4   Parameter Learning

We have assumed so far that the parameters and structure of our probabilistic models were known. This chapter addresses the problem of *learning* or *fitting* model parameters from data.[1] We begin by introducing an approach where we identify the parameters of a model that maximize the likelihood of observing the data. After discussing the limitations of such an approach, we introduce an alternative Bayesian approach, in which we start with a probability distribution over the unknown parameters and then update that distribution based on the observed data using the laws of probability. We then discuss probabilistic models that avoid committing to a fixed number of parameters.

4.1   Maximum Likelihood Parameter Learning

In *maximum likelihood parameter learning*, we attempt to find the parameters of a distribution that maximize the likelihood of observing the data. If θ represents the parameters of a distribution, then the *maximum likelihood estimate* is

$$\hat{\theta} = \arg\max_{\theta} P(D \mid \theta) \tag{4.1}$$

where $P(D \mid \theta)$ is the likelihood that our probability model assigns to the data D occurring when the model parameters are set to θ.[2] We often use the "hat" accent ("$\hat{\ }$") to indicate an estimate of a parameter.

There are two challenges associated with maximum likelihood parameter learning. One is to choose an appropriate probability model by which we define $P(D \mid \theta)$. We often assume that the samples in our data D are *independently and identically distributed*, which means that our samples $D = o_{1:m}$ are drawn from a

distribution $o_i \sim P(\cdot \mid \theta)$ with

$$P(D \mid \theta) = \prod_i P(o_i \mid \theta) \tag{4.2}$$

Probability models could include, for example, the categorical distributions or Gaussian distributions mentioned in earlier chapters.

The other challenge is performing the maximization in equation (4.1). For many common probability models, we can perform this optimization analytically. Others may be difficult. A common approach is to maximize the *log-likelihood*, often denoted as $\ell(\theta)$. Since the log-transformation is monotonically increasing, maximizing the log-likelihood is equivalent to maximizing the likelihood:[3]

$$\hat{\theta} = \arg\max_\theta \sum_i \log P(o_i \mid \theta) \tag{4.3}$$

[3] Although it does not matter whether we maximize the natural logarithm (base e) or the common logarithm (base 10) in this equation, throughout this book we will use $\log(x)$ to mean the logarithm of x with base e.

Computing the sum of log-likelihoods is typically much more numerically stable compared to computing the product of many small probability masses or densities. The remainder of this section will demonstrate how to optimize equation (4.1) for different types of distributions.

4.1.1   Maximum Likelihood Estimates for Categorical Distributions

Suppose that the random variable C represents whether a flight will result in a midair collision, and we are interested in estimating the distribution $P(C)$. Because C is either 0 or 1, it is sufficient to estimate the parameter $\theta = P(c^1)$. What we want to do is infer θ from data D. We have a historical database spanning a decade consisting of m flights with n midair collisions. Our intuition, of course, tells us that a good estimate for θ, given the data D, is n/m. Under the assumption of independence of outcomes between flights, the probability of a sequence of m outcomes in D with n midair collisions is:

$$P(D \mid \theta) = \theta^n (1 - \theta)^{m-n} \tag{4.4}$$

The maximum likelihood estimate $\hat{\theta}$ is the value for θ that maximizes equation (4.4), which is equivalent to maximizing the logarithm of the likelihood:

$$\ell(\theta) = \log\left(\theta^n (1 - \theta)^{m-n}\right) \tag{4.5}$$

$$= n \log \theta + (m - n) \log(1 - \theta) \tag{4.6}$$

We can use the standard technique for finding the maximum of a function by setting the first derivative of ℓ to 0 and then solving for θ. The derivative is given by

$$\frac{\partial}{\partial \theta}\ell(\theta) = \frac{n}{\theta} - \frac{m-n}{1-\theta} \tag{4.7}$$

We can solve for $\hat{\theta}$ by setting the derivative to 0:

$$\frac{n}{\hat{\theta}} - \frac{m-n}{1-\hat{\theta}} = 0 \tag{4.8}$$

After a few algebraic steps, we see that, indeed, $\hat{\theta} = n/m$.

Computing the maximum likelihood estimate for a variable X that can assume k values is also straightforward. If $n_{1:k}$ are the observed counts for the k different values, then the maximum likelihood estimate for $P(x^i \mid n_{1:k})$ is given by

$$\hat{\theta}_i = \frac{n_i}{\sum_{j=1}^{k} n_j} \tag{4.9}$$

4.1.2   Maximum Likelihood Estimates for Gaussian Distributions

In a Gaussian distribution, the log-likelihood of the mean μ and variance σ^2 with m samples is given by

$$\ell(\mu, \sigma^2) \propto -m \log \sigma - \frac{\sum_i (o_i - \mu)^2}{2\sigma^2} \tag{4.10}$$

Again, we can set the derivative to 0 with respect to the parameters and solve for the maximum likelihood estimate:

$$\frac{\partial}{\partial \mu}\ell(\mu, \sigma^2) = \frac{\sum_i (o_i - \hat{\mu})}{\hat{\sigma}^2} = 0 \tag{4.11}$$

$$\frac{\partial}{\partial \sigma}\ell(\mu, \sigma^2) = -\frac{m}{\hat{\sigma}} + \frac{\sum_i (o_i - \hat{\mu})^2}{\hat{\sigma}^3} = 0 \tag{4.12}$$

After some algebraic manipulation, we get

$$\hat{\mu} = \frac{\sum_i o_i}{m} \qquad \hat{\sigma}^2 = \frac{\sum_i (o_i - \hat{\mu})^2}{m} \tag{4.13}$$

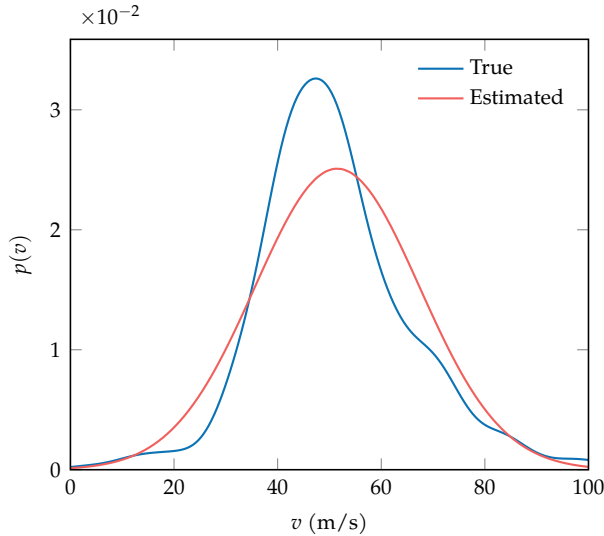Figure 4.1 provides an example of fitting a Gaussian to data.

Figure 4.1. Suppose that we have airspeed measurements $o_{1:m}$ from m aircraft tracks, and we want to fit a Gaussian model. This figure shows a Gaussian with the maximum likelihood estimates $\hat{\mu} = 51.5 \, \text{m/s}$ and $\hat{\sigma} = 15.9 \, \text{m/s}$. The "true" distribution is shown for comparison. In this case, the Gaussian is a fairly reasonable approximation of the true distribution.

4.1.3   Maximum Likelihood Estimates for Bayesian Networks

We can apply maximum likelihood parameter learning to Bayesian networks. Here, we will assume that our network is composed of a set of n discrete variables that we denote as $X_{1:n}$. Our data $D = \{\mathbf{o}_1, \ldots, \mathbf{o}_m\}$ consists of observed samples from those variables. In our network with structure G, r_i is the number of instantiations of X_i, and q_i is the number of instantiations of the parents of X_i. If X_i has no parents, then $q_i = 1$. The jth instantiation of the parents of X_i is denoted as π_{ij}.

The factor table for X_i thus has $r_i q_i$ entries, resulting in a total of $\sum_{i=1}^{n} r_i q_i$ parameters in our Bayesian network. Each parameter is written as θ_{ijk} and determines

$$P(X_i = k \mid \pi_{ij}) = \theta_{ijk} \qquad (4.14)$$

Although there are $\sum_{i=1}^{n} r_i q_i$ parameters, only $\sum_{i=1}^{n} (r_i - 1) q_i$ are independent. We use $\boldsymbol{\theta}$ to represent the set of all parameters.

We use m_{ijk} to represent the number of times $X_i = k$ given parental instantiation j in the data set. Algorithm 4.1 provides an implementation of a function for extracting these counts or statistics from a data set. The likelihood is given in

terms of m_{ijk}:

$$P(D \mid \boldsymbol{\theta}, G) = \prod_{i=1}^{n} \prod_{j=1}^{q_i} \prod_{k=1}^{r_i} \theta_{ijk}^{m_{ijk}} \tag{4.15}$$

Similar to the maximum likelihood estimate for the univariate distribution in equation (4.9), the maximum likelihood estimate in our discrete Bayesian network model is

$$\hat{\theta}_{ijk} = \frac{m_{ijk}}{\sum_{k'} m_{ijk'}} \tag{4.16}$$

Example 4.1 illustrates this process.

```
function sub2ind(siz, x)
    k = vcat(1, cumprod(siz[1:end-1]))
    return dot(k, x .- 1) + 1
end

function statistics(vars, G, D::Matrix{Int})
    n = size(D, 1)
    r = [vars[i].r for i in 1:n]
    q = [prod([r[j] for j in inneighbors(G,i)]) for i in 1:n]
    M = [zeros(q[i], r[i]) for i in 1:n]
    for o in eachcol(D)
        for i in 1:n
            k = o[i]
            parents = inneighbors(G,i)
            j = 1
            if !isempty(parents)
                j = sub2ind(r[parents], o[parents])
            end
            M[i][j,k] += 1.0
        end
    end
    return M
end
```

Algorithm 4.1. A function for extracting the statistics, or counts, from a discrete data set `D`, assuming a Bayesian network with variables `vars` and structure `G`. The data set is an $n \times m$ matrix, where n is the number of variables and m is the number of data points. This function returns an array `M` of length n. The ith component consists of a $q_i \times r_i$ matrix of counts. The `sub2ind(siz, x)` function returns a linear index into an array with dimensions specified by `siz` given coordinates `x`. It is used to identify which parental instantiation is relevant to a particular data point and variable.

4.2   Bayesian Parameter Learning

Bayesian parameter learning addresses some of the drawbacks of maximum likelihood estimation, especially when the amount of data is limited. For example, suppose that our aviation safety database was limited to the events of the past week, and we found no recorded midair collisions. If θ is the probability that a flight results in a midair collision, then the maximum likelihood estimate would

Suppose that we have a small network, $A \rightarrow B \leftarrow C$, and we want to extract the statistics from data matrix `D`. We can use the following code:

```
G = SimpleDiGraph(3)
add_edge!(G, 1, 2)
add_edge!(G, 3, 2)
vars = [Variable(:A,2), Variable(:B,2), Variable(:C,2)]
D = [1 2 2 1; 1 2 2 1; 2 2 2 2]
M = statistics(vars, G, D)
```

The output is an array `M` consisting of these three count matrices, each of size $q_i \times r_i$:

$$\begin{bmatrix} 2 & 2 \end{bmatrix} \quad \begin{bmatrix} 0 & 0 \\ 0 & 0 \\ 2 & 0 \\ 0 & 2 \end{bmatrix} \quad \begin{bmatrix} 0 & 4 \end{bmatrix}$$

We can compute the maximum likelihood estimate by normalizing the rows in the matrices in `M`:

```
θ = [mapslices(x→normalize(x,1), Mi, dims=2) for Mi in M]
```

which produces

$$\begin{bmatrix} 0.5 & 0.5 \end{bmatrix} \quad \begin{bmatrix} \text{NAN} & \text{NAN} \\ \text{NAN} & \text{NAN} \\ 1 & 0 \\ 0 & 1 \end{bmatrix} \quad \begin{bmatrix} 0 & 1 \end{bmatrix}$$

As we can see, the first and second parental instantiations of the second variable B leads to NAN ("not a number") estimates. Because there are no observations of those two parental instantiations in the data, the denominator of equation (4.16) equals zero, making the parameter estimate undefined. Most of the other parameters are not NAN. For example, the parameter $\theta_{112} = 0.5$ means that the maximum likelihood estimate of $P(a^2)$ is 0.5.

Example 4.1.   Using the `statistics` function for extracting the statistics from a data set. Bayesian parameter learning can be used to avoid NAN values, but we must specify a prior.

be $\hat{\theta} = 0$. Believing that there is zero chance of a midair collision is not a reasonable conclusion unless our prior hypothesis was, for example, that all flights were perfectly safe.

The Bayesian approach to parameter learning involves estimating $p(\theta \mid D)$, the posterior distribution over θ given our data D. Instead of obtaining a point estimate $\hat{\theta}$ as in maximum likelihood estimation, we obtain a distribution. This distribution can help us quantify our uncertainty about the true value of θ. We can convert this distribution into a point estimate by computing the expectation:

$$\hat{\theta} = \mathbb{E}_{\theta \sim p(\cdot \mid D)}[\theta] = \int \theta p(\theta \mid D) \, d\theta \tag{4.17}$$

In some cases, however, the expectation may not be an acceptable estimate, as illustrated in figure 4.2. An alternative is to use the *maximum a posteriori* estimate:

$$\hat{\theta} = \arg\max_{\theta} p(\theta \mid D) \tag{4.18}$$

This estimate corresponds to a value of θ that is assigned the greatest density. This is often referred to as the *mode* of the distribution. As shown in figure 4.2, the mode may not be unique.

Bayesian parameter learning can be viewed as inference in a Bayesian network with the structure in figure 4.3, which makes the assumption that the observed variables are conditionally independent of each other. As with any Bayesian network, we must specify $p(\theta)$ and $P(O_i \mid \theta)$. We often use a uniform prior $p(\theta)$. The remainder of this section discusses how to apply Bayesian parameter learning to different models of $P(O_i \mid \theta)$.

4.2.1   Bayesian Learning for Binary Distributions

Suppose we want to learn the parameters of a binary distribution. Here, we will use $P(o^1 \mid \theta) = \theta$. To infer the distribution over θ in the Bayesian network in figure 4.3, we can proceed with the standard method for performing inference
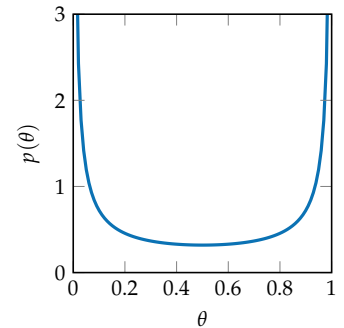

Figure 4.2. An example of a distribution where the expected value of θ is not a good estimate. The expected value of 0.5 has a lower density than occurs at the extreme values of 0 or 1. This distribution happens to be a beta distribution, a type of distribution we will discuss shortly, with parameters $(0.2, 0.2)$.
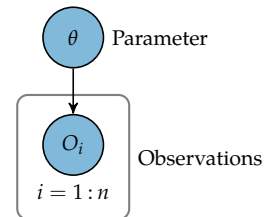

Figure 4.3. Bayesian network representing parameter learning.

discussed in chapter 3. Here, we will assume a uniform prior:

$$p(\theta \mid o_{1:m}) \propto p(\theta, o_{1:m}) \tag{4.19}$$

$$= p(\theta) \prod_{i=1}^{m} P(o_i \mid \theta) \tag{4.20}$$

$$= \prod_{i=1}^{m} P(o_i \mid \theta) \tag{4.21}$$

$$= \prod_{i=1}^{m} \theta^{o_i} (1-\theta)^{1-o_i} \tag{4.22}$$

$$= \theta^n (1-\theta)^{m-n} \tag{4.23}$$

The posterior is proportional to $\theta^n (1-\theta)^{m-n}$, where n is the number of times $O_i = 1$. To find the normalization constant, we integrate

$$\int_0^1 \theta^n (1-\theta)^{m-n} \, d\theta = \frac{\Gamma(n+1)\Gamma(m-n+1)}{\Gamma(m+2)} \tag{4.24}$$

where Γ is the *gamma function*. The gamma function is a real-valued generalization of the factorial. If m is an integer, then $\Gamma(m) = (m-1)!$. Taking normalization into account, we have

$$p(\theta \mid o_{1:m}) = \frac{\Gamma(m+2)}{\Gamma(n+1)\Gamma(m-n+1)} \theta^n (1-\theta)^{m-n} \tag{4.25}$$

$$= \text{Beta}(\theta \mid n+1, m-n+1) \tag{4.26}$$

The *beta distribution* $\text{Beta}(\alpha, \beta)$ is defined by parameters α and β, and curves for this distribution are shown in figure 4.4. The distribution $\text{Beta}(1, 1)$ corresponds to the uniform distribution spanning 0 to 1.

The distribution $\text{Beta}(\alpha, \beta)$ has mean

$$\frac{\alpha}{\alpha + \beta} \tag{4.27}$$

When α and β are both greater than 1, the mode is

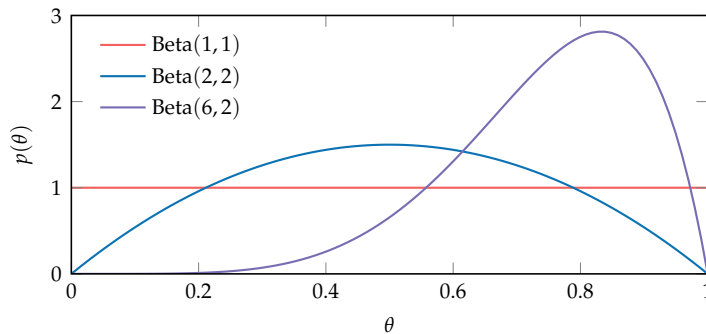$$\frac{\alpha - 1}{\alpha + \beta - 2} \tag{4.28}$$

Figure 4.4. An overlay of several beta probability densities.

Conveniently, if a beta distribution is used as a prior over a parameter of a binary distribution, then the posterior is also a beta distribution. In particular, if the prior is given by $\text{Beta}(\alpha, \beta)$ and we make an observation o_i, then we get a posterior of $\text{Beta}(\alpha + 1, \beta)$ if $o_i = 1$ and $\text{Beta}(\alpha, \beta + 1)$ if $o_i = 0$. Hence, if we started with a prior given by $\text{Beta}(\alpha, \beta)$ and our data showed that there were n collisions out of m flights, then the posterior would be given by $\text{Beta}(\alpha + n, \beta + m - n)$. The α and β parameters in the prior are sometimes called *pseudocounts* because they are treated similarly to the observed counts of the two outcome classes in the posterior, although the pseudocounts need not be integers.

Choosing the prior, in principle, should be done without knowledge of the data used to compute the posterior. Uniform priors often work well in practice, although if expert knowledge is available, then it can be encoded into the prior. For example, suppose that we had a slightly bent coin and we wanted to estimate θ, the probability that the coin would land heads. Before we collected any data by flipping the coin, we would start with a belief θ that is likely to be around 0.5. Instead of starting with a uniform prior $\text{Beta}(1, 1)$, we might use $\text{Beta}(2, 2)$ (shown in figure 4.4), which gives more weight to values near 0.5. If we were more confident in an estimate near 0.5, then we could reduce the variance of the prior by increasing the pseudocounts. The prior $\text{Beta}(10, 10)$ is much more peaked than $\text{Beta}(2, 2)$. In general, however, the importance of the prior diminishes with the amount of data used to compute the posterior. If we observe m flips and n were heads, then the difference between $\text{Beta}(1 + n, 1 + m - n)$ and $\text{Beta}(10 + n, 10 + m - n)$ is negligible if we observe thousands of coin flips.

4.2.2 *Bayesian Learning for Categorical Distributions*

The *Dirichlet distribution*[4] is a generalization of the beta distribution and can be used to estimate the parameters of categorical distributions. Suppose that X is a discrete random variable that takes integer values from 1 to n. We define the parameters of the distribution to be $\theta_{1:n}$, where $P(x^i) = \theta_i$. Of course, the parameters must sum to 1, and so only the first $n - 1$ parameters are independent. The Dirichlet distribution can be used to represent both the prior and the posterior distribution and is parameterized by $\alpha_{1:n}$. The density is given by

$$\mathrm{Dir}(\theta_{1:n} \mid \alpha_{1:n}) = \frac{\Gamma(\alpha_0)}{\prod_{i=1}^n \Gamma(\alpha_i)} \prod_{i=1}^n \theta_i^{\alpha_i - 1} \tag{4.29}$$

where α_0 is used to denote the summation of the parameters $\alpha_{1:n}$.[5] If $n = 2$, then it is easy to see that equation (4.29) is equivalent to the beta distribution.

It is common to use a uniform prior where all the Dirichlet parameters $\alpha_{1:n}$ are set to 1. As with the beta distribution, the parameters in the Dirichlet are often referred to as *pseudocounts*. If the prior over $\theta_{1:n}$ is given by $\mathrm{Dir}(\alpha_{1:n})$ and there are m_i observations of $X = i$, then the posterior is given by

$$p(\theta_{1:n} \mid \alpha_{1:n}, m_{1:n}) = \mathrm{Dir}(\theta_{1:n} \mid \alpha_1 + m_1, \dots, \alpha_n + m_n) \tag{4.30}$$

The distribution $\mathrm{Dir}(\alpha_{1:n})$ has a mean vector whose ith component is

$$\frac{\alpha_i}{\sum_{j=1}^n \alpha_j} \tag{4.31}$$

When $\alpha_i > 1$, the ith component of the mode is

$$\frac{\alpha_i - 1}{\sum_{j=1}^n \alpha_j - n} \tag{4.32}$$

As we have seen, Bayesian parameter estimation is straightforward for binary and discrete random variables because it involves simply counting the various outcomes in the data. Bayes' rule can be used to infer the distribution over the parameters for other parametric distributions. Depending on the choice of prior and the form of the parametric distribution, calculating the posterior over the space of parameters also might be done analytically.

[4] This distribution is named after the German mathematician Johann Peter Gustav Lejeune Dirichlet (1805–1859).

[5] See appendix B for plots of Dirichlet distribution densities for different parameters.

4.2.3   *Bayesian Learning for Bayesian Networks*

We can apply Bayesian parameter learning to discrete Bayesian networks. The prior over the Bayesian network parameters $\boldsymbol{\theta}$ can be factorized as follows:

$$p(\boldsymbol{\theta} \mid G) = \prod_{i=1}^{n} \prod_{j=1}^{q_i} p(\boldsymbol{\theta}_{ij}) \tag{4.33}$$

where $\boldsymbol{\theta}_{ij} = (\theta_{ij1}, \ldots, \theta_{ijr_i})$. The prior $p(\boldsymbol{\theta}_{ij})$, under some weak assumptions, can be shown to follow a Dirichlet distribution $\text{Dir}(\alpha_{ij1}, \ldots, \alpha_{ijr_i})$. Algorithm 4.2 provides an implementation for creating a data structure holding α_{ijk}, where all entries are 1, corresponding to a uniform prior.

After observing data in the form of m_{ijk} counts (as introduced in section 4.1.3), the posterior is then

$$p(\boldsymbol{\theta}_{ij} \mid \alpha_{ij}, m_{ij}) = \text{Dir}(\boldsymbol{\theta}_{ij} \mid \alpha_{ij1} + m_{ij1}, \ldots, \alpha_{ijr_i} + m_{ijr_i}) \tag{4.34}$$

similar to equation (4.30). Example 4.2 demonstrates this process.

```
function prior(vars, G)
    n = length(vars)
    r = [vars[i].r for i in 1:n]
    q = [prod([r[j] for j in inneighbors(G,i)]) for i in 1:n]
    return [ones(q[i], r[i]) for i in 1:n]
end
```

Algorithm 4.2. A function for generating a prior α_{ijk} where all entries are 1. The array of matrices that this function returns takes the same form as the statistics generated by algorithm 4.1. To determine the appropriate dimensions, the function takes as input the list of variables `vars` and structure `G`.

We can compute the parameters of the posterior associated with a Bayesian network through simple addition of the prior parameters and counts (equation (4.34)). If we use the matrix of counts `M` obtained in example 4.1, we can add it to the matrices of prior parameters `α = prior(vars, G)` to obtain the set of posterior parameters `M + α`:

Example 4.2. Computing the posterior parameters in a Bayesian network. Note that unlike example 4.1, here we do not have NAN values.

$$\begin{bmatrix} 3 & 3 \end{bmatrix} \qquad \begin{bmatrix} 1 & 1 \\ 1 & 1 \\ 3 & 1 \\ 1 & 3 \end{bmatrix} \qquad \begin{bmatrix} 1 & 5 \end{bmatrix}$$

4.3   Nonparametric Learning

The previous two sections assumed that the probabilistic model was of a fixed form and that a fixed set of parameters were to be learned from the data. An alternative approach is based on *nonparametric* methods in which the number of parameters scales with the amount of data. A common nonparametric method is *kernel density estimation* (algorithm 4.3). Given observations $o_{1:m}$, kernel density estimation represents the density as follows:

$$p(x) = \frac{1}{m} \sum_{i=1}^{m} \phi(x - o_i) \qquad (4.35)$$

where ϕ is a *kernel function*, which integrates to 1. The kernel function is used to assign greater density to values near the observed data points. A kernel function is generally symmetric, meaning that $\phi(x) = \phi(-x)$. A common kernel is the zero-mean Gaussian distribution. When such a kernel is used, the standard deviation is often referred to as the *bandwidth*, which can be tuned to control the smoothness of the density function. Larger bandwidths generally lead to smoother densities. Bayesian methods can be applied to the selection of the appropriate bandwidth based on the data. The effect of the bandwidth choice is shown in figure 4.5.

```
gaussian_kernel(b) = x→pdf(Normal(0,b), x)

function kernel_density_estimate(φ, O)
    return x → sum([φ(x - o) for o in O])/length(O)
end
```

Algorithm 4.3.   The method `gaussian_kernel` returns a zero-mean Gaussian kernel $\phi(x)$ with bandwidth `b`. Kernel density estimation is also implemented for a kernel `φ` and list of observations `O`.

4.4   Learning with Missing Data

When learning the parameters of our probabilistic model we may have *missing* entries in our data.[6] For example, if we are conducting a survey, some respondents may decide to skip a question. Table 4.1 shows an example of a data set with missing entries involving three binary variables: *A*, *B*, and *C*. One approach to handling missing data is to discard all the instances that are *incomplete*, where there are one or more missing entries. Depending on how much of the data is missing, we might have to discard much of it. In table 4.1, we would have to discard all but one of the rows, which can be wasteful.

[6] Learning with missing data is the subject of a large body of literature. A comprehensive introduction and review is provided by G. Molenberghs, G. Fitzmaurice, M. G. Kenward, A. Tsiatis, and G. Verbeke, eds., *Handbook of Missing Data Methodology*. CRC Press, 2014.
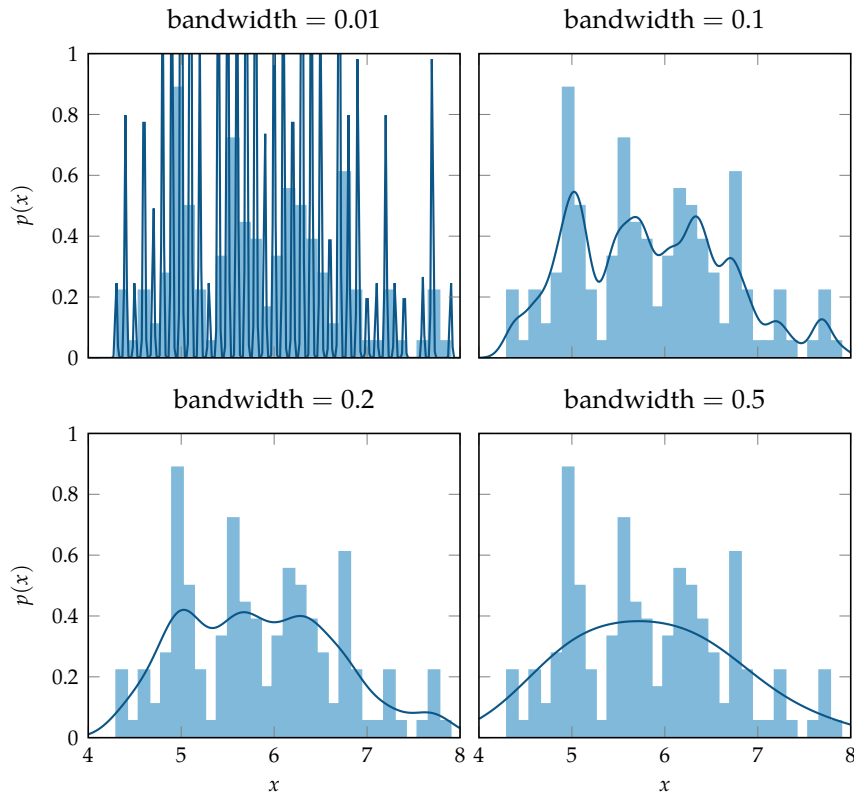
Figure 4.5. Kernel density estimation applied to the same data set using zero-mean Gaussian kernels with different bandwidths. The histogram in blue shows the underlying data set frequencies, and the black lines indicate the probability density from kernel density estimation. Larger bandwidths smooth out the estimate, whereas smaller bandwidths can overfit to specific samples.

We can learn model parameters from missing data using either a maximum likelihood or a Bayesian approach. If taking a Bayesian maximum a posteriori approach, we want to find the estimate

$$\hat{\theta} = \arg\max_{\theta} p(\theta \mid D_{\text{obs}}) \tag{4.36}$$

$$= \arg\max_{\theta} \sum_{D_{\text{mis}}} p(\theta \mid D_{\text{obs}}, D_{\text{mis}}) P(D_{\text{mis}} \mid D_{\text{obs}}) \tag{4.37}$$

A	B	C
1	1	0
?	1	1
1	?	?
?	?	?

Table 4.1. Example of data consisting of four instances with six missing entries.

where D_{obs} and D_{mis} consist of all the observed and missing data, respectively. If the data is continuous, then the sum would be replaced by an integral. The marginalization over the missing data can be computationally expensive. The same marginalization also affects the computational tractability of a Bayesian approach.

This section discusses two general approaches for learning with missing data without having to enumerate over all the possible combinations of missing values. The first involves learning the distribution parameters using predicted values of the missing entries. The second involves an iterative approach for improving our parameter estimates.

We will focus on the context where data is *missing at random*, meaning that the probability that an entry is missing is conditionally independent of its value, given the values of the observed variables. An example of a situation that does not adhere to this assumption might include radar data containing measurements of the distance to a target, but the measurement may be missing either due to noise or because the target is beyond the sensing range. The fact that an entry is missing is an indication that the value is more likely to be high. Accounting for this form of missingness requires different models and algorithms from what we discuss here.[7]

4.4.1 Data Imputation

An alternative to discarding incomplete instances is to impute the values of missing entries. *Data imputation* is the process of inferring values for missing entries. One way to view imputation is as an approximation of equation (4.37), where we find

$$\hat{D}_{\text{mis}} = \arg\max_{D_{\text{mis}}} p(D_{\text{mis}} \mid D_{\text{obs}}) \tag{4.38}$$

[7] Different *missingness mechanisms* and associated inference techniques are reviewed by R. J. A. Little and D. B. Rubin, *Statistical Analysis with Missing Data*, 3rd ed. Wiley, 2020.

Once we have the imputed missing values, we can use that data to produce a maximum posteriori estimate:

$$\hat{\theta} = \arg\max_{\theta} p(\theta \mid D_{\mathrm{obs}}) \approx \arg\max_{\theta} p(\theta \mid D_{\mathrm{obs}}, \hat{D}_{\mathrm{mis}}) \qquad (4.39)$$

or, alternatively, we can take a maximum likelihood approach.

Solving equation (4.38) may still be computationally challenging. One simple approach for discrete data sets is to replace missing entries with the most commonly observed value, called the *marginal mode*. For example, in table 4.1, we might replace all the missing values for A with its marginal mode of 1.

Continuous data often lacks duplicates. However, we can fit a distribution to continuous values and then use the mode of the resulting distribution. For example, we might fit a Gaussian distribution to the data in table 4.2, and then fill in the missing entries with the mean of the observed values associated with each variable. The top-left plot in figure 4.6 illustrates the effect of this approach on two-dimensional data. The red lines show how values with missing first or second components are paired with their imputed counterparts. We can then use the observed and imputed data to arrive at a maximum likelihood estimate of the parameters of a joint Gaussian distribution. As we can see, this method of imputation does not always produce sensible predictions and the learned model is quite poor.

We can often do better if we account for the probabilistic relationships between the observed and unobserved variables. In figure 4.6, there is clearly correlation between the two variables; hence, knowing the value of one variable can help predict the value of the other variable. A common approach to imputation, called *nearest-neighbor imputation*, is to use the values associated with the instance that is nearest with respect to a distance measure defined on the observed variables. The top-right plot in figure 4.6 uses the Euclidean distance for imputation. This approach tends to lead to better imputations and learned distributions.

An alternative approach is to fit a distribution to the fully observed data and then use that distribution to infer the missing values. We can use the inference algorithms from the previous chapter to perform this inference. For example, if our data is discrete and we can assume a Bayesian network structure, we can use variable elimination or Gibbs sampling to produce a distribution over the missing variables for an instance from the observed variables. From this distribution, we might use the mean or mode to impute the missing values. Alternatively, we can

A	B	C
−6.5	0.9	4.2
?	4.4	9.2
7.8	?	?
?	?	?

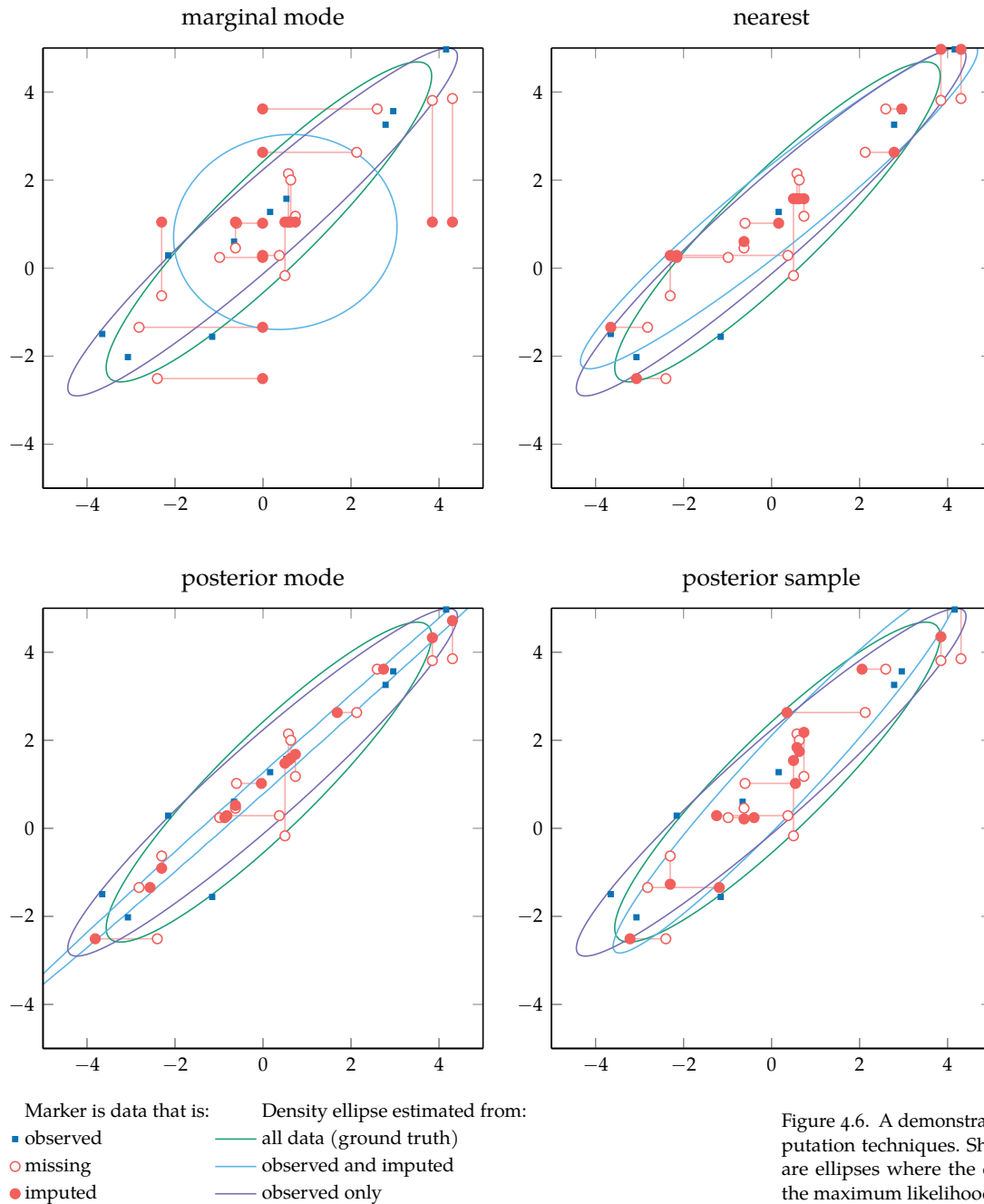Table 4.2. Example of data with continuous values.

Figure 4.6. A demonstration of imputation techniques. Shown here are ellipses where the density of the maximum likelihood estimate of the joint distribution equals 0.02.

pull a sample from this distribution. If our data is continuous and we can assume that the data is jointly Gaussian, we can use algorithm 3.11 to infer the posterior distribution. The bottom plots in figure 4.6 demonstrate imputation using these posterior mode and posterior sampling approaches.

4.4.2   Expectation-Maximization

The *expectation-maximization* (*EM*) category of approaches involves iterative improvement of the distribution parameter estimate $\hat{\theta}$.[8] We begin with an initial $\hat{\theta}$, which may be a guess, randomly sampled from a prior distribution over distribution parameters, or estimated using one of the methods discussed in section 4.4.1. At each iteration, we perform a two-step process to update $\hat{\theta}$.

The first step is called the *expectation step* (*E-step*), where we use the current estimate of θ to infer completions of the data. For example, if we are modeling our data using a discrete Bayesian network, we can use one of our inference algorithms to infer a distribution over the missing entries for each instance. When extracting the counts, we apply a weighting proportional to the likelihood of the completions as shown in example 4.3. In cases where there are many missing variables, there may be too many possible completions to practically enumerate, making a sampling-based approach attractive. We may also want to use sampling as an approximation method when our variables are continuous.

The second step is called the *maximization step* (*M-step*), where we attempt to find a new $\hat{\theta}$ that maximizes the likelihood of the completed data. If we have a discrete Bayesian network with the weighted counts in the form shown in example 4.3, then we can perform the same maximum likelihood estimate as discussed earlier in this chapter. Alternatively, we can use a maximum a posteriori estimate if we want to incorporate a prior.

This approach is not guaranteed to converge to model parameters that maximize the likelihood of the observed data, but it can work well in practice. To reduce the risk of the algorithm converging to only a local optimum, we can run the algorithm to convergence from many different initial points in the parameter space. We simply choose the resulting parameter estimate in the end that maximizes likelihood.

Expectation-maximization can even be used to impute values for variables that are not observed at all in the data. Such variables are called *latent variables*. To illustrate, suppose we have a Bayesian network $Z \rightarrow X$, where X is continuous

[8] Expectation-maximization was introduced by A. P. Dempster, N. M. Laird, and D. B. Rubin, "Maximum Likelihood from Incomplete Data via the EM Algorithm," *Journal of the Royal Statistical Society, Series B (Methodological)*, vol. 39, no. 1, pp. 1–38, 1977.

Suppose that we have a binary Bayesian network with $A \rightarrow B$. We start by assuming that $\hat{\theta}$ implies

$$P(a^1) = 0.5 \qquad P(b^1 \mid a^0) = 0.2 \qquad P(b^1 \mid a^1) = 0.6$$

Using these parameters, we can expand the data set with missing values (left) to a weighted data set with all possible individual completions (right):

A	B
1	1
0	1
0	?
?	0

A	B	weight
1	1	1
0	1	1
0	0	$1 - P(b^1 \mid a^0) = 0.8$
0	1	$P(b^1 \mid a^0) = 0.2$
0	0	$\alpha P(a^0)P(b^0 \mid a^0) = \alpha 0.4 = 2/3$
1	0	$\alpha P(a^1)P(b^0 \mid a^1) = \alpha 0.2 = 1/3$

The α in the calculation here is a normalization constant, which enforces that each instance is expanded to instances whose weights sum to 1. The count matrices are then

$$\begin{bmatrix} (2+2/3) & (1+1/3) \end{bmatrix} \qquad \begin{bmatrix} (0.8 + 2/3) & 1.2 \\ 1/3 & 1 \end{bmatrix}$$

and Z is discrete and can take on one of three values. Our model assumes $p(x \mid z)$ is conditional Gaussian. Our data set contains only values for X, but none for Z. We start with an initial $\hat{\theta}$ and use it to infer a probability distribution over the values of Z, given the value of X for each instance. The distribution over entry completions are then used to update our estimate of the parameters of $P(Z)$ and $P(X \mid Z)$ as illustrated in example 4.4. We iterate to convergence, which often occurs very quickly. The parameters that we obtain in this example define a Gaussian mixture model, which was introduced in section 2.2.2.

4.5   Summary

- Parameter learning involves inferring the parameters of a probabilistic model from data.

- A maximum likelihood approach to parameter learning involves maximizing a likelihood function, which can be done analytically for some models.

- A Bayesian approach to parameter learning involves inferring a probability distribution over the underlying parameter using Bayes' rule.

- The beta and Dirichlet distributions are examples of Bayesian priors that are easily updated with evidence.

- In contrast with parametric learning, which assumes a fixed parameterization of a probability model, nonparametric learning uses representations that grow with the amount of data.

- We can approach the problem of learning parameters from missing data using methods such as data imputation or expectation-maximization, where we make inferences based on observed values.

4.6   Exercises

Exercise 4.1. Suppose that Anna is shooting basketball free throws. Before we see her play, we start with an independent uniform prior over the probability that she successfully makes a basket per shot. We observe her take three shots, with two of them resulting in successful baskets. What is the probability that we assign to her making the next basket?
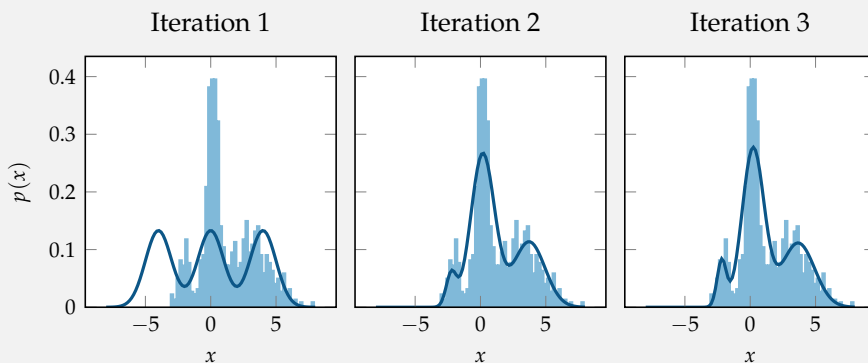
We have a Bayesian network $Z \rightarrow X$, where Z is a discrete latent variable with three values and X is continuous with $p(x \mid z)$ modeled as a conditional Gaussian. Hence, we have parameters defining $P(z^1)$, $P(z^2)$, and $P(z^3)$, as well as μ_i and σ_i for each of the three Gaussian distributions associated with different values of Z. In this example, we use an initial parameter vector $\hat{\theta}$ that specifies $P(z^i) = 1/3$ and $\sigma_i = 1$ for all i. We spread out the means with $\mu_1 = -4$, $\mu_2 = 0$, and $\mu_3 = 4$.

Suppose our first instance in our data has $X = 4.2$. We want to infer the distribution over Z for that instance:

$$P(z^i \mid X = 4.2) = \frac{P(z^i)\,\mathcal{N}(4.2 \mid \mu_i, \sigma_i^2)}{\sum_j P(z^j)\,\mathcal{N}(4.2 \mid \mu_j, \sigma_j^2)}$$

We compute this distribution for all the instances in our data set. For the weighted completions, we can obtain a new estimate for $\hat{\theta}$. We estimate $P(z^i)$ by taking the mean across the instances in our data set. To estimate μ_i and σ_i, we use the mean and standard deviation of the values for X over the instances in our data set, weighted by the probability of z^i associated with the various instances.

We repeat the process until convergence occurs. The plot here shows three iterations. The histogram was generated from the values of X. The dark blue function indicates the inferred density. By the third iteration, our parameters of the Gaussian mixture model closely represent the data distribution.

Example 4.4. Expectation maximization applied to learning the parameters of a Gaussian mixture model.

© 2022 Massachusetts Institute of Technology, shared under a Creative Commons CC-BY-NC-ND license.
2024-02-06 20:54:49-08:00, comments to bugs@algorithmsbook.com

Solution: We denote the probability of making a basket as θ. Since we start with a uniform prior Beta$(1, 1)$ and observe two baskets and one miss, our posterior is then Beta$(1 + 2, 1 + 1) =$ Beta$(3, 2)$. We want to compute the probability of a basket as follows:

$$P(\text{basket}) = \int P(\text{basket} \mid \theta) \, \text{Beta}(\theta \mid 3, 2) \, d\theta = \int \theta \, \text{Beta}(\theta \mid 3, 2) \, d\theta$$

This expression is just the expectation (or mean) of a beta distribution, which gives us $P(\text{basket}) = 3/5$.

Exercise 4.2. Consider a continuous random variable X that follows the *Laplace distribution* parameterized by μ and b, with density

$$p(x \mid \mu, b) = \frac{1}{2b} \exp\left(-\frac{|x - \mu|}{b}\right)$$

Compute the maximum likelihood estimates of the parameters of a Laplace distribution given a data set D of m independent observations $x_{1:m}$. Note that $\partial|u|/\partial x = \text{sign}(u)\partial u/\partial x$, where the sign function returns the sign of its argument.

Solution: Since the observations are independent, we can write the log-likelihood function as the summation:

$$\ell(\mu, b) = \sum_{i=1}^{m} \log\left[\frac{1}{2b} \exp\left(-\frac{|x_i - \mu|}{b}\right)\right]$$

$$= -\sum_{i=1}^{m} \log 2b - \sum_{i=1}^{m} \frac{|x_i - \mu|}{b}$$

$$= -m \log 2b - \frac{1}{b} \sum_{i=1}^{m} |x_i - \mu|$$

To obtain the maximum likelihood estimates of the true parameters μ and b, we take the partial derivatives of the log-likelihood with respect to each of the parameters, set them to zero, and solve for each parameter. First, we solve for $\hat{\mu}$:

$$\frac{\partial}{\partial \mu} \ell(\mu, b) = \frac{1}{\hat{b}} \sum_{i=1}^{m} \text{sign}(x_i - \mu)$$

$$0 = \frac{1}{\hat{b}} \sum_{i=1}^{m} \text{sign}(x_i - \hat{\mu})$$

$$0 = \sum_{i=1}^{m} \text{sign}(x_i - \hat{\mu})$$

$$\hat{\mu} = \text{median}(x_{1:m})$$

Now, solving for $\hat{b}$:

$$\frac{\partial}{\partial b}\ell(\mu, b) = -\frac{m}{b} + \frac{1}{b^2}\sum_{i=1}^{m}|x_i - \hat{\mu}|$$

$$0 = -\frac{m}{\hat{b}} + \frac{1}{\hat{b}^2}\sum_{i=1}^{m}|x_i - \hat{\mu}|$$

$$\frac{m}{\hat{b}} = \frac{1}{\hat{b}^2}\sum_{i=1}^{m}|x_i - \hat{\mu}|$$

$$\hat{b} = \frac{1}{m}\sum_{i=1}^{m}|x_i - \hat{\mu}|$$

Thus, the maximum likelihood estimates for the parameters of a Laplace distribution are $\hat{\mu}$, the median of the observations, and $\hat{b}$, the mean of absolute deviations from the median.

Exercise 4.3. This question explores the application of maximum likelihood estimation to *censored data*, where some measurements are only partially known. Suppose that we are building electric motors for a quadcopter drone, and we want to produce a model of how long they last until failure. Although there may be more suitable distributions for modeling the reliability of components,[9] we will use an *exponential distribution* parameterized by λ with probability density function $\lambda\exp(-\lambda x)$ and cumulative distribution function $1 - \exp(-\lambda x)$. We fly five drones. Three have motor failures after 132 hours, 42 hours, and 89 hours. We stopped testing the other two after 200 hours without failure; we do not know their failure times; we just know that they are greater than 200 hours. What is the maximum likelihood estimate for λ given this data?

[9] K. S. Trivedi and A. Bobbio, *Reliability and Availability Engineering*. Cambridge University Press, 2017.

Solution: This problem has $n = 3$ fully observed measurements and $m = 2$ censored measurements. We use t_i to represent the ith fully observed measurement and $\underline{t}_j$ to represent the jth censored measurement. The likelihood of a single measurement above $\underline{t}_j$ is the complement of the cumulative distribution function, which is simply $\exp(-\lambda\underline{t}_j)$. Hence, the likelihood of the data is

$$\left(\prod_{i=1}^{n}\lambda e^{-\lambda t_i}\right)\left(\prod_{j=1}^{m}e^{-\lambda\underline{t}_j}\right)$$

We use our standard approach of maximizing the log-likelihood, which is given by

$$\ell(\lambda) = \sum_{i=1}^{n}(\log\lambda - \lambda t_i) + \sum_{j=1}^{m}-\lambda\underline{t}_j$$

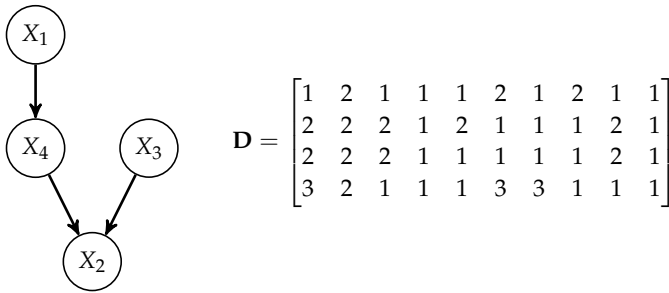The derivative with respect to λ is

$$\frac{\partial\ell}{\partial\lambda} = \frac{n}{\lambda} - \sum_{i=1}^{n}t_i - \sum_{j=1}^{m}\underline{t}_j$$

Setting this derivative to 0, we can solve for λ to obtain the maximum likelihood estimate:

$$\hat{\lambda} = \frac{n}{\sum_{i=1}^{n} t_i + \sum_{j=1}^{m} \underline{t}_j} = \frac{3}{132 + 42 + 89 + 200 + 200} \approx 0.00452$$

The mean of the exponential distribution is $1/\lambda$, making the mean in our problem 221 hours.

Exercise 4.4. We have a Bayesian network where the variables $X_{1:3}$ can take on values in $\{1, 2\}$ and X_4 can take on values in $\{1, 2, 3\}$. Given the data set D of observations $\mathbf{o}_{1:m}$, as illustrated here, generate the maximum likelihood estimates of the associated conditional distribution parameters θ.

$$\mathbf{D} = \begin{bmatrix} 1 & 2 & 1 & 1 & 1 & 2 & 1 & 2 & 1 & 1 \\ 2 & 2 & 2 & 1 & 2 & 1 & 1 & 1 & 2 & 1 \\ 2 & 2 & 2 & 1 & 1 & 1 & 1 & 1 & 2 & 1 \\ 3 & 2 & 1 & 1 & 1 & 3 & 3 & 1 & 1 & 1 \end{bmatrix}$$

Solution: We can generate count matrices $\mathbf{M}_i$ of size $q_i \times r_i$ for each node by iterating through the data set and storing the counts. We then normalize each row in the count matrices to yield the matrices containing the maximum likelihood estimates of the parameters:

$$\mathbf{M}_1 = \begin{bmatrix} 7 & 3 \end{bmatrix} \qquad \mathbf{M}_2 = \begin{bmatrix} 3 & 1 \\ 0 & 0 \\ 2 & 0 \\ 0 & 2 \\ 0 & 1 \\ 0 & 1 \end{bmatrix} \qquad \mathbf{M}_3 = \begin{bmatrix} 6 & 4 \end{bmatrix} \qquad \mathbf{M}_4 = \begin{bmatrix} 5 & 0 & 2 \\ 1 & 1 & 1 \end{bmatrix}$$

$$\hat{\theta}_1 = \begin{bmatrix} 0.7 & 0.3 \end{bmatrix} \qquad \hat{\theta}_2 = \begin{bmatrix} 0.75 & 0.25 \\ \text{NAN} & \text{NAN} \\ 1.0 & 0.0 \\ 0.0 & 1.0 \\ 0.0 & 1.0 \\ 0.0 & 1.0 \end{bmatrix} \qquad \hat{\theta}_3 = \begin{bmatrix} 0.6 & 0.4 \end{bmatrix} \qquad \hat{\theta}_4 \approx \begin{bmatrix} 0.71 & 0.0 & 0.29 \\ 0.33 & 0.33 & 0.34 \end{bmatrix}$$

Exercise 4.5. We have a biased coin, and we want to estimate the Bernoulli parameter ϕ that specifies the probability the coin lands on heads. If the first toss lands on heads ($o_1 = 1$), answer the following questions:

- What is the maximum likelihood estimate of ϕ?

- Using a uniform prior, what is the maximum a posteriori estimate of ϕ?
- Using a uniform prior, what is the expectation of our posterior distribution over ϕ?

Solution: Since our first toss lands on heads, we have $n = 1$ successes and $m = 1$ trials.

- The maximum likelihood estimate of ϕ is $n/m = 1$.
- Using a uniform $\mathrm{Beta}(1,1)$ prior, the posterior distribution is $\mathrm{Beta}(1 + n, 1 + m - n) = \mathrm{Beta}(2,1)$. The maximum a posteriori estimate of ϕ or mode of the posterior distribution is

$$\frac{\alpha - 1}{\alpha + \beta - 2} = \frac{2 - 1}{2 + 1 - 2} = 1$$

- The mean of the posterior distribution is

$$\frac{\alpha}{\alpha + \beta} = \frac{2}{2 + 1} = \frac{2}{3}$$

Exercise 4.6. Suppose we are given the following data set, with one missing value. What is the value that will be imputed using marginal mode imputation, assuming that the marginal distribution is a Gaussian? What is the value that will be imputed using nearest-neighbor imputation?

X_1	X_2
0.5	1.0
?	0.3
−0.6	−0.3
0.1	0.2

Solution: Assuming that the marginal distribution over X_1 is a Gaussian, we can compute the marginal mode, which is the mean parameter of the Gaussian distribution:

$$\mu = \frac{1}{m} \sum_{i=1}^{m} x_i = \frac{0.5 - 0.6 + 0.1}{3} = 0$$

Thus, for marginal mode imputation, the missing value will be set to 0. For nearest-neighbor imputation, the nearest sample to $X_2 = 0.3$ is the fourth sample, so the missing value will be set to 0.1.

Exercise 4.7. Suppose we are given a data set over two variables $X_{1:2}$, with several missing values. We assume that $X_{1:2}$ are jointly Gaussian and use the fully-observed samples to fit the following distribution:

$$\begin{bmatrix} X_1 \\ X_2 \end{bmatrix} \sim \mathcal{N}\left(\begin{bmatrix} 5 \\ 2 \end{bmatrix}, \begin{bmatrix} 4 & 1 \\ 1 & 2 \end{bmatrix} \right)$$

What is the value that will be imputed for X_1 for the sample $X_2 = 1.5$ using posterior mode imputation? What distribution do we need to sample from for posterior sample imputation?

Solution: Since we assumed that $X_{1:2}$ are jointly Gaussian, the posterior distribution over X_1 given X_2 is also Gaussian, and its mode is the mean parameter of the posterior distribution. We can compute the mean of the posterior distribution as follows:

$$p(x_1 \mid x_2) = \mathcal{N}\left(x_1 \mid \mu_{x_1 \mid x_2}, \sigma^2_{x_1 \mid x_2}\right)$$

$$\mu_{x_1 \mid x_2 = 1.5} = 5 + (1)(2)^{-1}(1.5 - 2) = 4.75$$

Thus, for posterior mode imputation, the missing value will be set to 4.75. For posterior sample imputation, we will sample a value $X_1 \sim \mathcal{N}(4.75, 3.5)$.