of nonsingular, d-by-d matrices $\mathbf{A}$ and $\mathbf{B}$. After a little manipulation (Problem 16), we obtain the final results:

$$\boldsymbol{\mu}_n = \boldsymbol{\Sigma}_0\Big(\boldsymbol{\Sigma}_0 + \frac{1}{n}\boldsymbol{\Sigma}\Big)^{-1}\hat{\boldsymbol{\mu}}_n + \frac{1}{n}\boldsymbol{\Sigma}\Big(\boldsymbol{\Sigma}_0 + \frac{1}{n}\boldsymbol{\Sigma}\Big)^{-1}\boldsymbol{\mu}_0 \tag{46}$$

(which, as in the univariate case, is a linear combination of $\hat{\boldsymbol{\mu}}_n$ and $\boldsymbol{\mu}_0$) and

$$\boldsymbol{\Sigma}_n = \boldsymbol{\Sigma}_0\Big(\boldsymbol{\Sigma}_0 + \frac{1}{n}\boldsymbol{\Sigma}\Big)^{-1}\frac{1}{n}\boldsymbol{\Sigma}. \tag{47}$$

The proof that $p(\mathbf{x}|\mathcal{D}) \sim N(\boldsymbol{\mu}_n, \boldsymbol{\Sigma}+\boldsymbol{\Sigma}_n)$ can be obtained as before by performing the integration

$$p(\mathbf{x}|\mathcal{D}) = \int p(\mathbf{x}|\boldsymbol{\mu})p(\boldsymbol{\mu}|\mathcal{D})\; d\boldsymbol{\mu}. \tag{48}$$

However, this result can be obtained with less effort by observing that $\mathbf{x}$ can be viewed as the sum of two mutually independent random variables, a random vector $\boldsymbol{\mu}$ with $p(\boldsymbol{\mu}|\mathcal{D}) \sim N(\boldsymbol{\mu}_n, \boldsymbol{\Sigma}_n)$ and an independent random vector $\mathbf{y}$ with $p(\mathbf{y}) \sim N(\mathbf{0}, \boldsymbol{\Sigma})$. Since the sum of two independent, normally distibuted vectors is again a normally distributed vector whose mean is the sum of the means and whose covariance matrix is the sum of the covariance matrices (Chap. **??** Problem **??**), we have

$$p(\mathbf{x}|\mathcal{D}) \sim N(\boldsymbol{\mu}_n, \boldsymbol{\Sigma}+\boldsymbol{\Sigma}_n), \tag{49}$$

and the generalization is complete.

3.5 Bayesian Parameter Estimation: General Theory

We have just seen how the Bayesian approach can be used to obtain the desired density $p(\mathbf{x}|\mathcal{D})$ in a special case — the multivariate Gaussian. This approach can be generalized to apply to any situation in which the unknown density can be parameterized. The basic assumptions are summarized as follows:

- The form of the density $p(\mathbf{x}|\boldsymbol{\theta})$ is assumed to be known, but the value of the parameter vector $\boldsymbol{\theta}$ is not known exactly.
- Our initial knowledge about $\boldsymbol{\theta}$ is assumed to be contained in a known a priori density $p(\boldsymbol{\theta})$.
- The rest of our knowledge about $\boldsymbol{\theta}$ is contained in a set $\mathcal{D}$ of n samples $\mathbf{x}_1, ..., \mathbf{x}_n$ drawn independently according to the unknown probability density $p(\mathbf{x})$.

The basic problem is to compute the posterior density $p(\boldsymbol{\theta}|\mathcal{D})$, since from this we can use Eq. 26 to compute $p(\mathbf{x}|\mathcal{D})$:

$$p(\mathbf{x}|\mathcal{D}) = \int p(\mathbf{x}|\boldsymbol{\theta})p(\boldsymbol{\theta}|\mathcal{D})\ d\boldsymbol{\theta}. \tag{50}$$

By Bayes' formula we have

$$p(\boldsymbol{\theta}|\mathcal{D}) = \frac{p(\mathcal{D}|\boldsymbol{\theta})p(\boldsymbol{\theta})}{\int p(\mathcal{D}|\boldsymbol{\theta})p(\boldsymbol{\theta})\ d\boldsymbol{\theta}}, \tag{51}$$

and by the independence assumption

$$p(\mathcal{D}|\boldsymbol{\theta}) = \prod_{k=1}^{n} p(\mathbf{x}_k|\boldsymbol{\theta}). \tag{52}$$

This constitutes the solution to the problem, and Eqs. 51 & 52 illuminate its relation to the maximum likelihood solution. Suppose that $p(\mathcal{D}|\boldsymbol{\theta})$ reaches a sharp peak at $\boldsymbol{\theta} = \hat{\boldsymbol{\theta}}$. If the prior density $p(\boldsymbol{\theta})$ is not zero at $\boldsymbol{\theta} = \hat{\boldsymbol{\theta}}$ and does not change much in the surrounding neighborhood, then $p(\boldsymbol{\theta}|\mathcal{D})$ also peaks at that point. Thus, Eq. 26 shows that $p(\mathbf{x}|\mathcal{D})$ will be approximately $p(\mathbf{x}|\hat{\boldsymbol{\theta}})$, the result one would obtain by using the maximum likelihood estimate as if it were the true value. If the peak of $p(\mathcal{D}|\boldsymbol{\theta})$ is very sharp, then the influence of prior information on the uncertainty in the true value of $\boldsymbol{\theta}$ can be ignored. In this and even the more general case, though, the Bayesian solution tells us how to use *all* the available information to compute the desired density $p(\mathbf{x}|\mathcal{D})$.

While we have obtained the formal Bayesian solution to the problem, a number of interesting questions remain. One concerns the difficulty of carrying out these computations. Another concerns the convergence of $p(\mathbf{x}|\mathcal{D})$ to $p(\mathbf{x})$. We shall discuss the matter of convergence briefly, and later turn to the computational question.

To indicate explicitly the number of samples in a set for a single category, we shall write $\mathcal{D}^n = \{\mathbf{x}_1, ..., \mathbf{x}_n\}$. Then from Eq. 52, if $n > 1$

$$p(\mathcal{D}^n|\boldsymbol{\theta}) = p(\mathbf{x}_n|\boldsymbol{\theta})p(\mathcal{D}^{n-1}|\boldsymbol{\theta}). \tag{53}$$

Substituting this in Eq. 51 and using Bayes' formula, we see that the posterior density satisfies the recursion relation

$$p(\boldsymbol{\theta}|\mathcal{D}^n) = \frac{p(\mathbf{x}_n|\boldsymbol{\theta})p(\boldsymbol{\theta}|\mathcal{D}^{n-1})}{\int p(\mathbf{x}_n|\boldsymbol{\theta})p(\boldsymbol{\theta}|\mathcal{D}^{n-1})\, d\boldsymbol{\theta}}. \tag{54}$$

With the understanding that $p(\boldsymbol{\theta}|\mathcal{D}^0) = p(\boldsymbol{\theta})$, repeated use of this equation produces the sequence of densities $p(\boldsymbol{\theta})$, $p(\boldsymbol{\theta}|\mathbf{x}_1)$, $p(\boldsymbol{\theta}|\mathbf{x}_1, \mathbf{x}_2)$, and so forth. (It should be obvious from Eq. 54 that $p(\boldsymbol{\theta}|\mathcal{D}^n)$ depends only on the points in $\mathcal{D}^n$, not the sequence in which they were selected.) This is called the *recursive Bayes* approach to parameter estimation. This is, too, our first example of an *incremental* or *on-line* learning method, where learning goes on as the data is collected. When this sequence of densities converges to a Dirac delta function centered about the true parameter value — *Bayesian learning* (Example 1). We shall come across many other, non-incremental learning schemes, where all the training data must be present before learning can take place.

RECURSIVE BAYES

INCREMENTAL LEARNING

In principle, Eq. 54 requires that we preserve all the training points in $\mathcal{D}^{n-1}$ in order to calculate $p(\boldsymbol{\theta}|\mathcal{D}^n)$ but for some distributions, just a few parameters associated with $p(\boldsymbol{\theta}|\mathcal{D}^{n-1})$ contain all the information needed. Such parameters are the *sufficient statistics* of those distributions, as we shall see in Sect. 3.6. Some authors reserve the term recursive learning to apply to only those cases where the sufficient statistics are retained — not the training data — when incorporating the information from a new training point. We could call this more restrictive usage *true recursive Bayes learning.*

Example 1: Recursive Bayes learning

Suppose we believe our one-dimensional samples come from a uniform distribution

$$p(x|\theta) \sim U(0,\theta) = \begin{cases} 1/\theta & 0 \le x \le \theta \\ 0 & \text{otherwise,} \end{cases}$$

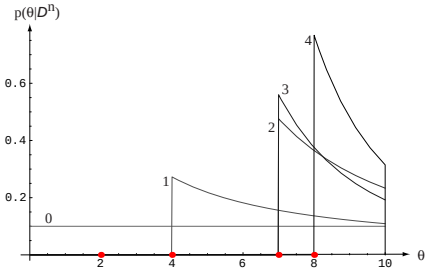
but initially we know only that our parameter is bounded. In particular we assume $0 < \theta \le 10$ (a non-informative or "flat prior" we shall discuss in Sect. 3.5.2). We will use recursive Bayes methods to estimate θ and the underlying densities from the data $\mathcal{D} = \{4, 7, 2, 8\}$, which were selected randomly from the underlying distribution. Before any data arrive, then, we have $p(\theta|\mathcal{D}^0) = p(\theta) = U(0, 10)$. When our first data point $x_1 = 4$ arrives, we use Eq. 54 to get an improved estimate:

$$p(\theta|\mathcal{D}^1) \propto p(x|\theta)p(\theta|\mathcal{D}^0) = \begin{cases} 1/\theta & \text{for } 4 \le \theta \le 10 \\ 0 & \text{otherwise,} \end{cases}$$

where throughout we will ignore the normalization. When the next data point $x_2 = 7$ arrives, we have

$$p(\theta|\mathcal{D}^2) \propto p(x|\theta)p(\theta|\mathcal{D}^1) = \begin{cases} 1/\theta^2 & \text{for } 7 \le \theta \le 10 \\ 0 & \text{otherwise,} \end{cases}$$

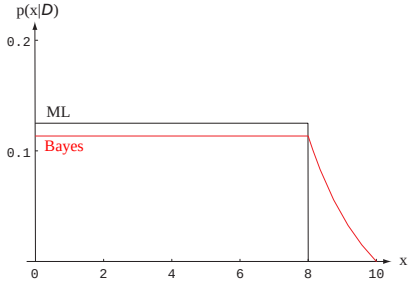
and similarly for the remaining sample points. It should be clear that since each successive step introduces a factor of $1/\theta$ into $p(x|\theta)$, and the distribution is nonzero only for x values above the largest data point sampled, the general form of our solution is $p(\theta|\mathcal{D}^n) \propto 1/\theta^n$ for $\max_x[\mathcal{D}^n] \le \theta \le 10$, as shown in the figure. Given our full data



The posterior $p(\theta|\mathcal{D}^n)$ for the model and n points in the data set in this Example. The posterior begins $p(\theta) \sim U(0, 10)$, and as more points are incorporated it becomes increasingly peaked at the value of the highest data point.

set, the maximum likelihood solution here is clearly $\hat{\theta} = 8$, and this implies a uniform $p(x|\mathcal{D}) \sim U(0, 8)$.

According to our Bayesian methodology, which requires the integration in Eq. 50, the density is uniform up to $x = 8$, but has a tail at higher values — an indication that the influence of our prior $p(\theta)$ has not yet been swamped by the information in the training data.



Given the full set of four points, the distribution based on the maximum likelihood solution is $p(x|\hat{\theta}) \sim U(0, 8)$, whereas the distribution derived from Bayesian methods has a small tail above $x = 8$, reflecting the prior information that values of x near 10 are possible.

Whereas the maximum likelihood approach estimates a *point* in $\boldsymbol{\theta}$ space, the Bayesian approach instead estimates a *distribution*. Technically speaking, then, we cannot directly compare these estimates. It is only when the second stage of inference is done — that is, we compute the distributions $p(x|\mathcal{D})$, as shown in the above figure — that the comparison is fair.

IDENTIFIABILITY

For most of the typically encountered probability densities $p(\mathbf{x}|\boldsymbol{\theta})$, the sequence of posterior densities does indeed converge to a delta function. Roughly speaking, this implies that with a large number of samples there is only one value for $\boldsymbol{\theta}$ that causes $p(\mathbf{x}|\boldsymbol{\theta})$ to fit the data, i.e., that $\boldsymbol{\theta}$ can be determined uniquely from $p(\mathbf{x}|\boldsymbol{\theta})$. When this is the case, $p(\mathbf{x}|\boldsymbol{\theta})$ is said to be *identifiable*. A rigorous proof of convergence under these conditions requires a precise statement of the properties required of $p(\mathbf{x}|\boldsymbol{\theta})$ and $p(\boldsymbol{\theta})$ and considerable care, but presents no serious difficulties (Problem 21).

There are occasions, however, when more than one value of $\boldsymbol{\theta}$ may yield the same value for $p(\mathbf{x}|\boldsymbol{\theta})$. In such cases, $\boldsymbol{\theta}$ cannot be determined uniquely from $p(\mathbf{x}|\boldsymbol{\theta})$, and $p(\mathbf{x}|\mathcal{D}^n)$ will peak near all of the values of $\boldsymbol{\theta}$ that explain the data. Fortunately, this ambiguity is erased by the integration in Eq. 26, since $p(\mathbf{x}|\boldsymbol{\theta})$ is the same for all of

these values of $\boldsymbol{\theta}$. Thus, $p(\mathbf{x}|\mathcal{D}^n)$ will typically converge to $p(\mathbf{x})$ whether or not $p(\mathbf{x}|\boldsymbol{\theta})$ is identifiable. While this might make the problem of identifiabilty appear to be moot, we shall see in Chap. **??** that identifiability presents a genuine problem in the case of unsupervised learning.

3.5.1 When do Maximum Likelihood and Bayes methods differ?

In virtually every case, maximum likelihood and Bayes solutions are equivalent in the asymptotic limit of infinite training data. However since practical pattern recognition problems invariably have a limited set of training data, it is natural to ask when maximum likelihood and Bayes solutions may be expected to differ, and then which we should prefer.

There are several criteria that will influence our choice. One is computational complexity (Sec. 3.7.2), and here maximum likelhood methods are often to be preferred since they require merely differential calculus techniques or gradient search for $\hat{\boldsymbol{\theta}}$, rather than a possibly complex multidimensional integration needed in Bayesian estimation. This leads to another consideration: interpretability. In many cases the maximum likelihood solution will be easier to interpret and understand since it returns the single best model from the set the designer provided (and presumably understands). In contrast Bayesian methods give a weighted average of models (parameters), often leading to solutions more complicated and harder to understand than those provided by the designer. The Bayesian approach reflects the remaining uncertainty in the possible models.

Another consideration is our confidence in the prior information, such as in the *form* of the underlying distribution $p(\mathbf{x}|\boldsymbol{\theta})$. A maximum likelihood solution $p(\mathbf{x}|\hat{\boldsymbol{\theta}})$ must of course be of the assumed parametric form; not so for the Bayesian solution. We saw this difference in Example 1, where the Bayes solution was not of the parametric form originally assumed, i.e., a uniform $p(x|\mathcal{D})$. In general, through their use of the full $p(\boldsymbol{\theta}|\mathcal{D})$ distribution Bayesian methods use more of the information brought to the problem than do maximum likelihood methods. (For instance, in Example 1 the addition of the third training point did not change the maximum likelihood solution, but did refine the Bayesian estimate.) If such information is reliable, Bayes methods can be expected to give better results. Further, general Bayesian methods with a "flat" or uniform prior (i.e., where no prior information is explicitly imposed) are equivalent to maximum likelihood methods. If there is much data, leading to a strongly peaked $p(\boldsymbol{\theta}|\mathcal{D})$, and the prior $p(\boldsymbol{\theta})$ is uniform or flat, then the MAP estimate is essentially the same as the maximum likelihood estimate.

When $p(\boldsymbol{\theta}|\mathcal{D})$ is broad, or asymmetric around $\hat{\boldsymbol{\theta}}$, the methods are quite likely to yield $p(\mathbf{x}|\mathcal{D})$ distributions that differ from one another. Such a strong asymmetry (when not due to rare statistical irregularities in the selection of the training data) generally convey some information about the distribution, just as did the asymmetric role of the threshold θ in Example 1. Bayes methods would exploit such information; not so maximum likelihood ones (at least not directly). Further, Bayesian methods make more explicit the crucial problem of bias and variance tradeoffs — roughly speaking the balance between the accuracy of the estimation and its variance, which depend upon the amount of traning data. This important matter was irrelevant in Chap. **??**, where there was no notion of a finite training set, but it will be crucial in our considerations of the theory of machine learning in Chap. **??**.

When designing a classifier by either of these methods, we determine the posterior densities for each category, and classify a test point by the maximum posterior. (If

there are costs, summarized in a cost matrix, these can be incorporated as well.) There are three sources of classification error in our final system:

Bayes or indistinguisability error: the error due to overlapping densities $p(\mathbf{x}|\omega_i$ for different values of i. This error is an inherent property of the problem and can never be eliminated.

Model error: the error due to having an incorrect model. This error can only be eliminated if the designer specifies a model that includes the true model which generated the data. Designers generally choose the model based on knowledge of the problem domain rather than on the subsequent estimation method, and thus the model error in maximum likelihood and Bayes methods rarely differ.

Estimation error: the error arising from the fact that the parameters are estimated from a finite sample. This error can best be reduced by increasing the training data, a topic we shall revisit in greater detail in Chap. **??**.

The relative contributions of these sources depend upon problem, of course. In the limit of infinite training data, the estimation error vanishes, and the total classification error will be the same for both maximum likelihodd and Bayes methods.

In summary, there are strong theoretical and methodological arguments supporting Bayesian estimation, though in practice maximum likelihood estimation is simpler, and when used for designing classifiers, can lead to classifiers nearly as accurate.

3.5.2 Non-informative Priors and Invariance

Generally speaking, the information about the prior $p(\boldsymbol{\theta})$ derives from the designer's knowledge of the problem domain and as such is beyond our study of the design of classifiers. Nevertheless in some cases we have guidence in how to create priors that do not impose structure when we believe none exists, and this leads us to the notion of non-informative priors.

Recall our discussion of the role of prior category probabilities in Chap. **??**, where in the absense of other information, we assumed each of c categories equally likely. Analogously, in a Bayesian framework we can have a "non-informative" prior over a parameter for a single category's distribution. Suppose for instance that we are using Bayesian methods to infer from data the mean and variance of a Gaussian. What prior might we put on these parameters? Surely the unit of spatial measurement — meters, feet, inches — is an historical accident and irrelevant to the functional form of the prior. Thus there is an implied *scale invariance*, formally stated as

SCALE INVARIANCE

$$p(\theta) = \alpha p(\theta/\alpha) \tag{55}$$

for some constant α. Such scale invariance here leads to priors such as $p(\mu) \propto \mu^{-k}$ for some undermined constant k (Problem 20). (Such a prior is *improper*; it does not integrate to unity, and hence cannot strictly be interpreted as representing our actual prior belief.) In general, then, if there is known or assumed invariance — such as translation, or for discrete distributions invariance to the sequential order of data selection — there will be constraints on the form of the prior. If we can find a prior that satisfies such constraints, the resulting prior is "non-informative" with respect to that invariance.

IMPROPER PRIOR

It is tempting to assert that the use of non-informative priors is somehow "objective" and lets the data speak for themselves, but such a view is a bit naive. For

example, we may seek a non-informative prior when estimating the standard deviation σ of a Gaussian. But this requirement might not lead to the non-informative prior for estimating the variance, σ^2. Which should we use? In fact, the greatest benefit of this approach is that it forces the designer to acknowledge and be clear about the assumed invariance — the choice of which generally lies outside our methodology. It may be more difficult to accommodate such arbitrary transformations in a maximum a posteriori (MAP) estimator (Sec. 3.2.1), and hence considerations of invariance are of greatest use in Bayesian estimation, or when the posterior is very strongly peaked and the mode not influenced by transformations of the density (Problem 19).

3.6 *Sufficient Statistics

From a practical viewpoint, the formal solution provided by Eqs. 26, 51 & 52 is not computationally attractive. In pattern recognition applications it is not unusual to have dozens or hundreds of parameters and thousands of training samples, which makes the direct computation and tabulation of $p(\mathcal{D}|\boldsymbol{\theta})$ or $p(\boldsymbol{\theta}|\mathcal{D})$ quite out of the question. We shall see in Chap. **??** how neural network methods avoid many of the difficulties of setting such a large number of parameters in a classifier, but for now we note that the only hope for an analytic, computationally feasible maximum likelihood solution lies in being able to find a parametric form for $p(\mathbf{x}|\boldsymbol{\theta})$ that on the one hand matches the characteristics of the problem and on the other hand allows a reasonably tractable solution.

Consider the simplification that occurred in the problem of learning the parameters of a multivariate Gaussian density. The basic data processing required was merely the computation of the sample mean and sample covariance. This easily computed and easily updated statistic contained all the information in the samples relevant to estimating the unknown population mean and covariance. One might suspect that this simplicity is just one more happy property of the normal distribution, and that such good fortune is not likely to occur in other cases. While this is largely true, there are distributions for which computationally feasible solutions can be obtained, and the key to their simplicity lies in the notion of a sufficient statistic.

To begin with, any function of the samples is a statistic. Roughly speaking, a *sufficient statistic* is a (possibly vector-valued) function $\mathbf{s}$ of the samples $\mathcal{D}$ that contains all of the information relevant to estimating some parameter $\boldsymbol{\theta}$. Intuitively, one might expect the definition of a sufficient statistic to involve the requirement that $p(\boldsymbol{\theta}|\mathbf{s},\mathcal{D}) = p(\boldsymbol{\theta}|\mathbf{s})$. However, this would require treating $\boldsymbol{\theta}$ as a random variable, limiting the definition to a Bayesian domain. To avoid such a limitation, the conventional definition is as follows: A statistic $\mathbf{s}$ is said to be *sufficient* for $\boldsymbol{\theta}$ if $p(\mathcal{D}|\mathbf{s},\boldsymbol{\theta})$ is independent of $\boldsymbol{\theta}$. If we think of $\boldsymbol{\theta}$ as a random variable, we can write

$$p(\boldsymbol{\theta}|\mathbf{s},\mathcal{D}) = \frac{p(\mathcal{D}|\mathbf{s},\boldsymbol{\theta})p(\boldsymbol{\theta}|\mathbf{s})}{p(\mathcal{D}|\mathbf{s})}, \tag{56}$$

whereupon it becomes evident that $p(\boldsymbol{\theta}|\mathbf{s},\mathcal{D}) = p(\boldsymbol{\theta}|\mathbf{s})$ if $\mathbf{s}$ is sufficient for $\boldsymbol{\theta}$. Conversely, if $\mathbf{s}$ is a statistic for which $p(\boldsymbol{\theta}|\mathbf{s},\mathcal{D}) = p(\boldsymbol{\theta}|\mathbf{s})$, and if $p(\boldsymbol{\theta}|\mathbf{s}) \neq 0$, it is easy to show that $p(\mathcal{D}|\mathbf{s},\boldsymbol{\theta})$ is independent of $\boldsymbol{\theta}$ (Problem 27). Thus, the intuitive and the conventional definitions are basically equivalent. As one might expect, for a Gaussian distribution the sample mean and covariance, taken together, represent a sufficient statistic for the true mean and covariance; if these are known, all other statistics

such as the mode, range, higher-order moments, number of data points, etc., are superfluous when estimating the true mean and covariance.

A fundamental theorem concerning sufficient statistics is the *Factorization Theorem*, which states that $\mathbf{s}$ is sufficient for $\boldsymbol{\theta}$ if and only if $p(\mathcal{D}|\boldsymbol{\theta})$ can be factored into the product of two functions, one depending only on $\mathbf{s}$ and $\boldsymbol{\theta}$, and the other depending only on the training samples. The virtue of the Factorization Theorem is that it allows us to shift our attention from the rather complicated density $p(\mathcal{D}|\mathbf{s},\boldsymbol{\theta})$, used to define a sufficient statistic, to the simpler function

$$p(\mathcal{D}|\boldsymbol{\theta}) = \prod_{k=1}^{n} p(\mathbf{x}_k|\boldsymbol{\theta}). \tag{57}$$

In addition, the Factorization Theorem makes it clear that the characteristics of a sufficient statistic are completely determined by the density $p(\mathbf{x}|\boldsymbol{\theta})$, and have nothing to do with a felicitous choice of an a priori density $p(\boldsymbol{\theta})$. A proof of the Factorization Theorem in the continuous case is somewhat tricky because degenerate situations are involved. Since the proof has some intrinsic interest, however, we include one for the simpler discrete case.

Theorem 3.1 (Factorization) *A statistic* $\mathbf{s}$ *is sufficient for* $\boldsymbol{\theta}$ *if and only if the probability* $P(\mathcal{D}|\boldsymbol{\theta})$ *can be written as the product*

$$P(\mathcal{D}|\boldsymbol{\theta}) = g(\mathbf{s},\boldsymbol{\theta})h(\mathcal{D}), \tag{58}$$

for some function $h(\cdot)$.

Proof:

(a) We begin by showing the "if" part of the theorem. Suppose first that $\mathbf{s}$ is sufficient for $\boldsymbol{\theta}$, so that $P(\mathcal{D}|\mathbf{s},\boldsymbol{\theta})$ is independent of $\boldsymbol{\theta}$. Since we want to show that $P(\mathcal{D}|\boldsymbol{\theta})$ can be factored, our attention is directed toward computing $P(\mathcal{D}|\boldsymbol{\theta})$ in terms of $P(\mathcal{D}|\mathbf{s},\boldsymbol{\theta})$. We do this by summing the joint probability $P(\mathcal{D},\mathbf{s}|\boldsymbol{\theta})$ over all values of $\mathbf{s}$:

$$\begin{aligned} P(\mathcal{D}|\boldsymbol{\theta}) &= \sum_{\mathbf{s}} P(\mathcal{D},\mathbf{s}|\boldsymbol{\theta}) \\ &= \sum_{\mathbf{s}} P(\mathcal{D}|\mathbf{s},\boldsymbol{\theta})P(\mathbf{s}|\boldsymbol{\theta}). \end{aligned} \tag{59}$$

But since $\mathbf{s} = \varphi(\mathcal{D})$ for some $\varphi(\cdot)$, there is only one possible value for $\mathbf{s}$ for the given data, and thus

$$P(\mathcal{D}|\boldsymbol{\theta}) = P(\mathcal{D}|\mathbf{s},\boldsymbol{\theta})P(\mathbf{s}|\boldsymbol{\theta}). \tag{60}$$

Moreover, since by hypothesis $P(\mathcal{D}|\mathbf{s},\boldsymbol{\theta})$ is independent of $\boldsymbol{\theta}$, the first factor depends only on $\mathcal{D}$. Identifying $P(\mathbf{s}|\boldsymbol{\theta})$ with $g(\mathbf{s},\boldsymbol{\theta})$, we see that $P(\mathcal{D}|\boldsymbol{\theta})$ factors, as desired.
(b) We now consider the "only if" part of the theorem. To show that the ability to factor $P(\mathcal{D}|\boldsymbol{\theta})$ as the product $g(\mathbf{s},\boldsymbol{\theta})h(\mathcal{D})$ implies that $\mathbf{s}$ is sufficient for $\boldsymbol{\theta}$, we must show that such a factoring implies that the conditional probability $P(\mathcal{D}|\mathbf{s},\boldsymbol{\theta})$ is independent of $\boldsymbol{\theta}$. Because $\mathbf{s} = \varphi(\mathcal{D})$, specifying a value for $\mathbf{s}$ constrains the possible sets of samples to some set $\bar{\mathcal{D}}$. Formally, $\bar{\mathcal{D}} = \{\mathcal{D}|\varphi(\mathcal{D}) = \mathbf{s}\}$. If $\bar{\mathcal{D}}$ is empty, no assignment of values to the samples can yield that value of $\mathbf{s}$, and $P(\mathbf{s}|\boldsymbol{\theta}) = 0$. Excluding such cases, i.e., considering only values of $\mathbf{s}$ that can arise, we have

$$P(\mathcal{D}|\mathbf{s},\boldsymbol{\theta}) = \frac{P(\mathcal{D},\mathbf{s}|\boldsymbol{\theta})}{P(\mathbf{s}|\boldsymbol{\theta})}. \tag{61}$$

The denominator can be computed by summing the numerator over all values of $\mathcal{D}$. Since the numerator will be zero if $\mathcal{D} \notin \bar{\mathcal{D}}$, we can restrict the summation to $\mathcal{D} \in \bar{\mathcal{D}}$. That is,

$$P(\mathcal{D}|\mathbf{s},\boldsymbol{\theta}) = \frac{P(\mathcal{D}|\mathbf{s},\boldsymbol{\theta})}{\sum_{\mathcal{D}\in\bar{\mathcal{D}}} P(\mathcal{D}|\mathbf{s},\boldsymbol{\theta})} = \frac{P(\mathcal{D}|\boldsymbol{\theta})}{\sum_{\mathcal{D}\in\bar{\mathcal{D}}} P(\mathcal{D}|\boldsymbol{\theta})} = \frac{g(\mathbf{s},\boldsymbol{\theta})h(\mathcal{D})}{\sum_{\mathcal{D}\in\bar{\mathcal{D}}} g(\mathbf{s},\boldsymbol{\theta})h(\mathcal{D})} = \frac{h(\mathcal{D})}{\sum_{\mathcal{D}\in\bar{\mathcal{D}}} h(\mathcal{D})}, \tag{62}$$

which is independent of $\boldsymbol{\theta}$. Thus, by definition, $\mathbf{s}$ is sufficient for $\boldsymbol{\theta}$. ∎

It should be pointed out that there are trivial ways of constructing sufficient statistics. For example we can define $\mathbf{s}$ to be a vector whose components are the n samples themselves: $\mathbf{x}_1, ..., \mathbf{x}_n$. In that case $g(\mathbf{s},\boldsymbol{\theta}) = p(\mathcal{D}|\boldsymbol{\theta})$ and $h(\mathcal{D}) = 1$. One can even produce a scalar sufficient statistic by the trick of interleaving the digits in the decimal expansion of the components of the n samples. Sufficient statistics such as these are of little interest, since they do not provide us with simpler results. The ability to factor $p(\mathcal{D}|\boldsymbol{\theta})$ into a product $g(\mathbf{s},\boldsymbol{\theta})h(\mathcal{D})$ is interesting only when the function g and the sufficient statistic $\mathbf{s}$ are simple. It should be noted that sufficiency is an integral notion. That is, if $\mathbf{s}$ is a sufficient statistic for $\boldsymbol{\theta}$, this does not necessarily imply that their corresponding components are sufficient, i.e., that s_1 is sufficient for θ_1, or s_2 for θ_2, and so on (Problem 26).

An obvious fact should also be mentioned: the factoring of $p(\mathcal{D}|\boldsymbol{\theta})$ into $g(\mathbf{s},\boldsymbol{\theta})h(\mathcal{D})$ is not unique. If $f(\mathbf{s})$ is any function of $\mathbf{s}$, then $g'(\mathbf{s},\boldsymbol{\theta}) = f(\mathbf{s})g(\mathbf{s},\boldsymbol{\theta})$ and $h'(\mathcal{D}) = h(\mathcal{D})/f(\mathbf{s})$ are equivalent factors. This kind of ambiguity can be eliminated by defining the *kernel density*

KERNEL DENSITY

$$\bar{g}(\mathbf{s},\boldsymbol{\theta}) = \frac{g(\mathbf{s},\boldsymbol{\theta})}{\int g(\mathbf{s},\boldsymbol{\theta})\, d\boldsymbol{\theta}} \tag{63}$$

which is invariant to this kind of scaling.

What is the importance of sufficient statistics and kernel densities for parameter estimation? The general answer is that the most practical applications of classical parameter estimation to pattern classification involve density functions that possess simple sufficient statistics and simple kernel densities. Moreover, it can be shown that for any clasification rule, we can find another based solely on sufficient statistics that has equal or better performance. Thus — in principle at least — we need only consider decisions based on sufficient statistics. It is, in essence, the ultimate in data reduction: we can reduce an extremely large data set down to a few numbers — the sufficient statistics — confident that all relevant information has been preserved. This means, too, that we can always create the Bayes classifier from sufficient statistics, as for example our Bayes classifiers for Gaussian distributions were functions solely of the sufficient statistics, estimates of $\boldsymbol{\mu}$ and $\boldsymbol{\Sigma}$.

In the case of maximum likelihood estimation, when searching for a value of $\boldsymbol{\theta}$ that maximizes $p(\mathcal{D}|\boldsymbol{\theta}) = g(\mathbf{s},\boldsymbol{\theta})h(\mathcal{D})$, we can restrict our attention to $g(\mathbf{s},\boldsymbol{\theta})$. In this case, the normalization provided by Eq. 63 is of no particular value unless $\bar{g}(\mathbf{s},\boldsymbol{\theta})$ is simpler than $g(\mathbf{s},\boldsymbol{\theta})$. The significance of the kernel density is revealed however in the Bayesian case. If we substitute $p(\mathcal{D}|\boldsymbol{\theta}) = g(\mathbf{s},\boldsymbol{\theta})h(\mathcal{D})$ in Eq. 51, we obtain

$$p(\boldsymbol{\theta}|\mathcal{D}) = \frac{g(\mathbf{s},\boldsymbol{\theta})p(\boldsymbol{\theta})}{\int g(\mathbf{s},\boldsymbol{\theta})p(\boldsymbol{\theta})\ d\boldsymbol{\theta}}. \tag{64}$$

If our prior knowledge of $\boldsymbol{\theta}$ is very vague, $p(\boldsymbol{\theta})$ will tend to be uniform, or changing very slowly as a function of $\boldsymbol{\theta}$. For such an essentially uniform $p(\boldsymbol{\theta})$, Eq. 64 shows that $p(\boldsymbol{\theta}|\mathcal{D})$ is approximately the same as the kernel density. Roughly speaking, the kernel density is the posterior distribution of the parameter vector when the prior distribution is uniform. Even when the a priori distribution is far from uniform, the kernel density typically gives the asymptotic distribution of the parameter vector. In particular, when $p(\mathbf{x}|\boldsymbol{\theta})$ is identifiable and when the number of samples is large, $g(\mathbf{s},\boldsymbol{\theta})$ usually peaks sharply at some value $\boldsymbol{\theta} = \hat{\boldsymbol{\theta}}$. If the a priori density $p(\boldsymbol{\theta})$ is continuous at $\boldsymbol{\theta} = \hat{\boldsymbol{\theta}}$ and if $p(\hat{\boldsymbol{\theta}})$ is not zero, $p(\boldsymbol{\theta}|\mathcal{D})$ will approach the kernel density $\bar{g}(\mathbf{s},\boldsymbol{\theta})$.

3.6.1 Sufficient Statistics and the Exponential Family

To see how the Factorization Theorem can be used to obtain sufficient statistics, consider once again the familiar d-dimensional normal case with fixed covariance but unknown mean, i.e., $p(\mathbf{x}|\boldsymbol{\theta}) \sim N(\boldsymbol{\theta},\boldsymbol{\Sigma})$. Here we have

$$\begin{aligned} p(\mathcal{D}|\boldsymbol{\theta}) &= \prod_{k=1}^{n} \frac{1}{(2\pi)^{d/2}|\boldsymbol{\Sigma}|^{1/2}} \exp\Big[-\frac{1}{2}(\mathbf{x}_k-\boldsymbol{\theta})^t\boldsymbol{\Sigma}^{-1}(\mathbf{x}_k-\boldsymbol{\theta})\Big] \\ &= \frac{1}{(2\pi)^{nd/2}|\boldsymbol{\Sigma}|^{n/2}} \exp\Big[-\frac{1}{2}\sum_{k=1}^{n}(\boldsymbol{\theta}^t\boldsymbol{\Sigma}^{-1}\boldsymbol{\theta} - 2\boldsymbol{\theta}^t\boldsymbol{\Sigma}^{-1}\mathbf{x}_k + \mathbf{x}_k^t\boldsymbol{\Sigma}^{-1}\mathbf{x}_k)\Big] \\ &= \exp\left[-\frac{n}{2}\boldsymbol{\theta}^t\boldsymbol{\Sigma}^{-1}\boldsymbol{\theta} + \boldsymbol{\theta}^t\boldsymbol{\Sigma}^{-1}\left(\sum_{k=1}^{n}\mathbf{x}_k\right)\right] \\ &\quad \times\frac{1}{(2\pi)^{nd/2}|\boldsymbol{\Sigma}|^{n/2}} \exp\left[-\frac{1}{2}\sum_{k=1}^{n}\mathbf{x}_k^t\boldsymbol{\Sigma}^{-1}\mathbf{x}_k\right]. \end{aligned} \tag{65}$$

This factoring isolates the $\boldsymbol{\theta}$ dependence of $p(\mathcal{D}|\boldsymbol{\theta})$ in the first term, and hence from the Factorization Theorem we conclude that $\sum_{k=1}^{n}\mathbf{x}_k$ is sufficient for $\boldsymbol{\theta}$. Of course, any one-to-one function of this statistic is also sufficient for $\boldsymbol{\theta}$; in particular, the sample mean

$$\hat{\boldsymbol{\mu}}_n = \frac{1}{n}\sum_{k=1}^{n}\mathbf{x}_k \tag{66}$$

is also sufficient for $\boldsymbol{\theta}$. Using this statistic, we can write

$$g(\hat{\boldsymbol{\mu}}_n,\boldsymbol{\theta}) = \exp\Big[-\frac{n}{2}\big(\boldsymbol{\theta}^t\boldsymbol{\Sigma}^{-1}\boldsymbol{\theta} - 2\boldsymbol{\theta}^t\boldsymbol{\Sigma}^{-1}\hat{\boldsymbol{\mu}}_n\big)\Big]. \tag{67}$$

From using Eq. 63, or by completing the square, we can obtain the kernel density:

$$\bar{g}(\hat{\boldsymbol{\mu}}_n,\boldsymbol{\theta}) = \frac{1}{(2\pi)^{d/2}|\frac{1}{n}\boldsymbol{\Sigma}|^{1/2}} \exp\Big[-\frac{1}{2}(\boldsymbol{\theta}-\hat{\boldsymbol{\mu}}_n)^t\Big(\frac{1}{n}\boldsymbol{\Sigma}\Big)^{-1}(\boldsymbol{\theta}-\hat{\boldsymbol{\mu}}_n)\Big]. \tag{68}$$

These results make it immediately clear that $\hat{\boldsymbol{\mu}}_n$ is the maximum likelihood estimate for $\boldsymbol{\theta}$. The Bayesian posterior density can be obtained from $\bar{g}(\hat{\boldsymbol{\mu}}_n,\boldsymbol{\theta})$ by performing

the integration indicated in Eq. 64. If the a priori density is essentially uniform, $p(\boldsymbol{\theta}|\mathcal{D}) = \bar{g}(\hat{\boldsymbol{\mu}}_n, \boldsymbol{\theta})$.

This same general approach can be used to find sufficient statistics for other density functions. In particular, it applies to any member of the *exponential family*, a group of probability and probability density functions that possess simple sufficient statistics. Members of the exponential family include the Gaussian, exponential, Rayleigh, Poisson, and many other familiar distributions. They can all be written in the form

$$p(\mathbf{x}|\boldsymbol{\theta}) = \alpha(\mathbf{x}) \exp\left[\mathbf{a}(\boldsymbol{\theta}) + \mathbf{b}(\boldsymbol{\theta})^t \mathbf{c}(\mathbf{x})\right]. \tag{69}$$

If we multiply n terms of the form in Eq. 69 we find

$$p(\mathcal{D}|\boldsymbol{\theta}) = \exp\left[n\mathbf{a}(\boldsymbol{\theta}) + \mathbf{b}(\boldsymbol{\theta})^t \sum_{k=1}^{n} \mathbf{c}(\mathbf{x}_k)\right] \prod_{k=1}^{n} \alpha(\mathbf{x}_k) = g(\mathbf{s}, \boldsymbol{\theta}) h(\mathcal{D}), \tag{70}$$

where we can take

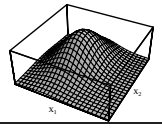
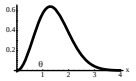
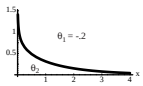
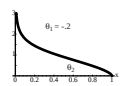
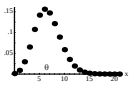
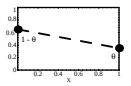
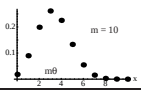
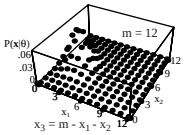
$$\begin{aligned} \mathbf{s} &= \frac{1}{n} \sum_{k=1}^{n} \mathbf{c}(\mathbf{x}), \\ g(\mathbf{s}, \boldsymbol{\theta}) &= \exp\left[n\{\mathbf{a}(\boldsymbol{\theta}) + \mathbf{b}(\boldsymbol{\theta})^t \mathbf{s}\}\right], \end{aligned}$$

and

$$h(\mathcal{D}) = \prod_{k=1}^{n} \alpha(\mathbf{x}_k).$$

The distributions, sufficient statistics, and unnormalized kernels for a number of commonly encountered members of the exponential family are given in Table **??**. It is a fairly routine matter to derive maximum likelihood estimates and Bayesian a posteriori distributions from these solutions. With two exceptions, the solutions given are for univariate cases, though they can be used in multivariate situations if statistical independence can be assumed. Note that a few well-known probability distributions, such as the Cauchy, do *not* have sufficient statistics, so that the sample mean can be a very poor estimator of the true mean (Problem 28).

Table 3.1: Common Exponential Distributions and their Sufficient Statistics.

Name	Distribution	Domain		s	$[g(\mathbf{s},\boldsymbol{\theta})]^{1/n}$
Normal	$p(x\|\boldsymbol{\theta}) = \sqrt{\frac{\theta_2}{2\pi}} e^{-(1/2)\theta_2(x-\theta_1)^2}$	$\theta_2 > 0$		$\begin{bmatrix} \frac{1}{n}\sum_{k=1}^{n} x_k \\ \frac{1}{n}\sum_{k=1}^{n} x_k^2 \end{bmatrix}$	$\sqrt{\theta_2} e^{-\frac{1}{2}\theta_2(s_2 - 2\theta_1 s_1 + \theta_1^2)}$
Multivariate Normal	$p(\mathbf{x}\|\boldsymbol{\theta}) = \frac{\|\boldsymbol{\Theta}_2\|^{1/2}}{(2\pi)^{d/2}} e^{-(1/2)(\mathbf{x}-\boldsymbol{\theta}_1)^t \boldsymbol{\Theta}_2 (\mathbf{x}-\boldsymbol{\theta}_1)}$	$\boldsymbol{\Theta}_2$ positive definite		$\begin{bmatrix} \frac{1}{n}\sum_{k=1}^{n} \mathbf{x}_k \\ \frac{1}{n}\sum_{k=1}^{n} \mathbf{x}_k \mathbf{x}_k^t \end{bmatrix}$	$\|\boldsymbol{\Theta}_2\|^{1/2} e^{-\frac{1}{2}[\mathrm{tr}\boldsymbol{\Theta}_2 \mathbf{s}_2 - 2\boldsymbol{\theta}_1^t \boldsymbol{\Theta}_2 \mathbf{s}_1 + \boldsymbol{\theta}_1^t \boldsymbol{\Theta}_2 \boldsymbol{\theta}_1]}$
Exponential	$p(x\|\theta) = \begin{cases} \theta e^{-\theta x} & x \geq 0 \\ 0 & \text{otherwise} \end{cases}$	$\theta > 0$		$\frac{1}{n}\sum_{k=1}^{n} x_k$	$\theta e^{-\theta s}$
Rayleigh	$p(x\|\theta) = \begin{cases} 2\theta x e^{-\theta x^2} & x \geq 0 \\ 0 & \text{otherwise} \end{cases}$	$\theta > 0$		$\frac{1}{n}\sum_{k=1}^{n} x_k^2$	$\theta e^{-\theta s}$
Maxwell	$p(x\|\theta) = \begin{cases} \frac{4}{\sqrt{\pi}} \theta^{3/2} x^2 e^{-\theta x^2} & x \geq 0 \\ 0 & \text{otherwise} \end{cases}$	$\theta > 0$		$\frac{1}{n}\sum_{k=1}^{n} x_k^2$	$\theta^{3/2} e^{-\theta s}$
Gamma	$p(x\|\boldsymbol{\theta}) = \begin{cases} \frac{\theta_2^{\theta_1+1}}{\Gamma(\theta_1+1)} x^{\theta_1} e^{-\theta_2 x} & x \geq 0 \\ 0 & \text{otherwise} \end{cases}$	$\theta_1 > -1$ $\theta_2 > 0$		$\begin{bmatrix} \left(\prod_{k=1}^{n} x_k\right)^{1/n} \\ \frac{1}{n}\sum_{k=1}^{n} x_k \end{bmatrix}$	$\frac{\theta_2^{\theta_1+1}}{\Gamma(\theta_1+1)} s_1^{\theta_1} e^{-\theta_2 s_2}$
Beta	$p(x\|\boldsymbol{\theta}) = \begin{cases} \frac{\Gamma(\theta_1+\theta_2+2)}{\Gamma(\theta_1+1)\Gamma(\theta_2+1)} x^{\theta_1}(1-x)^{\theta_2} & 0 \leq x \leq 1 \\ 0 & \text{otherwise} \end{cases}$	$\theta_1 > -1$ $\theta_2 > -1$		$\begin{bmatrix} \left(\prod_{k=1}^{n} x_k\right)^{1/n} \\ \left(\prod_{k=1}^{n} (1-x_k)\right)^{1/n} \end{bmatrix}$	$\frac{\Gamma(\theta_1+\theta_2+2)}{\Gamma(\theta_1+1)\Gamma(\theta_2+1)} s_1^{\theta_1} s_2^{\theta_2}$
Poisson	$P(x\|\theta) = \frac{\theta^x}{x!} e^{-\theta} \quad x = 0, 1, 2, ...$	$\theta > 0$		$\frac{1}{n}\sum_{k=1}^{n} x_k$	$\theta^s e^{-\theta}$
Bernoulli	$P(x\|\theta) = \theta^x (1-\theta)^{1-x} \quad x = 0, 1$	$0 < \theta < 1$		$\frac{1}{n}\sum_{k=1}^{n} x_k$	$\theta^s (1-\theta)^{1-s}$
Binomial	$P(x\|\theta) = \frac{m!}{x!(m-x)!} \theta^x (1-\theta)^{m-x}$ $x = 0, 1, ..., m$	$0 < \theta < 1$		$\frac{1}{n}\sum_{k=1}^{n} x_k$	$\theta^s (1-\theta)^{m-s}$
Multinomial	$P(\mathbf{x}\|\boldsymbol{\theta}) = \frac{m! \prod_{i=1}^{d} \theta_i^{x_i}}{\prod_{i=1}^{d} x_i!}$ $x_i = 0, 1, ..., m$ $\sum_{i=1}^{d} x_i = m$	$0 < \theta_i < 1$ $\sum_{i=1}^{d} \theta_i = 1$		$\frac{1}{n}\sum_{k=1}^{n} \mathbf{x}_k$	$\prod_{i=1}^{d} \theta_i^{s_i}$

3.7 Problems of Dimensionality

In practical multicategory applications, it is not at all unusual to encounter problems involving fifty or a hundred features, particularly if the features are binary valued. We might typically believe that each feature is useful for at least some of the discriminations; while we may doubt that each feature provides independent information, intentionally superfluous features have not been included. There are two issues that must be confronted. The most important is how classification accuracy depends upon the dimensionality (and amount of training data); the second is the computational complexity of designing the classifier.

3.7.1 Accuracy, Dimension, and Training Sample Size

If the features are statistically independent, there are some theoretical results that suggest the possibility of excellent performance. For example, consider the two-class multivariate normal case with the same covariance where $p(\mathbf{x}|\omega_j) \sim N(\boldsymbol{\mu}_j, \boldsymbol{\Sigma})$, $j = 1, 2$. If the a priori probabilities are equal, then it is not hard to show (Chap. **??**, Problem **??**) that the Bayes error rate is given by

$$P(e) = \frac{1}{\sqrt{2\pi}} \int_{r/2}^{\infty} e^{-u^2/2}\, du, \tag{71}$$

where r^2 is the squared Mahalanobis distance (Chap. **??**, Sect. **??**):

$$r^2 = (\boldsymbol{\mu}_1 - \boldsymbol{\mu}_2)^t \boldsymbol{\Sigma}^{-1} (\boldsymbol{\mu}_1 - \boldsymbol{\mu}_2). \tag{72}$$

Thus, the probability of error decreases as r increases, approaching zero as r approaches infinity. In the conditionally independent case, $\boldsymbol{\Sigma} = diag(\sigma_1^2, ..., \sigma_d^2)$, and

$$r^2 = \sum_{i=1}^{d} \left(\frac{\mu_{i1} - \mu_{i2}}{\sigma_i} \right)^2. \tag{73}$$

This shows how each feature contributes to reducing the probability of error. Naturally, the most useful features are the ones for which the difference between the means is large relative to the standard deviations. However no feature is useless if its means for the two classes differ. An obvious way to reduce the error rate further is to introduce new, independent features. Each new feature need not add much, but if r can be increased without limit, the probability of error can be made arbitrarily small.

In general, if the performance obtained with a given set of features is inadequate, it is natural to consider adding new features, particularly ones that will help separate the class pairs most frequently confused. Although increasing the number of features increases the cost and complexity of both the feature extractor and the classifier, it is often reasonable to believe that the performance will improve. After all, if the probabilistic structure of the problem were completely known, the Bayes risk could not possibly be increased by adding new features. At worst, the Bayes classifer would ignore the new features, but if the new features provide any additional information, the performance must improve (Fig. 3.3).

Unfortunately, it has frequently been observed in practice that, beyond a certain point, the inclusion of additional features leads to worse rather than better performance. This apparent paradox presents a genuine and serious problem for classifier

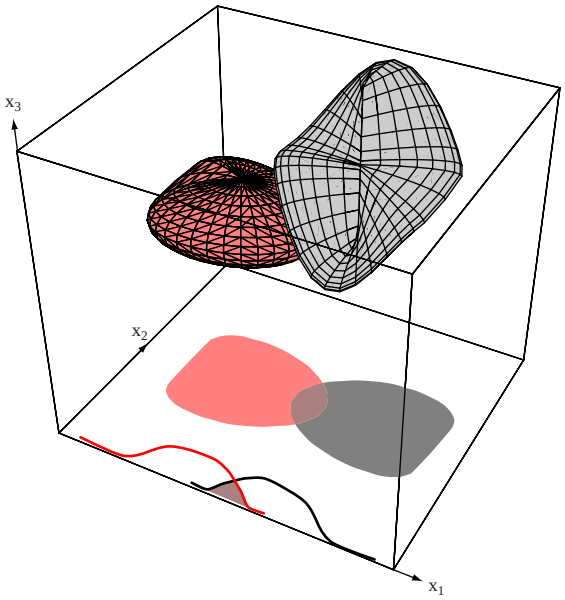


Figure 3.3: Two three-dimensional distributions have nonoverlapping densities, and thus in three dimensions the Bayes error vanishes. When projected to a subspace — here, the two-dimensional $x_1 - x_2$ subspace or a one-dimensional x_1 subspace — there can be greater overlap of the projected distributions, and hence greater Bayes errors.

design. The basic source of the difficulty can always be traced to the fact that we have the wrong model — e.g., the Gaussian assumption or conditional assumption are wrong — or the number of design or training samples is finite and thus the distributions are not estimated accurately. However, analysis of the problem is both challenging and subtle. Simple cases do not exhibit the experimentally observed phenomena, and more realistic cases are difficult to analyze. In an attempt to provide some rigor, we shall return to topics related to problems of dimensionality and sample size in Chap. **??**.

3.7.2 Computational Complexity

We have mentioned that one consideration affecting our design methodology is that of the computational difficulty, and here the technical notion of computational complexity can be useful. First, we will will need to understand the notion of the *order* of a function $f(x)$: we say that the $f(x)$ is "of the order of $h(x)$" — written $f(x) = O(h(x))$ and generally read "big oh of $h(x)$" — if there exist constants c_0 and x_0 such that $|f(x)| \leq c_0|h(x)|$ for all $x > x_0$. This means simply that for sufficiently large x, an upper bound on the function grows no worse than $h(x)$. For instance, suppose $f(x) = a_0 + a_1x + a_2x^2$; in that case we have $f(x) = O(x^2)$ because for sufficiently large x, the constant, linear and quadratic terms can be "overcome" by proper choice of c_0 and x_0. The generalization to functions of two or more variables is straightforward. It should be clear that by the definition above, the big oh order of a function is not unique. For instance, we can describe our particular $f(x)$ as being $O(x^2)$, $O(x^3)$, $O(x^4)$, $O(x^2 \ln x)$.

ORDER

BIG OH

Because of the non-uniqueness of the big oh notation, we occasionally need to be

more precise in describing the order of a function. We say that $f(x) = \mathbf{\Theta}(h(x))$ "big theta of $h(x)$" if there are constants x_0, c_1 and c_2 such that for $x > x_0$, $f(x)$ always lies between $c_1 h(x)$ and $c_2 h(x)$. Thus our simple quadratic function above would obey $f(x) = \mathbf{\Theta}(x^2)$, but would *not* obey $f(x) = \mathbf{\Theta}(x^3)$. (A fuller explanation is provided in the Appendix.)

In describing the computational complexity of an algorithm we are generally interested in the number of basic mathematical operations, such as additions, multiplications and divisions it requires, or in the time and memory needed on a computer. To illustrate this concept we consider the complexity of a maximum likelihood estimation of the parameters in a classifier for Gaussian priors in d dimensions, with n training samples for each of c categories. For each category it is necessary to calculate the discriminant function of Eq. 74, below. The computational complexity of finding the sample mean $\hat{\boldsymbol{\mu}}$ is $O(nd)$, since for each of the d dimensions we must add n component values. The required division by n in the mean calculation is a single computation, independent of the number of points, and hence does not affect this complexity. For each of the $d(d+1)/2$ independent components of the sample covariance matrix $\widehat{\mathbf{\Sigma}}$ there are n multiplications and additions (Eq. 19), giving a complexity of $O(d^2 n)$. Once $\widehat{\mathbf{\Sigma}}$ has been computed, its determinant is an $O(d^2)$ calculation, as we can easily verify by counting the number of operations in matrix "sweep" methods. The inverse can be calculated in $O(d^3)$ calculations, for instance by Gaussian elimination.* The complexity of estimating $P(\omega)$ is of course $O(n)$. Equation 74 illustrates these individual components for the problem of setting the parameters of normal distributions via maximum lielihood:

$$g(\mathbf{x}) = -\frac{1}{2}(\mathbf{x} - \overbrace{\hat{\boldsymbol{\mu}}}^{O(dn)})^t \overbrace{\widehat{\mathbf{\Sigma}}^{-1}}^{O(nd^2)} (\mathbf{x} - \hat{\boldsymbol{\mu}}) - \overbrace{\frac{d}{2} \ln 2\pi}^{O(1)} - \overbrace{\frac{1}{2} \ln |\widehat{\mathbf{\Sigma}}|}^{O(d^2 n)} + \overbrace{\ln P(\omega)}^{O(n)}. \tag{74}$$

Naturally we assume that $n > d$ (otherwise our covariance matrix will not have a well defined inverse), and thus for large problems the overall complexity of calculating an individual discriminant function is dominated by the $O(d^2 n)$ term in Eq. 74. This is done for each of the categories, and hence our overall computational complexity for learning in this Bayes classifer is $O(cd^2 n)$. Since c is typically a constant much smaller than d^2 or n, we can call our complexity $O(d^2 n)$. We saw in Sect. 3.7 that it was generally desirable to have more training data from a larger dimensional space; our complexity analysis shows the steep cost in so doing.

We next reconsider the matter of estimating a covariance matrix in a bit more detail. This requires the estimation of $d(d+1)/2$ parameters — the d diagonal elements and $d(d-1)/2$ independent off-diagonal elements. We observe first that the appealing maximum likelihood estimate

$$\widehat{\mathbf{\Sigma}} = \frac{1}{n} \sum_{k=1}^{n} (\mathbf{x}_k - \mathbf{m}_n)(\mathbf{x}_k - \mathbf{m}_n)^t, \tag{75}$$

is an $O(nd^2)$ calculation, is the sum of $n-1$ independent d-by-d matrices of rank one, and thus is guaranteed to be singular if $n \leq d$. Since we must invert $\widehat{\mathbf{\Sigma}}$ to obtain the discriminant functions, we have an algebraic requirement for at least $d+1$ samples. To smooth our statistical fluctuations and obtain a really good estimate, it would not be surprising if several times that number of samples were needed.

* We mention for the afficionado that there are more complex matrix inversion algorithms that are $O(d^{2.376\cdots})$, and there may be algorithms with even lower complexity yet to be discovered.

The computational complexity for *classification* is less, of course. Given a test point $\mathbf{x}$ we must compute $(\mathbf{x} - \hat{\boldsymbol{\mu}})$, an $O(d)$ calculation. Moreover, for each of the categories we must multiply the inverse covariance matrix by the separation vector, an $O(d^2)$ calculation. The $\max_i g_i(\mathbf{x})$ decision is a separate $O(c)$ operation. For small c then, recall is an $O(d^2)$ operation. Here, as throughout virtually all pattern classification, recall is much simpler (and faster) than learning. The complexity of the corresponding case for Bayesian learning, summarized in Eq. 49, yields the same computational complexity as in maximum likelihood. More generally, however, Bayesian learning has higher complexity as a consequence of integrating over model parameters $\boldsymbol{\theta}$.

Such a rough analysis did not tell us the constants of proportionality. For a finite size problem it is possible (though not particularly likely) that a particular $O(n^3)$ algorithm is simpler than a particular $O(n^2)$ algorithm, and it is occasionally necessary for us to determine these constants to find which of several implemementations is the simplest. Nevertheless, big oh and big theta analyses, as just described, are generally the best way to describe the computational complexity of an algorithm.

Sometimes we stress space and time complexities, which are particularly relevant when contemplating parallel implementations. For instance, the sample mean of a category could be calculated with d separate processors, each adding n sample values. Thus we can describe this implementation as $O(d)$ in *space* (i.e., the amount of memory or possibly the number of processors) and $O(n)$ in *time* (i.e., number of sequential steps). Of course for any particular algorithm there may be a number of time-space tradeoffs, for instance using a single processor many times, or using many processors in parallel for a shorter time. Such tradeoffs are important considerations can be important in neural network implementations, as we shall see in Chap. **??**.

SPACE COMPLEXITY

TIME COMPLEXITY

A common qualitative distinction is made between *polynomially* complex and *exponentially* complex algorithms — $O(a^k)$ for some constant a and aspect or variable k of the problem. Exponential algorithms are generally so complex that for reasonable size cases we avoid them altogether, and resign ourselves to approximate solutions that can be found by polynomially complex algorithms.

3.7.3 Overfitting

It frequently happens that the number of available samples is inadequate, and the question of how to proceed arises. One possibility is to reduce the dimensionality, either by redesigning the feature extractor, by selecting an appropriate subset of the existing features, or by combining the existing features in some way (Chap **??**). Another possibility is to assume that all c classes share the same covariance matrix, and to pool the available data. Yet another alternative is to look for a better estimate for $\boldsymbol{\Sigma}$. If any reasonable a priori estimate $\boldsymbol{\Sigma}_0$ is available, a Bayesian or pseudo-Bayesian estimate of the form $\lambda\boldsymbol{\Sigma}_0 + (1 - \lambda)\widehat{\boldsymbol{\Sigma}}$ might be employed. If $\boldsymbol{\Sigma}_0$ is diagonal, this diminishes the troublesome effects of "accidental" correlations. Alternatively, one can remove chance correlations heuristically by thresholding the sample covariance matrix. For example, one might assume that all covariances for which the magnitude of the correlation coefficient is not near unity are actually zero. An extreme of this approach is to assume statistical independence, thereby making all the off-diagonal elements be zero, regardless of empirical evidence to the contrary — an $O(nd)$ calculation. Even though such assumptions are almost surely incorrect, the resulting heuristic estimates sometimes provide better performance than the maximum likelihood estimate of the full parameter space.

Here we have another apparent paradox. The classifier that results from assuming independence is almost certainly suboptimal. It is understandable that it will perform better if it happens that the features actually are independent, but how can it provide better performance when this assumption is untrue? The answer again involves the problem of insufficient data, and some insight into its nature can be gained from considering an analogous problem in curve fitting. Figure 3.4 shows a set of ten data points and two candidate curves for fitting them. The data points were obtained by adding zero-mean, independent noise to a parabola. Thus, of all the possible polynomials, presumably a parabola would provide the best fit, assuming that we are interested in fitting data obtained in the future as well as the points at hand. Even a straight line could fit the training data fairly well. The parabola provides a better fit, but one might wonder whether the data are adequate to fix the curve. The best parabola for a larger data set might be quite different, and over the interval shown the straight line could easily be superior. The tenth-degree polynomial fits the given data perfectly. However, we do not expect that a tenth-degree polynomial is required here. In general, reliable interpolation or extrapolation can not be obtained unless the solution is overdetermined, i.e., there are more points than function parameters to be set.

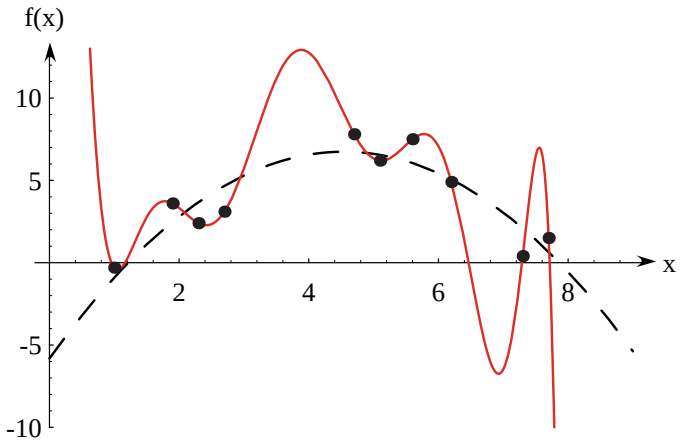


Figure 3.4: The "training data" (black dots) were selected from a quadradic function plus Gaussian noise, i.e., $f(x) = ax^2 + bx + c + \epsilon$ where $p(\epsilon) \sim N(0, \sigma^2)$. The 10th degree polynomial shown fits the data perfectly, but we desire instead the second-order function $f(x)$, since it would lead to better predictions for new samples.

In fitting the points in Fig. 3.4, then, we might consider beginning with a high-order polynomial (e.g., 10th order), and successively smoothing or simplifying our model by eliminating the highest-order terms. While this would in virtually all cases lead to greater error on the "training data," we might expect the generalization to improve.

Analogously, there are a number of heuristic methods that can be applied in the Gaussian classifier case. For instance, suppose we wish to design a classifier for distributions $N(\boldsymbol{\mu}_1, \boldsymbol{\Sigma}_1)$ and $N(\boldsymbol{\mu}_2, \boldsymbol{\Sigma}_2)$ and we have reason to believe that we have insufficient data for accurately estimating the parameters. We might make the simplification that they have the same covariance, i.e., $N(\boldsymbol{\mu}_1, \boldsymbol{\Sigma})$ and $N(\boldsymbol{\mu}_2, \boldsymbol{\Sigma})$, and estimate $\boldsymbol{\Sigma}$ accordingly. Such estimation requires proper normalization of the data

(Problem 36).

SHRINKAGE

An intermediate approach is to assume a weighted combination of the equal and individual covariances, a technique known as *shrinkage*, (also called regularized discriminant analysis) since the individual covariances "shrink" toward a common one. If i is an index on the c categories in question, we have

$$\mathbf{\Sigma}_i(\alpha) = \frac{(1-\alpha)n_i\mathbf{\Sigma}_i + \alpha n\mathbf{\Sigma}}{(1-\alpha)n_i + \alpha n}, \tag{76}$$

for $0 < \alpha < 1$. Additionally, we could "shrink" the estimate of the (assumed) common covariance matrix toward the identity matrix, as

$$\mathbf{\Sigma}(\beta) = (1-\beta)\mathbf{\Sigma} + \beta\mathbf{I}, \tag{77}$$

for $0 < \beta < 1$ (Computer exercise 8). (Such methods for simplifying classifiers have counterparts in regression, generally known as *ridge regression*.)

Our short, intuitive descussion here will have to suffice until Chap. **??**, where we will explore the crucial issue of controlling the complexity or expressive power of a classifer for optimum performance.

3.8 *Expectation-Maximization (EM)

We saw in Chap. **??** Sec. **??** how we could classify a test point even when it has missing features. We can now extend our application of maximum likelihood techniques to permit the *learning* of parameters governing a distribution from training points, some of which have missing features. If we had uncorrupted data, we could use maximum likelihood, i.e., find $\hat{\boldsymbol{\theta}}$ that maximized the log-likelihood $l(\boldsymbol{\theta})$. The basic idea in the expectation maximization or EM algorithm, is to iteratively estimate the likelihood given the data that is present. The method has precursors in the Baum-Welch algorithm we will consider in Sec. 3.10.6.

Consider a full sample $\mathcal{D} = \{\mathbf{x}_1, ..., \mathbf{x}_n\}$ of points taken from a single distribution. Suppose, though, that here some features are missing; thus any sample point can be written as $\mathbf{x}_k = \{\mathbf{x}_{kg}, \mathbf{x}_{kb}\}$, i.e., comprising the "good" features and the missing, or "bad" ones (Chapt. **??**, Sect. **??**). For notational convenience we separate these individual *features* (not samples) into two sets, $\mathcal{D}_g$ and $\mathcal{D}_b$ with $\mathcal{D} = \mathcal{D}_g \cup \mathcal{D}_b$ being the union of such features.

Next we form the function

$$Q(\boldsymbol{\theta};\ \boldsymbol{\theta}^i) = \mathcal{E}_{\mathcal{D}_b}[\ln\, p(\mathcal{D}_g, \mathcal{D}_b;\ \boldsymbol{\theta})|\mathcal{D}_g;\ \boldsymbol{\theta}^i], \tag{78}$$

where the use of the semicolon denotes, for instance on the left hand side, that $Q(\boldsymbol{\theta};\ \boldsymbol{\theta}^i)$ is a function of $\boldsymbol{\theta}$ with $\boldsymbol{\theta}^i$ assumed fixed; on the right hand side it denotes that the expected value is over the missing features assuming $\boldsymbol{\theta}^i$ are the true parameters describing the (full) distribution. The simplest way to interpret this, the central equation in expectation maximization, is the following. The parameter vector $\boldsymbol{\theta}^i$ is the current (best) estimate for the full distribution; $\boldsymbol{\theta}$ is a candidate vector for an improved estimate. Given such a candidate $\boldsymbol{\theta}$, the right hand side of Eq. 78 calculates the likelihood of the data, including the unknown feature $\mathcal{D}_b$ *marginalized* with respect to the current best distribution, which is described by $\boldsymbol{\theta}^i$. Different candidate $\boldsymbol{\theta}$s will of course lead to different such likelihoods. Our algorithm will select the best such candidate $\boldsymbol{\theta}$ and call it $\boldsymbol{\theta}^{i+1}$ — the one corresponding to the greatest $Q(\boldsymbol{\theta};\ \boldsymbol{\theta}^i)$.

If we continue to let i be an interation counter, and now let T be a preset convergence criterion, our algorithm is as follows and illustrated in Fig. 3.5:

Algorithm 1 (Expectation-Maximization)

1 **begin** **initialize** $\boldsymbol{\theta}^0, T, i = 0$
2 **do** $i \leftarrow i + 1$
3 **E step** : compute $Q(\boldsymbol{\theta};\ \boldsymbol{\theta}^i)$
5 **M step** : $\boldsymbol{\theta}^{i+1} \leftarrow \arg\max_{\theta} Q(\boldsymbol{\theta};\ \boldsymbol{\theta}^i)$
6 **until** $Q(\boldsymbol{\theta}^{i+1};\ \boldsymbol{\theta}^i) - Q(\boldsymbol{\theta}^i;\ \boldsymbol{\theta}^{i-1}) \leq T$
7 **return** $\hat{\boldsymbol{\theta}} \leftarrow \boldsymbol{\theta}^{i+1}$
8 **end**

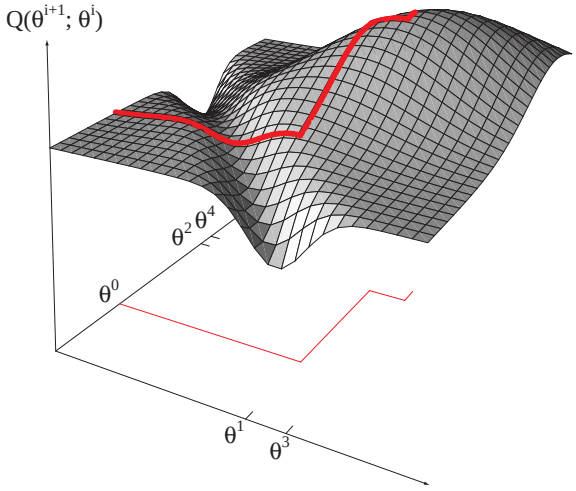


Figure 3.5: The search for the best model via the EM algorithm starts with some initial value of the model parameters, θ^0. Then, via the **M step** the optimal θ^1 is found. Next, θ^1 is held constant and the value θ^2 found which optimizes $Q(\cdot,\cdot)$. This process iterates until no value of θ can be found that will increase $Q(\cdot,\cdot)$. Note in particular that this is different from a gradient search. For example here θ^1 is the global optimum (given fixed θ^0), and would not necessarily have been found via gradient search. (In this illustration, $Q(\cdot,\cdot)$ is shown symmetric in its arguments; this need not be the case in general, however.)

This so-called Expectation-Maximization or EM algorithm is most useful when the optimization of $Q(\cdot,\cdot)$ is simpler than that of $l(\cdot)$. Most importantly, the algorithm guarantees that the log-likelihood of the good data (with the bad data marginalized) will increase monotonically, as explored in Problem 37. This is not the same as finding the particular value of the bad data that gives the maximum likelihood of the full (completed) data, as can be seen in Example 2.

Example 2: Expectation-Maximization for a 2D normal model

Suppose our data consists of four points in two dimensions, one point of which is missing a feature: $\mathcal{D} = \{\mathbf{x}_1, \mathbf{x}_2, \mathbf{x}_3, \mathbf{x}_4\} = \left\{\binom{0}{2}, \binom{1}{0}, \binom{2}{2}, \binom{*}{4}\right\}$, where $*$ represents the unknown value of the first feature of point $\mathbf{x}_4$. Thus our bad data $\mathcal{D}_b$ consists of

the single feature x_{41}, and the good data $\mathcal{D}_g$ all the rest. We assume our model is a Gaussian with diagonal covariance and arbitrary mean, and thus can be described by the parameter vector

$$\boldsymbol{\theta} = \begin{pmatrix} \mu_1 \\ \mu_2 \\ \sigma_1^2 \\ \sigma_2^2 \end{pmatrix}.$$

We take our initial guess to be a Gaussian centered on the origin having $\boldsymbol{\Sigma} = \mathbf{I}$, that is:

$$\boldsymbol{\theta}^0 = \begin{pmatrix} 0 \\ 0 \\ 1 \\ 1 \end{pmatrix}.$$

In finding our first improved estimate, $\boldsymbol{\theta}^1$, we must calculate $Q(\boldsymbol{\theta}, \boldsymbol{\theta}^0)$ or, by Eq. 78,

$$\begin{aligned} Q(\boldsymbol{\theta};\, \boldsymbol{\theta}^0) &= \mathcal{E}_{x_{41}}[\ln\, p(\mathbf{x}_g, \mathbf{x}_b;\, \boldsymbol{\theta}|\boldsymbol{\theta}^0;\, \mathcal{D}_g)] \\ &= \int_{-\infty}^{\infty} \left[\sum_{k=1}^{3} \ln p(\mathbf{x}_k|\boldsymbol{\theta}) + \ln p(\mathbf{x}_4|\boldsymbol{\theta})\right] p(x_{41}|\boldsymbol{\theta}^0;\, x_{42} = 4)\, dx_{41} \\ &= \sum_{k=1}^{3} [\ln p(\mathbf{x}_k|\boldsymbol{\theta})] + \int_{-\infty}^{\infty} \ln p\left(\begin{pmatrix} x_{41} \\ 4 \end{pmatrix}\middle|\boldsymbol{\theta}\right) \underbrace{\frac{p\left(\begin{pmatrix} x_{41} \\ 4 \end{pmatrix}|\boldsymbol{\theta}^0\right)}{\left(\int_{-\infty}^{\infty} p\left(\begin{pmatrix} x'_{41} \\ 4 \end{pmatrix}\middle|\boldsymbol{\theta}^0\right) dx'_{41}\right)}}_{\equiv K} dx_{41}, \end{aligned}$$

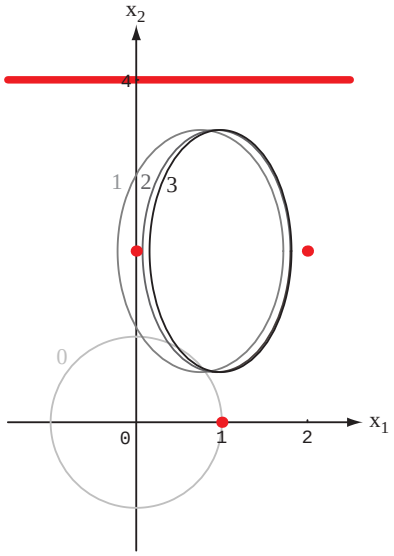
where x_{41} is the unknown first feature of point $\mathbf{x}_4$, and K is a constant that can be brought out of the integral. We focus on the integral, substitute the equation for a general Gaussian, and find

$$\begin{aligned} Q(\boldsymbol{\theta};\, \boldsymbol{\theta}^0) &= \sum_{k=1}^{3} [\ln\, p(\mathbf{x}_k|\boldsymbol{\theta})] + \frac{1}{K} \int_{-\infty}^{\infty} \ln\, p\left(\begin{pmatrix} x_{41} \\ 4 \end{pmatrix}\middle|\boldsymbol{\theta}\right) \frac{1}{2\pi \left|\begin{pmatrix} 1 & 0 \\ 0 & 1 \end{pmatrix}\right|} \exp\left[-\frac{1}{2}(x_{41}^2 + 4^2)\right] dx_{41} \\ &= \sum_{k=1}^{3} [\ln\, p(\mathbf{x}_k|\boldsymbol{\theta})] - \frac{1 + \mu_1^2}{2\sigma_1^2} - \frac{(4 - \mu_2)^2}{2\sigma_2^2} - \ln\, (2\pi\sigma_1\sigma_2). \end{aligned}$$

This completes the expectation or **E step**. Through a straightforward calculation, we find the values of $\boldsymbol{\theta}$ (that is, μ_1, μ_2, σ_1 and σ_2 that maximize $Q(,\cdot)$, to get the next estimate:

$$\boldsymbol{\theta}^1 = \begin{pmatrix} 0.75 \\ 2.0 \\ 0.938 \\ 2.0 \end{pmatrix}.$$

This new mean and the $1/e$ ellipse of the new covariance matrix are shown in the figure. Subsequent iterations are conceptually the same, but require a bit more extensive calculation. The mean will remain at $\mu_2 = 2$. After three iterations the algorithm converges at the solution $\boldsymbol{\mu} = \binom{1.0}{2.0}$, and $\boldsymbol{\Sigma} = \begin{pmatrix} 0.667 & 0 \\ 0 & 2.0 \end{pmatrix}$.



The four data points, one of which is missing the value of x_1 component, are shown in red. The initial estimate is a circularly symmetric Gaussian, centered on the origin (gray). (A better initial estimate could have been derived from the three known points.) Each iteration leads to an improved estimate, labelled by the iteration number i; here, after three iterations the algorithm has converged.

We must be careful and note that the EM algorithm leads to the greatest log-likelihood of the *good* data, with the bad data marginalized. There may be particular values of the bad data that give a different solution and an even greater log-likelihood. For instance, in this Example if the missing feature had value $x_{41} = 2$, so that $\mathbf{x}_4 = \binom{2}{4}$, we would have a solution

$$\boldsymbol{\theta} = \begin{pmatrix} 1.0 \\ 2.0 \\ 0.5 \\ 2.0 \end{pmatrix}$$

and a log-likelihood for the *full* data (good plus bad) that is greater than for the good alone. Such an optimization, however, is not the goal of the canonical EM algorithm. Note too that if no data is missing, the calculation of $Q(\boldsymbol{\theta};\ \boldsymbol{\theta}^i)$ is simple since no integrals are involved.

GENERALIZED EXPECTATION-MAXIMIZATION

Generalized Expectation-Maximization or GEM algorithms are a bit more lax than the EM algorithm, and require merely that an *improved* $\boldsymbol{\theta}^{i+1}$ be set in the **M step** (line 5) of the algorithm — not necessarily the *optimal*. Naturally, convergence will not be as rapid as for a proper EM algorithm, but GEM algorithms afford greater freedom to choose computationally simpler steps. One version of GEM is to find the maximum likelihood value of unknown features at each iteration step, then recalculate $\boldsymbol{\theta}$ in light of these new values — if indeed they lead to a greater likelihood.

In practice, the term Expectation-Maximization has come to mean loosely any iterative scheme in which the likelihood of *some* data increases with each step, even if such methods are not, technically speaking, the true EM algorithm as presented here.

3.9 Bayesian Belief Networks

The methods we have described up to now are fairly general — all that we assumed, at base, was that we could parameterize the distributions by a feature vector $\boldsymbol{\theta}$. If we had prior information about the distribution of $\boldsymbol{\theta}$, this too could be used. Sometimes our knowledge about a distribution is not directly of this type, but instead about the statistical dependencies (or independencies) among the component features. Recall that for some multidimensional distribution $p(\mathbf{x})$, if for two features we have $p(x_i, x_j) = p(x_i)p(x_j)$, we say those variables are statistically independent (Fig. 3.6).

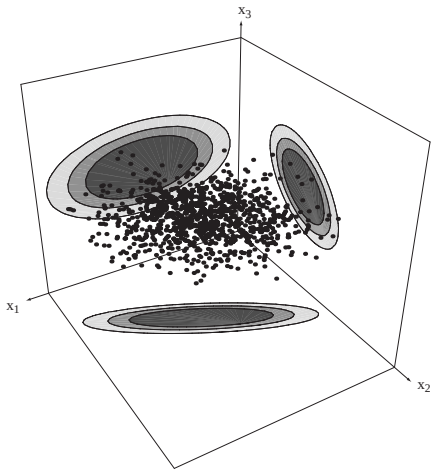


Figure 3.6: A three-dimensional distribution which obeys $p(x_1, x_3) = p(x_1)p(x_3)$; thus here x_1 and x_3 are statistically independent but the other feature pairs are not.

There are many cases where we know or can safely assume which variables are or are not independent, even without sampled data. Suppose for instance we are describing the state of an automobile — temperature of the engine, pressures of the fluids and in the tires, voltages in the wires, and so on. Our basic knowledge of cars includes the fact that the oil pressure in the engine and the air pressure in a tire are functionally unrelated, and hence can be safely assumed to be statistically independent. However the oil temperature and engine temperature are *not* independent (but could be conditionally independent). Furthermore we may know *several* variables that might influence another: the coolant temperature is affected by the engine temperature, the speed of the radiator fan (which blows air over the coolant-filled radiator), and so on.

We will represent these dependencies graphically, by means of *Bayesian belief nets*, also called *causal networks*, or simply *belief nets*. They take the topological form of a directed acyclic graph (DAG), where each link is directional, and there are no loops. (More general networks permit such loops, however.) While such nets can represent continuous multidimensional distributions, they have enjoyed greatest application and

success for discrete variables. For this reason, and because the formal properties are simpler, we shall concentrate on the discrete case.

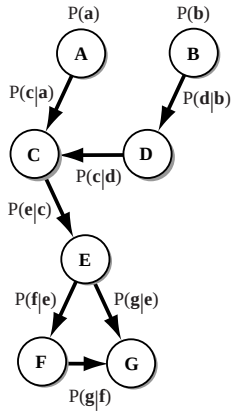


Figure 3.7: A belief network consists of nodes (labelled with upper case bold letters) and their associated discrete states (in lower-case). Thus node **A** has states a_1, a_2, ..., denoted simply **a**; node **B** has states b_1, b_2, ..., denoted **b**, and so forth. The links between nodes represent conditional probabilities. For example, $P(\mathbf{c}|\mathbf{a})$ can be described by a matrix whose entries are $P(c_i|a_j)$.

Each *node* (or unit) represents one of the system variables, and here takes on discrete values. We will label nodes with **A**, **B**, ..., and the variables at each node by the corresponding lower-case letter. Thus, while there are a discrete number of possible values of node **A** — here two, a_1 and a_2 — there may be continuous-valued *probabilities* on these discrete states. For example, if node **A** represents the state of a binary lamp switch — $a_1 = on$, $a_2 = off$ — we might have $P(a_1) = 0.739, P(a_2) = 0.261$, or indeed any other probabilities. A link joining node **A** to node **C** in Fig. 3.7 is directional, and represents the conditional probabilities $P(c_i|a_j)$, or simply $P(\mathbf{c}|\mathbf{a})$. For the time being we shall not be concerned with how these conditional probabilities are determined, except to note that in some cases human experts provide the values. NODE

Suppose we have a belief net, complete with conditional probabilities, and know the values or probabilities of some of the states. Through careful application of Bayes rule or Bayesian inference, we will be able to determine the maximum posterior value of the unknown variables in the net. We first consider how to determine the state of just one node from the states in units with which it is connected. The connected nodes are the only ones we need to consider directly — the others are conditionally independent. This is, at base, the simplification provided by our knowledge of the dependency structure of the system.

In considering a single node **X** in the simple net of Fig. 3.8, it is extremely useful to distinguish the set of nodes *before* **X** — called its *parents* $\mathcal{P}$ — and the set of those *after* it — called its *children* $\mathcal{C}$. When we evaluate the probabilities at **X**, we must treat the parents of **X** differently from its children. Thus, in Fig. 3.8, **A** and **B** are in $\mathcal{P}$ of **X** while **C** and **D** are in $\mathcal{C}$. PARENT CHILD

The *belief* of a set of propositions $\mathbf{x} = (x_1, x_2, ...)$ on node **X** describes the relative probabilities of the variables given all the evidence **e** throughout the rest of the network, i.e., $P(\mathbf{x}|\mathbf{e})$.* We can divide the dependency of the belief upon the parents and BELIEF

* While this is sometimes denoted BEL(**x**), we keep a notation that clarifies the dependencies and is more similar to that in our previous discussions.

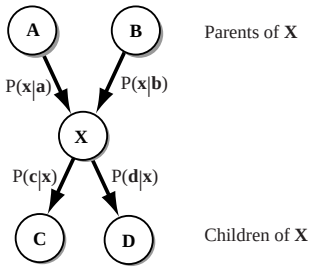


Figure 3.8: A portion of a belief network, consistsing of a node $\mathbf{X}$, having variable values $(x_1, x_2, ...)$, its parents ($\mathbf{A}$ and $\mathbf{B}$), and its children ($\mathbf{C}$ and $\mathbf{D}$).

the children in the following way:

$$P(\mathbf{x}|\mathbf{e}) \propto P(\mathbf{e}^{\mathcal{C}}|\mathbf{x})P(\mathbf{x}|\mathbf{e}^{\mathcal{P}}), \tag{79}$$

where $\mathbf{e}$ represents all evidence (i.e., values of variables on nodes other than $\mathbf{X}$), $\mathbf{e}^{\mathcal{P}}$ the evidence on the parent nodes, and $\mathbf{e}^{\mathcal{C}}$ the children nodes. In Eq. 79 we show only a proportionality — at the end of our calculation we will normalize the probabilities over the states at $\mathbf{X}$.

The first term in Eq. 79 is quite simple, and is a manifestation of Bayes' formula. We can expand the dependency upon the evidence of the children nodes as follows:

$$\begin{aligned} P(\mathbf{e}^{\mathcal{C}}|\mathbf{x}) &= P(\mathbf{e}_{\mathcal{C}_1}, \mathbf{e}_{\mathcal{C}_2}, ..., \mathbf{e}_{\mathcal{C}_{|\mathcal{C}|}}|\mathbf{x}) \\ &= P(\mathbf{e}_{\mathcal{C}_1}|\mathbf{x})P(\mathbf{e}_{\mathcal{C}_2}|\mathbf{x}) \cdots P(\mathbf{e}_{\mathcal{C}_{|\mathcal{C}|}}|\mathbf{x}) \\ &= \prod_{j=1}^{|\mathcal{C}|} P(\mathbf{e}_{\mathcal{C}_j}|\mathbf{x}), \end{aligned} \tag{80}$$

where $\mathcal{C}_j$ represents the jth child node and $\mathbf{e}_{\mathcal{C}_j}$ the values of the probabilities of its states. Note too our convention that $|\mathcal{C}|$ denotes the *cardinality* of set $\mathcal{C}$ — the number of elements in the set — a convenient notation for indicating the full range of summations or products. In the last step of Eq. 80 we used our knowledge that since the child nodes cannot be joined by a line, then they are conditionally independent given $\mathbf{x}$. Equation 80 simply states that the probability of a given set of states throughout all the children nodes of $\mathbf{X}$ is the product of the (independent) probabilities in the individual children nodes. For instance, in the simple example in Fig. 3.8, we have

CARDINALITY

$$P(\mathbf{e_C}, \mathbf{e_D}|\mathbf{x}) = P(\mathbf{e_C}|\mathbf{x})P(\mathbf{e_D}|\mathbf{x}). \tag{81}$$

Incorporating evidence from parent nodes is a bit more subtle. We have:

$$\begin{aligned} P(\mathbf{x}|\mathbf{e}^{\mathcal{P}}) &= P(\mathbf{x}|\mathbf{e}_{\mathcal{P}_1}, \mathbf{e}_{\mathcal{P}_2}, ..., \mathbf{e}_{\mathcal{P}_{|\mathcal{P}|}}) \\ &= \sum_{all\ i,j,...,k} P(\mathbf{x}|\mathcal{P}_{1i}, \mathcal{P}_{2j}, ..., \mathcal{P}_{|\mathcal{P}|k})P(\mathcal{P}_{1i}, \mathcal{P}_{2j}, ..., \mathcal{P}_{|\mathcal{P}|k}|\mathbf{e}_{\mathcal{P}_1}, ..., \mathbf{e}_{\mathcal{P}_{|\mathcal{P}|}}) \\ &= \sum_{all\ i,j,...,k} P(\mathbf{x}|\mathcal{P}_{1i}, \mathcal{P}_{2j}, ..., \mathcal{P}_{|\mathcal{P}|k})P(\mathcal{P}_{1i}|\mathbf{e}_{\mathcal{P}_1}) \cdots P(\mathcal{P}_{|\mathcal{P}|k}|\mathbf{e}_{\mathcal{P}_{|\mathcal{P}|k}}), \end{aligned} \tag{82}$$

where the summation is over all possible configurations of values on the different parent nodes. Here $\mathcal{P}_{mn}$ denotes a particular value for state n on parent node $\mathcal{P}_m$. In the last step of Eq. 82 we have again used our assumption that the (unconnected) parent nodes are statistically independent.

While Eq. 82 and its unavoidable notational complexities may appear intimidating, it is actually just a logical consequence of Bayes' rule. For the purposes of clarity and for computing $\mathbf{x}$, each term at the extreme right, $P(\mathcal{P}_{1i}|\mathbf{e}_{\mathcal{P}_1})$ can be considered to be $P(\mathcal{P}_{1i})$ — the probability of state i on the first parent node. Our notation shows that this probability depends upon the evidence at $\mathcal{P}_1$, including from *its* parents, but for the sake of computing the probabilities at $\mathbf{X}$ we temporarily ignore the dependencies beyond the parents and children of $\mathbf{X}$.

Thus we rewrite Eq. 82 as

$$P(\mathbf{x}|\mathbf{e}^{\mathcal{P}}) = \sum_{all\ \mathcal{P}_{mn}} P(\mathbf{x}|\mathcal{P}_{mn}) \prod_{i=1}^{|\mathcal{P}|} P(\mathcal{P}_i|\mathbf{e}_{\mathcal{P}_i}) \tag{83}$$

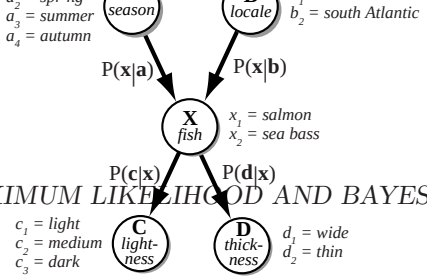
We put these results together for the general case with $|\mathcal{P}|$ parent nodes and $|\mathcal{C}|$ children nodes, Eqs. 80 & 83, and find

$$P(\mathbf{x}|\mathbf{e}) \propto \underbrace{\prod_{j=1}^{|\mathcal{C}|} P(\mathbf{e}_{\mathcal{C}_j}|\mathbf{x})}_{P(\mathbf{e}^{\mathcal{C}}|\mathbf{x})} \underbrace{\left[\sum_{all\ \mathcal{P}_{mn}} P(\mathbf{x}|\mathcal{P}_{mn}) \prod_{i=1}^{|\mathcal{P}|} P(\mathcal{P}_i|\mathbf{e}_{\mathcal{P}_i}) \right]}_{P(\mathbf{x}|\mathbf{e}^{\mathcal{P}})} . \tag{84}$$

In words, Eq. 84 states that the probability of a particular values for node $\mathbf{X}$ is the product of two factors. The first is due to the children (the product of their independent likelihoods). The second is the sum over all possible configurations of states on the parent nodes of the prior probabilities of their values and the conditional probabilities of the $\mathbf{x}$ variables given those parent values. The final values must be normalized to represent probabilities.

Example 3: Belief network for fish

Suppose we are again interested in classifying fish, but now we want to use more information. Imagine that a human expert has constructed the simple belief network in the figure, where node $\mathbf{A}$ represents the time of year, and can have four values: a_1 = *winter*, a_2 = *spring*, a_3 = *summer* and a_4 = *autumn*. Node $\mathbf{B}$ represents the geographical area where the fish was caught: b_1 = *north Atlantic* and b_2 = *south Atlantic*. $\mathbf{A}$ and $\mathbf{B}$ are the parents of the node $\mathbf{X}$, which represents the fish and has just two possible values: x_1 = *salmon* and x_2 = *sea bass*. Similarly, our expert tells us that the children nodes represent lightness, $\mathbf{C}$, with c_1 = *dark*, c_2 = *medium* and c_3 = *light* as well as thickness, $\mathbf{D}$, with d_1 = *thick* and d_2 = *thin*. The direction of the links (from $\mathbf{A}$ and $\mathbf{B}$ to $\mathbf{X}$ and likewise from $\mathbf{X}$ to $\mathbf{C}$ and $\mathbf{D}$) is meant to describe the influences among the variables, as shown in the figure.



A simple belief net for the fish example. The season and the fishing locale are statistically independent, but the type of fish caught does depend on these factors. Further, the width of the fish and its color depend upon the fish.

The following probability matrixes (here, given by an expert) describe the influence of time of year and fishing area on the identity of the fish:

$$P(x_i|a_j): \begin{array}{c} \\ \text{winter} \\ \text{spring} \\ \text{summer} \\ \text{autumn} \end{array} \begin{array}{c} \begin{array}{cc} \text{salmon} & \text{sea bass} \end{array} \\ \begin{pmatrix} .9 & .1 \\ .3 & .7 \\ .4 & .6 \\ .8 & .2 \end{pmatrix} \end{array}, \qquad P(x_i|b_j): \begin{array}{c} \\ \text{north} \\ \text{south} \end{array} \begin{array}{c} \begin{array}{cc} \text{salmon} & \text{sea bass} \end{array} \\ \begin{pmatrix} .65 & .35 \\ .25 & .75 \end{pmatrix} \end{array}$$

Thus salmon are best found in the north fishing areas in the winter and autumn, sea bass in the south fishing areas in the spring and summer, and so forth. Recall that in our belief networks the variables are discrete, and all influences are cast as probabilites, rather than probability densities. Given that we have any particular feature value on a parent node, we must have some fish; thus each row is normalized, as for instance $P(x_1|a_1) + P(x_2|a_1) = 1$.

Suppose our expert tells us that the conditional probabilities for the variables in the children nodes are as follows:

$$P(c_i|x_j): \begin{array}{c} \\ \text{salmon} \\ \text{sea bass} \end{array} \begin{array}{c} \begin{array}{ccc} \text{light} & \text{medium} & \text{dark} \end{array} \\ \begin{pmatrix} .33 & .33 & .34 \\ .8 & .1 & .1 \end{pmatrix} \end{array}, \qquad P(d_i|x_j): \begin{array}{c} \\ \text{salmon} \\ \text{sea bass} \end{array} \begin{array}{c} \begin{array}{cc} \text{wide} & \text{thin} \end{array} \\ \begin{pmatrix} .4 & .6 \\ .95 & .05 \end{pmatrix} \end{array}$$

Thus salmon come in the full range of lightnesses, while sea bass are primarily light in color and are primarily wide.

Now we turn to the problem of using such a belief net to infer the identity of a fish. We have no direct information about the identity of the fish, and thus $P(x_1) = P(x_2) = 0.5$. This might be a reasonable starting point, expressing our lack of knowledge of the identity of the fish. Our goal now is to estimate the probabilities $P(x_1|\mathbf{e})$ and $P(x_2|\mathbf{e})$. Note that without any evidence we have

$$\begin{aligned} P(x_1) &= \sum_{i,j,k,l} P(x_1, a_i, b_j, c_k, d_l) \\ &= \sum_{i,j,k,l} P(a_i)P(b_j)P(x_1|a_i, b_j)P(c_k|x_1)P(d_l|x_1) \\ &= \sum_{i,j} P(a_i)P(b_j)P(x_1|a_i, b_j) \\ &= (0.25)(0.5)\sum_{i,j} P(x_1|a_i, b_j) \\ &= (0.25)(0.5)(0.9 + 0.3 + 0.4 + 0.7 + 0.8 + 0.2 + 0.1 + 0.6) \\ &= 0.5, \end{aligned}$$

and thus $P(x_1) = P(x_2)$, as we would expect.

Now we collect evidence for each node, $\{e_{\mathbf{A}}, e_{\mathbf{B}}, e_{\mathbf{C}}, e_{\mathbf{D}}\}$, assuming they are independent of each other. Suppose we know that it is winter, i.e., $P(a_1|e_{\mathbf{A}}) = 1$ and $P(a_i|e_{\mathbf{A}}) = 0$ for $i = 2, 3, 4$. Suppose we do not know which fishing area the boat came from but found that the particular fishing crew prefers to fish in the south Atlantic; we assume, then, that $P(b_1|e_{\mathbf{B}}) = 0.2$ and $P(b_2|e_{\mathbf{B}}) = 0.8$. We measure the fish and find that it is fairly light, and set by hand to be $P(e_{\mathbf{C}}|c_1) = 1$, $P(e_{\mathbf{C}}|c_2) = 0.5$, and $P(e_{\mathbf{C}}|c_3) = 0$. Suppose that due to occlusion, we cannot measure the width of the fish; we thus set $P(e_{\mathbf{D}}|d_1) = P(e_{\mathbf{D}}|d_2)$.

By Eq. 82, we have the estimated probability of each fish due to the parents $\mathcal{P}$ is, in full expanded form

$$\begin{aligned} P_{\mathcal{P}}(x_1) &\propto P(x_1|a_1,b_1)P(a_1)P(b_1) \\ &\quad +P(x_1|a_1,b_2)P(a_1)P(b_2) \\ &\quad +P(x_1|a_2,b_1)P(a_2)P(b_1) \\ &\quad +P(x_1|a_2,b_2)P(a_2)P(b_2) \\ &\quad +P(x_1|a_3,b_1)P(a_3)P(b_1) \\ &\quad +P(x_1|a_3,b_2)P(a_3)P(b_2) \\ &\quad +P(x_1|a_4,b_1)P(a_4)P(b_1) \\ &\quad +P(x_1|a_4,b_2)P(a_4)P(b_2) \\ &= 0.82. \end{aligned}$$

A similar calculation gives $P_{\mathcal{P}}(x_2) = 0.18$.

We now turn to the children nodes and find by Eq. 84

$$\begin{aligned} P_{\mathcal{C}}(x_1) &\propto P(e_{\mathbf{C}}|x_1)P(e_{\mathbf{D}}|x_1) \\ &= [P(e_{\mathbf{C}}|c_1)P(c_1|x_1) + P(e_{\mathcal{C}}|c_2)P(c_2|x_1) + P(e_{\mathbf{C}}|c_3)P(c_3|x_1)] \\ &\quad \times[P(e_{\mathbf{D}}|d_1)P(d_1|x_1) + P(e_{\mathbf{D}}|d_2)P(d_2|x_1)] \\ &= [(1.0)(0.33) + (0.5)(0.33) + (0)(0.34)] \times [(1.0)(0.4) + (1.0)(0.6)] \\ &= 0.495. \end{aligned}$$

A similar calculation gives $P_{\mathcal{C}}(x_2) \propto 0.85$. We put these estimates together by Eq. 79 as products $P(x_i) \propto P_{\mathcal{C}}(x_i)P_{\mathcal{P}}(x_i)$ and renormalize (i.e., divide by their sum). Thus our final estimates for node $\mathbf{X}$ are

$$P(x_1|\mathbf{e}) = \frac{(0.82)(0.495)}{(0.82)(0.495) + (0.18)(0.85)} = 0.726$$
$$P(x_2|\mathbf{e}) = \frac{(0.18)(0.85)}{(0.82)(0.495) + (0.18)(0.85)} = 0.274.$$

Thus given all the evidence throughout the belief net, the most probable outcome is $x_1 = salmon$.

A given belief net can be used to infer any of the unknown variables. In Example 3, we used information about the time of year, fishing location and some measured

properties of the fish to infer its identity (salmon or sea bass). The same network could instead be used to infer the probability that a fish is thin, or dark in color, based on probabilities of the identity of the fish, time of year, and so on (Problem 42).

When the dependency relationships among the features used by a classifier are unknown, we generally proceed by taking the simplest assumption, i.e., that the features are conditionally independent given the category, i.e.,

$$p(\omega_k|\mathbf{x}) \propto \prod_{i=1}^{d} p(x_i|\omega_k). \tag{85}$$

NAIVE BAYES RULE

In practice, this so-called *naive Bayes rule* or *idiot Bayes rule* often works quite well in practice, and can be expressed by a very simple belief net (Problem 43).

In Example 3 our entire belief net consisted of $\mathbf{X}$, its parents and children, and we needed to update only the values on $\mathbf{X}$. In the more general case, where the network is large, there may be many nodes whose values are unknown. In that case we may have to visit nodes randomly and update the probabilites until the entire configuration of probabilities is stable. It can be shown that under weak conditions, this process will converge to consistent values of the variables throughout the entire network (Problem 44).

Belief nets have found increasing use in complicated problems such as medical diagnosis. Here the upper-most nodes (ones without their own parents) represent a fundamental biological agent such as the presence of a virus or bacteria. Intermediate nodes then describe diseases, such as flu or emphysema, and the lower-most nodes the symptoms, such as high temperature or coughing. A physician enters measured values into the net and finds the most likely disease or cause. Such networks can be used in a somewhat more sophisticated way, automatically computing which unknown variable (node) should be measured to best reveal the identity of the disease.

We will return in Chap. **??** to address the problem of learning in such belief net models.

3.10 Hidden Markov Models

While belief nets are a powerful method for representing the dependencies and independencies among variables, we turn now to the problem of representing a particular but extremely important dependencies. In problems that have an inherent temporality — that is, consist of a process that unfolds in time — we may have states at time t that are influenced directly by a state at $t-1$. Hidden Markov models (HMMs) have found greatest use in such problems, for instance speech recognition or gesture recognition. While the notation and description is aunavoidably more complicated than the simpler models considered up to this point, we stress that the same underlying ideas are exploited. Hidden Markov models have a number of parameters, whose values are set so as to best explain training patterns for the known category. Later, a test pattern is classified by the model that has the highest posterior probability, i.e., that best "explains" the test pattern.

3.10.1 First-order Markov models

We consider a sequence of states at successive times; the state at any time t is denoted $\omega(t)$. A particular sequence of length T is denoted by $\boldsymbol{\omega}^T = \{\omega(1), \omega(2), ..., \omega(T)\}$ as

for instance we might have $\boldsymbol{\omega}^6 = \{\omega_1, \omega_4, \omega_2, \omega_2, \omega_1, \omega_4\}$. Note that the system can revisit a state at different steps, and not every state need be visited.

Our model for the production of any sequence is described by *transition probabilities* $P(\omega_j(t+1)|\omega_i(t)) = a_{ij}$ — the time-independent probability of having state ω_j at step $t+1$ given that the state at time t was ω_i. There is no requirement that the transition probabilities be symmetric ($a_{ij} \neq a_{ji}$, in general) and a particular state may be visited in succession ($a_{ii} \neq 0$, in general), as illustrated in Fig. 3.9.

TRANSITION PROBABILITY

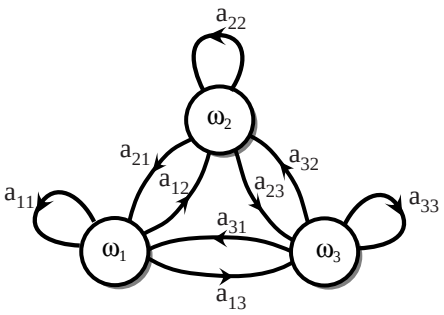


Figure 3.9: The discrete states, ω_i, in a basic Markov model are represented by nodes, and the transition probabilities, a_{ij}, by links. In a first-order discrete time Markov model, at any step t the full system is in a particular state $\omega(t)$. The state at step $t+1$ is a random function that depends solely on the state at step t and the transition probabilities.

Suppose we are given a particular model $\boldsymbol{\theta}$ — that is, the full set of a_{ij} — as well as a particular sequence $\boldsymbol{\omega}^T$. In order to calculate the probability that the model generated the particular sequence we simply multiply the successive probabilities. For instance, to find the probability that a particular model generated the sequence described above, we would have $P(\boldsymbol{\omega}^T|\boldsymbol{\theta}) = a_{14}a_{42}a_{22}a_{21}a_{14}$. If there is a prior probability on the first state $P(\omega(1) = \omega_i)$, we could include such a factor as well; for simplicity, we will ignore that detail for now.

Up to here we have been discussing a Markov model, or technically speaking, a *first-order* discrete time Markov model, since the probability at $t+1$ depends only on the states at t. For instance, in a Markov model for the production of spoken words, we might have states representing phonemes, and a Markov model for the production of a spoken work might have states representing phonemes. Such a Markov model for the word "cat" would have states for /k/, /a/ and /t/, with transitions from /k/ to /a/; transitions from /a/ to /t/; and transitions from /t/ to a final silent state.

Note however that in speech recognition the perceiver does not have access to the states $\omega(t)$. Instead, we measure some properties of the emitted sound. Thus we will have to augment our Markov model to allow for *visible states* — which are directly accessible to external measurement — as separate from the ω states, which are not.

3.10.2 First-order hidden Markov models

We continue to assume that at every time step t the system is in a state $\omega(t)$ but now we also assume that it emits some (visible) symbol $v(t)$. While sophisticated Markov models allow for the emission of continuous functions (e.g., spectra), we will restrict ourselves to the case where a discrete symbol is emitted. As with the states, we define

a particular sequence of such visible states as $\mathbf{V}^T = \{v(1), v(2), ..., v(T)\}$ and thus we might have $\mathbf{V}^6 = \{v_5, v_1, v_1, v_5, v_2, v_3\}$.

Our model is then that in any state $\omega(t)$ we have a probability of emitting a particular visible state $v_k(t)$. We denote this probability $P(v_k(t)|\omega_j(t)) = b_{jk}$. Because we have access only to the visible states, while the ω_i are unobservable, such a full model is called a *hidden Markov model* (Fig. 3.10)

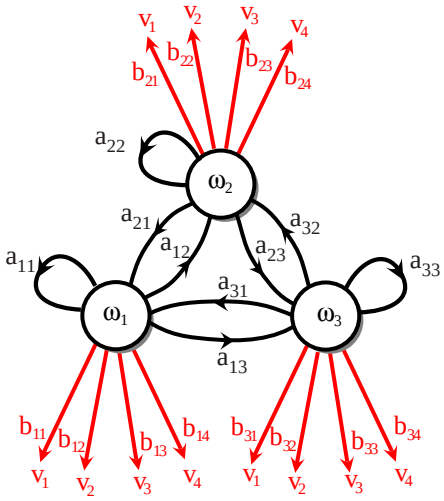


Figure 3.10: Three hidden units in an HMM and the transitions between them are shown in black while the visible states and the emission probabilities of visible states are shown in red. This model shows all transitions as being possible; in other HMMs, some such candidate transitions are not allowed.

3.10.3 Hidden Markov Model Computation

Now we define some new terms and clarify our notation. In general networks such as those in Fig. 3.10 are finite-state machines, and when they have associated transition probabilities, they are called Markov networks. They are strictly *causal* — the probabilities depend only upon *previous* states. A Markov model is called *ergodic* if every one of the states has a non-zero probability of occuring given some starting state. A *final* or *absorbing state* ω_0 is one which, if entered, is never left (i.e., $a_{00} = 1$).

ABSORBING STATE

As mentioned, we denote the transition probabilities a_{ij} among hidden states and for the probability b_{jk} of the emission of a visible state:

$$\begin{aligned} a_{ij} &= P(\omega_j(t+1)|\omega_i(t)) \\ b_{jk} &= P(v_k(t)|\omega_j(t)). \end{aligned} \tag{86}$$

We demand that some transition occur from step $t \to t+1$ (even if it is to the same state), and that some visible symbol be emitted after every step. Thus we have the normalization conditions:

$$\sum_j a_{ij} = 1 \quad \text{for all } i \text{ and}$$

$$\sum_k b_{jk} \quad = \quad 1 \text{ for all } j, \tag{87}$$

where the limits on the summations are over all hidden states and all visible symbols, respectively.

With these preliminaries behind us, we can now focus on the three central issues in hidden Markov models:

The Evaluation problem. Suppose we have an HMM, complete with transition probabilites a_{ij} and b_{jk}. Determine the probability that a particular sequence of visible states $\mathbf{V}^T$ was generated by that model.

The Decoding problem. Suppose we have an HMM as well as a set of observations $\mathbf{V}^T$. Determine the most likely sequence of *hidden* states $\boldsymbol{\omega}^T$ that led to those observations.

The Learning problem. Suppose we are given the coarse structure of a model (the number of states and the number of visible states) but *not* the probabilities a_{ij} and b_{jk}. Given a set of training observations of visible symbols, determine these parameters.

We consider each of these problems in turn.

3.10.4 Evaluation

The probability that the model produces a sequence $\mathbf{V}^T$ of visible states is:

$$P(\mathbf{V}^T) = \sum_{r=1}^{r_{max}} P(\mathbf{V}^T|\boldsymbol{\omega}_r^T)P(\boldsymbol{\omega}_r^T), \tag{88}$$

where each r indexes a particular sequence $\boldsymbol{\omega}_r^T = \{\omega(1), \omega(2), ..., \omega(T)\}$ of T hidden states. In the general case of c hidden states, there will be $r_{max} = c^T$ possible terms in the sum of Eq. 88, corresponding to all possible sequences of length T. Thus, according to Eq. 88, in order to compute the probability that the model generated the particular sequence of T visible states $\mathbf{V}^T$, we should take each conceivable sequence of hidden states, calculate the probability they produce $\mathbf{V}^T$, and then add up these probabilities. The probability of a particular visible sequence is merely the product of the corresponding (hidden) transition probabilities a_{ij} and the (visible) output probabilities b_{jk} of each step.

Because we are dealing here with a first-order Markov process, the second factor in Eq. 88, which describes the transition probability for the hidden states, can be rewritten as:

$$P(\boldsymbol{\omega}_r^T) = \prod_{t=1}^{T} P(\omega(t)|\omega(t-1)) \tag{89}$$

that is, a product of the a_{ij}'s according to the hidden sequence in question. In Eq. 89, $\omega(T) = \omega_0$ is some final absorbing state, which uniquely emits the visible state v_0. In speech recognition applications, ω_0 typically represents a null state or lack of utterance, and v_0 is some symbol representing silence. Because of our assumption that the output probabilities depend only upon the hidden state, we can write the first factor in Eq. 88 as

$$P(\mathbf{V}^T|\boldsymbol{\omega}_r^T) = \prod_{t=1}^{T} P(v(t)|\omega(t)), \tag{90}$$

that is, a product of b_{jk}'s according to the hidden state and the corresponding visible state. We can now use Eqs. 89 & 90 to express Eq. 88 as

$$P(\mathbf{V}^T) = \sum_{r=1}^{r_{max}} \prod_{t=1}^{T} P(v(t)|\omega(t))P(\omega(t)|\omega(t-1)). \tag{91}$$

Despite its formal complexity, Eq. 91 has a straightforward interpretation. The probability that we observe the particular sequence of T visible states $\mathbf{V}^T$ is equal to the sum over all r_{max} possible sequences of hidden states of the conditional probability that the system has made a particular transition multiplied by the probability that it then emitted the visible symbol in our target sequence. All these are captured in our paramters a_{ij} and b_{kj}, and thus Eq. 91 can be evaluated directly. Alas, this is an $O(c^T T)$ calculation, which is quite prohibitive in practice. For instance, if $c = 10$ and $T = 20$, we must perform on the order of 10^{21} calculations.

A computationaly simpler algorithm for the same goal is as follows. We can calculate $P(\mathbf{V}^T)$ recursively, since each term $P(v(t)|\omega(t))P(\omega(t)|\omega(t-1))$ involves only $v(t)$, $\omega(t)$ and $\omega(t-1)$. We do this by defining

$$\alpha_i(t) = \begin{cases} 0 & t = 0 \text{ and } i \neq \text{initial state} \\ 1 & t = 0 \text{ and } i = \text{initial state} \\ \sum_j \alpha(t-1)a_{ij}b_{jk}v(t) & \text{otherwise,} \end{cases} \tag{92}$$

where the notation $b_{jk}v(t)$ means the transition probability b_{jk} selected by the visible state emitted at time t. thus the only non-zero contribution to the sum is for the index k which matches the visible state $v(t)$. Thus $\alpha_i(t)$ represents the probability that our HMM is in hidden state ω_i at step t having generated the first t elements of $\mathbf{V}^T$. This calculation is implemented in the *Forward algorithm* in the following way:

Algorithm 2 (HMM Forward)

1 **initialize** $\omega(1), t = 0, a_{ij}, b_{jk}$, visible sequence $\mathbf{V}^T, \alpha(0) = 1$
2 **for** $t \leftarrow t + 1$
3 $\alpha_j(t) \leftarrow \sum_{i=1}^{c} \alpha_i(t-1)a_{ij}b_{jk}$
4 **until** $t = T$
5 **return** $P(\mathbf{V}^T) \leftarrow \alpha_0(T)$
6 **end**

where in line 5, α_0 denotes the probability of the associated sequence ending to the known final state. The *Forward algorithm* has, thus, a computational complexity of $O(c^2T)$ — *far* more efficient than the complexity associated with exhaustive enumeration of paths of Eq. 91 (Fig. 3.11). For the illustration of $c = 10$, $T = 20$ above, we would need only on the order of 2000 calculations — more than 17 orders of magnitude faster than that to examine each path individually.

We shall have cause to use the *Backward algorithm*, which is the time-reversed version of the *Forward algorithm*.

Algorithm 3 (HMM Backward)

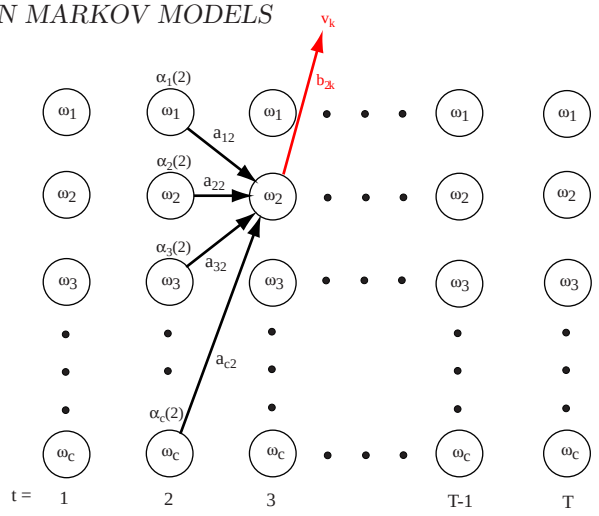


Figure 3.11: The computation of probabilities by the Forward algorithm can be visualized by means of a trellis — a sort of "unfolding" of the HMM through time. Suppose we seek the probability that the HMM was in state ω_2 at $t = 3$ and generated the observed visible up through that step (including the observed visible symbol v_k). The probability the HMM was in state $\omega_j(t = 2)$ and generated the observed sequence through $t = 2$ is $\alpha_j(2)$ for $j = 1, 2, ..., c$. To find $\alpha_2(3)$ we must sum these and multiply the probability that state ω_2 emitted the observed symbol v_k. Formally, for this particular illustration we have $\alpha_2(3) = b_{2k} \sum_{j=1}^{c} \alpha_j(2) a_{j2}$.

1 **initialize** $\omega(T), t = T, a_{ij}, b_{jk}$, visible sequence V^T
2 **for** $t \leftarrow t - 1$;
4 $\beta_j(t) \leftarrow \sum_{i=1}^{c} \beta_i(t+1) a_{ij} b_{jk} v(t+1)$
5 **until** $t = 1$
7 **return** $P(V^T) \leftarrow \beta_i(0)$ for the known initial state
8 **end**

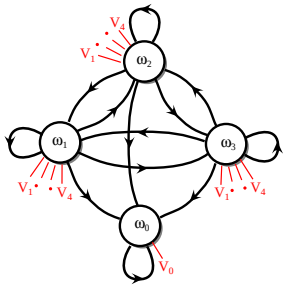
Example 4: Hidden Markov Model

To clarify the evaluation problem, consider an HMM such as shown in Fig. 3.10, but with an explicit absorber state and unique null visible symbol V_0 with the following transition probabilities (where the matrix indexes begin at 0):

$$a_{ij} = \begin{pmatrix} 1 & 0 & 0 & 0 \\ 0.2 & 0.3 & 0.1 & 0.4 \\ 0.2 & 0.5 & 0.2 & 0.1 \\ 0.8 & 0.1 & 0.0 & 0.1 \end{pmatrix} \text{ and}$$

$$b_{jk} = \begin{pmatrix} 1 & 0 & 0 & 0 & 0 \\ 0 & 0.3 & 0.4 & 0.1 & 0.2 \\ 0 & 0.1 & 0.1 & 0.7 & 0.1 \\ 0 & 0.5 & 0.2 & 0.1 & 0.2 \end{pmatrix}.$$

What is the probability it generates the particular sequence $\mathbf{V}^5 = \{v_3, v_1, v_3, v_2, v_0\}$? Suppose we know the initial hidden state at $t = 0$ to be ω_1. The visible symbol at each step is shown above, and the $\alpha_i(t)$ in each unit. The circles show the value for $\alpha_i(t)$ as we progress left to right. The product $a_{ij}b_{jk}$ is shown along each transition link for the step $t = 1$ to $t = 2$. The final probability, $P(\mathbf{V}^T|\boldsymbol{\theta})$ is hence 0.0011.



The HMM (above) consists of four hidden states (one of which is an absorber state, ω_0), each emitting one of five visible states; only the allowable transitions to visible states are shown. The trellis for this HMM is shown below. In each node is $\alpha_i(t)$ — the probability the model generated the observed visible sequence up to t. For instance, we know that the system was in hidden state ω_1 at $t = 1$, and thus $\alpha_1(0) = 1$ and $\alpha_i(0) = 0$ for $i \neq 1$. The arrows show the calculation of $\alpha_i(1)$. for instance, since visible state v_1 was emitted at $t = 1$, we have $\alpha_0(1) = \alpha_1(0)a_{10}b_{01} = 1[0.2 \times 0] = 0$. as shown by the top arrow. Likewise the nest highest arrow corresponds to the calculation $\alpha_1(1) = \alpha_1(0)a_{11}b_{11} = 1[0.3 \times 0.3] = 0.09$. In this example, the calculation of $\alpha_i(1)$ is particularly simple, since only transitions from the known initial hidden state need be considered; all other transitions have zero contribution to $\alpha_i(1)$. For subsequent times, however, the caculation requires a *sum* over all hidden states at the previous time, as given by line 3 in the Forward algorithm. The probability shown in the final (absorbing) state gives the probability of the full sequence observed, $P(\mathbf{V}^T|\boldsymbol{\theta}) = 0.0011$.

If we denote our model — the a's and b's — by $\boldsymbol{\theta}$, we have by Bayes' formula that the probability of the model given the observed sequence is:

$$P(\boldsymbol{\theta}|\mathbf{V}^T) = \frac{P(\mathbf{V}^T|\boldsymbol{\theta})P(\boldsymbol{\theta})}{P(\mathbf{V}^T)} \tag{93}$$

In HMM pattern recognition we would have a number of HMMs, one for each category and classify a test sequence according to the model with the highest probability. Thus in HMM speech recognition we could have a model for "cat" and another one for "dog" and for a test utterance determine which model has the highest probability. In practice, nearly all HMMs for speech are *left-to-right* models (Fig. 3.12).

LEFT-TO-RIGHT MODEL

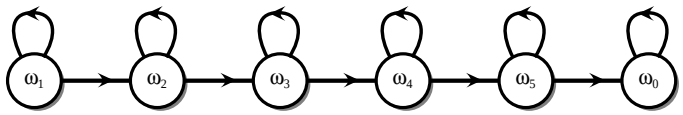


Figure 3.12: A left-to-right HMM commonly used in speech recognition. For instance, such a model could describe the utterance "viterbi," where ω_1 represents the phoneme /v/, ω_2 represents /i/, ..., and ω_0 a final silent state. Such a left-to-right model is more restrictive than the general HMM in Fig. 3.10, and precludes transitions "back" in time.

The `Forward algorithm` gives us $P(V^T|\boldsymbol{\theta})$. The prior probability of the model, $P(\boldsymbol{\theta})$, is given by some external source, such as a *language model* in the case of speech. This prior probability might depend upon the semantic context, or the previous words, or yet other information. In the absence of such information, it is traditional to assume a uniform density on $P(\boldsymbol{\theta})$, and hence ignore it in any classification problem. (This is an example of a "non-informative" prior.)

3.10.5 Decoding

Given a sequence of visible states $\mathbf{V}^T$, the decoding problem is to find the most probable sequence of hidden states. While we might consider enumerating every possible path and calculating the probability of the visible sequence observed, this is an $O(c^T T)$ calculation and prohibitive. Instead, we use perhaps the simplest decoding algorithm:

Algorithm 4 (HMM decoding)

```
1  begin initialize  Path = {}, t = 0
2        for  t ← t + 1
4            k = 0, α_0 = 0
5            for  k ← k + 1
7                α_k(t) ← b_jk v(t) Σ_{i=1}^{c} α_i(t − 1) a_ij
8            until k = c
10           j' ← arg max_j α_j(t)
11           AppendTo Path ω_j'
12       until t = T
13   return Path
14 end
```

A closely related algorithm uses logarithms of the probabilities and calculates total probabilities by addition of such logarithms; this method has complexity $O(c^2 T)$ (Problem 48).

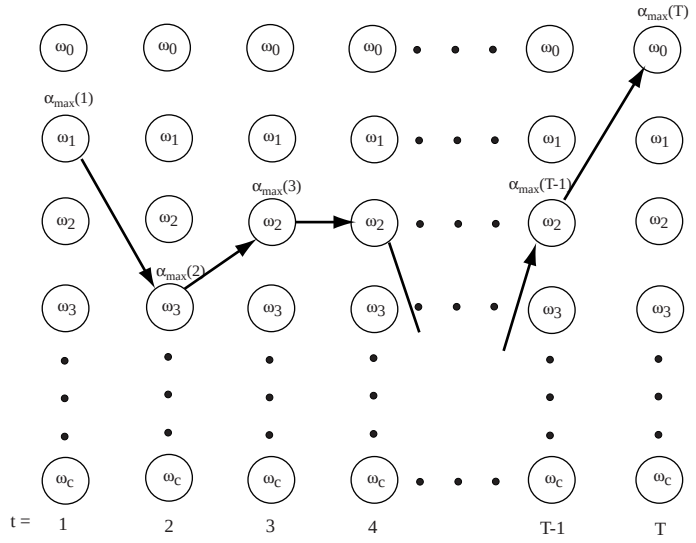
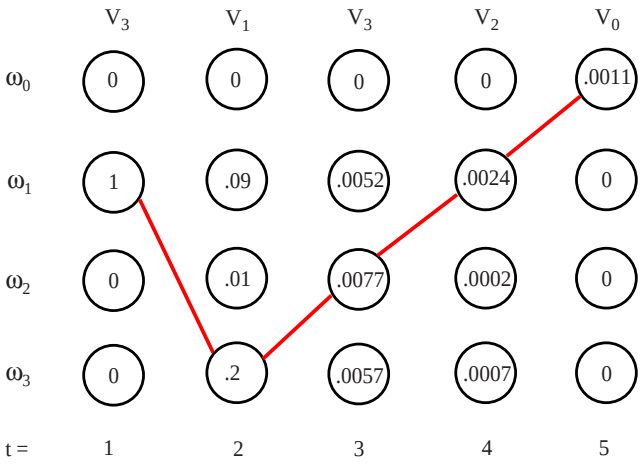


Figure 3.13: The decoding algorithm finds at each time step t the state that has the highest probability of having come from the previous step and generated the observed visible state v_k. The full path is the sequence of such states. Because this is a local optimization (dependent only upon the single previous time step, not the full sequence), the algorithm does not guarantee that the path is indeed allowable. For instance, it might be possible that the maximum at $t = 5$ is ω_1 and at $t = 6$ is ω_2, and thus these would appear in the path. This can even occur if $a_{12} = P(\omega_2(t+1)|\omega_1(t)) = 0$, precluding that transition.

The red line in Fig. 3.13 corresponds to Path, and connects the hidden states with the highest value of α_i at each step t. There is a difficulty, however. Note that there is no guarantee that the path is in fact a *valid* one — it might not be consistent with the underlying models. For instance, it is possible that the path actually implies a transition that is forbidden by the model, as illustrated in Example 5.

Example 5: HMM decoding

We find the path for the data of Example 4 for the sequence $\{\omega_1, \omega_3, \omega_2, \omega_1, \omega_0\}$. Note especially that the transition from ω_3 to ω_2 is not allowed according to the transition probabilities a_{ij} given in Example 4. The path *locally* optimizes the probability through the trellis.



The locally optimal path through the HMM trellis of Example 4.

HMMs address the problem of rate invariance in the following two ways. The first is that the transition probabilities themselves incorporate probabilistic structure of the durations. Moreover, using postprocessing, we can delete repeated states and just get the sequence somewhat independent of variations in rate. Thus in post-processing we can convert the sequence $\{\omega_1, \omega_1, \omega_3, \omega_2, \omega_2, \omega_2\}$ to $\{\omega_1, \omega_3, \omega_2\}$, which would be appropriate for speech recognition, where the fundamental phonetic units are not repeated in natural speech.

3.10.6 Learning

The goal in HMM learning is to determine model parameters — the transition probabilities a_{ij} and b_{jk} — from an ensemble of training samples. There is no known method for obtaining the optimal or most likely set of parameters from the data, but we can nearly always determine a good solution by a straightforward technique.

The Forward-backward Algorithm

The Forward-backward algorithm is an instance of a generalized Expectation-Maximization algorithm. The general approach will be to iteratively update the weights in order to better explain the observed training sequences.

Above, we defined $\alpha_i(t)$ as the probability that the model is in state $\omega_i(t)$ and has generated the target sequence up to step t. We can analogously define $\beta_i(t)$ to be the probability that the model is in state $\omega_i(t)$ and *will generate* the remainder of the given target sequence, i.e., from $t+1 \rightarrow T$. We express $\beta_i(t)$ as:

$$\beta_i(t) = \begin{cases} 0 & \omega_i(t) \neq \text{sequence's final state and } t = T \\ 1 & \omega_i(t) = \text{sequence's final state and } t = T \\ \sum_j a_{ij} b_{jk} v(t+1) \beta_j(t+1) & \text{otherwise,} \end{cases} \tag{94}$$

To understand Eq. 94, imagine we knew $\alpha_i(t)$ up to step $T-1$, and we wanted to calculate the probability that the model would generate the remaining single visible

symbol. This probability, $\beta_i(T)$, is just the probability we make a transition to state $\omega_i(T)$ multiplied by the probability that this hidden state emitted the correct final visible symbol. By the definition of $\beta_i(T)$ in Eq. 94, this will be either 0 (if $\omega_i(T)$ is not the final hidden state) or 1 (if it is). Thus it is clear that $\beta_i(T-1) = \sum_j a_{ij} b_{ij} v(T) \beta_i(T)$. Now that we have determined $\beta_i(T-1)$, we can repeat the process, to determine $\beta_i(T-2)$, and so on, *backward* through the trellis of Fig. **??**.

But the $\alpha_i(t)$ and $\beta_i(t)$ we determined are merely *estimates* of their true values, since we don't know the actual value of the transition probabilities a_{ij} and b_{ij} in Eq. 94. We can calculate an improved value by first defining $\gamma_{ij}(t)$ — the probability of transition between $\omega_i(t-1)$ and $\omega_j(t)$, given the model generated the entire training sequence $\mathbf{V}^T$ by *any* path. We do this by defining $\gamma_{ij}(t)$, as follows:

$$\gamma_{ij}(t) = \frac{\alpha_i(t-1) a_{ij} b_{ij} \beta_i(t)}{P(V^T|\boldsymbol{\theta})}, \tag{95}$$

where $P(\mathbf{V}^T|\boldsymbol{\theta})$ is the probability that the model generated sequence $\mathbf{V}^T$ by any path. Thus $\gamma_{ij}(t)$ is the probability of a transition from state $\omega_i(t-1)$ to $\omega_j(t)$ given that the model generated the complete visible sequence V^T.

We can now calculate an improved estimate for a_{ij}. The expected number of transitions between state $\omega_i(t-1)$ and $\omega_j(t)$ at *any* time in the sequence is simply $\sum_{t=1}^{T} \gamma_{ij}(t)$, whereas at step t it is $\sum_{t=1}^{T} \sum_k \gamma_{ik}(t)$. Thus $\hat{a}_{ij}$ (the estimate of the probability of a transition from $\omega_i(t-1)$ to $\omega_j(t)$) can be found by taking the ratio between the expected number of transitions from ω_i to ω_j and the total expected number of *any* transitions from ω_i. That is:

$$\hat{a}_{ij} = \frac{\sum_{t=1}^{T} \gamma_{ij}(t)}{\sum_{t=1}^{T} \sum_k \gamma_{ik}(t)}. \tag{96}$$

In the same way, we can obtain an improved estimate $\hat{b}_{ij}$ by calculating the ratio between the frequency that any particular symbol v_k is emitted and that for any symbol. Thus we have

$$\hat{b}_{jk} = \frac{\sum \gamma_{jk}(t)}{\sum_{t=1}^{T} \gamma_{jk}(t)}. \tag{97}$$

In short, then, we start with rough or arbitrary estimates of a_{ij} and b_{jk}, calculate improved estimates by Eqs. 96 & 97, and repeat until some convergence criterion is met (e.g., sufficiently small change in the estimated values of the parameters on subsequent iterations). This is the *Baum-Welch* or *Forward-backward algorithm* — an example of a Generalized Expectation-Maximumization algorithm (Sec. 3.8):

Algorithm 5 (Forward-backward)

1 **begin initialize** a_{ij}, b_{jk}, training sequence V^T, convergence criterion θ
2 **do** $z \leftarrow z + 1$
3 Compute $\hat{a}(z)$ from $a(z-1)$ and $b(z-1)$ by Eq. 96
4 Compute $\hat{b}(z)$ from $a(z-1)$ and $b(z-1)$ by Eq. 97
5 $a_{ij}(z) \leftarrow \hat{a}_{ij}(z-1)$
6 $b_{jk}(z) \leftarrow \hat{b}_{jk}(z-1)$

7 **until** $\max_{i,j,k}[a_{ij}(z) - a_{ij}(z-1), b_{jk}(z) - b_{jk}(z-1)] < \theta$; convergence achieved$ln : ForBackstop$
8 **return** $a_{ij} \leftarrow a_{ij}(z)$; $b_{jk} \leftarrow b_{jk}(z)$
9 **end**

The stopping or convergence criterion in line **??** halts learning when no estimated transition probability changes more than a predetermined amount, θ. In typical speech recognition applications, convergence requires several presentations of each training sequence (fewer than five is common). Other popular stopping criteria are based on overall probability that the learned model could have generated the full training data.

Summary

If we know a parametric form of the class-conditional probability densities, we can reduce our learning task from one of finding the distribution itself, to that of finding the *parameters* (represented by a vector $\boldsymbol{\theta}_i$ for each category ω_i), and use the resulting distributions for classification. The maximum likelihood method seeks to find the parameter value that is best supported by the training data, i.e., maximizes the probability of obtaining the samples actually observed. (In practice, for computational simplicity one typically uses log-likelihood.) In Bayesian estimation the parameters are considered random variables having a known a priori density; the training data convert this to an a posteriori density. The recursive Bayes method updates the Bayesian parameter estimate incrementally, i.e., as each training point is sampled. While Bayesian estimation is, in principle, to be preferred, maximum likelihood methods are generally easier to implement and in the limit of large training sets give classifiers nearly as accurate.

A sufficient statistic $\mathbf{s}$ for $\boldsymbol{\theta}$ is a function of the samples that contains all information needed to determine $\boldsymbol{\theta}$. Once we know the sufficient statistic for models of a given form (e.g., exponential family), we need only estimate their value from data to create our classifier — no other functions of the data are relevant.

Expectation-Maximization is an iterative scheme to maximize model parameters, even when some data are missing. Each iteration employs two steps: the expectation or **E step** which requires marginalizing over the missing variables given the current model, and the maximization or **M step**, in which the optimum parameters of a new model are chosen. Generalized Expectation-Maximization algorithms demand merely that parameters be *improved* — not optimized — on each iteration and have been applied to the training of a large range of models.

Bayesian belief nets allow the designer to specify, by means of connection topology, the functional dependences and independencies among model variables. When any subset of variables is clamped to some known values, each node comes to a probability of its value through a Bayesian inference calculation. Parameters representing conditional dependences can be set by an expert.

Hidden Markov models consist of nodes representing hidden states, interconnected by links describing the conditional probabilities of a transition between the states. Each hidden state also has an associated set of probabilities of emiting a particular visible states. HMMs can be useful in modelling sequences, particularly context dependent ones, such as phonemes in speech. All the transition probabilities can be learned (estimated) iteratively from sample sequences by means of the *Forward-backward* or *Baum-Welch* algorithm, an example of a generalized EM algorithm. Classification

proceeds by finding the single model among candidates that is most likely to have produced a given observed sequence.

Bibliographical and Historical Remarks

Maximum likelihood and Bayes estimation have a long history. The Bayesian approach to learning in pattern recognition began by the suggestion that the proper way to use samples when the conditional densities are unknown is the calculation of $P(\omega_i|\mathbf{x}, \mathcal{D})$, [6]. Bayes himself appreciated the role of non-informative priors. An analysis of different priors from statistics appears in [21, 15] and [4] has an extensive list of references.

The origins of Bayesian belief nets traced back to [33], and a thorough literature review can be found in [8]; excellent modern books such as [24, 16] and tutorials [7] can be recommended. An important dissertation on the theory of belief nets, with an application to medical diagnosis is [14], and a summary of work on diagnosis of machine faults is [13]. While we have focussed on directed acyclic graphs, belief nets are of broader use, and even allow loops or arbitrary topologies — a topic that would lead us far afield here, but which is treated in [16].

The Expectation-Maximization algorithm is due to Dempster et al.[11] and a thorough overview and history appears in [23]. On-line or incremental versions of EM are described in [17, 31]. The definitive compendium of work on missing data, including much beyond our discussion here, is [27].

Markov developed what later became called the Markov framework [22] in order to analyze the the text of his fellow Russian Pushkin's masterpiece **Eugene Onegin**. Hidden Markov models were introduced by Baum and collaborators [2, 3], and have had their greatest applications in the speech recognition [25, 26], and to a lesser extent statistical language learning [9], and sequence identification, such as in DNA sequences [20, 1]. Hidden Markov methods have been extended to two-dimensions and applied to recognizing characters in optical document images [19]. The decoding algorithm is related to pioneering work of Viterbi and followers [32, 12]. The relationship between hidden Markov models and graphical models such as Bayesian belief nets is explored in [29].

Knuth's classic [18] was the earliest compendium of the central results on computational complexity, the majority due to himself. The standard books [10], which inspired several homework problems below, are a bit more accessible for those without deep backgrounds in computer science. Finally, several other pattern recognition textbooks, such as [28, 5, 30] which take a somewhat different approach to the field can be recommended.

Problems

⨁ *Section 3.2*

1. Let x have an exponential density

$$p(x|\theta) = \begin{cases} \theta e^{-\theta x} & x \geq 0 \\ 0 & \text{otherwise.} \end{cases}$$

(a) Plot $p(x|\theta)$ versus x for $\theta = 1$. Plot $p(x|\theta)$ versus θ, $(0 \leq \theta \leq 5)$, for $x = 2$.

(b) Suppose that n samples $x_1, ..., x_n$ are drawn independently according to $p(x|\theta)$. Show that the maximum likelihood estimate for θ is given by

$$\hat{\theta} = \frac{1}{\frac{1}{n}\sum_{k=1}^{n} x_k}.$$

(c) On your graph generated with $\theta = 1$ in part (a), mark the maximum likelihood estimate $\hat{\theta}$ for large n.

2. Let x have a uniform density

$$p(x|\theta) \sim U(0, \theta) = \begin{cases} 1/\theta & 0 \le x \le \theta \\ 0 & \text{otherwise.} \end{cases}$$

(a) Suppose that n samples $\mathcal{D} = \{x_1, ..., x_n\}$ are drawn independently according to $p(x|\theta)$. Show that the maximum likelihood estimate for θ is $\max[\mathcal{D}]$, i.e., the value of the maximum element in $\mathcal{D}$.

(b) Suppose that $n = 5$ points are drawn from the distribution and the maximum value of which happens to be $\max_k x_k = 0.6$. Plot the likelihood $p(\mathcal{D}|\theta)$ in the range $0 \le \theta \le 1$. Explain in words why you do not need to know the values of the other four points.

3. Maximum likelihood methods apply to estimates of prior probabilities as well. Let samples be drawn by successive, independent selections of a state of nature ω_i with unknown probability $P(\omega_i)$. Let $z_{ik} = 1$ if the state of nature for the kth sample is ω_i and $z_{ik} = 0$ otherwise.

(a) Show that

$$P(z_{i1}, \ldots, z_{in}|P(\omega_i)) = \prod_{k=1}^{n} P(\omega_i)^{z_{ik}} (1 - P(\omega_i))^{1-z_{ik}}.$$

(b) Show that the maximum likelihood estimate for $P(\omega_i)$ is

$$\hat{P}(\omega_i) = \frac{1}{n}\sum_{k=1}^{n} z_{ik}.$$

Interpret your result in words.

4. Let $\mathbf{x}$ be a d-dimensional binary (0 or 1) vector with a multivariate Bernoulli distribution

$$P(\mathbf{x}|\boldsymbol{\theta}) = \prod_{i=1}^{d} \theta_i^{x_i}(1 - \theta_i)^{1-x_i},$$

where $\boldsymbol{\theta} = (\theta_1, ..., \theta_d)^t$ is an unknown parameter vector, θ_i being the probability that $x_i = 1$. Show that the maximum likelihood estimate for $\boldsymbol{\theta}$ is

$$\hat{\boldsymbol{\theta}} = \frac{1}{n}\sum_{k=1}^{n} \mathbf{x}_k.$$

5. Let each component x_i of $\mathbf{x}$ be binary valued (0 or 1) in a two-category problem with $P(\omega_1) = P(\omega_2) = 0.5$. Suppose that the probability of obtaining a 1 in any component is

$$\begin{aligned} p_{i1} &= p \\ p_{i2} &= 1-p, \end{aligned}$$

and we assume for definiteness $p > 1/2$. The probability of error is known to approach zero as the dimensionality d approaches infinity. This problem asks you to explore the behavior as we increase the number of *features* in a *single* sample — a complementary situation.

(a) Suppose that a single sample $\mathbf{x} = (x_1, ..., x_d)^t$ is drawn from category ω_1. Show that the maximum likelihood estimate for p is given by

$$\hat{p} = \frac{1}{d}\sum_{i=1}^{d} x_i.$$

(b) Describe the behavior of $\hat{p}$ as d approaches infinity. Indicate why such behavior means that by letting the number of features increase without limit we can obtain an error-free classifier even though we have only one sample from each class.

(c) Let $T = 1/d \sum_{j=1}^{d} x_j$ represent the proportion of 1's in a single sample. Plot $P(T|\omega_i)$ vs. T for the case $P = 0.6$, for small d and for large d (e.g., $d = 11$ and $d = 111$, respectively). Explain your answer in words.

6. Derive Eqs. 18 & 19 for the maximum likelihood estimation of the mean and covariance of a multidimensional Gaussian. State clearly any assumptions you need to invoke.

7. Show that if our model is poor, the maximum likelihood classifier we derive is not the best — even among our (poor) model set — by exploring the following example. Suppose we have two equally probable categories (i.e., $P(\omega_1) = P(\omega_2) = 0.5$). Further, we know that $p(x|\omega_1) \sim N(0, 1)$ but *assume* that $p(x|\omega_2) \sim N(\mu, 1)$. (That is, the parameter θ we seek by maximum likelihood techniques is the mean of the second distribution.) Imagine however that the *true* underlying distribution is $p(x|\omega_2) \sim N(1, 10^6)$.

(a) What is the value of our maximum likelihood estimate $\hat{\mu}$ in our poor model, given a large amount of data?

(b) What is the decision boundary arising from this maximum likelihood estimate in the poor model?

(c) Ignore for the moment the maximum likelihood approach, and use the methods from Chap. **??** to derive the Bayes optimal decision boundary given the *true* underlying distributions — $p(x|\omega_1) \sim N(0, 1)$ and $p(x|\omega_2) \sim N(1, 10^6)$. Be careful to include all portions of the decision boundary.

(d) Now consider again classifiers based on the (poor) model assumption of $p(x|\omega_2) \sim N(\mu, 1)$. Using your result immediately above, find a *new* value of μ that will give lower error than the maximum likelihood classifier.

(e) Discuss these results, with particular attention to the role of knowledge of the underlying model.

8. Consider an extreme case of the general issue discussd in Problem 7, one in which it is possible that the maximum likelihood solution leads to the *worst* possible classifier, i.e., one with an error that approaches 100% (in probability). Suppose our data in fact comes from two one-dimensional distributions of the forms

$$\begin{aligned} p(x|\omega_1) &\sim [(1-k)\delta(x-1) + k\delta(x+X)] \quad \text{and} \\ p(x|\omega_2) &\sim [(1-k)\delta(x+1) + k\delta(x-X)], \end{aligned}$$

where X is positive, $0 \leq k < 0.5$ represents the portion of the total probability mass concentrated at the point $\pm X$, and $\delta(\cdot)$ is the Dirac delta function. Suppose our poor models are of the form $p(x|\omega_1, \mu_1) \sim N(\mu_1, \sigma_1^2)$ and $p(x|\omega_2, \mu_2) \sim N(\mu_2, \sigma_2^2)$ and we form a maximum likelihood classifier.

(a) Consider the symmetries in the problem and show that in the infinite data case the decision boundary will always be at $x = 0$, regardless of k and X.

(b) Recall that the maximum likelihood estimate of either mean, $\hat{\mu}_i$, is the mean of its distribution. For a fixed k, find the value of X such that the maximum likelihood estimates of the means "switch," i.e., where $\hat{\mu}_1 \geq \hat{\mu}_2$.

(c) Plot the true distributions and the Gaussian estimates for the particular case $k = .2$ and $X = 5$. What is the classification error in this case?

(d) Find a dependence $X(k)$ which will guarantee that the estimated mean $\hat{\mu}_1$ of $p(x|\omega_1)$ is less than zero. (By symmetry, this will also insure $\hat{\mu}_2 > 0$.)

(e) Given your $X(k)$ just derived, state the classification error in terms of k.

(f) Suppose we constrained our model space such that $\sigma_1^2 = \sigma_2^2 = 1$ (or indeed any other constant). Would that change the above results?

(g) Discuss how if our model is wrong (here, does not include the delta functions), the error can approaches 100% (in probability). Does this surprising answer arise because we have found some local minimum in parameter space?

9. Prove the invariance property of maximum likelihood estimators, i.e., that if $\hat{\theta}$ is the maximum likelihood estimate of θ, then for any differentiable function $\tau(\cdot)$, the maximum likelihood estimate of $\tau(\theta)$ is $\tau(\hat{\theta})$.

10. Suppose we employ a novel method for estimating the mean of a data set $\mathcal{D} = \{\mathbf{x}_1, \mathbf{x}_2, ..., \mathbf{x}_n\}$: we assign the mean to be the value of the first point in the set, i.e., $\mathbf{x}_1$.

(a) Show that this method is unbiased.

(b) State why this method is nevertheless highly undesirable.

11. One measure of the difference between two distributions in the same space is the *Kullback-Leibler divergence* of Kullback-Leibler "distance":

$$D_{KL}(p_1(\mathbf{x}), p_2(\mathbf{x})) = \int p_1(\mathbf{x}) \ln \frac{p_1(\mathbf{x})}{p_2(\mathbf{x})} d\mathbf{x}.$$

(This "distance," does not obey the requisite symmetry and triangle inequalities for a metric.) Suppose we seek to approximate an arbitrary distribution $p_2(\mathbf{x})$ by a normal $p_1(\mathbf{x}) \sim N(\boldsymbol{\mu}, \boldsymbol{\Sigma})$. Show that the values that lead to the smallest Kullback-Leibler divergence are the obvious ones:

$$\begin{aligned} \boldsymbol{\mu} &= \mathcal{E}_2[\mathbf{x}] \\ \boldsymbol{\Sigma} &= \mathcal{E}_2[(\mathbf{x}-\boldsymbol{\mu})(\mathbf{x}-\boldsymbol{\mu})^t], \end{aligned}$$

where the expectation taken is over the density $p_2(\mathbf{x})$.

⊕ *Section 3.3*

12. Justify all the statements in the text leading from Eq. 25 to Eq. 26.

⊕ *Section 3.4*

13. Let $p(\mathbf{x}|\boldsymbol{\Sigma}) \sim N(\boldsymbol{\mu}, \boldsymbol{\Sigma})$ where $\boldsymbol{\mu}$ is known and $\boldsymbol{\Sigma}$ is unknown. Show that the maximum likelihood estimate for $\boldsymbol{\Sigma}$ is given by

$$\widehat{\boldsymbol{\Sigma}} = \frac{1}{n}\sum_{k=1}^{n}(\mathbf{x}_k - \boldsymbol{\mu})(\mathbf{x}_k - \boldsymbol{\mu})^t$$

by carrying out the following argument:

(a) Prove the matrix identity $\mathbf{a}^t\mathbf{A}\mathbf{a} = \mathrm{tr}[\mathbf{A}\mathbf{a}\mathbf{a}^t]$, where the trace, $\mathrm{tr}[\mathbf{A}]$, is the sum of the diagonal elements of $\mathbf{A}$.

(b) Show that the likelihood function can be written in the form

$$p(\mathbf{x}_1, ..., \mathbf{x}_n|\boldsymbol{\Sigma}) = \frac{1}{(2\pi)^{nd/2}}|\boldsymbol{\Sigma}^{-1}|^{n/2}\exp\left[-\frac{1}{2}\mathrm{tr}\left[\boldsymbol{\Sigma}^{-1}\sum_{k=1}^{n}(\mathbf{x}_k-\boldsymbol{\mu})(\mathbf{x}_k-\boldsymbol{\mu})^t\right]\right].$$

(c) Let $\mathbf{A} = \boldsymbol{\Sigma}^{-1}\widehat{\boldsymbol{\Sigma}}$ and $\lambda_1, ..., \lambda_n$ be the eigenvalues of $\mathbf{A}$; show that your result above leads to

$$p(\mathbf{x}_1, ..., \mathbf{x}_n|\boldsymbol{\Sigma}) = \frac{1}{(2\pi)^{nd/2}|\widehat{\boldsymbol{\Sigma}}|^{n/2}}(\lambda_1\cdots\lambda_d)^{n/2}\exp\left[-\frac{n}{2}(\lambda_1+\cdots+\lambda_d)\right].$$

(d) Complete the proof by showing that the likelihood is maximized by the choice $\lambda_1 = \cdots = \lambda_d = 1$. Explain your reasoning.

14. Suppose that $p(\mathbf{x}|\boldsymbol{\mu}_i, \boldsymbol{\Sigma}, \omega_i) \sim N(\boldsymbol{\mu}_i, \boldsymbol{\Sigma})$, where $\boldsymbol{\Sigma}$ is a common covariance matrix for all c classes. Let n samples $\mathbf{x}_1, ..., \mathbf{x}_n$ be drawn as usual, and let $l_1, ..., l_n$ be their labels, so that $l_k = i$ if the state of nature for $\mathbf{x}_k$ was ω_i.

(a) Show that

$$p(\mathbf{x}_1, ..., \mathbf{x}_n, l_1, ..., l_n|\boldsymbol{\mu}_1, ..., \boldsymbol{\mu}_c, \boldsymbol{\Sigma}) = \frac{\prod_{k=1}^{n}P(\omega_{l_k})}{(2\pi)^{nd/2}|\boldsymbol{\Sigma}|^{n/2}} \exp\left[-\tfrac{1}{2}\sum_{k=1}^{n}(\mathbf{x}_k-\boldsymbol{\mu}_{l_k})^t\boldsymbol{\Sigma}^{-1}(\mathbf{x}_k-\boldsymbol{\mu}_{l_k})\right].$$

(b) Using the results for samples drawn from a single normal population, show that the maximum likelihood estimates for $\boldsymbol{\mu}_i$ and $\boldsymbol{\Sigma}$ are given by

$$\hat{\boldsymbol{\mu}} = \frac{\sum\limits_{l_k=i} \mathbf{x}_k}{\sum\limits_{l_k=1} 1}$$

and

$$\widehat{\boldsymbol{\Sigma}} = \frac{1}{n}\sum_{k=1}^{n}(\mathbf{x}_k - \hat{\boldsymbol{\mu}}_{l_k})(\mathbf{x}_k - \hat{\boldsymbol{\mu}}_{l_k})^t.$$

Interpret your answer in words.

15. Consider the problem of learning the mean of a univariate normal distribution. Let $n_0 = \sigma^2/\sigma_0^2$ be the dogmatism, and imagine that μ_0 is formed by averaging n_0 fictitious samples x_k, $k = -n_0 + 1, -n_0 + 2, ..., 0$.

(a) Show that Eqs. 32 & 33 for μ_n and σ_n^2 yield

$$\mu_n = \frac{1}{n + n_0}\sum_{k=-n_0+1}^{n} x_k$$

and

$$\sigma_n^2 = \frac{\sigma^2}{n + n_0}.$$

(b) Use this result to give an interpretation of the a priori density $p(\mu) \sim N(\mu_0, \sigma_0^2)$.

16. Suppose that $\mathbf{A}$ and $\mathbf{B}$ are nonsingular matrices of the same order.

(a) Prove the matrix identity

$$(\mathbf{A}^{-1} + \mathbf{B}^{-1})^{-1} = \mathbf{A}(\mathbf{A} + \mathbf{B})^{-1}\mathbf{B} = \mathbf{B}(\mathbf{A} + \mathbf{B})^{-1}\mathbf{A}.$$

(b) Must these matrixes be square for this identity to hold?

(c) Use this result in showing that Eqs. 46 & 47 do indeed follow from Eqs. 42 & 43.

⊕ *Section 3.5*

17. The purpose of this problem is to derive the Bayesian classifier for the d-dimensional multivariate Bernoulli case. As usual, work with each class separately, interpreting $P(\mathbf{x}|\mathcal{D})$ to mean $P(\mathbf{x}|\mathcal{D}_i, \omega_i)$. Let the conditional probability for a given category be given by

$$P(\mathbf{x}|\boldsymbol{\theta}) = \prod_{i=1}^{d} \theta_i^{x_i}(1 - \theta_i)^{1-x_i},$$

and let $\mathcal{D} = \{\mathbf{x}_1, ..., \mathbf{x}_n\}$ be a set of n samples independently drawn according to this probability density.

(a) If $\mathbf{s} = (s_1, ..., s_d)^t$ is the sum of the n samples, show that

$$P(\mathcal{D}|\boldsymbol{\theta}) = \prod_{i=1}^{d} \theta_i^{s_i}(1-\theta_i)^{n-s_i}.$$

(b) Assuming a uniform a priori distribution for $\boldsymbol{\theta}$ and using the identity

$$\int_0^1 \theta^m (1-\theta)^n \, d\theta = \frac{m!n!}{(m+n+1)!},$$

show that

$$p(\boldsymbol{\theta}|\mathcal{D}) = \prod_{i=1}^{d} \frac{(n+1)!}{s_i!(n-s_i)!}\theta_i^{s_i}(1-\theta_i)^{n-s_i}.$$

(c) Plot this density for the case $d = 1, n = 1$, and for the two resulting possibilities for s_1.

(d) Integrate the product $P(\mathbf{x}|\boldsymbol{\theta})p(\boldsymbol{\theta}|\mathcal{D})$ over $\boldsymbol{\theta}$ to obtain the desired conditional probability

$$P(\mathbf{x}|\mathcal{D}) = \prod_{i=1}^{d} \left(\frac{s_i+1}{n+2}\right)^{x_i} \left(1 - \frac{s_i+1}{n+2}\right)^{1-x_i}.$$

(e) If we think of obtaining $P(\mathbf{x}|\mathcal{D})$ by substituting an estimate $\hat{\boldsymbol{\theta}}$ for $\boldsymbol{\theta}$ in $P(\mathbf{x}|\boldsymbol{\theta})$, what is the effective Bayesian estimate for $\boldsymbol{\theta}$?

18. Consider how knowledge of an invariance can guide our creation of a prior in the following case. Suppose we have a binary (0 or 1) variable x, chosen independently with a probability $p(\theta) = p(x = 1)$. Imagine we have observed $\mathcal{D}^n = \{x_1, x_2, ..., x_n\}$, and now wish to evaluate the probability that $x_{n+1} = 1$, which we express as a ratio:

$$\frac{P(x_{n+1} = 1|\mathcal{D}^n)}{P(x_{n+1} = 0|\mathcal{D}^n)}.$$

(a) Define $s = x_1 + \cdots + x_n$ and $p(t) = P(x_1 + \cdots + x_{n+1} = t)$. Assume now invariance of exchangeability, i.e., that the samples in any set $\mathcal{D}^n$ could have been selected in an arbitrary order and it would not affect any probabilities. Show how this assumption of exchangeability implies the ratio in question can be written

$$\frac{p(s+1)/\binom{n+1}{s+1}}{p(s)/\binom{n+1}{s}},$$

where $\binom{n+1}{s} = \frac{(n+1)!}{s!(n+1-s)!}$ is the binomial coefficient.

(b) Evaluate this ratio given the assumption $p(s) \simeq p(s+1)$, when n and $n-s$ and s are not too small. Interpret your answer in words.

(c) In the binomial framework, we now seek a prior $p(\theta)$ such that $p(s)$ does not depend upon s, where

$$p(s) = \int_0^1 \binom{n}{s} \theta^s (1-\theta)^{n-s} p(\theta)\, d\theta.$$

Show that this requirement is satisfied if $p(\theta)$ is uniform, i.e., $p(\theta) \sim U(0,1)$.

19. Assume we have training data from a Gaussian distribution of known covariance $\boldsymbol{\Sigma}$ but unknown mean $\boldsymbol{\mu}$. Suppose further that this mean itself is random, and characterized by a Gaussian density having mean $\mathbf{m}_0$ and covariance $\boldsymbol{\Sigma}_0$.

(a) What is the MAP estimator for $\boldsymbol{\mu}$?

(b) Suppose we transform our coordinates by a linear transform $\mathbf{x}' = \mathbf{A}\mathbf{x}$, for non-singular matrix $\mathbf{A}$, and accordingly for other terms. Determine whether your MAP estimator gives the appropriate estimate for the transformed mean $\boldsymbol{\mu}'$. Explain.

20. Suppose for a given class with parameter s the density can be written as:

$$p(x|\alpha) = \frac{1}{\alpha} f\left(\frac{x}{\alpha}\right).$$

In such a case we say that α is a *scale* parameter. For instance, the standard deviation σ is a scale parameter for a one-dimensional Gaussian.

(a) Imagine that we measure $x' = \alpha x$ instead of x, for some constant α. Show that the density now can be written as

$$p(x'|\alpha') = \frac{1}{\alpha'} f\left(\frac{x'}{\alpha'}\right).$$

Find α'.

(b) Find the non-informative prior for α', written as $p'(\alpha')$. You will need to note that for any interval $\Delta \in (0, \infty)$ the following equation should hold:

$$\int_\Delta p(\alpha) d\alpha = \int_\Delta p'(\alpha')\, d\alpha'.$$

21. State the conditions on $p(\mathbf{x}|\boldsymbol{\theta})$, on $p(\boldsymbol{\theta})$, and on $\mathcal{D}^n$ that insure that the estimate $p(\boldsymbol{\theta}|\mathcal{D}^n)$ in Eq. 54 converges in the limit $n \to \infty$.

⨁ *Section 3.6*

22. Employ the notation of the chapter and suppose $\mathbf{s}$ is a sufficient statistic statistics for which $p(\boldsymbol{\theta}|\mathbf{s}, \mathcal{D}) = p(\boldsymbol{\theta}|\mathbf{s})$. Assume $p(\boldsymbol{\theta}|\mathbf{s}) \neq 0$ and prove that $p(\mathcal{D}|\mathbf{s}, \boldsymbol{\theta})$ is independent of $\boldsymbol{\theta}$.

23. Using the results given in Table 3.1, show that the maximum likelihood estimate for the parameter θ of a Rayleigh distribution is given by

$$\hat{\theta} = \frac{1}{\frac{1}{n}\sum_{k=1}^{n} x_k^2}.$$

24. Using the results given in Table 3.1, show that the maximum likelihood estimate for the parameter θ of a Maxwell distribution is given by

$$\hat{\theta} = \frac{3/2}{\frac{1}{n}\sum_{k=1}^{n} x_k^2}.$$

25. Using the results given in Table 3.1, show that the maximum likelihood estimate for the parameter θ of a multinomial distribution is given by

$$\hat{\theta}_i = \frac{s_i}{\sum_{j=1}^{d} s_j}.$$

where the vector $\mathbf{s} = (s_1, ..., s_d)^t$ is the average of the n samples $\mathbf{x}_1, ..., \mathbf{x}_n$.

26. Demonstrate that sufficiency is an integral concept, i.e., that if $\mathbf{s}$ is sufficient for $\boldsymbol{\theta}$, then corresponding components of $\mathbf{s}$ and $\boldsymbol{\theta}$ need not be sufficient. Do this for the case of a univariate Gaussian $p(x) \sim N(\mu, \sigma^2)$ where $\boldsymbol{\theta} = \binom{\mu}{\sigma^2}$ is the full vector of parameters.

(a) Verify that the statistic

$$\mathbf{s} = \begin{pmatrix} s_1 \\ s_2 \end{pmatrix} = \begin{bmatrix} \frac{1}{n}\sum_{k=1}^{n} x_k \\ \frac{1}{n}\sum_{k=1}^{n} x_k^2 \end{bmatrix}$$

is indeed sufficient for $\boldsymbol{\theta}$, as given in Table 3.1.

(b) Show that s_1 taken alone is not sufficient for μ. Does your answer depend upon whether σ^2 is known?

(c) Show that s_2 taken alone is not sufficient for σ^2. Does your answer depend upon whether μ is known?

27. Suppose $\mathbf{s}$ is a statistic for which $p(\boldsymbol{\theta}|\mathbf{x}, \mathcal{D}) = p(\boldsymbol{\theta}|\mathbf{s})$.

(a) Assume $p(\boldsymbol{\theta}|\mathbf{s}) \neq 0$, and prove that $p(\mathcal{D}|\mathbf{s}, \boldsymbol{\theta})$ is independent of $\boldsymbol{\theta}$.

(b) Create an example to show that the inequality $p(\boldsymbol{\theta}|\mathbf{s}) \neq 0$ is required for your proof above.

28. Consider the Cauchy distribution,

$$p(x) = \frac{1}{\pi b} \cdot \frac{1}{1 + \left(\frac{x-a}{b}\right)^2},$$

for $b > 0$ and arbitrary real a.

(a) Confirm that the distribution is indeed normalized.

(b) For a fixed a and b, try to calculate the mean and the standard deviation of the distribution. Explain your results.

(c) Prove that this distribution has no sufficient statistics for the mean and standard deviation.

⊕ *Section 3.7*

29. In the following, suppose a and b are constants and n a variable parameter.

(a) Is $a^{n+1} = O(a^n)$?

(b) Is $a^{bn} = O(a^n)$?

(c) Is $a^{n+b} = O(a^n)$?

(d) Prove $f(n) = O(f(n))$.

30. Consider the evaluation of a polynomial function $f(x) = \sum_{i=0}^{n-1} a_i x^i$, where the n coefficients a_i are given.

(a) Write pseudocode for a simple $\mathbf{\Theta}(n^2)$-time algorithm for evaluating $f(x)$.

(b) Show that such a polynomial can be rewritten as:

$$f(x) = \sum_{i=0}^{n-1} a_i x^i = (\cdots(a_{n-1}x + a_{n-2})x + \cdots + a_1)x + a_0,$$

and so forth — a method known as *Horner's rule*. Use the rule to write pseudocode for a $\mathbf{\Theta}(n)$-time algorithm for evaluating $f(x)$.

31. For each of the short procedures, state the computational complexity in terms of the variables N, M, P, and K, as appropriate. Assume that all data structures are defined, that those without indexes are scalars and that those with indexes have the number of dimensions shown.

Algorithm 6

```
1 begin for i ← i + 1
2           s ← s + i^3
3   until i = N
4 return s
5 end
```

Algorithm 7

```
1 begin for i ← i + 1
2           s ← s + x_i × x_i
3   until i = N
4 return √s
5 end
```

Algorithm 8

```
1  begin for j ← j + 1
2            for i ← i + 1
3                 s_j ← s_j + w_ij x_i
4            until i = I
5    until j = J
6    for k ← k + 1
7        for j ← j + 1
8            r_k ← r_k + w_jk s_j
9        until j = J
10   until k = K
11 end
```

32. Consider a computer having a uniprocessor that can perform one operation per nanosecond (10^{-9} sec). The left column of the table shows the functional dependence of such operations in different hypothetical algorithms. For each such function, fill in the number of operations n that can be performed in the total time listed along the top.

$f(n)$	1 sec	1 hour	1 day	1 year
$\log_2 n$				
$\sqrt{n}$				
n				
$n\log_2 n$				
n^2				
n^3				
2^n				
e^n				
$n!$				

33. Show that the estimator of Eq. 21 is indeed unbiased for:

(a) Normal distributions.

(b) Cauchy distributions.

(c) Binomial distributions.

(d) Prove that the estimator of Eq. 20 is asymptotically unbiased.

34. Let the sample mean $\hat{\boldsymbol{\mu}}_n$ and the sample covariance matrix $\mathbf{C}_n$ for a set of n samples $\mathbf{x}_1, ..., \mathbf{x}_n$ (each of which is d-dimensional) be defined by

$$\hat{\boldsymbol{\mu}}_n = \frac{1}{n}\sum_{k=1}^{n}\mathbf{x}_k$$

and

$$\mathbf{C}_n = \frac{1}{n-1}\sum_{k=1}^{n}(\mathbf{x}_k - \hat{\boldsymbol{\mu}}_n)(\mathbf{x}_k - \hat{\boldsymbol{\mu}}_n)^t.$$

We call these the "non-recursive" formulas.

(a) What is the computational complexity of calculating $\hat{\boldsymbol{\mu}}_n$ and $\mathbf{C}_n$ by these formulas?

(b) Show that alternative, "recursive" techniques for calculating $\hat{\boldsymbol{\mu}}_n$ and $\mathbf{C}_n$ based on the successive addition of new samples $\mathbf{x}_{n+1}$ can be derived using the recursion relations

$$\hat{\boldsymbol{\mu}}_{n+1} = \hat{\boldsymbol{\mu}}_n + \frac{1}{n+1}(\mathbf{x}_{n+1} - \hat{\boldsymbol{\mu}}_n)$$

and

$$\mathbf{C}_{n+1} = \frac{n-1}{n}\mathbf{C}_n + \frac{1}{n+1}(\mathbf{x}_{n+1} - \hat{\boldsymbol{\mu}}_n)(\mathbf{x}_{n+1} - \hat{\boldsymbol{\mu}}_n)^t.$$

(c) What is the computational complexity of finding $\hat{\boldsymbol{\mu}}_n$ and $\mathbf{C}_n$ by these recursive methods?

(d) Describe situations where you might prefer to use the recursive method for computing $\hat{\boldsymbol{\mu}}_n$ and $\mathbf{C}_n$, and ones where you might prefer the non-recursive method.

35. In pattern classification, one is often interested in the inverse of the covariance matrix, for instance when designing a Bayes classifier for Gaussian distributions. Note that the non-recursive calculation of $\mathbf{C}_n^{-1}$ (the inverse of the covariance matrix based on n samples, cf., Problem 34) might require the $O(n^3)$ inversion of $\mathbf{C}_n$ by standard matrix methods. We now explore an alternative, "recursive" method for computing $\mathbf{C}_n^{-1}$.

(a) Prove the so-called Sherman-Morrison-Woodbury matrix identity

$$(\mathbf{A} + \mathbf{x}\mathbf{y}^t)^{-1} = \mathbf{A}^{-1} - \frac{\mathbf{A}^{-1}\mathbf{x}\mathbf{y}^t\mathbf{A}^{-1}}{1 + \mathbf{y}^t\mathbf{A}^{-1}\mathbf{x}}.$$

(b) Use this and the results of Problem 34 to show that

$$\mathbf{C}_{n+1}^{-1} = \frac{n}{n-1}\left[\mathbf{C}_n^{-1} - \frac{\mathbf{C}_n^{-1}(\mathbf{x}_{n+1} - \hat{\boldsymbol{\mu}}_n)(\mathbf{x}_{n+1} - \hat{\boldsymbol{\mu}}_n)^t\mathbf{C}_n^{-1}}{\frac{n^2-1}{n} + (\mathbf{x}_{n+1} - \hat{\boldsymbol{\mu}}_n)^t\mathbf{C}_n^{-1}(\mathbf{x}_{n+1} - \hat{\boldsymbol{\mu}}_n)}\right].$$

(c) What is the computational complexity of this calculation?

(d) Describe situations where you would use the recursive method, and ones where you would use instead the non-recursive method.

36. Suppose we wish to simplify (or regularize) a Gaussian classifier for two categories by means of shrinkage. Suppose that the estimated distributions are $N(\boldsymbol{\mu}_1, \boldsymbol{\Sigma}_1)$ and $N(\boldsymbol{\mu}_2, \boldsymbol{\Sigma}_2)$. In order to employ shrinkage of an assumed common covariance toward the identity matrix as given in Eq. 77, show that one must first normalize the data to have unit variance.

⊕ *Section 3.8*

37. Consider the convergence of the Expectation-Maximization algorithm, i.e., that if $l(\boldsymbol{\theta}, \mathcal{D}_g) = \ln p(\mathcal{D}_g;\ \boldsymbol{\theta})$ is not already optimum, then the EM algorithm increases it. Prove this as follows:

(a) First note that

$$l(\boldsymbol{\theta};\ \mathcal{D}_g) = \ln p(\mathcal{D}_g, \mathcal{D}_b;\ \boldsymbol{\theta}) - \ln p(\mathcal{D}_b|\mathcal{D}_g;\ \boldsymbol{\theta}).$$

Let $\mathcal{E}'[\cdot]$ denote the expectation with respect to the distribution $p(\mathcal{D}_b|\mathcal{D}_g;\ \boldsymbol{\theta}')$. Take such an expectation of $l(\boldsymbol{\theta};\ \mathcal{D}_g)$, and express your answer in terms of $Q(\boldsymbol{\theta}, \boldsymbol{\theta}')$ of Eq. 78.

(b) Define $\phi(\mathcal{D}_b) = p(\mathcal{D}_b|\mathcal{D}_g;\ \boldsymbol{\theta})/p(\mathcal{D}_b|\mathcal{D}_g;\ \boldsymbol{\theta}')$ to be the ratio of expectations assuming the two distributions. Show that $\mathcal{E}'[\ln\phi(\mathcal{D}_b)] \leq \mathcal{E}'[\phi(\mathcal{D}_b)] - 1 = 0$.

(c) Use this result to show that if $Q(\boldsymbol{\theta}^{t+1}, \boldsymbol{\theta}^t) > Q(\boldsymbol{\theta}^t, \boldsymbol{\theta}^t)$, achieved by the **M step** in Algorithm **??**, then $l(\boldsymbol{\theta}^{t+1};\ \mathcal{D}_g) > l(\boldsymbol{\theta}^t;\ \mathcal{D}_g)$.

38. Suppose we seek to estimate $\boldsymbol{\theta}$ describing a multidimensional distribution from data $\mathcal{D}$, some of whose points are missing features. Consider an iterative algorithm in which the maximum likelihood value of the missing values is calculated, then assumed to be correct for the purposes of restinating $\boldsymbol{\theta}$ and iterated.

(a) Is this always equivalent to an Expectation-Maximization algorithm, or just a generalized Expectation-Maximization algorithm?

(b) If it is an Expectation-Maximization algorithm, what is $Q(\boldsymbol{\theta}, \boldsymbol{\theta}^t)$, as described by Eq. 78?

39. Consider data $\mathcal{D} = \left\{\binom{2}{3}, \binom{3}{1}, \binom{5}{4}, \binom{4}{*}, \binom{*}{6}\right\}$, sampled from a two-dimensional uniform distribution

$$p(\mathbf{x}) \sim U(\mathbf{x}_l, \mathbf{x}_u) = \begin{cases} \frac{1}{|x_{u1} - x_{l1}||x_{u2} - x_{l2}|} & \text{if } x_{l1} \leq x_1 \leq x_{u1} \\ & \text{and } x_{l2} \leq x_2 \leq x_{u2} \\ 0 & \text{otherwise,} \end{cases}$$

where $*$ represents missing feature values.

(a) Start with an initial estimate

$$\boldsymbol{\theta}^0 = \begin{pmatrix} \mathbf{x}_l \\ \mathbf{x}_u \end{pmatrix} = \begin{pmatrix} 0 \\ 0 \\ 10 \\ 10 \end{pmatrix},$$

and analytically calculate $Q(\boldsymbol{\theta}, \boldsymbol{\theta}^0)$ — the **E step** in the EM algorithm.

(b) Find the $\boldsymbol{\theta}$ that maximizes your $Q(\boldsymbol{\theta}, \boldsymbol{\theta}^0)$ — the **M step**.

(c) Plot your data and the bounding rectangle.

(d) Without having to iterate further, state the estimate of $\boldsymbol{\theta}$ that would result after convergence of the EM algorithm.

40. Consider data $\mathcal{D} = \left\{\binom{1}{1}, \binom{3}{3}, \binom{2}{*}\right\}$, sampled from a two-dimensional (separable) distribution $p(x_1, x_2) = p(x_1)p(x_2)$, with

$$p(x_1) \sim \begin{cases} \frac{1}{\theta_1} e^{-\theta_1 x_1} & \text{if } x_1 \geq 0 \\ 0 & \text{otherwise,} \end{cases}$$

and

$$p(x_2) \sim U(0, \theta_2) = \begin{cases} \frac{1}{\theta_2} & \text{if } 0 \leq x_2 \leq \theta \\ 0 & \text{otherwise.} \end{cases}$$

As usual, $*$ represents a missing feature value.

(a) Start with an initial estimate $\boldsymbol{\theta}^0 = \binom{2}{4}$ and analytically calculate $Q(\boldsymbol{\theta}, \boldsymbol{\theta}^0)$ — the **E step** in the EM algorithm. Be sure to consider the normalization of your distribution.

(b) Find the $\boldsymbol{\theta}$ that maximizes your $Q(\boldsymbol{\theta}, \boldsymbol{\theta}^0)$ — the **M step**.

(c) Plot your data on a two-dimensional graph and indicate the new parameter estimates.

41. Repeat Problem 40 but with data $\mathcal{D} = \left\{\binom{1}{1}, \binom{3}{3}, \binom{*}{2}\right\}$.

⨁ *Section 3.9*

42. Use the conditional probability matrices in Example 3 to answer the following separate problems.

(a) Suppose it is December 20 — the end of autumn and the beginning of winter — and thus let $P(a_1) = P(a_4) = 0.5$. Furthermore, it is known that the fish was caught in the north Atlantic, i.e., $P(b_1) = 1$. Suppose the lightness has not been measured but it is known that the fish is thin, i.e., $P(d_2) = 1$. Classify the fish as salmon or sea bass. What is the expected error rate?

(b) Suppse all we know is that a fish is thin and medium lightness. What season is it now, most likely? What is your probability of being correct?

(c) Suppose we know a fish is thin and medium lightness and that it was caught in the north Atlantic. What season is it, most likely? What is the probability of being correct?

43. One of the simplest assumptions is that of the naive Bayes rule or idiot Bayes rule expressed in Eq. 85. Draw the belief net for a three-category problem with five features x_i, $i = 1, 2, ...5$.

44. Consider a Bayesian belief net with several nodes having unspecified values. Suppose that one such node is selected at random, the probabilities of its nodes computed by the formulas described in the text. Next another such node is chosen at random (possibly even a node already visited), and the probabilities similarly updated. Prove that this procedure will converge to the desired probabilities throughout the full network.

⨁ *Section 3.10*

45. Consider training an HMM by the *Forward-backward algorithm*, for a single sequence of length T where each symbol could be one of c values. What is the computational complexity of a single revision of all values $\hat{a}_{ij}$ and $\hat{b}_{jk}$?

46. The standard method for calculating the probability of a sequence in a given HMM is to use the forward probabilities $\alpha_i(t)$.

(a) Show by a simple substitution that a symmetric method can be derived using the backward probabilities $\beta_i(t)$.

(b) Prove that one can get the probability by combining the forward and the backward probabilities at any place in the middle of the sequence. That is, show that

$$P(\boldsymbol{\omega}^{T'}) = \sum_{i=1}^{T'} \alpha_i(t)\beta_i(t),$$

where $\boldsymbol{\omega}^{T'}$ is a particular sequence of length $T' < T$.

(c) Show that your formula reduces to the known values at the beginning and end of the sequence.

47. Suppose we have a large number of symbol sequences emitted from an HMM that has a particular transition probability $a_{i'j'} = 0$ for some single value of i' and j'. We use such sequences to train a new HMM, one that happens also to start with its $a_{i'j'} = 0$. Prove that this parameter will remain 0 throughout training by the *Forward-backward* algorithm. In other words, if the topology of the trained model (pattern of non-zero connections) matches that of the generating HMM, it will remain so after training.

48. Consider the decoding algorithm (*Algorithm* 4) in the text.

(a) Take logarithms of HMM model parameters and write pseudocode for an equivalent algorithm.

(b) Explain why taking logarithms is an $O(n)$ calculation, and thus the complexity of your algorithm in (a) is $O(c^2T)$.

49. Explore the close relationship between Bayesian belief nets and hidden Markov models as follows.

(a) Prove that the forward and the backward equations for hidden Markov models are special cases of Eq. 84.

(b) Use your answer to explain the relationship between these two general classes of models.

Computer exercises

Several exercises will make use of the following three-dimensional data sampled from three categories, denoted ω_i.

	ω_1			ω_2			ω_3		
point	x_1	x_2	x_3	x_1	x_2	x_3	x_1	x_2	x_3
1	0.42	-0.087	0.58	-0.4	0.58	0.089	0.83	1.6	-0.014
2	-0.2	-3.3	-3.4	-0.31	0.27	-0.04	1.1	1.6	0.48
3	1.3	-0.32	1.7	0.38	0.055	-0.035	-0.44	-0.41	0.32
4	0.39	0.71	0.23	-0.15	0.53	0.011	0.047	-0.45	1.4
5	-1.6	-5.3	-0.15	-0.35	0.47	0.034	0.28	0.35	3.1
6	-0.029	0.89	-4.7	0.17	0.69	0.1	-0.39	-0.48	0.11
7	-0.23	1.9	2.2	-0.011	0.55	-0.18	0.34	-0.079	0.14
8	0.27	-0.3	-0.87	-0.27	0.61	0.12	-0.3	-0.22	2.2
9	-1.9	0.76	-2.1	-0.065	0.49	0.0012	1.1	1.2	-0.46
10	0.87	-1.0	-2.6	-0.12	0.054	-0.063	0.18	-0.11	-0.49

⨁ *Section 3.2*

1. Consider Gaussian density models in different dimensions.

(a) Write a program to find the maximum likelihood values $\hat{\mu}$ and $\hat{\sigma}^2$. Apply your program individually to each of the three features x_i of category ω_1 in the table above.

(b) Modify your program to apply to two-dimensional Gaussian data $p(\mathbf{x}) \sim N(\boldsymbol{\mu}, \boldsymbol{\Sigma})$. Apply your data to each of the three possible pairings of two features for ω_1.

(c) Modify your program to apply to three-dimensional Gaussian data. Apply your data to the full three-dimensional data for ω_1.

(d) Assume your three-dimensional model is separable, so that $\boldsymbol{\Sigma} = diag(\sigma_1^2, \sigma_2^2, \sigma_3^2)$. Write a program to estimate the mean and the diagonal components of $\boldsymbol{\Sigma}$. Apply your program to the data in ω_2.

(e) Compare your results for the mean of each feature μ_i calculated in the above ways. Explain why they are the same or different.

(f) Compare your results for the variance of each feature σ_i^2 calculated in the above ways. Explain why they are the same or different.

⨁ *Section 3.3*

2. Consider a one-dimensional model of a triangular density governed by two scalar parameters:

$$p(x|\boldsymbol{\theta}) \equiv T(\mu, \delta) = \begin{cases} (\delta - |x - \mu|)/\delta^2 & \text{for } |x - \mu| < \delta \\ 0 & \text{otherwise,} \end{cases}$$

where $\boldsymbol{\theta} = \binom{\mu}{\delta}$. Write a program to calculate a density $p(x|\mathcal{D})$ via Bayesian methods (Eq. 26) and apply it to the x_2 feature of category ω_2. Plot your resulting posterior density $p(x|\mathcal{D})$.

⨁ *Section 3.4*

3. Consider Bayesian estimation of the mean of a one-dimensional Gaussian. Suppose you are given the prior for the mean is $p(\mu) \sim N(\mu_0, \sigma_0)$.

(a) Write a program that plots the density $p(x|\mathcal{D})$ given μ_0, σ_0, σ and training set $\mathcal{D} = \{x_1, x_2, ..., x_n\}$.

(b) Estimate σ for the x_2 component of ω_3 in the table above. Now assume $\mu_0 = -1$ and plot your estimated densities $p(x|\mathcal{D})$ for each of the following values of the dogmatism, σ^2/σ_0^2: 0.1, 1.0, 10, 100.

⊕ *Section 3.5*

4. Suppose we have reason to believe that our data is sampled from a two-dimensional uniform density

$$p(\mathbf{x}|\boldsymbol{\theta}) \sim U(\mathbf{x}_l, \mathbf{x}_u) = \begin{cases} \frac{1}{|x_{u1}-x_{l1}||x_{u2}-x_{l2}|} & \text{for } x_{l1} \le x_1 \le x_{u1} \text{ and } x_{l2} \le x_2 \le x_{u2} \\ 0 & \text{otherwise,} \end{cases}$$

where x_{l1} is the x_1 component of the "lower" bounding point $\mathbf{x}_l$, and analogously for the x_2 component and for the upper point. Suppose we have reliable prior information that the density is zero outside the box defined by $\mathbf{x}_l = \binom{-6}{-6}$ and $\mathbf{x}_u = \binom{+6}{+6}$. Write a program that calculates $p(\mathbf{x}|\mathcal{D})$ via recursive Bayesian estimation and apply it to the $x_1 - x_2$ components of ω_1, in sequence, from the table above. For each expanding data set $\mathcal{D}^n$ $(2 \le n \le 10)$ plot your posterior density.

⊕ *Section 3.6*

5. Write a single program to calculate sufficient statistics for any members of the exponential family (Eq. 69). Assume that the x_3 data from ω_3 in the table come from an exponential density, and use your program to calculate the sufficient statistics for each of the following exponential forms: Gaussian, Rayleigh and Maxwell.

⊕ *Section 3.7*

6. Consider error rates in different dimensions.

(a) Use maximum likelihood to train a dichotomizer using the three-dimensional data for categories ω_1 and ω_2 in the Table above. Numerically integrate to estimate the classification error rate.

(b) Now consider the data projected into a two-dimensional subspace. For each of the three subspaces — defined by $x_1 = 0$ or $x_2 = 0$ or $x_3 = 0$ — train a Gaussian dichotomizer. Numerically integrate to estimate the error rate.

(c) Now consider the data projected onto one-dimensional subspaces, defined by each of the three axes. Train a Gaussian classifier, and numerically integrate to estimate the error rate.

(d) Discuss the rank order of the error rates you find.

(e) Assuming that you resestimate the distribution in the different dimensions, logically must the Bayes error be higher in the projected spaces.

7. Repeat the steps in Exercise 6 but for categories ω_1 and ω_3.

8. Consider the classification of Gaussian data employing shrinkage of covariance matrixes to a common one.

(a) Generate 20 training points from each of three equally probable three-dimensional Gaussian distributions $N(\boldsymbol{\mu}_i, \boldsymbol{\Sigma}_i)$ with the following parameters:

$$\begin{aligned} \boldsymbol{\mu}_1 &= (0,0,0)^t, & \boldsymbol{\Sigma}_1 &= diag[3,5,2] \\ \boldsymbol{\mu}_2 &= (1,5,-3)^t, & \boldsymbol{\Sigma}_2 &= \begin{pmatrix} 1 & 0 & 0 \\ 0 & 4 & 1 \\ 0 & 1 & 6 \end{pmatrix} \\ \boldsymbol{\mu}_3 &= (0,0,0)^t, & \boldsymbol{\Sigma}_3 &= 10\mathbf{I}. \end{aligned}$$

(b) Write a program to estimate the means and covariances of your data.

(c) Write a program that takes α and shrinks these estimated covariance matrixes according to Eq. 76.

(d) Plot the training error as a function of α, where $0 < \alpha < 1$.

(e) Use your program from part (a) to generate 50 test points from each category. Plot the test error as a function of α.

⊕ *Section 3.8*

9. Suppose we know that the ten data points in category ω_1 in the table above come from a three-dimensional Gaussian. Suppose, however, that we do not have access to the x_3 components for the even-numbered data points.

(a) Write an EM program to estimate the mean and covariance of the distribution. Start your estimate with $\boldsymbol{\mu}^0 = \mathbf{0}$ and $\boldsymbol{\Sigma}^0 = \mathbf{I}$, the three-dimensional identity matrix.

(b) Compare your final esimate with that for the case when there is no missing data.

10. Suppose we know that the ten data points in category ω_2 in the table above come from a three-dimensional uniform distribution $p(\mathbf{x}|\omega_2) \sim U(\mathbf{x}_l, \mathbf{x}_u)$. Suppose, however, that we do not have access to the x_3 components for the even-numbered data points.

(a) Write an EM program to estimate the six scalars comprising $\mathbf{x}_l$ and $\mathbf{x}_u$ of the distribution. Start your estimate with $\mathbf{x}_l = (-2,-2,-2)^t$ and $\mathbf{x}_u = (+2,+2,+2)^t$.

(b) Compare your final esimate with that for the case when there is no missing data.

⊕ *Section 3.9*

Write a program to evaluate the Bayesian belief net for fish in Example 3, including the information in $P(x_i|a_j)$,P(x$_i$|b_j), $P(c_i|x_j)$, and $P(d_i|x_j)$. Test your program on the calculation given in the Example. Apply your program to the following cases, and state any assumptions you need to make.

(a) A dark, thin fish is caught in the north Atlantic in summer. What is the probability it is a salmon?

(b) A thin, medium fish is caught in the north Atlantic. What is the probability it is winter? spring? summer? autumn?

(c) A light, wide fish is caught in the autumn. What is the probability it came from the north Atlantic?

⨁ *Section 3.10*

11. Consider the use of hidden Markov models for classifying sequences of four visible states, `A-D`. Train two hidden Markov models, each consisting of three hidden states (plus a null initial state and a null final state), fully connected, with the following data. Assume that each sequence starts with a null symbol and ends with an end null symbol (not listed).

sample	ω_1	ω_2
1	AABBCCDD	DDCCBBAA
2	ABBCBBDD	DDABCBA
3	ACBCBCD	CDCDCBABA
4	AD	DDBBA
5	ACBCBABCDD	DADACBBAA
6	BABAADDD	CDDCCBA
7	BABCDCC	BDDBCAAAA
8	ABDBBCCDD	BBABBDDDCD
9	ABAAACDCCD	DDADDBCAA
10	ABD	DDCAAA

(a) Print out the full transition matrices for each of the models.

(b) Assume equal prior probabilities for the two models and classify each of the following sequences: `ABBBCDDD`, `DADBCBAA`, `CDCBABA`, and `ADBBBCD`.

(c) As above, classify the test pattern `BADBDCBA`. Find the prior probabilities for your two trained models that would lead to equal posteriors for your two categories when applied to this pattern.

Bibliography

[1] Pierre Baldi, Søren Brunak, Yves Chauvin, Jacob Engelbrecht, and Anders Krogh. Hidden Markov models for human genes. In Stephen J. Hanson, Jack D. Cowan, and C. Lee Giles, editors, *Neural Information Processing Systems*, volume 6, pages 761–768, San Mateo, CA, 1994. Morgan Kaufmann.

[2] Leonard E. Baum and Ted Petrie. Statistical inference for probabilistic functions of finite state Markov chains. *Annals of Mathematical Statistics*, 37:1554–1563, 1966.

[3] Leonard E. Baum, Ted Petrie, George Soules, and Norman Weiss. A maximization technique occurring in the statistical analysis of probabilistic functions of Markov chains. *Annals of Mathematical Statistics*, 41(1):164–171, 1970.

[4] José M. Bernardo and Adrian F. M. Smith. *Bayesian Theory*. John Wiley, New York, NY, 1996.

[5] Christopher M. Bishop. *Neural Networks for Pattern Recognition*. Oxford University Press, Oxford, UK, 1995.

[6] David Braverman. Learning filters for optimum pattern recognition. *IRE Transactions on Information Theory*, IT-8:280–285, 1962.

[7] Wray L. Buntine. Operations for learning with graphical models. *Journal of Artificial Intelligence Research*, 2:159–225, 1994.

[8] Wray L. Buntine. A guide to the literature on learning probabilistic networks from data. *IEEE Transactions on Knowledge and Data Engineering*, 8(2):195–210, 1996.

[9] Eugene Charniak. *Statistical Language Learning*. MIT Press, Cambridge, MA, 1993.

[10] Thomas H. Cormen, Charles E. Leiserson, and Ronald L. Rivest. *Introduction to Algorithms*. MIT Press, Cambridge, MA, 1990.

[11] Arthur P. Dempster, Nan M. Laird, and Donald B. Rubin. Maximum likelihood from incomplete data via the EM algorithm (with discussion). *Journal of the Royal Statistical Society, Series B*, 39:1–38, 1977.

[12] G. David Forney, Jr. The Viterbi algorithm. *Proceedings of the IEEE*, 61:268–278, 1973.

[13] Peter E. Hart and Jamey Graham. Query-free information retrieval. *IEEE Expert: Intelligent Systems and Their Application*, 12(5):32–37, 1997.

[14] David Heckerman. *Probabilistic Similarity Networks*. ACM Doctoral Dissertation Award Series. MIT Press, Cambridge, MA, 1991.

[15] Harold Jeffreys. *Theory of Probability*. Oxford University Press, Oxford, UK, 1961 reprint edition, 1939.

[16] Michael I. Jordan, editor. *Learning in Graphical Models*. Kluwer, Dortrecht, Netherlands, 1998.

[17] Michael I. Jordan and Robert A. Jacobs. Hierarchical mixtures of experts and the EM algorithm. *Neural Computation*, 6:181–214, 1994.

[18] Donald E. Knuth. *The Art of Computer Programming*, volume 1. Addison-Wesley, Reading, MA, 1 edition, 1973.

[19] Gary E. Kopec and Phil A. Chou. Document image decoding using Markov source models. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 16(6):602–617, 1994.

[20] Anders Krogh, Michael Brown, I. Saira Mian, Kimmen Sjölander, and David Haussler. Hidden Markov models in computational biology: Applications to protein modelling. *Journal of Molecular Biology*, 235:1501–1531, 1994.

[21] Dennis Victor Lindley. The use of prior probability distributions in statistical inference and decision. In Jerzy Neyman and Elizabeth L. Scott, editors, *Proceedings Fourth Berkeley Symposium on Mathematical Statistics and Probability*, Berkeley, CA, 1961. U. California Press.

[22] Andrei Andreivich Markov. xxx. *xxx*, xxx(xxx):xxx–xxx, 1907.

[23] Geoffrey J. McLachlan and Thiriyambakam Krishnan. *The EM Algorithm and Extensions*. Wiley Interscience, New York, NY, 1996.

[24] Judea Pearl. *Probabilistic Reasoning in Intelligent Systems: Networks of Plausible Inference*. Morgan Kaufmann, San Mateo, CA, 1988.

[25] Lawrence Rabiner and Biing-Hwang Juang. *Fundamentals of Speech Recognition*. Prentice Hall, Englewood Cliffs, NJ, 1993.

[26] Lawrence R. Rabiner. A tutorial on hidden Markov models and selected applications in speech recognition. *Proceedings of IEEE*, 77(2):257–286, 1989.

[27] Donald B. Rubin and Roderick J. A. Little. *Statistical Analysis with Missing Data*. John Wiley, New York, NY, 1987.

[28] Jürgen Schürmann. *Pattern Classification: A unified view of statistical and neural approaches*. John Wiley and Sons, New York, NY, 1996.

[29] Padhraic Smyth, David Heckerman, and Michael Jordan. Probabilistic independence networks for hidden Markov probability models. *Neural Computation*, 9:227–269, 1997.

[30] Charles W. Therrien. *Decision Estimation and Classification: An Introduction to Pattern Recognition and Related Topics*. Wiley Interscience, New York, NY, 1989.

[31] D. Michael Titterington. Recursive parameter estimation using incomplete data. *Jounal of the Royal Statistical Society series B*, 46:257–267, 1984.

[32] Andrew J. Viterbi. Error bounds for convolutional codes and an asymptotically optimal decoding algorithm. *IEEE Transactions on Information Theory*, IT-13:260–269, 1967.

[33] Sewal Wright. Correlation and causation. *Journal of Agricultural Research*, 20:557–585, 1921.

Index