17 Parameter Estimation

In this chapter, we discuss the problem of estimating parameters for a Bayesian network. We assume that the network structure is fixed and that our data set $\mathcal{D}$ consists of fully observed instances of the network variables: $\mathcal{D} = \{\xi[1], \ldots, \xi[M]\}$. This problem arises fairly often in practice, since numerical parameters are harder to elicit from human experts than structure is. It also plays a key role as a building block for both structure learning and learning from incomplete data. As we will see, despite the apparent simplicity of our task definition, there is surprisingly much to say about it.

As we will see, there are two main approaches to dealing with the parameter-estimation task: one based on *maximum likelihood estimation*, and the other using Bayesian approaches. For each of these approaches, we first discuss the general principles, demonstrating their application in the simplest context: a Bayesian network with a single random variable. We then show how the structure of the distribution allows the techniques developed in this very simple case to generalize to arbitrary network structures. Finally, we show how to deal with parameter estimation in the context of structured CPDs.

17.1 Maximum Likelihood Estimation

In this section, we describe the basic principles behind maximum likelihood estimation.

17.1.1 The Thumbtack Example

We start with what may be considered the simplest learning problem: parameter learning for a single variable. This is a classical Statistics 101 problem that illustrates some of the issues that we will encounter in more complex learning problems. Surprisingly, this simple problem already contains some interesting issues that we need to tackle.

Imagine that we have a thumbtack, and we conduct an experiment whereby we flip the thumbtack in the air. It comes to land as either heads or tails, as in figure 17.1. We toss the thumbtack several times, obtaining a data set consisting of *heads* or *tails* outcomes. Based on this data set, we want to estimate the probability with which the next flip will land heads or tails. In this description, we already made the implicit assumption that the thumbtack tosses are controlled by an (unknown) *parameter* θ, which describes the frequency of heads in thumbtack tosses. In addition, we also assume that the data instances are independent and identically distributed (IID).

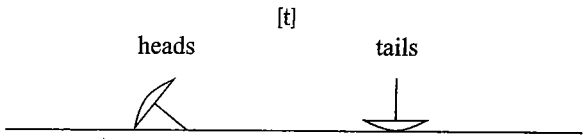


Figure 17.1 A simple thumbtack tossing experiment

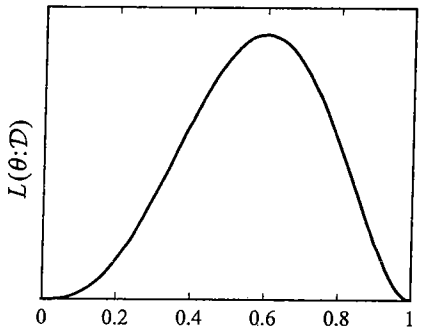


Figure 17.2 The likelihood function for the sequence of tosses H, T, T, H, H

Assume that we toss the thumbtack 100 times, of which 35 come up heads. What is our estimate for θ? Our intuition suggests that the best estimate is 0.35. Had θ been 0.1, for example, our chances of seeing $35/100$ heads would have been much lower. In fact, we examined a similar situation in our discussion of sampling methods in section 12.1, where we used samples from a distribution to estimate the probability of a query. As we discussed, the central limit theorem shows that, as the number of coin tosses grows, it is increasingly unlikely to sample a sequence of IID thumbtack flips where the fraction of tosses that come out heads is very far from θ. Thus, for sufficiently large M, the fraction of heads among the tosses is a good estimate with high probability.

To formalize this intuition, assume that we have a set of thumbtack tosses $x[1], \ldots, x[M]$ that are IID, that is, each is sampled independently from the same distribution in which $X[m]$ is equal to H (heads) or T (tails) with probability θ and $1 - \theta$, respectively. Our task is to find a good value for the parameter θ. As in many formulations of learning tasks, we define a *hypothesis space* Θ — a set of possibilities that we are considering — and an *objective function* that tells us how good different hypotheses in the space are relative to our data set $\mathcal{D}$. In this case, our hypothesis space Θ is the set of all parameters $\theta \in [0, 1]$.

hypothesis space

objective function

How do we score different possible parameters θ? As we discussed in section 16.3.1, one way of evaluating θ is by how well it predicts the data. In other words, if the data are likely given the parameter, the parameter is a good predictor. For example, suppose we observe the sequence of outcomes H, T, T, H, H. If we know θ, we could assign a probability to observing this particular sequence. The probability of the first toss is $P(X[1] = H) = \theta$. The probability of the second toss is $P(X[2] = T \mid X[1] = H)$, but our assumption that the coin tosses are independent allows us to conclude that this probability is simply $P(X[2] = T) = 1 - \theta$. This

is also the probability of the third outcome, and so on. Thus, the probability of the sequence is

$$P(\langle H,T,T,H,H\rangle : \theta) = \theta(1-\theta)(1-\theta)\theta\theta = \theta^3(1-\theta)^2.$$

As expected, this probability depends on the particular value θ. As we consider different values of θ, we get different probabilities for the sequence. Thus, we can examine how the probability of the data changes as a *function* of θ. We thus define the *likelihood function* to be

likelihood function

$$L(\theta : \langle H,T,T,H,H\rangle) = P(\langle H,T,T,H,H\rangle : \theta) = \theta^3(1-\theta)^2.$$

Figure 17.2 plots the likelihood function in our example.

Clearly, parameter values with higher likelihood are more likely to generate the observed sequences. Thus, we can use the likelihood function as our measure of quality for different parameter values and select the parameter value that maximizes the likelihood; this value is called the *maximum likelihood estimator (MLE)*. By viewing figure 17.2 we see that $\hat{\theta} = 0.6 = 3/5$ maximizes the likelihood for the sequence H,T,T,H,H.

maximum likelihood estimator

Can we find the MLE for the general case? Assume that our data set $\mathcal{D}$ of observations contains $M[1]$ heads and $M[0]$ tails. We want to find the value $\hat{\theta}$ that maximizes the likelihood of θ relative to $\mathcal{D}$. The likelihood function in this case is:

$$L(\theta : \mathcal{D}) = \theta^{M[1]}(1-\theta)^{M[0]}.$$

It turns out that it is easier to maximize the logarithm of the likelihood function. In our case, the *log-likelihood* function is:

log-likelihood

$$\ell(\theta : \mathcal{D}) = M[1]\log\theta + M[0]\log(1-\theta).$$

Note that the log-likelihood is monotonically related to the likelihood. Therefore, maximizing the one is equivalent to maximizing the other. However, the log-likelihood is more convenient to work with, since products are converted to summations.

Differentiating the log-likelihood, setting the derivative to 0, and solving for θ, we get that the maximum likelihood parameter, which we denote $\hat{\theta}$, is

$$\hat{\theta} = \frac{M[1]}{M[1]+M[0]}, \tag{17.1}$$

as expected (see exercise 17.1).

As we will see, the maximum likelihood approach has many advantages. However, the approach also has some limitations. For example, if we get 3 heads out of 10 tosses, the MLE estimate is 0.3. We get the same estimate if we get 300 heads out of 1,000 tosses. Clearly, the two experiments are not equivalent. Our intuition is that, in the second experiment, we should be more confident of our estimate. Indeed, statistical estimation theory deals with *confidence intervals*. These are common in news reports, for example, when describing the results of election polls, where we often hear that "61 $\pm$ 2 percent" plan to vote for a certain candidate. The 2 percent is a confidence interval — the poll is designed to select enough people so that the MLE estimate will be within 0.02 of the true parameter, with high probability.

confidence interval

17.1.2 The Maximum Likelihood Principle

We now generalize the discussion of maximum likelihood estimation to a broader range of learning problems. We then consider how to apply it to the task of learning the parameters of a Bayesian network.

We start by describing the setting of the learning problem. Assume that we observe several IID samples of a set of random variables $\mathcal{X}$ from an unknown distribution $P^*(\mathcal{X})$. We assume we know in advance the sample space we are dealing with (that is, which random variables, and what values they can take). However, we do not make any additional assumptions about P^*.

training set

We denote the *training set* of samples as $\mathcal{D}$ and assume that it consists of M instances of $\mathcal{X}$: $\xi[1], \ldots \xi[M]$.

Next, we need to consider what exactly we want to learn. We assume that we are given a

parametric model

parametric model for which we wish to estimate *parameters*. Formally, a *parametric model* (also known as a parametric family; see section 8.2) is defined by a function $P(\xi : \boldsymbol{\theta})$, specified in

parameters

terms of a set of *parameters*. Given a particular set of parameter values $\boldsymbol{\theta}$ and an instance ξ of $\mathcal{X}$, the model assigns a probability (or density) to ξ. Of course, we require that for each choice of parameters $\boldsymbol{\theta}$, $P(\xi : \boldsymbol{\theta})$ is a legal distribution; that is, it is nonnegative and

$$\sum_{\xi} P(\xi : \boldsymbol{\theta}) = 1.$$

In general, for each model, not all parameter values are legal. Thus, we need to define the

parameter space

parameter space Θ, which is the set of allowable parameters.

To get some intuition, we consider concrete examples. The model we examined in section 17.1.1 has parameter space $\Theta_{thumbtack} = [0, 1]$ and is defined as

$$P_{thumbtack}(x : \theta) = \begin{cases} \theta & \text{if } x = H \\ 1 - \theta & \text{if } x = T. \end{cases}$$

There are many additional examples.

Example 17.1

multinomial

Suppose that X is a multinomial *variable that can take values $x^1, \ldots, x^K$. The simplest representation of a multinomial distribution is as a vector $\boldsymbol{\theta} \in \mathbb{R}^K$, such that*

$$P_{multinomial}(x : \boldsymbol{\theta}) = \theta_k \text{ if } x = x^k.$$

The parameter space of this model is

$$\Theta_{multinomial} = \left\{ \boldsymbol{\theta} \in [0, 1]^K : \sum_i \theta_i = 1 \right\}.$$

■

Example 17.2

Gaussian

Suppose that X is a continuous variable that can take values in the real line. A Gaussian *model for X is*

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

where $\boldsymbol{\theta} = \langle \mu, \sigma \rangle$. The parameter space for this model is $\Theta_{Gaussian} = \mathbb{R} \times \mathbb{R}^+$. That is, we allow any real value of μ and any positive real value for σ. ■

likelihood function

The next step in maximum likelihood estimation is defining the *likelihood function.* As we saw in our example, the likelihood function for a given choice of parameters $\boldsymbol{\theta}$ is the probability (or density) the model assigns the training data:

$$L(\boldsymbol{\theta} : \mathcal{D}) = \prod_m P(\xi[m] : \boldsymbol{\theta}).$$

In the thumbtack example, we saw that we can write the likelihood function using simpler terms. That is, using the counts $M[1]$ and $M[0]$, we managed to have a compact description of the likelihood. More precisely, once we knew the values of $M[1]$ and $M[0]$, we did not need to consider other aspects of training data (for example, the order of tosses). These are the *sufficient statistics* for the thumbtack learning problem. In a more general setting, a sufficient statistic is a function of the data that summarizes the relevant information for computing the likelihood.

Definition 17.1

sufficient statistics

A function $\tau(\xi)$ from instances of $\mathcal{X}$ to $\mathbb{R}^\ell$ (for some ℓ) is a sufficient statistic *if, for any two data sets $\mathcal{D}$ and $\mathcal{D}'$ and any $\boldsymbol{\theta} \in \Theta$, we have that*

$$\sum_{\xi[m] \in \mathcal{D}} \tau(\xi[m]) = \sum_{\xi'[m] \in \mathcal{D}'} \tau(\xi'[m]) \implies L(\boldsymbol{\theta} : \mathcal{D}) = L(\boldsymbol{\theta} : \mathcal{D}').$$

■

We often refer to the tuple $\sum_{\xi[m] \in \mathcal{D}} \tau(\xi[m])$ as the sufficient statistics of the data set $\mathcal{D}$.

Example 17.3

Let us reconsider the multinomial model of example 17.1. It is easy to see that a sufficient statistic for the data set is the tuple of counts *$\langle M[1], \ldots, M[K] \rangle$, such that $M[k]$ is number of times the value x^k appears in the training data. To obtain these counts by summing instance-level statistics, we define $\tau(x)$ to be a tuple of dimension K, such that $\tau(x)$ has a 0 in every position, except at the position k for which $x = x^k$, where its value is 1:*

$$\tau(x^k) = (\overbrace{0, \ldots, 0}^{k-1}, 1, \overbrace{0, \ldots, 0}^{n-k}).$$

Given the vector of counts, we can write the likelihood function as

$$L(\mathcal{D} : \boldsymbol{\theta}) = \prod_k \theta_k^{M[k]}.$$

■

Example 17.4

Let us reconsider the Gaussian model of example 17.2. In this case, it is less obvious how to construct sufficient statistics. However, if we expand the term $(x - \mu)^2$ in the exponent, we can rewrite the model as

$$P_{Gaussian}(x : \mu, \sigma) = e^{-x^2 \frac{1}{2\sigma^2} + x \frac{\mu}{\sigma^2} - \frac{\mu^2}{2\sigma^2} - \frac{1}{2}\log(2\pi) - \log(\sigma)}.$$

We then see that the function

$$s_{Gaussian}(x) = \langle 1, x, x^2 \rangle$$

is a sufficient statistic for this model. Note that the first element in the sufficient statistics tuple is "1," which does not depend on the value of the data item; it serves, as in the multinomial case, to count the number of data items. ■

We venture several comments about the likelihood function. First, we stress that the likelihood function measures the effect of the choice of parameters on the training data. Thus, for example, if we have two sets of parameters θ and θ', so that $L(\theta : \mathcal{D}) = L(\theta' : \mathcal{D})$, then we *cannot*, given only the data, distinguish between the two choices of parameters. Moreover, if $L(\theta : \mathcal{D}) = L(\theta' : \mathcal{D})$ for all possible choices of $\mathcal{D}$, then the two parameters are *indistinguishable* for any outcome. In such a situation, we can say in advance (that is, before seeing the data) that some distinctions cannot be resolved based on the data alone.

Second, since we are maximizing the likelihood function, we usually want it to be continuous (and preferably smooth) function of θ. To ensure these properties, most of the theory of statistical estimation requires that $P(\xi : \theta)$ is a continuous and differentiable function of θ, and moreover that Θ is a continuous set of points (which is often assumed to be convex).

maximum likelihood estimation

Once we have defined the likelihood function, we can use *maximum likelihood estimation* to choose the parameter values. Formally, we state this principle as follows.

Maximum Likelihood Estimation: Given a data set $\mathcal{D}$, choose parameters $\hat{\theta}$ that satisfy

$$L(\hat{\theta} : \mathcal{D}) = \max_{\theta \in \Theta} L(\theta : \mathcal{D}).$$

Example 17.5 *Consider estimating the parameters of the multinomial distribution of example 17.3. As one might guess, the maximum likelihood is attained when*

$$\hat{\theta}_k = \frac{M[k]}{M}$$

(see exercise 17.2). That is, the probability of each value of X corresponds to its frequency in the training data. ■

Example 17.6 *Consider estimating the parameters of a Gaussian distribution of example 17.4. It turns out that the maximum is attained when μ and σ correspond to the* empirical *mean and variance of the training data:*

empirical mean, variance

$$\hat{\mu} = \frac{1}{M} \sum_m x[m]$$

$$\hat{\sigma} = \sqrt{\frac{1}{M} \sum_m (x[m] - \hat{\mu})^2}$$

(see exercise 17.3). ■

17.2 MLE for Bayesian Networks

We now move to the more general problem of estimating parameters for a Bayesian network. It turns out that the structure of the Bayesian network allows us to reduce the parameter estimation problem to a set of unrelated problems, each of which can be addressed using the techniques of the previous section. We begin by considering a simple example to clarify our intuition, and then generalize to more complicated networks.

17.2.1 A Simple Example

The simplest example of a nontrivial network structure is a network consisting of two binary variables, say X and Y, with an arc $X \rightarrow Y$. (A network without such an arc trivially reduces to the cases we already discussed.)

As for a single parameter, our goal in maximum likelihood estimation is to maximize the likelihood (or log-likelihood) function. In this case, our network is parameterized by a parameter vector $\boldsymbol{\theta}$, which defines the set of parameters for all the CPDs in the network. In this example, our parameterization would consist of the following parameters: θ_{x^1}, and θ_{x^0} specify the probability of the two values of X; $\theta_{y^1|x^1}$, and $\theta_{y^0|x^1}$ specify the probability of Y given that $X = x^1$; and $\theta_{y^1|x^0}$, and $\theta_{y^0|x^0}$ describe the probability of Y given that $X = x^0$. For brevity, we also use the shorthand $\boldsymbol{\theta}_{Y|x^0}$ to refer to the set $\{\theta_{y^1|x^0}, \theta_{y^0|x^0}\}$, and $\boldsymbol{\theta}_{Y|X}$ to refer to $\boldsymbol{\theta}_{Y|x^1} \cup \boldsymbol{\theta}_{Y|x^0}$.

In this example, each training instance is a tuple $\langle x[m], y[m] \rangle$ that describes a particular assignment to X and Y. Our likelihood function is:

$$L(\boldsymbol{\theta} : \mathcal{D}) = \prod_{m=1}^{M} P(x[m], y[m] : \boldsymbol{\theta}).$$

Our network model specifies that $P(X, Y : \boldsymbol{\theta})$ has a product form. Thus, we can write

$$L(\boldsymbol{\theta} : \mathcal{D}) = \prod_m P(x[m] : \boldsymbol{\theta}) P(y[m] \mid x[m] : \boldsymbol{\theta}).$$

Exchanging the order of multiplication, we can equivalently write this term as

$$L(\boldsymbol{\theta} : \mathcal{D}) = \left(\prod_m P(x[m] : \boldsymbol{\theta}) \right) \left(\prod_m P(y[m] \mid x[m] : \boldsymbol{\theta}) \right).$$

That is, the likelihood decomposes into two separate terms, one for each variable. Moreover, each of these terms is a *local likelihood* function that measures how well the variable is predicted given its parents.

Now consider the two individual terms. Clearly, each one depends only on the parameters for that variable's CPD. Thus, the first is $\prod_m P(x[m] : \boldsymbol{\theta}_X)$. This term is identical to the multinomial likelihood function we discussed earlier. The second term is more interesting, since we can decompose it even further:

$$\begin{aligned}
&\prod_m P(y[m] \mid x[m] : \boldsymbol{\theta}_{Y|X}) \\
&\quad = \prod_{m:x[m]=x^0} P(y[m] \mid x[m] : \boldsymbol{\theta}_{Y|X}) \cdot \prod_{m:x[m]=x^1} P(y[m] \mid x[m] : \boldsymbol{\theta}_{Y|X}) \\
&\quad = \prod_{m:x[m]=x^0} P(y[m] \mid x[m] : \boldsymbol{\theta}_{Y|x^0}) \cdot \prod_{m:x[m]=x^1} P(y[m] \mid x[m] : \boldsymbol{\theta}_{Y|x^1}).
\end{aligned}$$

Thus, in this example, the likelihood function decomposes into a product of terms, one for each group of parameters in $\boldsymbol{\theta}$. This property is called the *decomposability* of the likelihood function.

likelihood decomposability

We can do one more simplification by using the notion of sufficient statistics. Let us consider one term in this expression:

$$\prod_{m:x[m]=x^0} P(y[m] \mid x[m] : \boldsymbol{\theta}_{Y|x^0}). \tag{17.2}$$

Each of the individual terms $P(y[m] \mid x[m] : \boldsymbol{\theta}_{Y|x^0})$ can take one of two values, depending on the value of $y[m]$. If $y[m] = y^1$, it is equal to $\theta_{y^1|x^0}$. If $y[m] = y^0$, it is equal to $\theta_{y^0|x^0}$. How many cases of each type do we get? First, we restrict attention only to those data cases where $x[m] = x^0$. These, in turn, partition into the two categories. Thus, we get $\theta_{y^1|x^0}$ in those data cases where $x[m] = x^0$ and $y[m] = y^1$; we use $M[x^0, y^1]$ to denote their number. We get $\theta_{y^0|x^0}$ in those data cases where $x[m] = x^0$ and $y[m] = x^0$, and use $M[x^0, y^0]$ to denote their number. Thus, the term in equation (17.2) is equal to:

$$\prod_{m:x[m]=x^0} P(y[m] \mid x[m] : \boldsymbol{\theta}_{Y|x^0}) \quad = \quad \theta_{y^1|x^0}^{M[x^0,y^1]} \cdot \theta_{y^0|x^0}^{M[x^0,y^0]}.$$

Based on our discussion of the multinomial likelihood in example 17.5, we know that we maximize $\theta_{Y|x^0}$ by setting:

$$\theta_{y^1|x^0} = \frac{M[x^0, y^1]}{M[x^0, y^1] + M[x^0, y^0]} = \frac{M[x^0, y^1]}{M[x^0]},$$

and similarly for $\theta_{y^0|x^0}$. Thus, we can find the maximum likelihood parameters in this CPD by simply counting how many times each of the possible assignments of X and Y appears in the training data. It turns out that these counts of the various assignments for some set of variables are useful in general. We therefore define:

Definition 17.2

Let $\boldsymbol{Z}$ be some set of random variables, and $\boldsymbol{z}$ be some instantiation to these random variables. Let $\mathcal{D}$ be a data set. We define $M[\boldsymbol{z}]$ to be the number of entries in $\mathcal{D}$ that have $\boldsymbol{Z}[m] = \boldsymbol{z}$

$$M[\boldsymbol{z}] = \sum_m \boldsymbol{1}\{\boldsymbol{Z}[m] = \boldsymbol{z}\} \tag{17.3}$$

■

17.2.2 Global Likelihood Decomposition

As we can expect, the arguments we used for deriving the MLE of $\theta_{Y|x^0}$ apply for the parameters of other CPDs in that example and indeed for other networks as well. We now develop, in several steps, the formal machinery for proving such properties in Bayesian networks.

We start by examining the likelihood function of a Bayesian network. Suppose we want to learn the parameters for a Bayesian network with structure $\mathcal{G}$ and parameters $\boldsymbol{\theta}$. This means that we agree in advance on the type of CPDs we want to learn (say table-CPDs, or noisy-ors). As we discussed, we are also given a data set $\mathcal{D}$ consisting of samples $\xi[1], \ldots, \xi[M]$. Writing

the likelihood, and repeating the steps we performed in our example, we get

$$\begin{aligned} L(\theta : \mathcal{D}) &= \prod_m P_{\mathcal{G}}(\xi[m] : \theta) \\ &= \prod_m \prod_i P(x_i[m] \mid \mathrm{pa}_{X_i}[m] : \theta) \\ &= \prod_i \left[\prod_m P(x_i[m] \mid \mathrm{pa}_{X_i}[m] : \theta) \right]. \end{aligned}$$

Note that each of the terms in the square brackets refers to the *conditional likelihood* of a particular variable given its parents in the network. We use $\theta_{X_i|\mathrm{Pa}_{X_i}}$ to denote the subset of parameters that determines $P(X_i \mid \mathrm{Pa}_{X_i})$ in our model. Then, we can write

$$L(\theta : \mathcal{D}) = \prod_i L_i(\theta_{X_i|\mathrm{Pa}_{X_i}} : \mathcal{D}),$$

local likelihood

where the *local likelihood* function for X_i is:

$$L_i(\theta_{X_i|\mathrm{Pa}_{X_i}} : \mathcal{D}) = \prod_m P(x_i[m] \mid \mathrm{pa}_{X_i}[m] : \theta_{X_i|\mathrm{Pa}_{X_i}}).$$

This form is particularly useful when the parameter sets $\theta_{X_i|\mathrm{Pa}_{X_i}}$ are *disjoint*. That is, each CPD is parameterized by a separate set of parameters that do not overlap. This assumption is quite natural in all our examples so far. (Although, as we will see in section 17.5, parameter sharing can be handy in many domains.) **This analysis shows that the likelihood decomposes as a product of independent terms, one for each CPD in the network. This important property is called the *global decomposition* of the likelihood function.**

global decomposability

We can now immediately derive the following result:

Proposition 17.1 *Let $\mathcal{D}$ be a complete data set for $X_1, \ldots, X_n$, let $\mathcal{G}$ be a network structure over these variables, and suppose that the parameters $\theta_{X_i|\mathrm{Pa}_{X_i}}$ are disjoint from $\theta_{X_j|\mathrm{Pa}_{X_j}}$ for all $j \neq i$. Let $\hat{\theta}_{X_i|\mathrm{Pa}_{X_i}}$ be the parameters that maximize $L_i(\theta_{X_i|\mathrm{Pa}_{X_i}} : \mathcal{D})$. Then, $\hat{\theta} = \langle \hat{\theta}_{X_1|\mathrm{Pa}_1}, \ldots, \hat{\theta}_{X_n|\mathrm{Pa}_n} \rangle$ maximizes $L(\theta : \mathcal{D})$.*

In other words, **we can maximize each local likelihood function *independently* of rest of the network, and then combine the solutions to get an MLE solution.** This decomposition of the global problem to independent subproblems allows us to devise efficient solutions to the MLE problem. Moreover, this decomposition is an immediate consequence of the network structure and does not depend on any particular choice of parameterization for the CPDs.

17.2.3 Table-CPDs

Based on the preceding discussion, we know that the likelihood of a Bayesian network decomposes into local terms that depend on the parameterization of CPDs. The choice of parameters determines how we can maximize each of the local likelihood functions. We now consider what is perhaps the simplest parameterization of the CPD: a *table-CPD*.

table-CPD

Suppose we have a variable X with parents $\boldsymbol{U}$. If we represent that CPD $P(X \mid \boldsymbol{U})$ as a table, then we will have a parameter $\theta_{x|\boldsymbol{u}}$ for each combination of $x \in Val(X)$ and $\boldsymbol{u} \in Val(\boldsymbol{U})$. In this case, we can rewrite the local likelihood function as follows:

$$\begin{aligned} L_X(\boldsymbol{\theta}_{X|\boldsymbol{U}} : \mathcal{D}) &= \prod_m \theta_{x[m]|\boldsymbol{u}[m]} \\ &= \prod_{\boldsymbol{u} \in Val(\boldsymbol{U})} \left[\prod_{x \in Val(X)} \theta_{x|\boldsymbol{u}}^{M[\boldsymbol{u},x]} \right], \end{aligned} \tag{17.4}$$

where $M[\boldsymbol{u}, x]$ is the number of times $\xi[m] = x$ and $\boldsymbol{u}[m] = \boldsymbol{u}$ in $\mathcal{D}$. That is, we grouped together all the occurrences of $\theta_{x|\boldsymbol{u}}$ in the product over all instances. This provides a further *local decomposition* of the likelihood function.

local decomposability

We need to maximize this term under the constraints that, for each choice of value for the parents $\boldsymbol{U}$, the conditional probability is legal, that is:

$$\sum \theta_{x|\boldsymbol{u}} = 1 \quad \text{for all } \boldsymbol{u}.$$

These constraints imply that the choice of value for $\theta_{x|\boldsymbol{u}}$ can impact the choice of value for $\theta_{x'|\boldsymbol{u}}$. However, the choice of parameters given different values $\boldsymbol{u}$ of $\boldsymbol{U}$ are independent of each other. Thus, we can maximize each of the terms in square brackets in equation (17.4) independently.

We can thus further decompose the local likelihood function for a tabular CPD into a product of simple likelihood functions. Each of these likelihood functions is a *multinomial* likelihood, of the type that we examined in example 17.3. The counts in the data for the different outcomes x are simply $\{M[\boldsymbol{u}, x] : x \in Val(X)\}$. We can then immediately use the maximum likelihood estimation for multinomial likelihood of example 17.5 and see that the MLE parameters are

$$\hat{\theta}_{x|\boldsymbol{u}} = \frac{M[\boldsymbol{u}, x]}{M[\boldsymbol{u}]}, \tag{17.5}$$

where we use the fact that $M[\boldsymbol{u}] = \sum_x M[\boldsymbol{u}, x]$.

This simple formula reveals a key challenge when estimating parameters for a Bayesian networks. Note that the number of data points used to estimate the parameter $\hat{\theta}_{x|\boldsymbol{u}}$ is $M[\boldsymbol{u}]$. Data points that do not agree with the parent assignment $\boldsymbol{u}$ play no role in this computation. As the number of parents $\boldsymbol{U}$ grows, the number of different parent assignments grows exponentially. Therefore, the number of data instances that we expect to have for a single parent assignment shrinks exponentially. This phenomenon is called *data fragmentation*, since the data set is partitioned into a large number of small subsets. Intuitively, when we have a very small number of data instances from which we estimate a parameter, the estimates we get can be very noisy (this intuition is formalized in section 17.6), leading to *overfitting*. We are also more likely to get a large number of zeros in the distribution, which can lead to very poor performance. **Our inability to estimate parameters reliably as the dimensionality of the parent set grows is one of the key limiting factors in learning Bayesian networks from data.** This problem is even more severe when the variables can take on a large number of values, for example, in text applications.

data fragmentation

overfitting

classification

Box 17.A — Concept: Naive Bayes Classifier. *One of the basic tasks of learning is* classification. *In this task, our goal is build a classifier — a procedure that assigns instances into two or more categories, for example, deciding whether an email message is junk mail that should be discarded or a relevant message that should be presented to the user. In the usual setting, we are given a training example of instances from each category, where instances are represented by various features. In our email classification example, a message might be analyzed by multiple features: its length, the type of attachments it contains, the domain of the sender, whether that sender appears in the user's address book, whether a particular word appears in the subject, and so on.*

Bayesian classifier

One general approach to this problem, which is referred to as Bayesian classifier, *is to learn a probability distribution of the features of instances of each class. In the language of probabilistic models, we use the random variables $\boldsymbol{X}$ to represent the instance, and the random variable C to represent the category of the instance. The distribution $P(\boldsymbol{X} \mid C)$ is the probability of a particular combination of features given the category. Using Bayes rule, we have that*

$$P(C \mid \boldsymbol{X}) \propto P(C)P(\boldsymbol{X} \mid C).$$

Thus, if we have a good model of how instances of each category behave (that is, of $P(\boldsymbol{X} \mid C)$), we can combine it with our prior estimate for the frequency of each category (that is, $P(C)$) to estimate the posterior probability of each of the categories (that is, $P(C \mid \boldsymbol{X})$). We can then decide either to predict the most likely category or to perform a more complex decision based on the strength of likelihood of each option. For example, to reduce the number of erroneously removed messages, a junk-mail filter might remove email messages only when the probability that it is junk mail is higher than a strict threshold.

This Bayesian classification approach is quite intuitive. Loosely speaking, it states that to classify objects successfully, we need to recognize the characteristics of objects of each category. Then, we can classify a new object by considering whether it matches the characteristic of each of the classes. More formally, we use the language of probability to describe each category, assigning higher probability to objects that are typical for the category and low probability to ones that are not.

naive Bayes

The main hurdle in constructing a Bayesian classifier is the question of representation of the multivariate distribution $p(\boldsymbol{X} \mid C)$. The naive Bayes *classifier is one where we use the simplest representation we can think of. That is, we assume that each feature X_i is independent of all the other features given the class variable C. That is,*

$$P(\boldsymbol{X} \mid C) = \prod_i P(X_i \mid C).$$

Learning the distribution $P(C)P(\boldsymbol{X} \mid C)$ is thus reduced to learning the parameters in the naive Bayes structure, with the category variable C rendering all other features as conditionally independent of each other.

As can be expected, learning this classifier is a straightforward application of the parameter estimation that we consider in this chapter. Moreover, classifying new examples requires simple computation, evaluating $P(c) \prod_i P(x_i \mid c)$ for each category c.

Although this simple classifier is often dismissed as naive, in practice it is often surprisingly effective. From a training perspective, this classifier is quite robust, since in most applications, even with relatively few training examples, we can learn the parameters of conditional distribution

$P(X_i \mid C)$. *However, one might argue that robust learning does not compensate for oversimplified independence assumption. Indeed, the strong independence assumption usually results in poor representation of the distribution of instances. However,* **errors in estimating the probability of an instance do not necessarily lead to classification errors. For classification, we are interested in the relative size of the conditional distribution of the instances given different categories. The ranking of different labels may not be that sensitive to errors in estimating the actual probability of the instance.** *Empirically, one often finds that the naive Bayes classifier correctly classifies an example to the right category, yet its posterior probability is very skewed and quite far from the correct distribution.*

In practice, the naive Bayes classifier is often a good baseline classifier to try before considering more complex solutions. It is easy to implement, it is robust, and it can handle different choices of descriptions of instances (for example, box 17.E).

17.2.4 Gaussian Bayesian Networks ⋆

Our discussion until now has focused on learning discrete-state Bayesian networks with multinomial parameters. However, the concepts we have developed in this section carry through to a wide variety of other types of Bayesian networks. In particular, the global decomposition properties we proved for a Bayesian network apply, without any change, to any other type of CPD. That is, if the data are complete, the learning problem reduces to a set of local learning problems, one for each variable. The main difference is in applying the maximum likelihood estimation process to a CPD of a different type: how we define the sufficient statistics, and how we compute the maximum likelihood estimate from them. In this section, we demonstrate how MLE principles can be applied in the setting of linear Gaussian Bayesian networks. In section 17.2.5 we provide a general procedure for CPDs in the exponential family.

Consider a variable X with parents $\boldsymbol{U} = \{U_1, \ldots, U_k\}$ with a linear Gaussian CPD:

$$P(X \mid \boldsymbol{u}) = \mathcal{N}\left(\beta_0 + \beta_1 u_1 + \ldots, \beta_k u_k; \sigma^2\right).$$

Our task is to learn the parameters $\boldsymbol{\theta}_{X|\boldsymbol{U}} = \langle \beta_0, \ldots, \beta_k, \sigma \rangle$. To find the MLE values of these parameters, we need to differentiate the likelihood and solve the equations that define a stationary point. As usual, it will be easier to work with the log-likelihood function. Using the definition of the Gaussian distribution, we have that

$$\begin{aligned}\ell_X(\boldsymbol{\theta}_{X|\boldsymbol{U}} : \mathcal{D}) &= \log L_X(\boldsymbol{\theta}_{X|\boldsymbol{U}} : \mathcal{D}) \\ &= \sum_m \left[-\frac{1}{2}\log(2\pi\sigma^2) - \frac{1}{2}\frac{1}{\sigma^2}\left(\beta_0 + \beta_1 u_1[m] + \ldots + \beta_k u_k[m] - x[m]\right)^2 \right].\end{aligned}$$

We start by considering the gradient of the log-likelihood with respect to β_0:

$$\begin{aligned}\frac{\partial}{\partial \beta_0}\ell_X(\boldsymbol{\theta}_{X|\boldsymbol{U}} : \mathcal{D}) &= \sum_m -\frac{1}{\sigma^2}\left(\beta_0 + \beta_1 u_1[m] + \ldots + \beta_k u_k[m] - x[m]\right) \\ &= -\frac{1}{\sigma^2}\left(\beta_0 + \beta_1 \sum_m u_1[m] + \ldots + \beta_k \sum_m u_k[m] - \sum_m x[m]\right).\end{aligned}$$

Equating this gradient to 0, and multiplying both sides with $\frac{\sigma^2}{M}$, we get the equation

$$\frac{1}{M}\sum_m x[m] = \beta_0 + \beta_1 \frac{1}{M}\sum_m u_1[m] + \ldots + \beta_k \frac{1}{M}\sum_m u_k[m].$$

Each of the terms is the average value of one of the variables in the data. We use the notation

$$\boldsymbol{E}_{\mathcal{D}}[X] = \frac{1}{M}\sum_m x[m]$$

to denote this expectation. Using this notation, we see that we get the following equation:

$$\boldsymbol{E}_{\mathcal{D}}[X] = \beta_0 + \beta_1 \boldsymbol{E}_{\mathcal{D}}[U_1] + \ldots + \beta_k \boldsymbol{E}_{\mathcal{D}}[U_k]. \tag{17.6}$$

Recall that theorem 7.3 specifies the mean of a linear Gaussian variable X in terms of the means of its parents $U_1, \ldots, U_k$, using an expression that has precisely this form. Thus, equation (17.6) tells us that the MLE parameters should be such that the mean of X in the data is consistent with the predicted mean of X according to the parameters.

Next, consider the gradient with respect to one of the parameters β_i. Using similar arithmetic manipulations, we see that the equation $0 = \frac{\partial}{\partial \beta_i} \ell_X(\boldsymbol{\theta}_{X|U} : \mathcal{D})$ can be formulated as:

$$\boldsymbol{E}_{\mathcal{D}}[X \cdot U_i] = \beta_0 \boldsymbol{E}_{\mathcal{D}}[U_i] + \beta_1 \boldsymbol{E}_{\mathcal{D}}[U_1 \cdot U_i] + \ldots + \beta_k \boldsymbol{E}_{\mathcal{D}}[U_k \cdot U_i]. \tag{17.7}$$

At this stage, we have $k+1$ linear equations with $k+1$ unknowns, and we can use standard linear algebra techniques for solving for the value of $\beta_0, \beta_1, \ldots, \beta_k$. We can get additional intuition, however, by doing additional manipulation of equation (17.7). Recall that the covariance $\boldsymbol{Cov}[X;Y] = \boldsymbol{E}[X \cdot Y] - \boldsymbol{E}[X] \cdot \boldsymbol{E}[Y]$. Thus, if we subtract $\boldsymbol{E}_{\mathcal{D}}[X] \cdot \boldsymbol{E}_{\mathcal{D}}[U_i]$ from the left-hand side of equation (17.7), we would get the empirical covariance of X and U_i. Using equation (17.6), we have that this term can also be written as:

$$\boldsymbol{E}_{\mathcal{D}}[X] \cdot \boldsymbol{E}_{\mathcal{D}}[U_i] = \beta_0 \boldsymbol{E}_{\mathcal{D}}[U_i] + \beta_1 \boldsymbol{E}_{\mathcal{D}}[U_1] \cdot \boldsymbol{E}_{\mathcal{D}}[U_i] + \ldots + \beta_k \boldsymbol{E}_{\mathcal{D}}[U_k] \cdot \boldsymbol{E}_{\mathcal{D}}[U_i].$$

Subtracting this equation from equation (17.7), we get:

$$\begin{aligned}\boldsymbol{E}_{\mathcal{D}}[X \cdot U_i] - \boldsymbol{E}_{\mathcal{D}}[X] \cdot \boldsymbol{E}_{\mathcal{D}}[U_i] \quad = \quad & \beta_1 \left(\boldsymbol{E}_{\mathcal{D}}[U_1 \cdot U_i] - \boldsymbol{E}_{\mathcal{D}}[U_1] \cdot \boldsymbol{E}_{\mathcal{D}}[U_i]\right) + \ldots + \\ & \beta_k \left(\boldsymbol{E}_{\mathcal{D}}[U_k \cdot U_i] - \boldsymbol{E}_{\mathcal{D}}[U_k] \cdot \boldsymbol{E}_{\mathcal{D}}[U_i]\right).\end{aligned}$$

Using $\boldsymbol{Cov}_{\mathcal{D}}[X;U_i]$ to denote the observed covariance of X and U_i in the data, we get:

$$\boldsymbol{Cov}_{\mathcal{D}}[X;U_i] = \beta_1 \boldsymbol{Cov}_{\mathcal{D}}[U_1;U_i] + \ldots + \beta_k \boldsymbol{Cov}_{\mathcal{D}}[U_k;U_i].$$

In other words, the observed covariance of X with U_i should be the one predicted by theorem 7.3 given the parameters and the observed covariances between the parents of X.

Finally, we need to find the value of the σ^2 parameter. Taking the derivative of the likelihood and equating to 0, we get an equation that, after suitable reformulation, can be written as

$$\sigma^2 = \boldsymbol{Cov}_{\mathcal{D}}[X;X] - \sum_i \sum_j \beta_i \beta_j \boldsymbol{Cov}_{\mathcal{D}}[U_i;U_j] \tag{17.8}$$

(see exercise 17.4). Again, we see that the MLE estimate has to match the constraints implied by theorem 7.3.

The global picture that emerges is as follows. To estimate $P(X \mid \boldsymbol{U})$, we estimate the means of X and $\boldsymbol{U}$ and covariance matrix of $\{X\} \cup \boldsymbol{U}$ from the data. The vector of means and covariance matrix defines a joint Gaussian distribution over $\{X\} \cup \boldsymbol{U}$. (In fact, this is the MLE estimate of the joint Gaussian; see exercise 17.5.) We then solve for the (unique) linear Gaussian that matches the joint Gaussian with these parameters. For this purpose, we can use the formulas provided by theorem 7.4. While these equations seem somewhat complex, they are merely describing the solution to a system of linear equations.

This discussion also identifies the sufficient statistics we need to collect to estimate linear Gaussians. These are the univariate terms of the form $\sum_m x[m]$ and $\sum_m u_i[m]$, and the interaction terms of the form $\sum_m x[m] \cdot u_i[m]$ and $\sum_m u_i[m] \cdot u_j[m]$. From these, we can estimate the mean and covariance matrix of the joint distribution.

Box 17.B — Concept: Nonparametric Models. *The discussion in this chapter has focused on estimating parameters for specific parametric models of CPDs: multinomials and linear Gaussians. However, a theory of maximum likelihood and Bayesian estimation exists for a wide variety of other parametric models. Moreover, in recent years, there has been a growing interest in the use of* nonparametric Bayesian estimation *methods, where a (conditional) distribution is not defined to be in some particular parametric class with a fixed number of parameters, but rather the complexity of the representation is allowed to grow as we get more data instances. In the case of discrete variables, any CPD can be described as a table, albeit perhaps a very large one; thus a nonparametric method is less essential (although see section 19.5.2.2 for a very useful example of a nonparametric method in the discrete case). In the case of continuous variables, we do not have a "universal" parametric distribution. While Gaussians are often the default, many distributions are not well fit by them, and it is often difficult to determine which parametric family (if any) will be appropriate for a given variable. In such cases, nonparametric methods offer a useful substitute. In such methods, we use the data points themselves as the basis for a probability distribution. Many nonparametric methods have been developed; we describe one simple variant that serves to illustrate this type of approach.*

nonparametric Bayesian estimation

Suppose we want to learn the distribution $P(X \mid \boldsymbol{U})$ from data. A reasonable assumption is that the CPD is smooth. Thus, if we observe $x, \boldsymbol{u}$ in a training sample, it should increase the probability of seeing similar values of X for similar values of $\boldsymbol{U}$. More precisely, we increase the density of $p(X = x + \epsilon \mid \boldsymbol{U} = \boldsymbol{u} + \delta)$ for small values of ϵ and δ.

kernel density estimation

One simple approach that captures this intuition is the use of kernel density estimation *(also known as* Parzen windows*). The idea is fairly simple: given the data $\mathcal{D}$, we estimate a "local" joint density $\tilde{p}_X(X, \boldsymbol{U})$ by spreading out density around each example $x[m], \boldsymbol{u}[m]$. Formally, we write*

$$\tilde{p}_X(x, \boldsymbol{u}) = \frac{1}{M} \sum_m K(x, \boldsymbol{u}; x[m], \boldsymbol{u}[m], \alpha),$$

where K is a kernel density function *and α is a parameter (or vector of parameters) controlling K. A common choice of kernel is a simple round Gaussian distribution with radius α around $x[m], \boldsymbol{u}[m]$:*

$$K(x, \boldsymbol{u}; x[m], \boldsymbol{u}[m], \alpha) = \mathcal{N}\left(\begin{pmatrix} x[m] \\ \boldsymbol{u}[m] \end{pmatrix}; \alpha^2 I\right),$$

where I is the identity matrix and α is the width of the window. Of course, many other choices for kernel function are possible; in fact, if K defines a probability measure (nonnegative and integrates to 1), then $\tilde{p}_X(x, \boldsymbol{u})$ is also a probability measure. Usually we choose kernel functions that are local, in that they put most of the mass in the vicinity of their argument. For such kernels, the resulting density $\tilde{p}_X(x, \boldsymbol{u})$ will have high mass in regions where we have seen many data instances $(x[m], \boldsymbol{u}[m])$ and low mass in regions where we have seen none.

Once we have estimated this local joint distribution, we can then reformulate it to produce a conditional distribution:

$$p(x \mid \boldsymbol{u}) = \frac{\sum_m K(x, \boldsymbol{u}; x[m], \boldsymbol{u}[m], \alpha)}{\sum_m K(\boldsymbol{u}; \boldsymbol{u}[m], \alpha)}.$$

Note that this learning procedure estimates virtually no parameters: the CPD is derived directly from the training instances. The only free parameter is α, which is the width of the window. Importantly, this parameter cannot be estimated using maximum likelihood: The α that maximizes the likelihood of the training set is $\alpha = 0$, which gives maximum density to the training instances themselves. This, of course, will simply memorize the training instances without any generalization. Thus, this parameter is generally selected using cross-validation.

The learned CPD here is essentially the list of training instances, which has both advantages and disadvantages. On the positive side, the estimates are very flexible and tailor themselves to the observations; indeed, as we get more training data, we can produce arbitrarily expressive representations of our joint density. On the negative side, there is no "compression" of the original data, which has both computational and statistical ramifications. Computationally, when there are many training samples the learned CPDs can become unwieldy. Statistically, this learning procedure makes no attempt to generalize beyond the data instances that we have seen. In high-dimensional spaces with limited data, most points in the space will be "far" from data instances, and therefore the estimated density will tend to be quite poor in most parts of the space. Thus, this approach is primarily useful in cases where we have a large number of training instances relative to the dimension of the space.

Finally, while these approaches help us avoid parametric assumptions on the learning side, we are left with the question of how to avoid them on the inference side. As we saw, most inference procedures are geared to working with parametric representations, mostly Gaussians. Thus, when performing inference with nonparametric CPDs, we must generally either use parametric approximations, or resort to sampling.

17.2.5 Maximum Likelihood Estimation as M-Projection ⋆

The MLE principle is a general one, in that it gives a recipe how to construct estimators for different statistical models (for example, multinomials and Gaussians). As we have seen, for simple examples the resulting estimators are quite intuitive. However, the same principle can be applied in a much broader range of parametric models. Indeed, as we now show, we have already discussed the framework that forms the basis for this generalization.

In section 8.5, we defined the notion of *projection*: finding the distribution, within a specified class, that is closest to a given target distribution. Parameter estimation is similar in the sense

that we select a distribution from a given class — all of those that can be described by the model — that is "closest" to our data. Indeed, we can show that maximum likelihood estimation aims to find the distribution that is "closest" to the empirical distribution $\hat{P}_{\mathcal{D}}$ (see equation (16.4)).

We start by rewriting the likelihood function in terms of the empirical distribution.

Proposition 17.2 *Let $\mathcal{D}$ be a data set, then*

$$\log L(\boldsymbol{\theta} : \mathcal{D}) = M \cdot \boldsymbol{E}_{\hat{P}_{\mathcal{D}}}[\log P(\mathcal{X} : \boldsymbol{\theta})].$$

PROOF We rewrite the likelihood by combining all identical instances in our training set and then writing the likelihood in terms of the empirical probability of each entry in our joint distribution:

$$\begin{aligned}
\log L(\boldsymbol{\theta} : \mathcal{D}) &= \sum_{m} \log P(\xi[m] : \boldsymbol{\theta}) \\
&= \sum_{\xi} \left[\sum m \boldsymbol{I}\{\xi[m] = \xi\}\right] \log P(\xi : \boldsymbol{\theta}) \\
&= \sum_{\xi} M \cdot \hat{P}_{\mathcal{D}}(\xi) \log P(\xi : \boldsymbol{\theta}) \\
&= M \cdot \boldsymbol{E}_{\hat{P}_{\mathcal{D}}}[\log P(\mathcal{X} : \boldsymbol{\theta})].
\end{aligned}$$

■

We can now apply proposition 16.1 to the empirical distribution to conclude that

$$\ell(\boldsymbol{\theta} : \mathcal{D}) = M \left(\boldsymbol{H}_{\hat{P}_{\mathcal{D}}}(\mathcal{X}) - \boldsymbol{D}(\hat{P}_{\mathcal{D}}(\mathcal{X}) \| P(\mathcal{X} \mid \boldsymbol{\theta}))\right). \tag{17.9}$$

From this result, we immediately derive the following relationship between MLE and M-projections.

Theorem 17.1 *The MLE $\hat{\boldsymbol{\theta}}$ in a parametric family relative to a data set $\mathcal{D}$ is the M-projection of $\hat{P}_{\mathcal{D}}$ onto the parametric family*

$$\hat{\boldsymbol{\theta}} = \arg\min_{\boldsymbol{\theta} \in \Theta} \boldsymbol{D}(\hat{P}_{\mathcal{D}} \| P_{\boldsymbol{\theta}}).$$

We see that MLE finds the distribution $P(\mathcal{X} : \boldsymbol{\theta})$ that is the M-projection of $\hat{P}_{\mathcal{D}}$ onto the set of distributions representable in our parametric family.

This result allows us to call upon our detailed analysis of M-projections in order to generalize MLE to other parametric classes in the exponential family. In particular, in section 8.5.2, we discussed the general notion of sufficient statistics and showed that the M-projection of a distribution P into a class of distributions $\mathcal{Q}$ was defined by the parameters $\boldsymbol{\theta}$ such that $\boldsymbol{E}_{Q_{\boldsymbol{\theta}}}[\tau(\mathcal{X})] = \boldsymbol{E}_{P}[\tau(\mathcal{X})]$. In our setting, we seek the parameters $\boldsymbol{\theta}$ whose expected sufficient statistics match those in $\hat{P}_{\mathcal{D}}$, that is, the sufficient statistics in $\mathcal{D}$.

If our CPDs are in an exponential family where the mapping *ess* from parameters to sufficient statistics is invertible, we can simply take the sufficient statistic vector from $\hat{P}_{\mathcal{D}}$, and invert this mapping to produce the MLE. Indeed, this process is precisely the one that gave rise to our MLE for multinomials and for linear Gaussians, as described earlier. However, the same process can be applied to many other classes of distributions in the exponential family.

This analysis provides us with a notion of sufficient statistics $\tau(\mathcal{X})$ and a clearly defined path to deriving MLE parameters for any distribution in the exponential family. Somewhat more surprisingly, it turns out that a parametric family has a sufficient statistic *only if* it is in the exponential family.

17.3 Bayesian Parameter Estimation

17.3.1 The Thumbtack Example Revisited

Although the MLE approach seems plausible, it can be overly simplistic in many cases. Assume again that we perform the thumbtack experiment and get 3 heads out of 10. It may be quite reasonable to conclude that the parameter θ is 0.3. But what if we do the same experiment with a standard coin, and we also get 3 heads? We would be much less likely to jump to the conclusion that the parameter of the coin is 0.3. Why? Because we have a lot more experience with tossing coins, so we have a lot more *prior knowledge* about their behavior. Note that we do not want our prior knowledge to be an absolute guide, but rather a reasonable starting assumption that allows us to counterbalance our current set of 10 tosses, under the assumption that they may not be typical. However, if we observe 1,000,000 tosses of the coin, of which 300,000 came out heads, then we may be more willing to conclude that this is a trick coin, one whose parameter is closer to 0.3.

Maximum likelihood allows us to make neither of these distinctions: between a thumbtack and a coin, and between 10 tosses and 1,000,000 tosses of the coin. There is, however, another approach, the one recommended by Bayesian statistics.

17.3.1.1 Joint Probabilistic Model

In this approach, we encode our prior knowledge about θ with a probability distribution; this distribution represents how likely we are a priori to believe the different choices of parameters. Once we quantify our knowledge (or lack thereof) about possible values of θ, we can create a joint distribution over the parameter θ and the data cases that we are about to observe $X[1], \ldots, X[M]$. This joint distribution is captures our assumptions about the experiment.

Let us reconsider these assumptions. Recall that we assumed that tosses are independent of each other. Note, however, that this assumption was made when θ was fixed. If we do not know θ, then the tosses are not marginally independent. Each toss tells us something about the parameter θ, and thereby about the probability of the next toss. However, once θ is known, we cannot learn about the outcome of one toss from observing the results of others. Thus, we assume that the tosses are *conditionally independent* given θ. We can describe these assumptions using the probabilistic model of figure 17.3.

Having determined the model structure, it remains to specify the local probability models in this network. We begin by considering the probability $P(X[m] \mid \theta)$. Clearly,

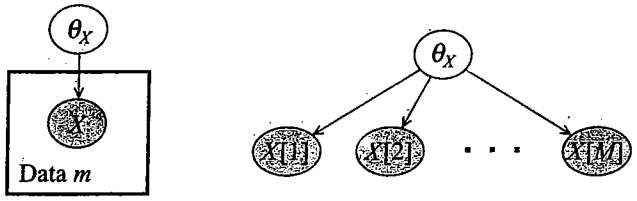
$$P(x[m] \mid \theta) = \begin{cases} \theta & \text{if } x[m] = x^1 \\ 1 - \theta & \text{if } x[m] = x^0. \end{cases}$$

Note that since we now treat θ as a random variable, we use the conditioning bar, instead of $P(x[m] : \theta)$.

prior parameter distribution

To finish the description of the joint distribution, we need to describe $P(\theta)$. This is our *prior distribution* over the value of θ. In our case, this is a continuous density over the interval $[0, 1]$. Before we discuss particular choices for this distribution, let us consider how we use it.

The network structure implies that the joint distribution of a particular data set and θ



(a) (b)

Figure 17.3 Meta-network for IID samples of a random variable X. (a) Plate model; (b) Ground Bayesian network.

factorizes as

$$\begin{aligned} P(x[1],\ldots,x[M],\theta) &= P(x[1],\ldots,x[M] \mid \theta)P(\theta) \\ &= P(\theta)\prod_{m=1}^{M} P(x[m] \mid \theta) \\ &= P(\theta)\theta^{M[1]}(1-\theta)^{M[0]}, \end{aligned}$$

where $M[1]$ is the number of heads in the data, and $M[0]$ is the number of tails. Note that the expression $P(x[1],\ldots,x[M] \mid \theta)$ is simply the likelihood function $L(\theta : \mathcal{D})$.

This network specifies a joint probability model over parameters and data. There are several ways in which we can use this network. Most obviously, we can take an observed data set $\mathcal{D}$ of M outcomes, and use it to instantiate the values of $x[1],\ldots,x[M]$; we can then compute the *posterior distribution* over θ:

posterior parameter distribution

$$P(\theta \mid x[1],\ldots,x[M]) = \frac{P(x[1],\ldots,x[M] \mid \theta)P(\theta)}{P(x[1],\ldots,x[M])}.$$

In this posterior, the first term in the numerator is the likelihood, the second is the prior over parameters, and the denominator is a normalizing factor that we will not expand on right now. We see that the posterior is (proportional to) a product of the likelihood and the prior. This product is normalized so that it will be a proper density function. In fact, if the prior is a uniform distribution (that is, $P(\theta) = 1$ for all $\theta \in [0,1]$), then the posterior is just the normalized likelihood function.

17.3.1.2 Prediction

If we do use a uniform prior, what then is the difference between the Bayesian approach and the MLE approach of the previous section? The main philosophical difference is in the use of the posterior. Instead of selecting from the posterior a single value for the parameter θ, we use it, in its entirety, for *predicting* the probability over the next toss.

To derive this prediction in a principled fashion, we introduce the value of the next coin toss $x[M+1]$ to our network. We can then compute the probability over $x[M+1]$ given the observations of the first M tosses. Note that, in this model, the parameter θ is unknown, and

we are considering all of its possible values. By reasoning over the possible values of θ and using the chain rule, we see that

$$\begin{aligned} &P(x[M+1] \mid x[1], \ldots, x[M]) = \\ &\quad = \int P(x[M+1] \mid \theta, x[1], \ldots, x[M]) P(\theta \mid x[1], \ldots, x[M]) d\theta \\ &\quad = \int P(x[M+1] \mid \theta) P(\theta \mid x[1], \ldots, x[M]) d\theta, \end{aligned}$$

where we use the conditional independencies implied by the meta-network to rewrite $P(x[M+1] \mid \theta, x[1], \ldots, x[M])$ as $P(x[M+1] \mid \theta)$. In other words, we are *integrating* our posterior over θ to predict the probability of heads for the next toss.

Let us go back to our thumbtack example. Assume that our prior is uniform over θ in the interval $[0, 1]$. Then $P(\theta \mid x[1], \ldots, x[M])$ is proportional to the likelihood $P(x[1], \ldots, x[M] \mid \theta) = \theta^{M[1]}(1-\theta)^{M[0]}$. Plugging this into the integral, we need to compute

$$P(X[M+1] = x^1 \mid x[1], \ldots, x[M]) = \frac{1}{P(x[1], \ldots, x[M])} \int \theta \cdot \theta^{M[1]}(1-\theta)^{M[0]} d\theta.$$

Doing all the math (see exercise 17.6), we get (for uniform priors)

$$P(X[M+1] = x^1 \mid x[1], \ldots, x[M]) = \frac{M[1]+1}{M[1]+M[0]+2}. \tag{17.10}$$

Bayesian estimator

Laplace's correction

This prediction, called the *Bayesian estimator*, is quite similar to the MLE prediction of equation (17.1), except that it adds one "imaginary" sample to each count. Clearly, as the number of samples grows, the Bayesian estimator and the MLE estimator converge to the same value. The particular estimator that corresponds to a uniform prior is often referred to as *Laplace's correction*.

17.3.1.3 Priors

We now want to consider nonuniform priors. The challenge here is to pick a distribution over this continuous space that we can represent compactly (for example, using an analytic formula), and update efficiently as we get new data. For reasons that we will discuss, an appropriate prior in this case is the *Beta distribution*.

Beta distribution

Definition 17.3 Beta hyperparameters

A Beta distribution *is parameterized by two* hyperparameters α_1, α_0, *which are positive reals. The distribution is defined as follows:*

$$\theta \sim \text{Beta}(\alpha_1, \alpha_0) \text{ if } p(\theta) = \gamma \theta^{\alpha_1 - 1}(1-\theta)^{\alpha_0 - 1}.$$

The constant γ is a normalizing constant, defined as follows:

$$\gamma = \frac{\Gamma(\alpha_1 + \alpha_0)}{\Gamma(\alpha_1)\Gamma(\alpha_0)},$$

Gamma function

where $\Gamma(x) = \int_0^\infty t^{x-1} e^{-t} dt$ is the Gamma function. ■

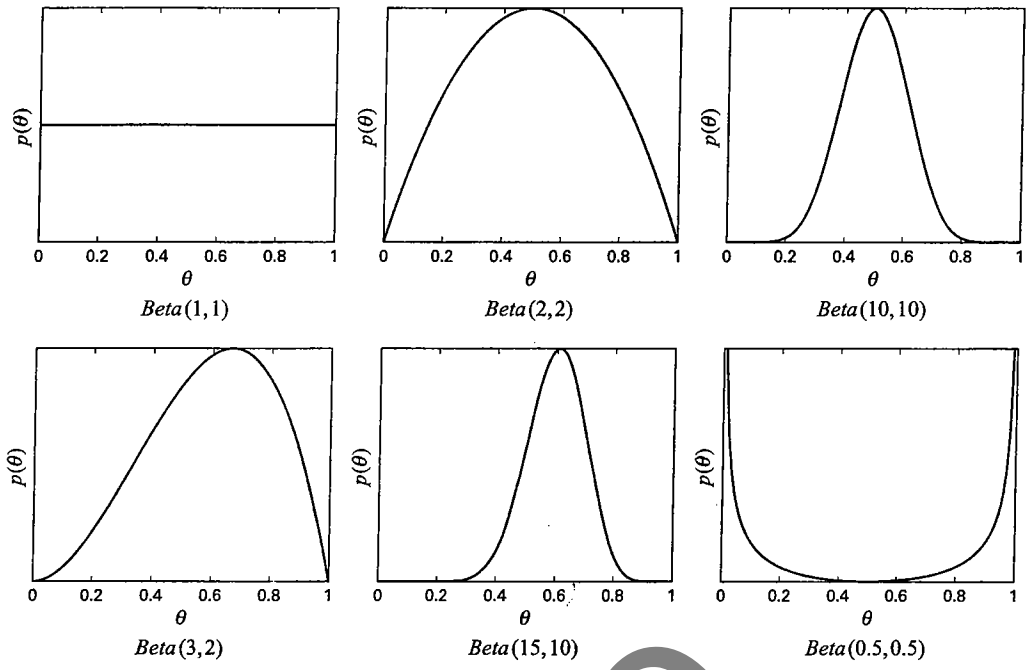


Figure 17.4 Examples of Beta distributions for different choices of hyperparameters

Intuitively, the hyperparameters α_1 and α_0 correspond to the number of imaginary heads and tails that we have "seen" before starting the experiment. Figure 17.4 shows Beta distributions for different values of α.

At first glance, the normalizing constant for the Beta distribution might seem somewhat obscure. However, the Gamma function is actually a very natural one: it is simply a continuous generalization of factorials. More precisely, it satisfies the properties $\Gamma(1) = 1$ and $\Gamma(x+1) = x\Gamma(x)$. As a consequence, we easily see that $\Gamma(n+1) = n!$ when n is an integer.

Beta distributions have properties that make them particularly useful for parameter estimation. Assume our distribution $P(\theta)$ is $Beta(\alpha_1, \alpha_0)$, and consider a single coin toss X. Let us compute the *marginal probability* over X, based on $P(\theta)$. To compute the marginal probability, we need to integrate out θ; standard integration techniques can be used to show that:

$$\begin{aligned} P(X[1] = x^1) &= \int_0^1 P(X[1] = x^1 \mid \theta) \cdot P(\theta) d\theta \\ &= \int_0^1 \theta \cdot P(\theta) d\theta = \frac{\alpha_1}{\alpha_1 + \alpha_0}. \end{aligned}$$

This conclusion supports our intuition that the Beta prior indicates that we have seen α_1 (imaginary) heads α_0 (imaginary) tails.

Now, let us see what happens as we get more observations. Specifically, we observe $M[1]$ heads and $M[0]$ tails. It follows easily that:

$$\begin{aligned} P(\theta \mid x[1], \ldots, x[M]) &\propto P(x[1], \ldots, x[M] \mid \theta) P(\theta) \\ &\propto \theta^{M[1]}(1-\theta)^{M[0]} \cdot \theta^{\alpha_1 - 1}(1-\theta)^{\alpha_0 - 1} \\ &= \theta^{\alpha_1 + M[1] - 1}(1-\theta)^{\alpha_0 + M[0] - 1}, \end{aligned}$$

which is precisely $Beta(\alpha_1 + M[1], \alpha_0 + M[0])$. This result illustrates a key property of the Beta distribution: If the prior is a Beta distribution, then the posterior distribution, that is, the prior conditioned on the evidence, is also a Beta distribution. In this case, we say that the Beta distribution is *conjugate* to the Bernoulli likelihood function (see definition 17.4).

conjugate prior

An immediate consequence is that we can compute the probabilities over the next toss:

$$P(X[M+1] = x^1 \mid x[1], \ldots, x[M]) = \frac{\alpha_1 + M[1]}{\alpha + M},$$

where $\alpha = \alpha_1 + \alpha_0$. In this case, our posterior Beta distribution tells us that we have seen $\alpha_1 + M[1]$ heads (imaginary and real) and $\alpha_0 + M[0]$ tails.

It is interesting to examine the effect of the prior on the probability over the next coin toss. For example, the prior $Beta(1, 1)$ is very different than $Beta(10, 10)$: Although both predict that the probability of heads in the first toss is 0.5, the second prior is more entrenched, and it requires more observations to deviate from the prediction 0.5. To see this, suppose we observe 3 heads in 10 tosses. Using the first prior, our estimate is $\frac{3+1}{10+2} = \frac{1}{3} \approx 0.33$. On the other hand, using the second prior, our estimate is $\frac{3+10}{10+20} = \frac{13}{30} \approx 0.43$. However, as we obtain more data, the effect of the prior diminishes. If we obtain 1,000 tosses of which 300 are heads, the first prior gives us an estimate of $\frac{300+1}{1,000+2}$ and the second an estimate of $\frac{300+10}{1,000+20}$, both of which are very close to 0.3. Thus, **the Bayesian framework allows us to capture both of the relevant distinctions. The distinction between the thumbtack and the coin can be captured by the strength of the prior: for a coin, we might use $\alpha_1 = \alpha_0 = 100$, whereas for a thumbtack, we might use $\alpha_1 = \alpha_0 = 1$. The distinction between a few samples and many samples is captured by the peakedness of our posterior, which increases with the amount of data.**

17.3.2 Priors and Posteriors

We now turn to examine in more detail the Bayesian approach to dealing with unknown parameters. We start with a discussion of the general principle and deal with the case of Bayesian networks in the next section.

As before, we assume a general learning problem where we observe a training set $\mathcal{D}$ that contains M IID samples of a set of random variable $\mathcal{X}$ from an unknown distribution $P^*(\mathcal{X})$. We also assume that we have a parametric model $P(\xi \mid \theta)$ where we can choose parameters from a parameter space Θ.

Recall that the MLE approach attempts to find the parameters $\hat{\theta}$ in Θ that are "best" given the data. The Bayesian approach, on the other hand, does not attempt to find such a *point estimate*. Instead, the underlying principle is that we should keep track of our *beliefs* about θ's values, and use these beliefs for reaching conclusions. That is, we should quantify the subjective probability we assign to different values of θ after we have seen the evidence. Note that, in representing

point estimate

such subjective probabilities, we now treat $\boldsymbol{\theta}$ as a random variable. Thus, the Bayesian approach requires that we use probabilities to describe our initial uncertainty about the parameters $\boldsymbol{\theta}$, and then use probabilistic reasoning (that is, Bayes rule) to take into account our observations.

To perform this task, we need to describe a joint distribution $P(\mathcal{D}, \boldsymbol{\theta})$ over the data and the parameters. We can easily write

$$P(\mathcal{D}, \boldsymbol{\theta}) = P(\mathcal{D} \mid \boldsymbol{\theta})P(\boldsymbol{\theta}).$$

parameter prior

The first term is just the likelihood function we discussed earlier. The second term is the *prior distribution* over the possible values in Θ. This prior captures our initial uncertainty about the parameters. It can also capture our previous experience before starting the experiment. For example, if we study coin tossing, we might have prior experience that suggests that most coins are unbiased (or nearly unbiased).

parameter posterior

Once we have specified the likelihood function and the prior, we can use the data to derive the *posterior distribution* over the parameters. Since we have specified a joint distribution over all the quantities in question, the posterior is immediately derived by Bayes rule:

$$P(\boldsymbol{\theta} \mid \mathcal{D}) = \frac{P(\mathcal{D} \mid \boldsymbol{\theta})P(\boldsymbol{\theta})}{P(\mathcal{D})}.$$

marginal likelihood

The term $P(\mathcal{D})$ is the *marginal likelihood* of the data

$$P(\mathcal{D}) = \int_{\Theta} P(\mathcal{D} \mid \boldsymbol{\theta})P(\boldsymbol{\theta})d\boldsymbol{\theta},$$

that is, the integration of the likelihood over all possible parameter assignments. This is the a priori probability of seeing this particular data set given our prior beliefs.

As we saw, for some probabilistic models, the likelihood function can be compactly described by using sufficient statistics. Can we also compactly describe the posterior distribution? In general, this depends on the form of the prior. As we saw in the thumbtack example of section 17.1.1, we can sometimes find priors for which we have a description of the posterior.

As another example of the forms of priors and posteriors, let us examine the learning problem of example 17.3. Here we need to describe our uncertainty about the parameters of a multinomial distribution. The parameter space Θ is the space of all nonnegative vectors $\boldsymbol{\theta} = \langle \theta_1, \ldots, \theta_K \rangle$ such that $\sum_k \theta_k = 1$. As we saw in example 17.3, the likelihood function in this model has the form:

$$L(\mathcal{D} : \boldsymbol{\theta}) = \prod_k \theta_k^{M[k]}.$$

Since the posterior is a product of the prior and the likelihood, it seems natural to require that the prior also have a form similar to the likelihood.

Dirichlet distribution

Dirichlet hyperparameters

One such prior is the *Dirichlet distribution*, which generalizes the Beta distribution we discussed earlier. A Dirichlet distribution is specified by a set of *hyperparameters* $\alpha_1, \ldots, \alpha_K$, so that

$$\boldsymbol{\theta} \sim Dirichlet(\alpha_1, \ldots, \alpha_K) \text{ if } P(\boldsymbol{\theta}) \propto \prod_k \theta_k^{\alpha_k - 1}.$$

Dirichlet posterior

We use α to denote $\sum_j \alpha_j$. If we use a Dirichlet prior, then the *posterior* is also Dirichlet:

Proposition 17.3 *If $P(\theta)$ is Dirichlet$(\alpha_1, \ldots, \alpha_K)$ then $P(\theta \mid \mathcal{D})$ is Dirichlet$(\alpha_1 + M[1], \ldots, \alpha_K + M[K])$, where $M[k]$ is the number of occurrences of x^k.*

Priors such as the Dirichlet are useful, since they ensure that the posterior has a nice compact description. Moreover, this description uses the same representation as the prior. This phenomenon is a general one, and one that we strive to achieve, since it makes our computation and representation much easier.

Definition 17.4 conjugate prior

A family of priors $P(\theta : \alpha)$ is conjugate *to a particular model $P(\xi \mid \theta)$ if for any possible data set $\mathcal{D}$ of IID samples from $P(\xi \mid \theta)$, and any choice of legal hyperparameters α for the prior over θ, there are hyperparameters α' that describe the posterior. That is,*

$$P(\theta : \alpha') \propto P(\mathcal{D} \mid \theta) P(\theta : \alpha).$$

■

For example, Dirichlet priors are conjugate to the multinomial model. We note that this does not preclude the possibility of other families that are also conjugate to the same model. See exercise 17.7 for an example of such a prior for the multinomial model. We can find conjugate priors for other models as well. See exercise 17.8 and exercise 17.11 for the development of conjugate priors for the Gaussian distribution.

This discussion shows some examples where we can easily update our beliefs about θ after observing a set of instances $\mathcal{D}$. This update process results in a posterior that combines our prior knowledge and our observations. What can we do with the posterior? We can use the posterior to determine properties of the model at hand. For example, to assess our beliefs that a coin we experimented with is biased toward heads, we might compute the posterior probability that $\theta > t$ for some threshold t, say 0.6.

Another use of the posterior is to predict the probability of future examples. Suppose that we are about to sample a new instance $\xi[M+1]$. Since we already have observations over previous instances, the *Bayesian estimator* is the posterior distribution over a new example:

Bayesian estimator

$$\begin{aligned} P(\xi[M+1] \mid \mathcal{D}) &= \int P(\xi[M+1] \mid \mathcal{D}, \theta) P(\theta \mid \mathcal{D}) d\theta \\ &= \int P(\xi[M+1] \mid \theta) P(\theta \mid \mathcal{D}) d\theta \\ &= \mathbf{E}_{P(\theta \mid \mathcal{D})}[P(\xi[M+1] \mid \theta)], \end{aligned}$$

where, in the second step, we use the fact that instances are independent given θ. Thus, our prediction is the average over all parameters according to the posterior.

Let us examine prediction with the Dirichlet prior. We need to compute

$$P(x[M+1] = x^k \mid \mathcal{D}) = \mathbf{E}_{P(\theta \mid \mathcal{D})}[\theta_k].$$

To compute the prediction on a new data case, we need to compute the expectation of particular parameters with respect for a Dirichlet distribution over θ.

Proposition 17.4 *Let $P(\theta)$ be a Dirichlet distribution with hyperparameters $\alpha_1, \ldots, \alpha_k$, and $\alpha = \sum_j \alpha_j$, then $E[\theta_k] = \frac{\alpha_k}{\alpha}$.*

Recall that our posterior is $Dirichlet(\alpha_1 + M[1], \ldots, \alpha_K + M[K])$ where $M[1], \ldots, M[K]$ are the sufficient statistics from the data. Hence, the prediction with Dirichlet priors is

$$P(x[M+1] = x^k \mid \mathcal{D}) = \frac{M[k] + \alpha_k}{M + \alpha}.$$

This prediction is similar to prediction with the MLE parameters. The only difference is that we added the hyperparameters to our counts when making the prediction. For this reason the Dirichlet hyperparameters are often called *pseudo-counts*. We can think of these as the number of times we have seen the different outcomes in our prior experience before conducting our current experiment.

pseudo-counts

The total α of the pseudo-counts reflects how confident we are in our prior, and is often called the *equivalent sample size*. Using α, we can rewrite the hyperparameters as $\alpha_k = \alpha\theta'_k$, where $\boldsymbol{\theta}' = \{\theta'_k : k = 1, \ldots, K\}$ is a distribution describing the *mean prediction* of our prior. We can see that the prior prediction (before observing any data) is simply $\boldsymbol{\theta}'$. Moreover, we can rewrite the prediction given the posterior as:

equivalent sample size

mean prediction

$$P(x[M+1] = x^k \mid \mathcal{D}) = \frac{\alpha}{M+\alpha}\theta'_k + \frac{M}{M+\alpha} \cdot \frac{M[k]}{M}. \tag{17.11}$$

That is, the prediction is a weighted average (convex combination) of the prior mean and the MLE estimate. The combination weights are determined by the relative magnitude of α — the confidence of the prior (or total weight of the pseudo-counts) — and M — the number of observed samples. We see that the Bayesian prediction converges to the MLE estimate when $M \to \infty$. Intuitively, when we have a very large training set the contribution of the prior is negligible, and the prediction will be dominated by the frequency of outcomes in the data. We also get convergence to the MLE estimate when $\alpha \to 0$, so that we have only a very weak prior. Note that the case where $\alpha = 0$ is not achievable: the normalization constant for the Dirichlet prior grows to infinity when the hyperparameters are close to 0. Thus, the prior with $\alpha = 0$ (that is, $\alpha_k = 0$ for all k) is not well defined. The prior with $\alpha = 0$ is often called a *improper prior*. The difference between the Bayesian estimate and the MLE estimate arises when M is not too large, and α is not close to 0. In these situations, the Bayesian estimate is "biased" toward the prior probability $\boldsymbol{\theta}'$.

improper prior

To gain some intuition for the interaction between these different factors, figure 17.5 shows the effect of the strength and means of the prior on our estimates. We can see that, as the amount of real data grows, our estimate converges to the true underlying distribution, regardless of the starting point. The convergence time grows both with the difference between the prior mean and the empirical mean, and with the strength of the prior. We also see that the Bayesian estimate is *smoother* than the MLE estimate, because with few instances, even single samples will change the MLE estimate dramatically.

Example 17.7 *Suppose we are trying to estimate the parameter associated with a coin, and we observe one head and one tail. Our MLE estimate of θ_1 is $1/2 = 0.5$. Now, if the next observation is a head, we will change our estimate to be $2/3 \approx 0.66$. On the other hand, if our next observation is a tail, we will change our estimate to $1/3 \approx 0.33$. In contrast, consider the Bayesian estimate with a Dirichlet prior with $\alpha = 1$ and $\theta'_1 = 0.5$. With this estimator, our original estimate is $1.5/3 = 0.5$. If we observe another head, we revise to $2.5/4 = 0.625$, and if observe another tail, we revise to*

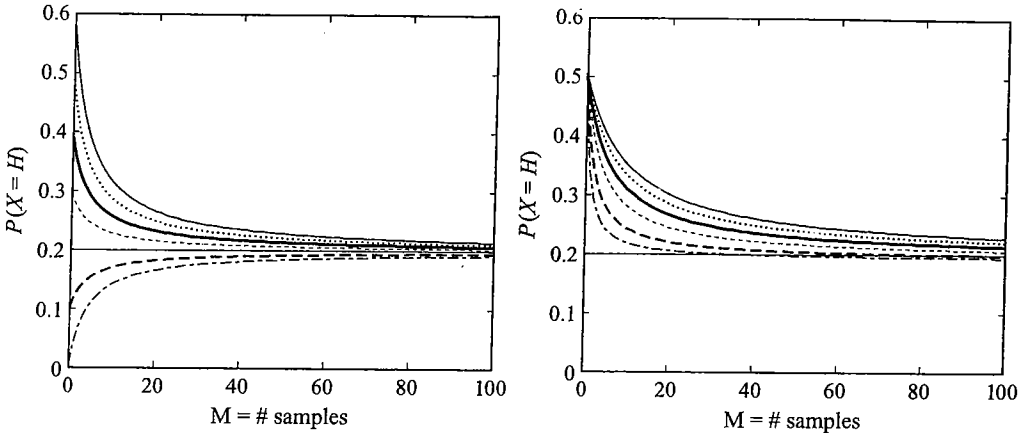


Figure 17.5 The effect of the strength and means of the Beta prior on our posterior estimates. Our data set is an idealized version of samples from a biased coin where the frequency of heads is 0.2: for a given data set size M, we assume that $\mathcal{D}$ contains $0.2M$ heads and $0.8M$ tails. The x axis represents the number of samples (M) in our data set $\mathcal{D}$, and the y axis the expected probability of heads according to the Bayesian estimate. (a) shows the effect of varying the prior means θ'_1, θ'_0, for a fixed prior strength α. (b) shows the effect of varying the prior strength for a fixed prior mean $\theta'_1 = \theta'_0 = 0.5$.

$1.5/4 = 0.375$. *We see that the estimate changes by slightly less after the update. If α is larger, then the smoothing is more aggressive. For example, when $\alpha = 5$, our estimate is $4.5/8 = 0.5625$ after observing a head, and $3.5/8 = 0.4375$ after observing a tail. We can also see this effect visually in figure 17.6, which shows our changing estimate for $P(\theta_H)$ as we observe a particular sequence of tosses.* ■

This smoothing effect results in more robust estimates when we do not have enough data to reach definite conclusions. If we have good prior knowledge, we revert to it. Alternatively, if we do not have prior knowledge, we can use a uniform prior that will keep our estimate from taking extreme values. In general, it is a bad idea to have extreme estimates (ones where some of the parameters are close to 0), since these might assign too small probability to new instances we later observe. In particular, as we already discussed, probability estimates that are actually 0 are dangerous, since no amount of evidence can change them. Thus, if we are unsure about our estimates, it is better to bias them away from extreme estimates. The MLE estimate, on the other hand, often assigns probability 0 to values that were not observed in the training data.

17.4 Bayesian Parameter Estimation in Bayesian Networks

We now turn to Bayesian estimation in the context of a Bayesian network. Recall that the Bayesian framework requires us to specify a joint distribution over the unknown parameters and the data instances. As in the single parameter case, we can understand the joint distribution over parameters and data as a Bayesian network.

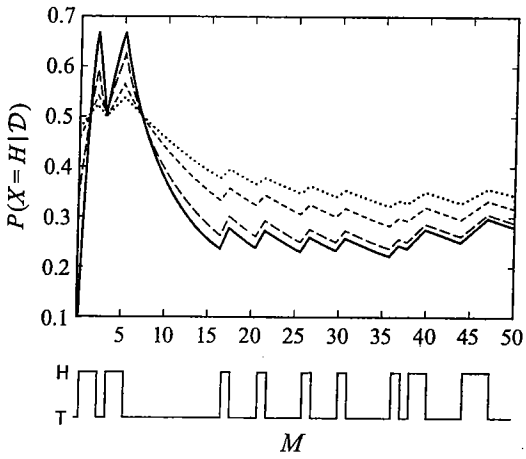


Figure 17.6 The effect of different priors on smoothing our parameter estimates. The graph shows the estimate of $P(X = H|\mathcal{D})$ (y-axis) after seeing different number of samples (x-axis). The graph below the x-axis shows the particular sequence of tosses. The solid line corresponds to the MLE estimate, and the remaining ones to Bayesian estimates with different strengths and uniform prior means. The large-dash line corresponds to *Beta*$(1, 1)$, the small-dash line to *Beta*$(5, 5)$, and the dotted line to *Beta*$(10, 10)$.

17.4.1 Parameter Independence and Global Decomposition

17.4.1.1 A Simple Example

Suppose we want to estimate parameters for a simple network with two variables X and Y so that X is the parent of Y. Our training data consist of observations $X[m], Y[m]$ for $m = 1, \ldots, M$. In addition, we have unknown parameter vectors $\boldsymbol{\theta}_X$ and $\boldsymbol{\theta}_{Y|X}$. The dependencies between these variables are described in the network of figure 17.7. This is the *meta-network* that describes our learning setup.

meta-network

This Bayesian network structure immediately reveals several points. For example, as in our simple thumbtack example, the instances are independent given the unknown parameters. A simple examination of active trails shows that $X[m]$ and $Y[m]$ are d-separated from $X[m']$ and $Y[m']$ once we observe the parameter variables.

In addition, the network structure embodies the assumption that the priors for the individual parameters variables are a priori independent. That is, we believe that knowing the value of one parameter tells us nothing about another. More precisely, we define

Definition 17.5

global parameter independence

Let $\mathcal{G}$ be a Bayesian network structure with parameters $\boldsymbol{\theta} = (\boldsymbol{\theta}_{X_1|\mathrm{Pa}_{X_1}}, \ldots, \boldsymbol{\theta}_{X_n|\mathrm{Pa}_{X_n}})$. A prior $P(\boldsymbol{\theta})$ is said to satisfy global parameter independence *if it has the form:*

$$P(\boldsymbol{\theta}) = \prod_i P(\boldsymbol{\theta}_{X_i|\mathrm{Pa}_{X_i}}).$$

■

This assumption may not be suitable for all domains, and it should be considered with care.

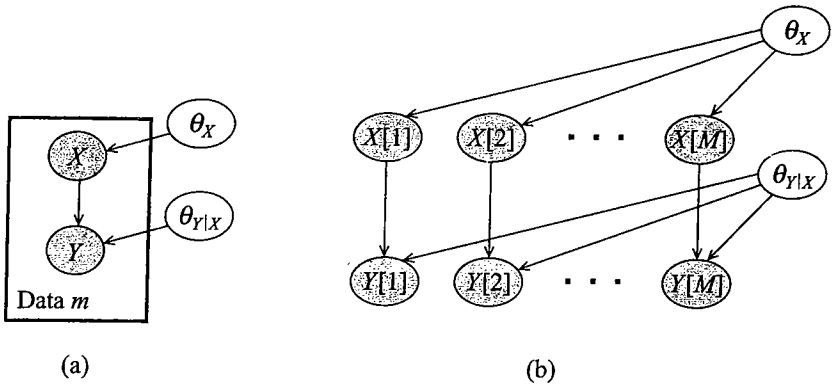


Figure 17.7 Meta-network for IID samples from a network $X \rightarrow Y$ with global parameter independence. (a) Plate model; (b) Ground Bayesian network.

Example 17.8

Consider an extension of our student *example, where our student takes multiple classes. For each class, we want to learn the distribution of Grade given the student's Intelligence and the course Difficulty. For classes taught by the same instructor, we might believe that the grade distribution is the same; for example, if two classes are both difficult, and the student is intelligent, his probability of getting an A is the same in both. However, under the global parameter independence assumption, these are two different random variables, and hence their parameters are independent.* ■

Thus, although we use the global parameter independence in this chapter, it is not always appropriate, and we relax it in later chapters.

If we accept global parameter independence, we can draw an important conclusion. Complete data d-separates the parameters for different CPDs. For example, if $x[m]$ and $y[m]$ are observed for all m, then $\boldsymbol{\theta}_X$ and $\boldsymbol{\theta}_{Y|X}$ are d-separated. To see this, note that any path between the two has the form

$$\boldsymbol{\theta}_X \rightarrow X[m] \rightarrow Y[m] \leftarrow \boldsymbol{\theta}_{Y|X},$$

so that the observation of $x[m]$ blocks the path. Thus, if these two parameter variables are independent a priori, they are also independent a posteriori. Using the definition of conditional independence, we conclude that

$$P(\boldsymbol{\theta}_X, \boldsymbol{\theta}_{Y|X} \mid \mathcal{D}) = P(\boldsymbol{\theta}_X \mid \mathcal{D}) P(\boldsymbol{\theta}_{Y|X} \mid \mathcal{D}).$$

This decomposition has immediate practical ramifications. Given the data set $\mathcal{D}$, we can determine the posterior over $\boldsymbol{\theta}_X$ independently of the posterior over $\boldsymbol{\theta}_{Y|X}$. Once we can solve each problem separately, we can combine the results. This is the analogous result to the likelihood decomposition for MLE estimation of section 17.2.2. In the Bayesian setting this property has additional importance. It tells us the posterior can be represented in a compact factorized form.

17.4.1.2 General Networks

We can generalize this conclusion to the general case of Bayesian network learning. Suppose we are given a network structure $\mathcal{G}$ with parameters $\boldsymbol{\theta}$. In the Bayesian framework, we need to specify a prior $P(\boldsymbol{\theta})$ over all possible parameterizations of the network. The posterior distribution over parameters given the data samples $\mathcal{D}$ is simply

$$P(\boldsymbol{\theta} \mid \mathcal{D}) = \frac{P(\mathcal{D} \mid \boldsymbol{\theta})P(\boldsymbol{\theta})}{P(\mathcal{D})}.$$

The term $P(\boldsymbol{\theta})$ is our prior distribution, $P(\mathcal{D} \mid \boldsymbol{\theta})$ is the probability of the data given a particular parameter settings, which is simply the likelihood function. Finally, $P(\mathcal{D})$ is the normalizing constant. As we discussed, this term is called the *marginal likelihood*; it will play an important role in the next chapter. For now, however, we can ignore it, since it does not depend on $\boldsymbol{\theta}$ and only serves to normalize the posterior.

marginal likelihood

As we discussed in section 17.2, we can decompose the likelihood into local likelihoods:

$$P(\mathcal{D} \mid \boldsymbol{\theta}) = \prod_i L_i(\boldsymbol{\theta}_{X_i|\mathrm{Pa}_{X_i}} : \mathcal{D}).$$

Moreover, if we assume that we have global parameter independence, then

$$P(\boldsymbol{\theta}) = \prod_i P(\boldsymbol{\theta}_{X_i|\mathrm{Pa}_{X_i}}).$$

Combining these two decompositions, we see that

$$P(\boldsymbol{\theta} \mid \mathcal{D}) = \frac{1}{P(\mathcal{D})} \prod_i \left[L_i(\boldsymbol{\theta}_{X_i|\mathrm{Pa}_{X_i}} : \mathcal{D}) P(\boldsymbol{\theta}_{X_i|\mathrm{Pa}_{X_i}}) \right].$$

Now each subset $\boldsymbol{\theta}_{X_i|\mathrm{Pa}_{X_i}}$ of $\boldsymbol{\theta}$ appears in just one term in the product. Thus, we have that the posterior can be represented as a product of local terms.

Proposition 17.5 *Let $\mathcal{D}$ be a complete data set for $\mathcal{X}$, let $\mathcal{G}$ be a network structure over these variables. If $P(\boldsymbol{\theta})$ satisfies global parameter independence, then*

$$P(\boldsymbol{\theta} \mid \mathcal{D}) = \prod_i P(\boldsymbol{\theta}_{X_i|\mathrm{Pa}_{X_i}} \mid \mathcal{D}).$$

The proof of this property follows from the steps we discussed. It can also be derived directly from the structure of the meta-Bayesian network (as in the network of figure 17.7).

17.4.1.3 Prediction

This decomposition of the posterior allows us to simplify various tasks. For example, suppose that, in our simple two-variable network, we want to compute the probability of another instance $x[M+1], y[M+1]$ based on our previous observations $x[1], y[1], \ldots, x[M], y[M]$. According to the structure of our meta-network, we need to sum out (or more precisely integrate out) the unknown parameter variables

$$P(x[M+1], y[M+1] \mid \mathcal{D}) = \int P(x[M+1], y[M+1] \mid \mathcal{D}, \boldsymbol{\theta}) P(\boldsymbol{\theta} \mid \mathcal{D}) d\boldsymbol{\theta},$$

where the integration is over all legal parameter values. Since $\boldsymbol{\theta}$ d-separates instances from each other, we have that

$$\begin{aligned} & P(x[M+1], y[M+1] \mid \mathcal{D}, \boldsymbol{\theta}) \\ &= P(x[M+1], y[M+1] \mid \boldsymbol{\theta}) \\ &= P(x[M+1] \mid \boldsymbol{\theta}_X) P(y[M+1] \mid x[M+1], \boldsymbol{\theta}_{Y|X}). \end{aligned}$$

Moreover, as we just saw, the posterior probability also decomposes into a product. Thus,

$$\begin{aligned} & P(x[M+1], y[M+1] \mid \mathcal{D}) \\ &= \int\int P(x[M+1] \mid \boldsymbol{\theta}_X) P(y[M+1] \mid x[M+1], \boldsymbol{\theta}_{Y|X}) \\ &\qquad P(\boldsymbol{\theta}_X \mid \mathcal{D}) P(\boldsymbol{\theta}_Y \mid \mathcal{D}) d\boldsymbol{\theta}_X d\boldsymbol{\theta}_Y \\ &= \left(\int P(x[M+1] \mid \boldsymbol{\theta}_X) P(\boldsymbol{\theta}_X \mid \mathcal{D}) d\boldsymbol{\theta}_X \right) \\ &\qquad \left(\int P(y[M+1] \mid x[M+1], \boldsymbol{\theta}_{Y|X}) P(\boldsymbol{\theta}_Y \mid \mathcal{D}) d\boldsymbol{\theta}_Y \right). \end{aligned}$$

In the second step, we use the fact that the double integral of two unrelated functions is the product of the integrals. That is:

$$\int\int f(x) g(y) dx dy = \left(\int f(x) dx \right) \left(\int g(y) dy \right).$$

Thus, we can solve the prediction problem for the two variables X and Y separately.

The same line of reasoning easily applies to the general case, and thus we can see that, in the setting of proposition 17.5, we have

$$\begin{aligned} & P(X_1[M+1], \ldots, X_n[M+1] \mid \mathcal{D}) = \\ & \qquad \prod_i \int P(X_i[M+1] \mid \mathrm{Pa}_{X_i}[M+1], \boldsymbol{\theta}_{X_i|\mathrm{Pa}_{X_i}}) P(\boldsymbol{\theta}_{X_i|\mathrm{Pa}_{X_i}} \mid \mathcal{D}) d\boldsymbol{\theta}_{X_i|\mathrm{Pa}_{X_i}}. \end{aligned} \tag{17.12}$$

We see that we can solve the prediction problem for each CPD independently and then combine the results.

We stress that the discussion so far was based on the assumption that the priors over parameters for different CPDs are independent. We see that, when learning from complete data, this assumption alone suffices to get a decomposition of the learning problem to several "local" problems, each one involving one CPD.

At this stage it might seem that the Bayesian framework introduces new complications that did not appear in the MLE setup. Note, however, that in deriving the MLE decomposition, we used the property that we can choose parameters for one CPD independently of the others. Thus, we implicitly made a similar assumption to get decomposition. The Bayesian treatment forces us to make such assumptions explicit, allowing us to more carefully evaluate their validity. We view this as a benefit of the Bayesian framework.

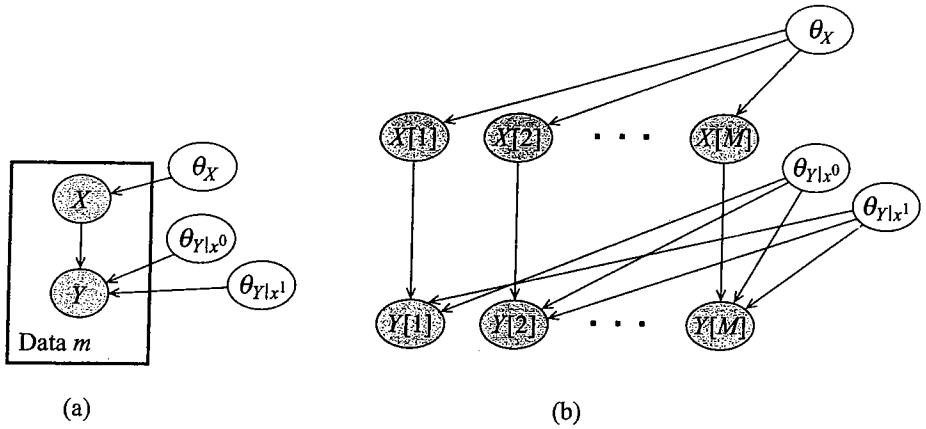


Figure 17.8 Meta-network for IID samples from a network $X \rightarrow Y$ with local parameter independence. (a) Plate model. (b) Ground Bayesian network.

17.4.2 Local Decomposition

Based on the preceding discussion, we now need to solve localized Bayesian estimation problems to get a global Bayesian solution. We now examine this localized estimation task for table-CPDs and tree-CPDs.

17.4.2.1 Table-CPDs

Consider, for example, the learning setting described in figure 17.7, where we take both X and Y to be binary. As we have seen, we need to represent the posterior $\boldsymbol{\theta}_X$ and $\boldsymbol{\theta}_{Y|X}$ given the data. We already know how to deal with the posterior over $\boldsymbol{\theta}_X$. If we use a Dirichlet prior over $\boldsymbol{\theta}_X$, then the posterior $P(\boldsymbol{\theta}_X \mid x[1], \ldots, x[M])$ is also represented as a Dirichlet distribution.

A less obvious question is how to deal with the posterior over $\boldsymbol{\theta}_{Y|X}$. If we are learning table-CPDs, this parameter vector contains four parameters $\theta_{y^0|x^0}, \ldots, \theta_{y^1|x^1}$. In our discussion of maximum likelihood estimation, we saw how the local likelihood over these parameters can be further decomposed into two terms, one over the parameters $\boldsymbol{\theta}_{Y|x^0}$ and one over the parameters $\boldsymbol{\theta}_{Y|x^1}$. Do we have a similar phenomenon in the Bayesian setting?

We start with the prior over $\boldsymbol{\theta}_{Y|X}$. One obvious choice is a Dirichlet prior over $\boldsymbol{\theta}_{Y|x^1}$ and another over $\boldsymbol{\theta}_{Y|x^0}$. More precisely, we have

$$P(\boldsymbol{\theta}_{Y|X}) = P(\boldsymbol{\theta}_{Y|x^1})P(\boldsymbol{\theta}_{Y|x^0}),$$

where each of the terms on the right is a Dirichlet prior. Thus, in this case, we assume that the two groups of parameters are independent a priori.

This independence assumption, in effect, allows us to replace the node $\boldsymbol{\theta}_{Y|X}$ in figure 17.7 with two nodes, $\boldsymbol{\theta}_{Y|x^1}$ and $\boldsymbol{\theta}_{Y|x^0}$ that are both roots (see figure 17.8). What can we say about the posterior distribution of these parameter groups? At first, it seems that the two are dependent

on each other given the data. Given an observation of $y[m]$, the path

$$\theta_{Y|x^0} \rightarrow Y[m] \leftarrow \theta_{Y|x^1}$$

is active (since we observe the sink of a v-structure), and thus the two parameters are not d-separated.

This, however, is not the end of the story. We get more insight if We examine how $y[m]$ depends on the two parameters. Clearly,

$$P(y[m] = y \mid x[m], \theta_{Y|x^0}, \theta_{Y|x^1}) = \begin{cases} \theta_{y|x^0} & \text{if } x[m] = x^0 \\ \theta_{y|x^1} & \text{if } x[m] = x^1. \end{cases}$$

We see that $y[m]$ does not depends on the value of $\theta_{Y|x^0}$ when $x[m] = x^1$. This example is an instance of the same type of context specific independence that we discussed in example 3.7. As discussed in section 5.3, we can perform a more refined form of d-separation test in such a situation by removing arcs that are ruled inactive in particular contexts. For the CPD of $y[m]$, we see that once we observe the value of $x[m]$, one of the two arcs into $y[m]$ is inactive. If $x[m] = x^0$, then the arc $\theta_{Y|x^1} \rightarrow y[m]$ is inactive, and if $x[m] = x^1$, then $\theta_{Y|x^0} \rightarrow y[m]$ is inactive. In either case, the v-structure $\theta_{Y|x^0} \rightarrow y[m] \leftarrow \theta_{Y|x^1}$ is removed. Since this removal occurs for every $m = 1, \ldots, M$, we conclude that no active path exists between $\theta_{Y|x^0}$ and $\theta_{Y|x^1}$ and thus, the two are independent given the observation of the data. In other words, we can write

$$P(\theta_{Y|X} \mid \mathcal{D}) = P(\theta_{Y|x^1} \mid \mathcal{D}) P(\theta_{Y|x^0} \mid \mathcal{D}).$$

Suppose that $P(\theta_{Y|x^0})$ is a Dirichlet prior with hyperparameters $\alpha_{y^0|x^0}$ and $\alpha_{y^1|x^0}$. As in our discussion of the local decomposition for the likelihood function in section 17.2.3, we have that the likelihood terms that involve $\theta_{Y|x^0}$ are those that measure the probability of $P(y[m] \mid x[m], \theta_{Y|X})$ when $x[m] = x^0$. Thus, we can decompose the joint distribution over parameters and data as follows:

$$\begin{aligned} P(\theta, \mathcal{D}) \;=\; & P(\theta_X) L_X(\theta_X : \mathcal{D}) \\ & P(\theta_{Y|x^1}) \prod_{m : x[m] = x^1} P(y[m] \mid x[m] : \theta_{Y|x^1}) \\ & P(\theta_{Y|x^0}) \prod_{m : x[m] = x^0} P(y[m] \mid x[m] : \theta_{Y|x^0}). \end{aligned}$$

Thus, this joint distribution is a product of three separate joint distributions with a Dirichlet prior for some multinomial parameter and data drawn from this multinomial. Our analysis for updating a single Dirichlet now applies, and we can conclude that the posterior $P(\theta_{Y|x^0} \mid \mathcal{D})$ is Dirichlet with hyperparameters $\alpha_{y^0|x^0} + M[x^0, y^0]$ and $\alpha_{y^1|x^0} + M[x^0, y^1]$.

We can generalize this discussion to arbitrary networks.

Definition 17.6 local parameter independence

Let X be a variable with parents $\mathbf{U}$. We say that the prior $P(\theta_{X|\mathbf{U}})$ satisfies local parameter independence *if*

$$P(\theta_{X|\mathbf{U}}) = \prod_{u} P(\theta_{X|u}).$$

■

The same pattern of reasoning also applies to the general case.

Proposition 17.6 *Let $\mathcal{D}$ be a complete data set for $\mathcal{X}$, let $\mathcal{G}$ be a network structure over these variables with table-CPDs. If the prior $P(\boldsymbol{\theta})$ satisfies global and local parameter independence, then*

$$P(\boldsymbol{\theta} \mid \mathcal{D}) = \prod_i \prod_{\mathrm{pa}_{X_i}} P(\boldsymbol{\theta}_{X_i|\mathrm{pa}_{X_i}} \mid \mathcal{D}).$$

Moreover, if $P(\boldsymbol{\theta}_{X|\boldsymbol{u}})$ is a Dirichlet prior with hyperparameters $\alpha_{x^1|\boldsymbol{u}}, \ldots, \alpha_{x^K|\boldsymbol{u}}$, then the posterior $P(\boldsymbol{\theta}_{X|\boldsymbol{u}} \mid \mathcal{D})$ is a Dirichlet distribution with hyperparameters $\alpha_{x^1|\boldsymbol{u}} + M[\boldsymbol{u}, x^1], \ldots, \alpha_{x^K|\boldsymbol{u}} + M[\boldsymbol{u}, x^K]$.

As in the case of a single multinomial, this result induces a predictive model in which, for the next instance, we have that

$$P(X_i[M+1] = x_i \mid \boldsymbol{U}[M+1] = \boldsymbol{u}, \mathcal{D}) = \frac{\alpha_{x_i|\boldsymbol{u}} + M[x_i, \boldsymbol{u}]}{\sum_i \alpha_{x_i|\boldsymbol{u}} + M[x_i, \boldsymbol{u}]}. \tag{17.13}$$

Plugging this result into equation (17.12), we see that for computing the probability of a new instance, we can use a single network parameterized as usual, via a set of multinomials, but ones computed as in equation (17.13).

17.4.3 Priors for Bayesian Network Learning

Bayesian network parameter prior

It remains only to address the question of assessing the set of *parameter priors* required for a Bayesian network. In a general Bayesian network, each node X_i has a set of multinomial distributions $\boldsymbol{\theta}_{X_i|\mathrm{pa}_{X_i}}$, one for each instantiation pa_{X_i} of X_i's parents Pa_{X_i}. Each of these parameters will have a separate Dirichlet prior, governed by hyperparameters

$$\boldsymbol{\alpha}_{X_i|\mathrm{pa}_{X_i}} = (\alpha_{x_i^1|\mathrm{pa}_{X_i}}, \ldots, \alpha_{x_i^{K_i}|\mathrm{pa}_{X_i}}),$$

where K_i is the number of values of X_i.

We can, of course, ask our expert to assign values to each of these hyperparameters based on his or her knowledge. This task, however, is rather unwieldy. Another approach, called the K2 prior, is to use a fixed prior, say $\alpha_{x_i^j|\mathrm{pa}_{X_i}} = 1$, for all hyperparameters in the network. As we discuss in the next chapter, this approach has consequences that are conceptually unsatisfying; see exercise 18.10.

A common approach to addressing the specification task uses the intuitions we described in our discussion of Dirichlet priors in section 17.3.1. As we showed, we can think of the hyperparameter α_{x^k} as an imaginary count in our prior experience. This intuition suggests the following representation for a prior over a Bayesian network. Suppose we have an imaginary data set $\mathcal{D}'$ of "prior" examples. Then, we can use counts from this imaginary data set as hyperparameters. More specifically, we set

$$\alpha_{x_i|\mathrm{pa}_{X_i}} = \alpha[x_i, \mathrm{pa}_{X_i}],$$

where $\alpha[x_i, \mathrm{pa}_{X_i}]$ is the number of times $X_i = x_i$ and $\mathrm{Pa}_{X_i} = \mathrm{pa}_{X_i}$ in $\mathcal{D}'$. We can easily see that prediction with this setting of hyperparameters is equivalent to MLE prediction from the combined data set that contains instances of both $\mathcal{D}$ and $\mathcal{D}'$.

One problem with this approach is that it requires storing a possibly large data set of pseudo-instances. Instead, we can store the size of the data set α and a representation $P'(X_1, \ldots, X_n)$ of the frequencies of events in this prior data set. If $P'(X_1, \ldots, X_n)$ is the distribution of events in $\mathcal{D}'$, then we get that

$$\alpha_{x_i|\mathrm{pa}_{X_i}} = \alpha \cdot P'(x_i, \mathrm{pa}_{X_i}).$$

How do we represent P'? Clearly, one natural choice is via a Bayesian network. Then, we can use Bayesian network inference to efficiently compute the quantities $P'(x_i, \mathrm{pa}_{X_i})$. Note that P' does not have to be structured in the same way as the network we learn (although it can be). It is, in fact, quite common to define P' as a set of independent marginals over the X_i's. A prior that can be represented in this manner (using α and P') is called a *BDe prior*. Aside from being philosophically pleasing, it has some additional benefits that we will discuss in the next chapter.

BDe prior

Box 17.C — Case Study: Learning the ICU-Alarm Network. *To give an example of the techniques described in this chapter, we evaluate them on a synthetic example. Figure 17.C.1 shows the graph structure of the real-world* ICU-Alarm *Bayesian network, hand-constructed by an expert, for monitoring patients in an Intensive Care Unit (ICU). The network has 37 nodes and a total of 504 parameters. We want to evaluate the ability of our parameter estimation algorithms to reconstruct the network parameters from data.*

ICU-Alarm

We generated a training set from the network, by sampling from the distribution specified by the network. We then gave the algorithm only the (correct) network structure, and the generated data, and measured the ability of our algorithms to reconstruct the parameters. We tested the MLE approach, and several Bayesian approaches. All of the approaches used a uniform prior mean, but different prior strengths α.

In performing such an experiment, there are many ways of measuring the quality of the learned network. One possible measure is the difference between the original values of the model parameters and the estimated ones. A related approach measures the distance between the original CPDs and the learned ones (in the case of table-CPDs, these two approaches are the same, but not for general parameterizations). These approaches place equal weights on different parameters, regardless of the extent to which they influence the overall distribution.

The approach we often take is the one described in section 16.2.1, where we measure the relative entropy between the generating distribution P^* and the learned distribution $\tilde{P}$ (see also section 8.4.2). This approach provides a global measure of the extent to which our learned distribution resembles the true distribution. Figure 17.C.2 shows the results for different stages of learning. As we might expect, when more instances are available, the estimation is better. The improvement is drastic in early stages of learning, where additional instances lead to major improvements. When the number of instances in our data set is larger, additional instances lead to improvement, but a smaller one.

More surprisingly, we also see that the MLE achieves the poorest results, a consequence of its extreme sensitivity to the specific training data used. The lowest error is achieved with a very weak prior — $\alpha = 5$ — which is enough to provide smoothing. As the strength of the prior grows, it starts to introduce a bias, not giving the data enough importance. Thus, the error of the estimated probability increases. However, we also note that the effect of the prior, even for $\alpha = 50$, disappears reasonably soon, and all of the approaches converge to the same line. Interestingly, the different

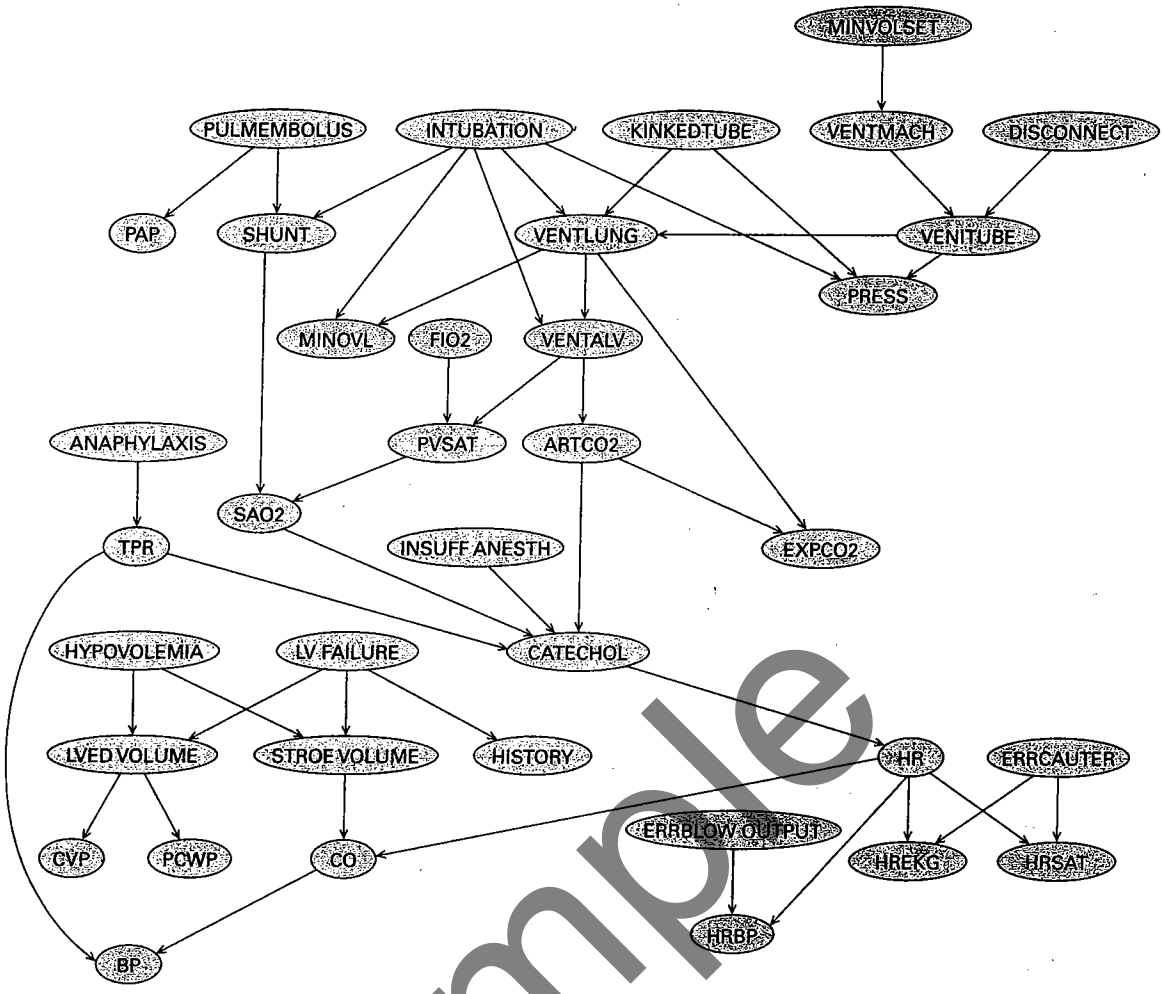


Figure 17.C.1 — The ICU-Alarm Bayesian network.

Bayesian approaches converge to this line long before the MLE approach. Thus, at least in this example, an overly strong bias provided by the prior is still a better compromise than the complete lack of smoothing of the MLE approach.

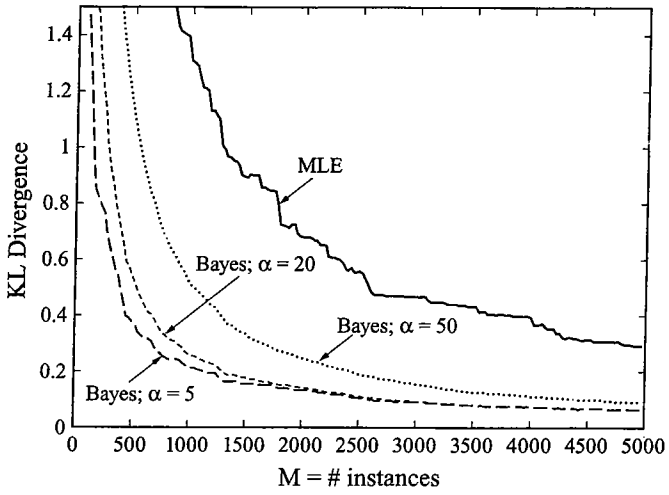


Figure 17.C.2 — Learning curve for parameter estimation for the ICU-Alarm network Relative entropy to true model as the amount of training data grows, for different values of the prior strength α.

17.4.4 MAP Estimation ⋆

Our discussion in this chapter has focused solely on Bayesian estimation for multinomial CPDs. Here, we have a closed form solution for the integral required for Bayesian prediction, and thus we can perform it efficiently. In many other representations, the situation is not so simple. In some cases, such as the noisy-or model or the logistic CPDs of section 5.4.2, we do not have a conjugate prior or a closed-form solution for the Bayesian integral. In those cases, Bayesian prediction requires numerical solutions for high-dimensional integrals. In other settings, such as the linear Gaussian CPD, we do have a conjugate prior (the *normal-Gamma distribution*), but we may prefer other priors that offer other desirable properties (such as the sparsity-inducing Laplacian prior described in section 20.4.1).

normal-Gamma distribution

MAP estimation

When a full Bayesian solution is impractical, we can resort to using *maximum a posteriori (MAP) estimation*. Here, we search for parameters that maximize the *posterior probability*:

$$\tilde{\boldsymbol{\theta}} = \arg\max_{\boldsymbol{\theta}} \log P(\boldsymbol{\theta} \mid \mathcal{D}).$$

When we have a large amount of data, the posterior is often sharply peaked around its maximum $\tilde{\boldsymbol{\theta}}$. In this case, the integral

$$P(X[M+1] \mid \mathcal{D}) = \int P(X[M+1] \mid \boldsymbol{\theta}) P(\boldsymbol{\theta} \mid \mathcal{D}) d\boldsymbol{\theta}$$

will be roughly $P(X[M+1] \mid \tilde{\boldsymbol{\theta}})$. More generally, we can view the MAP estimate as a way of using the prior to provide *regularization* over the likelihood function:

regularization

$$\begin{aligned}\arg\max_{\boldsymbol{\theta}} \log P(\boldsymbol{\theta} \mid \mathcal{D}) &= \arg\max_{\boldsymbol{\theta}} \log \left(\frac{P(\boldsymbol{\theta}) P(\mathcal{D} \mid \boldsymbol{\theta})}{P(\mathcal{D})} \right) \\ &= \arg\max_{\boldsymbol{\theta}} \left(\log P(\boldsymbol{\theta}) + \log P(\mathcal{D} \mid \boldsymbol{\theta}) \right). \end{aligned} \tag{17.14}$$

That is, $\tilde{\boldsymbol{\theta}}$ is the maximum of a function that sums together the log-likelihood function and $\log P(\boldsymbol{\theta})$. This latter term takes into account the prior on different parameters and therefore biases the parameter estimate away from undesirable parameter values (such as those involving conditional probabilities of 0) when we have few learning instances. When the number of samples is large, the effect of the prior becomes negligible, since $\ell(\boldsymbol{\theta} : \mathcal{D})$ grows linearly with the number of samples whereas the prior does not change.

Because our parameter priors are generally well behaved, MAP estimation is often no harder than maximum likelihood estimation, and is therefore often applicable in practice, even in cases where Bayesian estimation is not. Importantly, however, it does not offer all of the same benefits as a full Bayesian estimation. In particular, it does not attempt to represent the shape of the posterior and thus does not differentiate between a flat posterior and a sharply peaked one. As such, it does not give us a sense of our confidence in different aspects of the parameters, and the predictions do not average over our uncertainty. This approach also suffers from issues regarding representation independence; see box 17.D.

---

representation independence

Box 17.D — Concept: Representation Independence. *One important property we may want of an estimator is* representation independence. *To understand this concept better, suppose that in our thumbtack example, we choose to use a parameter η, so that $P'(X = H \mid \eta) = \frac{1}{1+e^{-\eta}}$. We have that $\eta = \log \frac{\theta}{1-\theta}$ where θ is the parameter we used earlier. Thus, there is a one-to-one correspondence between a choice θ and a choice η. Although one choice of parameters might seem more natural to us than another, there is no formal reason why we should prefer one over the other, since both can represent exactly the same set of distributions.*

More generally, a reparameterization of a given family is a new set of parameter values η in a space Υ and a mapping from the new parameters to the original one, that is, from η to $\boldsymbol{\theta}(\eta)$ so that $P(\cdot \mid \eta)$ in the new parameterization is equal to $P(\cdot \mid \boldsymbol{\theta}(\eta))$ in the original parameterization. In addition, we require that the reparameterization maintain the same set of distributions, that is, for each choice of $\boldsymbol{\theta}$ there is η such that $P(\cdot \mid \eta) = P(\cdot \mid \boldsymbol{\theta})$.

This concept immediately raises the question as to whether the choice of representation can impact our estimates. While we might prefer a particular way of parameterization because it is more intuitive or interpretable, we may not want this choice to bias our estimated parameters.

Fortunately, it is not difficult to see that maximum likelihood estimation is insensitive to reparameterization. If we have two different ways to represent the same family of the distribution, then the distributions *in the family that maximize the likelihood using one parameterization also maximize the likelihood with the other parameterization. More precisely, if $\hat{\eta}$ is MLE, then the matching parameter values $\boldsymbol{\theta}(\hat{\eta})$ are also MLE when we consider the likelihood function in the $\boldsymbol{\theta}$ space. This property is a direct consequence of the fact that the likelihood function is a function of the distribution induced by the parameter values, and not of the actual parameter values.*

The situation with Bayesian inference is subtler. Here, instead of identifying the maximum parameter value, we now perform integration over all possible parameter values. Naively, it seems that such an estimation is more sensitive to the parameterization than MLE, which depends only

on the maximum of the likelihood surface. However, a careful choice of prior can account for the representation change and thereby lead to representation independence. Intuitively, if we consider a reparameterization η with a function $\boldsymbol{\theta}(\eta)$ mapping to the original parameter space, then we would like the prior on η to maintain the probability of events. That is,

$$P(A) = P(\{\boldsymbol{\theta}(\eta) : \eta \in A\}), \ \forall A \subset \Upsilon. \tag{17.15}$$

This constraint implies that the prior over different regions of parameters is maintained. Under this assumption, Bayesian prediction will be identical under the two parameterizations.

We illustrate the notion of a reparameterized prior in the context of a Bernoulli distribution:

Example 17.9

Consider a Beta prior over the parameter θ of a Bernoulli distribution:

$$P(\theta : \alpha_0, \alpha_1) = c\theta^{\alpha_1 - 1}(1-\theta)^{\alpha_0 - 1}, \tag{17.16}$$

where c is the normalizing constant described in definition 17.3. Recall (example 8.5) that the natural parameter for a Bernoulli distribution is

$$\eta = \log \frac{\theta}{1-\theta}$$

with the transformation

$$\theta = \frac{1}{1+e^{-\eta}}, \quad 1-\theta = \frac{1}{1+e^{\eta}}.$$

What is the prior distribution on η? To preserve the probability of events, we want to make sure that for every interval $[a, b]$

$$\int_a^b c\theta^{\alpha_1 - 1}(1-\theta)^{\alpha_0 - 1} d\theta = \int_{\log \frac{a}{1-a}}^{\log \frac{b}{1-b}} P(\eta) d\eta.$$

To do so, we need to perform a change of variables. Using the relation between η and θ, we get

$$d\eta = \frac{1}{\theta(1-\theta)} d\theta.$$

Plugging this into the equation, we can verify that an appropriate prior is:

$$P(\eta) = c\left(\frac{1}{1+e^{-\eta}}\right)^{\alpha_1}\left(\frac{1}{1+e^{\eta}}\right)^{\alpha_0},$$

where c is the same constant as before. This means that the prior on η, when stated in terms of θ, is $\theta^{\alpha_1}(1-\theta)^{\alpha_0}$, in contrast to equation (17.16). At first this discrepancy seems like a contradiction. However, we have to remember that the transformation from θ to η takes the region $[0, 1]$ and stretches it to the whole real line. Thus, the matching prior cannot be uniform. ■

This example demonstrates that a uniform prior, which we consider to be unbiased or uninformative, can seem very different when we consider a different parameterization.

Thus, both MLE and Bayesian estimation (when carefully executed) are representation-independent. This property, unfortunately, does not carry through to MAP estimation. Here we are not interested in the integral over all parameters, but rather in the density of the prior at different values of the parameters. This quantity does change when we reparameterize the prior.

Example 17.10

Consider the setting of example 17.9 and develop the MAP parameters for the priors we considered there. When we use the θ parameterization, we can check that

$$\tilde{\theta} = \arg\max_{\theta} \log P(\theta) = \frac{\alpha_1 - 1}{\alpha_0 + \alpha_1 - 2}.$$

On the other hand,

$$\tilde{\eta} = \arg\max_{\eta} \log P(\eta) = \log \frac{\alpha_1}{\alpha_0}.$$

To compare the two, we can transform $\tilde{\eta}$ to θ representation and find

$$\theta(\tilde{\eta}) = \frac{\alpha_1}{\alpha_0 + \alpha_1}.$$

In other words, the MAP of the η parameterization gives the same predictions as the mean parameterization if we do the full Bayesian inference. ■

Thus, MAP estimation is more sensitive to choices in formalizing the likelihood and the prior than MLE or full Bayesian inference. This suggests that the MAP parameters involve, to some extent, an arbitrary choice. *Indeed, we can bias the MAP toward different solutions if we construct a specific reparameterization where the density is particularly large in specific regions of the parameter space. The parameterization dependency of MAP is a serious caveat we should be aware of.*

17.5 Learning Models with Shared Parameters

In the preceding discussion, we focused on parameter estimation for Bayesian networks with table-CPDs. In this discussion, we made the strong assumption that the parameters for each conditional distribution $P(X_i \mid \boldsymbol{u}_i)$ can be estimated separately from parameters of other conditional distributions. In the Bayesian case, we also assumed that the priors on these distributions are independent. This assumption is a very strong one, which often does not hold in practice. In real-life systems, we often have *shared parameters*: parameters that occur in multiple places across the network. In this section, we discuss how to perform parameter estimation in networks where the same parameters are used multiple times.

shared parameters

Analogously to our discussion of parameter estimation, we can exploit both global and local structure. Global structure occurs when the same CPD is used across multiple variables in the network. This type of sharing arises naturally from the template-based models of chapter 6. Local structure is finer-grained, allowing parameters to be shared even within a single CPD; it arises naturally in some types of structured CPDs. We discuss each of these scenarios in turn, focusing on the simple case of MLE.