Readings #8: Learning I: Introduction & Parameter Estimation*

Daphne Koller
Stanford University

Nir Friedman
Hebrew University

1 General Introduction

We now move to a completely different topic. So far, we have assumed that there is an expert who is willing to sit with us for hours to design a BN. This is not always the case. In some domains, there is no-one who has the necessary expertise. In other domains, the amount of knowledge required is just too large or the expert's time is too valuable. In yet others, the domain changes over time, and we can't expect an expert to sit and redesign the network every few weeks.

However, we often have access to data; e.g., in a medical domain, we may have access to a lot of patients' medical records. It would be nice if we could automatically discover the patterns in the data and use them to construct a model. Thus, we are interesting in the following task:

$$D = \{\mathbf{d}[1], \ldots, \mathbf{d}[N]\} \longrightarrow BN$$

For example, if we are trying to induce the Alarm network from data, our starting point might be something of the form:

$$\begin{array}{r|ccccc}
 & B & E & A & C & R \\
\mathbf{d}[1] & b^0 & e^1 & a^0 & c^1 & r^0 \\
\mathbf{d}[2] & b^1 & e^0 & a^0 & c^0 & r^0 \\
\mathbf{d}[3] & b^0 & e^0 & a^1 & c^1 & r^0 \\
\mathbf{d}[4] & b^0 & e^1 & a^1 & c^0 & r^1 \\
\cdots & & & & &
\end{array}$$

There are several variants of this problem, with different applications:

	fully observed	partially observed
fixed structure	(1)	(3)
fixed variables	(2)	(4)
hidden variables	—	(5)

(1) The easiest problem. The network structure is specified, and the inducer only needs to estimate the parameters. The problem is well-understood and the algorithms are computationally efficient. Despite its simplicity, this problem is still extremely useful, because numbers are very hard to elicit from people. Also, it forms the basis for everything else.

(2) In this case, the inducer is given the set of variables in the model, and needs to select the arcs between them and estimate parameters. This problem is useful for a variety of applications. In general, when we are given a new domain with no (available) domain expert, and want to get all of the benefits of a BN model. It is also useful in data-mining style applications, where we have masses of data that we want to interpret. In addition to providing a model that will allow us to predict

*This is based on draft of a text book, some of the references and citations are missing

Figure 1: A simple thumbtack tossing experiment

behavior of cases that we haven't seen, the structure also gives the expert some indication of what attributes are correlated. The algorithms here are combinatorially expensive. They essentially reduce to a heuristic search over the space of BN structures.

Missing data: Here, some data cases might have question marks in some entries, for values that were not observed. Dealing with such data is crucial in many circumstances, because our data cases often have missing values. For example, in medical settings, a patient rarely gets all of the possible tests. Also, in domains were getting full observations is easy, we wouldn't need to have a BN to give us probabilities over the things we don't observe. The ability to deal with partially observed data cases is crucial to adapting the BN after it's operational. Applications: insurance, credit cards. This type of problem is also useful for hidden variables, i.e., ones that are impossible to observe, e.g., a user's taste in TV shows. As we will see, this capacity is useful for automatic discovery of clusters in data, e.g., the clusters that we saw of users who watch the same TV shows.

(3) This task is not so easy, but fairly well-understood. It is a numerical optimization task. Unfortunately, all known algorithms involve nonlinear optimization and multiple calls to Bayesian network inference as a subroutine, making this process difficult for large networks.

(4) This is the problem that we would really like to solve, since it both discovers structure and deals with missing data. This problem encompasses two subproblems, each of which is already expensive to solve. Thus, this process is even more expensive. Until last year, it was so expensive that no-one could do it. But two years ago, a new algorithm was invented that makes this task feasible, although still expensive.

(5) Discovering hidden variables is very interesting, can even lead to scientific discovery or to comparing several "scientific" theories which hypothesize certain common causes for correlations (Tobacco institute, Bell curve). This task is very hard, and we have only some very preliminary answers.

2 Single parameter

Let's start with the simplest problem: parameter learning for a single variable. Surprisingly, this problem already contains some interesting issues that we will need to tackle.

Imagine that we have a thumbtack, and we conduct an experiment whereby we flip the thumbtack in the air, and it comes to land as either heads or tails, as in Figure 1. We assume, as usual, that the different tosses are independent, and that, in each one, the probability of the thumbtack landing heads is some real number θ. Our goal is to estimate θ.

2.1 Maximum likelihood

Assume that we toss the thumbtack 100 times, of which 35 come up heads. What is our estimate for θ? Intuitively, it seems obvious that the answer is 0.35: had θ had been 0.1, our chances of seeing 35/100 heads would be much lower.

Let's formalize this intuition. First, let's state our assumptions. We have a set of instances $\mathbf{d}[1], \ldots, \mathbf{d}[M]$ such that

- Each is sampled from the same distribution
- Each is sampled independently from the rest

Such samples are called *IID — Independent & Identically Distributed.*

We assume that the distribution that generated the data is parameterized by some parameter vector $\boldsymbol{\theta}$. Thus, it defines a probability $P(\mathbf{d} \mid \boldsymbol{\theta})$ over data cases. In the case of the thumbtack, each data case $X[m]$ takes value heads ($X[m] = x^1$) or tails $X[m] = x^0$). The single parameter θ is the probability with which an individual toss lands heads.

Our task is to find a good value for the parameter θ. How good is a particular value θ? A parameter is good if it predicts the data well. In other words, if the data is very likely given the parameter, the parameter is a good predictor.

As in many formulations of learning tasks, we define a *hypothesis space* — a set of possibilities that we are considering, and a scoring function that tells us how good different hypotheses in the space are relative to our data set D. In this case, our hypothesis space is the set of all parameters $\theta \in [0, 1]$. How good is a parameter θ? One possibility for evaluating θ is by how well it predicts the data. In other words, if the data is very likely given the parameter, the parameter is a good predictor. Assuming that we have M_h heads and M_t tails,

$$P(D \mid \theta) = \theta^{M_h} \cdot (1 - \theta)^{M_t}.$$

The higher this number, the better θ is at predicting D.

We can define this more generally, as it will be a very useful concept later on.

Definition 2.1: Let $\mathcal{M}$ be a probabilistic model, that defines a probability distribution over data sets D. We define the *likelihood* of a model given D as

$$L(\mathcal{M} : D) = P(D \mid \mathcal{M})$$

∎

Thus, the likelihood function for a sequence H, T, T, H, H is

$$L(\theta : D) = \theta(1 - \theta)(1 - \theta)\theta\theta = \theta^3(1 - \theta)^2.$$

Figure 2 shows this likelihood function.

This example brings up an important concept. To compute this likelihood, we only needed to know the number of heads N_h and the number of tails N_t. These are the *sufficient statistics* for this learning problem. A *sufficient statistic* is a function of the data that summarizes the relevant information for computing the likelihood.

Definition 2.2: A function $s(D)$ from data sets to $\mathbb{R}^k$ (for some k) is a *sufficient statistic* if, for any two data sets D and D', we have that

$$s(D) = s(D') \Longrightarrow L(\theta : D) = L(\theta : D').$$

∎

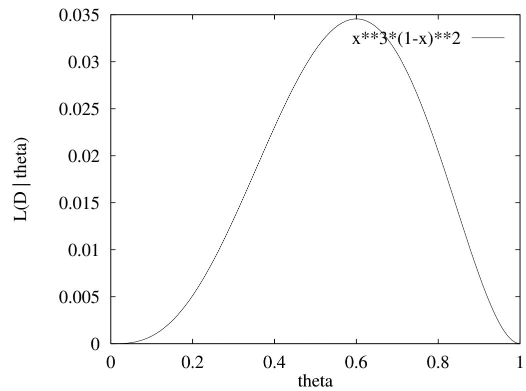

Figure 2: A likelihood function for a single parameter, for a data set consisting of three heads and two tails.

This concept will turn out to be very useful.

We can also define the *log-likelihood* of a model $\mathcal{M}$ given D as:

$$\ell(\mathcal{M} : D) = \log L(\mathcal{M} : D).$$

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, as products are converted to summations, reducing problems of numerical underflow. In particular, we have for our thumbtack example that

$$\ell(\theta : D) = M_h \log \theta + M_t \log(1 - \theta).$$

The *Maximum Likelihood Estimation (MLE)* principle tells us to choose θ that maximizes the likelihood function. This is one of the most commonly used estimators in statistics. In our thumbtack example, we would get

$$\hat{\theta} = \frac{M_h}{M_h + M_t}$$

as expected.

This approach seems plausible, but is overly simplistic in many cases. Assume that we do this experiment with the thumbtack, 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 dime, and also get 3 heads? We would be much less likely to jump to the conclusion that the parameter of the dime is 0.3. Why? Because we have a lot more experience with tossing dimes, so we have a lot more *prior knowledge* about their behavior. Note that we don't 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 dime, of which 300,000 came out heads, then we may be more willing to conclude that this is a trick dime, one whose parameter is closer to 0.3.

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

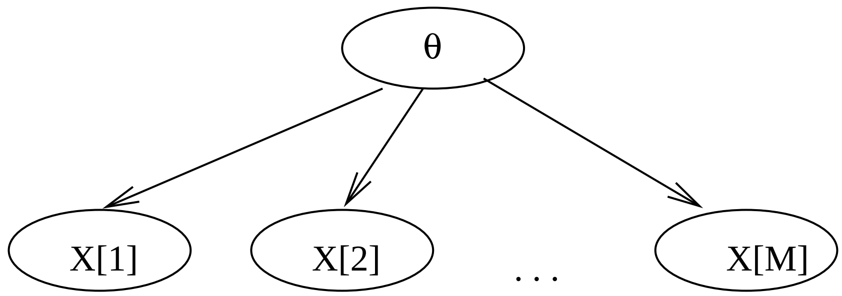

Figure 3: The Bayesian network for simple Bayesian parameter estimation.

2.2 The Bayesian approach

The underlying principle of Bayesian statistics is that we place a *prior distribution* over anything about which we have uncertainty, which expresses whatever prior beliefs about it we might have. In this case, since we are uncertain about the value of the parameter θ, we will place a prior distribution over its (uncountably many) possible values. That is, we define a distribution $P(\theta)$.

More formally, we now have a joint probability space that includes both the tosses and the parameter. Let $X[1], \ldots, X[M]$ be our coin tosses. The joint distribution $P(\theta, X[1], \ldots, X[M])$ can be viewed as a Bayesian network, as shown in Figure 3. The parameters of this joint distribution are easy to specify. The CPD of the root node is our prior over θ, and the conditional probabilities $P(X[m] \mid \theta)$ are according to θ, i.e., $P(X[m] = x^1 \mid \theta) = \theta$. In other words:

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

where M_h is the number of heads in the data, and M_t is the number of tails. Note that the expression $P(X[1], \ldots, X[M] \mid \theta) = P(D \mid \theta)$ is simply the likelihood function $L(\theta : D)$.

The network structure for the parameter estimation problem reflects that the coin tosses are independent given θ. Note, however, that the coin tosses are not marginally independent. In other words, if I don't know θ, then one coin toss gives me information about the outcome of another: Each coin toss tells me something about the probability over θ, and thereby about the probability of the next coin.

Let's see what happens. Assume we had a data set D, with observations for $X[1], \ldots, X[M]$. What is our predicted probability of heads for the $(M+1)$th data case.

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

where

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

In other words, we are using our posterior over θ to predict the probability of heads for the next toss. In our posterior, the first term in the numerator is the likelihood, the second is the prior over parameters, and the third is a normalizing factor which is the marginal probability of the data:

$$P(D) = \int_0^1 P(\theta) P(D \mid \theta) d\theta$$

Let's go back to our thumbtack. Assume that our prior is uniform over θ in the interval $[0,1]$. Then $P(\theta \mid D)$ is proportional to the likelihood $P(D \mid \theta) = \theta^{M_h}(1-\theta)^{M_t}$. Plugging this into the integral, we get

$$P(X[M+1] \mid D) \propto \int \theta \cdot \theta^{M_h}(1-\theta)^{M_t} d\theta$$

doing all the math and renormalizing, we get (for uniform priors)

$$P(X[M+1] \mid D) = \frac{M_h + 1}{M_h + M_t + 2}$$

I.e., we have added 1 imaginary sample to each count. Clearly, as the number of samples grows, the Bayesian estimator and the MLE estimator converge to each other. This will be a general phenomenon.

The challenge here is to pick a distribution over this continuous space that we can represent compactly in closed form, and update efficiently as we get data. For reasons that we discuss below, a very appropriate prior in this case is the *Beta* distribution. A Beta distribution is parameterized by two numbers α_h, α_t. Intuitively, these correspond to the number of imaginary heads and tails that the expert has seen. The Beta distribution $Beta(\alpha_h, \alpha_t)$ is proportional to:

$$\theta^{\alpha_h - 1}(1-\theta)^{\alpha_t - 1}.$$

More precisely, we have that

$$Beta(\alpha_h, \alpha_t) = \frac{\Gamma(\alpha)}{\Gamma(\alpha_h)\Gamma(\alpha_t)} \theta^{\alpha_h - 1}(1-\theta)^{\alpha_t - 1}$$

where $\alpha = \alpha_h + \alpha_t$ and $\Gamma(m)$ is the *Gamma function*, which is $(m-1)!$ but is also defined for non-integers. Figure 4 shows a few Beta distributions. As we can see, the use of an entire distribution rather than a single number to model our parameter θ has the advantage that it reflects not just our current estimate for the value of θ, but also our degree of confidence about that. This is very important in cases where the answers to our queries are very sensitive to the value of θ.

Beta distributions have properties that make them particularly useful for parameter estimation. Assume our distribution on θ is $Beta(\alpha_h, \alpha_t)$. First, let's compute the probability of the next coin toss X coming out heads (x^1):

$$\begin{aligned} P(x^1 \mid \theta) &= \int_0^1 P(\theta) P(x^1 \mid \theta) d\theta \\ &= \int_0^1 P(\theta)\theta d\theta = \alpha_h/(\alpha_h + \alpha_t). \end{aligned}$$

where the last expression is what you get if you do the integral. This supports our intuition that the beta distribution corresponds to having seen α_h heads (imaginary or real) and α_t tails.

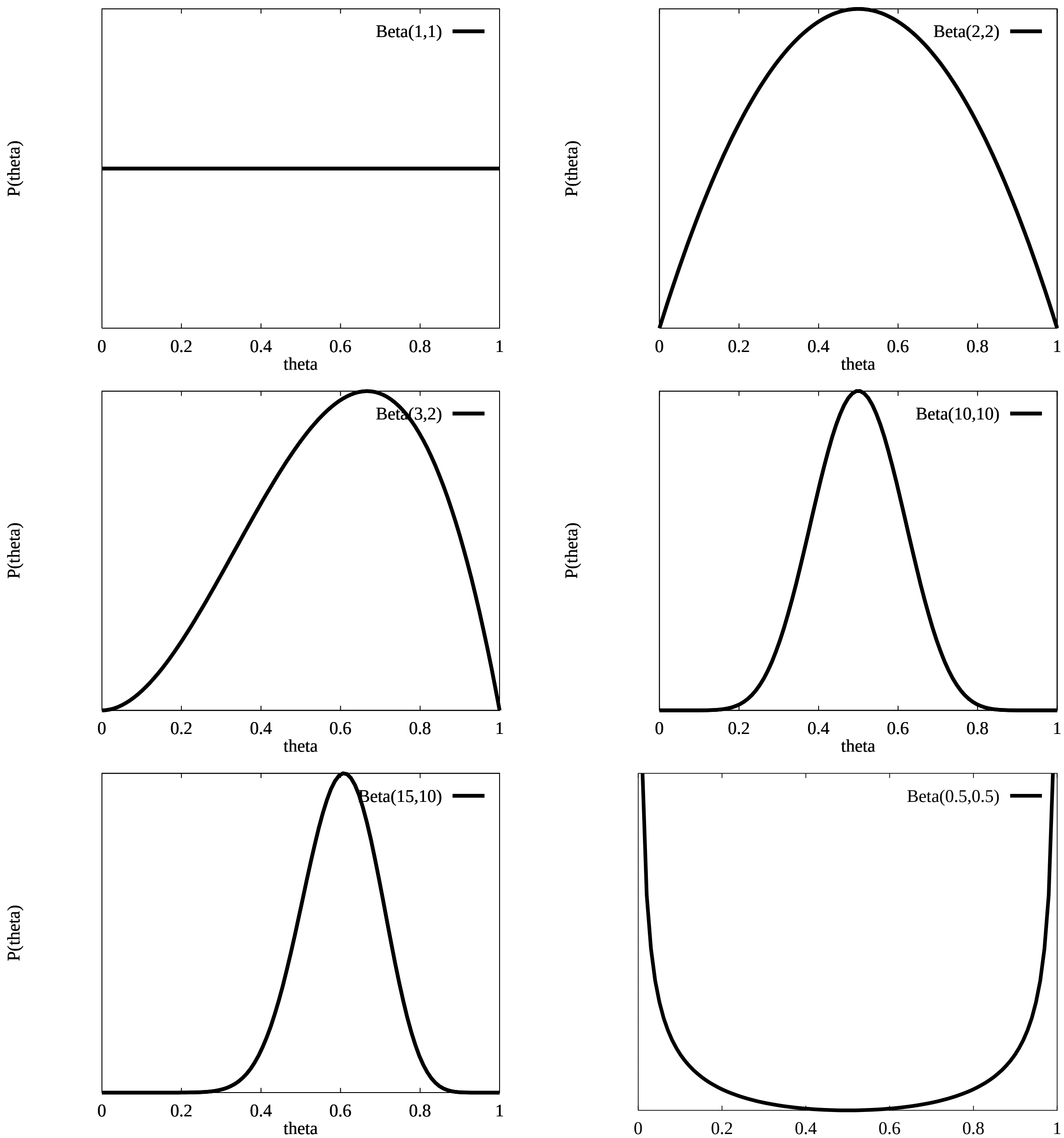

Figure 4: Examples of Beta distributions

Now, let's see what happens if we get more data. Specifically, we get a data set D with N_h heads and N_t tails. It turns out that we can show that:

$$\begin{aligned} P(\theta \mid D) &\propto P(D \mid \theta)P(\theta) \\ &\propto \theta^{N_h}(1-\theta)^{N_t} \cdot \theta^{\alpha_h - 1}(1-\theta)^{\alpha_t - 1} \\ &= \theta^{\alpha_h + N + h - 1}(1-\theta)^{\alpha_t + N_t - 1} \end{aligned}$$

which is precisely $Beta(\alpha_h + N_h, \alpha_t + N_t)$.

This last analysis shows us that the Beta distribution has an important property: the posterior distribution, i.e., the prior conditioned on with a likelihood function which is a Binomial distribution, is also a Beta distribution.

Definition 2.3: Consider a parametric model $P(\mathbf{X} \mid \boldsymbol{\theta})$ which defines a distribution for data instances $\mathbf{x}$ given some set of parameters $\boldsymbol{\theta}$. Let $P(\boldsymbol{\theta} \mid \alpha)$ be a parametric form for a prior over $\boldsymbol{\theta}$, parameterized by *hyperparameters* α. We say that $P(\mathbf{X} \mid \boldsymbol{\theta})$ and $P(\boldsymbol{\theta} \mid \alpha)$ are a *conjugate family* if the *posterior parameter distribution*

$$P(\boldsymbol{\theta} \mid \alpha, \mathbf{x}) \propto P(\mathbf{x} \mid \boldsymbol{\theta})P(\boldsymbol{\theta} \mid \alpha)$$

has the same parametric form as our parameter prior $P(\boldsymbol{\theta} \mid \alpha)$ (albeit with different hyperparameters). We also say that $P(\boldsymbol{\theta} \mid \alpha)$ is the *conjugate prior* for $P(\mathbf{X} \mid \boldsymbol{\theta})$. ∎

In our case, the Beta distribution is a conjugate prior for the Binomial distribution. Conjugate families are very useful for several reasons. First, in many cases we can represent them with a small finite set of hyperparameters. Second, they allow for sequential update of the parameter distribution, as new evidence is obtained, all within the same parametric representation. Finally, in many cases, we have a simple closed-form solution for the prediction task, i.e., for the marginal distribution over $\mathbf{X}$:

$$P(\mathbf{X} \mid \alpha) = \int P(\mathbf{X} \mid \boldsymbol{\theta})P(\boldsymbol{\theta} \mid \alpha)d\boldsymbol{\theta}$$

So far, we have focused attention on binary random variables. This discussion has a natural extension to multi-valued discrete random variables. The Binomial distribution has an extension to multi-valued variables called the *Multinomial distribution*. If our random variable X has k values, the Multinomial distribution is parameterized using a parameter vector $\boldsymbol{\theta} = (\theta_1, \ldots, \theta_k)$ such that all $\theta_i \geq 0$ and $\sum_{i=1}^{k} \theta_i = 1$; θ_i is the probability of the ith outcome.. Similarly, the Beta distribution has an extension called the *Dirichlet distribution*, parameterized using a vector of non-negative parameters $\alpha = (\alpha_1, \ldots, \alpha_k)$. It is defined as:

$$Dir(\alpha_1, \ldots, \alpha_k) = \frac{\Gamma(\alpha)}{\prod_i \Gamma(\alpha_i)} \prod_{i=1}^{k} \theta_i^{\alpha_i - 1}$$

where $\alpha = \sum_i \alpha_i$. As in the case of binary random variables, the Dirichlet distribution is conjugate to the Multinomial distribution. All of the results regarding prediction and computing the posterior extend in the obvious way. That is, if $\boldsymbol{\theta}$ is distributed as above $Dir(\alpha_1, \ldots, \alpha_k)$, and we assume that X is distributed as a multinomial with parameter $\boldsymbol{\theta}$, then

$$P(X = x_i) = \frac{\alpha_i}{\sum_j \alpha_j}.$$

Similarly, if we have a data set D whose sufficient statistics are $M_1, \ldots, M_k$, then

$$P(\boldsymbol{\theta} \mid D) = Dir(\alpha_1 + N_1, \ldots, \alpha_k + N_k).$$

One issue that remains to be resolved is the source of this prior distribution. This distribution, in the Bayesian approach, is completely subjective, so it can be whatever we want it to be. However, that answer is clearly not satisfying. The intuition about imaginary samples gives us some guidance on how to construct these priors. A standard approach is to elicit a Beta distribution from a user via a distribution and an *equivalent sample size*. In other words, we often elicit $p_1, \ldots, p_k$, which represent the user's prior beliefs about the parameters, and an equivalent sample size α, which indicates how many imaginary samples the user has seen. The hyperparameter α_i is then αp_i. This approach makes a lot of sense in medical diagnosis, for example, where the equivalent sample size would correspond to the number of patients the doctor has encountered, and the p_i's to the doctor's sense of how frequent the different cases are.

Let's revisit this issue in the context of our example. If we were tossing a coin, we probably believe fairly strongly that θ is $1/2$, because we have seen many coin tosses; hence, we would have $p_h = p_t = 1/2$, and α quite large. For the thumbtack, we may believe that θ is $1/2$, simply because of ignorance. In this case, we might have $p_h = p_t = 1/2$, but $\alpha = 2$, i.e., a prior of $Beta(1;1)$. Note that, unlike for maximum likelihood estimation, we would not conclude even with an uninformative prior that $\theta = 1$ based on two data cases. In fact, one of the tenets of the Bayesian approach is that only things that we know absolutely to be impossible have probability 0. Therefore, no matter how many heads we see, we will never believe that $\theta = 1$ with probability one.

In general, the Bayesian method is much more robust than maximum likelihood estimation, and much less sensitive to noise.

3 Parameter estimation in Bayesian networks

3.1 Maximum likelihood

It turns out that the generalization of these ideas to the problem of parameter estimation in full Bayesian networks is straightforward. Roughly speaking, a BN is simply a bunch of individual coins: one for each variable X_i and each assignment of values to its parents. Thus, for example, in the simple network $A \rightarrow B$ we have three coins (assuming A, B are both binary): one for A, one for $B|a^1$, and one for $B|a^0$. We simply treat each one as a separate coin, and estimate its value separately. The main difference is that we don't get to observe each coin in each of our samples. Thus, for example, in the datacase a^1, b^0, we get one observation for the A coin, one for the $B|a^1$ coin, and none for the $B|a^0$ coin. This means that our distribution for different parameters in the network may be more or less spread out, depending on how likely this particular instantiation of the parents is.

As we will see, this mapping between BN parameter estimation and estimation of a single parameter holds both for maximum likelihood estimation and for the Bayesian approach.

3.2 Maximum likelihood

Let's begin by considering an example. Consider again the simple Alarm BN, but where we drop the *Radio-report* variable for simplicity of presentation.

As for a single parameter, our goal in maximum likelihood estimation is to maximize the likelihood (or log-likelihood) function. In this case, our BN 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 be:

$$\begin{array}{l} \theta_E \\ \theta_B \\ \theta_{A|b^0,e^0}, \theta A|b^0, e^1, \theta_{A|b^1,e^0}, \theta A|b^1, e^1 \\ \theta_{C|a^0}, \theta_{C|a^1} \end{array} \tag{1}$$

For brevity (and consistency with later notation), we also use $\theta a^1|b^0, e^0$ to refer to $\theta_{A|b^0,e^0}$ and $\theta a^1|b^0, e^0$ to refer to $(1-\theta_{A|b^0,e^0})$.

Each instance is a tuple $B[m], E[m], A[m], C[m]$. Our likelihood function is:

$$\begin{aligned} L(\boldsymbol{\theta}:D) &= \prod_{m=1}^{M} P(B[m], E[m], A[m], C[m] \mid \boldsymbol{\theta}) \\ &= \prod_m P(B[m] \mid \boldsymbol{\theta}) P(E[m] \mid \boldsymbol{\theta}) P(A[m] \mid B[m], E[m], \boldsymbol{\theta}) P(C[m] \mid A[m], \boldsymbol{\theta}) \\ &= \prod_m P(B[m] \mid \boldsymbol{\theta}) \prod_m P(E[m] \mid \boldsymbol{\theta}) \prod_m P(A[m] \mid B[m], E[m], \boldsymbol{\theta}) \prod_m P(C[m] \mid A[m], \boldsymbol{\theta}) \end{aligned}$$

I.e., the likelihood decomposes into separate likelihoods for every variable.

In fact, we can do even better. Let's look at individual terms. Clearly, each one depends only on the parameters for that variable's CPT. Thus, the first is $\prod_m P(B[m] \mid \theta_B)$. The second is $\prod_m P(E[m] \mid \theta_E)$. The third and fourth are more interesting, as we can decompose them even further. Let's do the fourth first, as it's simpler.

$$\begin{aligned} &\prod_m P(C[m] \mid A[m], \theta_{C|a^0}, \theta_{C|a^1}) \\ &\quad= \prod_{m:A[m]=a^0} P(C[m] \mid A[m], \theta_{C|a^0}, \theta_{C|a^1}) \cdot \prod_{m:A[m]=a^1} P(C[m] \mid A[m], \theta_{C|a^0}, \theta_{C|a^1}) \\ &\quad= \prod_{m:A[m]=a^0} P(C[m] \mid A[m], \theta_{C|a^0}) \cdot \prod_{m:A[m]=a^1} P(C[m] \mid A[m], \theta_{C|a^1}) \end{aligned}$$

We can do a similar decomposition for the term corresponding to A. We will have one term for b^0, e^0, another for b^0, e^1, etc. Thus, the likelihood function decomposes into a product of terms, one for each parameter θ in $\boldsymbol{\theta}$. This property is called the *decomposability* of the likelihood function.

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

$$\prod_{m:A[m]=a^0} P(C[m] \mid A[m], \theta_{C|a^0}) \tag{2}$$

Each of the individual terms $P(C[m] \mid A[m], \theta_{C|a^0})$ can take one of two values, depending on the value of $C[m]$. If $C[m] = c^1$, it is equal to $\theta_{C|a^0}$. If $C[m] = c^0$, it is equal to $1 - \theta_{C|a^0}$. How many cases of each type do we get? First, we restrict attention only to those data cases where $A[m] = a^0$. These, in turn, partition into the two categories. Thus, we get $\theta_{C|a^0}$ in those data cases where $A[m] = a^0$ and $C[m] = c^1$; we get $1 - \theta_{C|a^0}$ in those data cases where $A[m] = a^0$ and $C[m] = c^0$. Thus, we are interested in the numbers $M[a^0, c^1]$ and $M[a^0, c^0]$, that correspond to the number of datacases of each type. More precisely, we define:

Definition 3.1: Let $\mathbf{Y}$ be some set of random variables, and $\mathbf{y}$ be some instantiation to these random variables. Let D be a data set. We define $M[\mathbf{y}]$ to be the number of entries $\mathbf{X}[m]$ in D that have $\mathbf{Y}[m] = \mathbf{y}$. ∎

We can finally finish simplifying our expression. The term in Eq. (2) is equal to:

$$\begin{aligned}\prod_{m:A[m]=a^0} P(C[m] \mid A[m], \theta_{C|a^0}) &= \prod_{m:A[m]=a^0;C[m]=c^1} \theta_{C|a^0} \cdot \prod_{m:A[m]=a^0;C[m]=c^0} (1-\theta_{C|a^0}) \\ &= \theta_{C|a^0}^{M[a^0,c^1]}(1-\theta_{C|a^0})^{M[a^0,c^0]}\end{aligned}$$

Doing the same process for the entire network, we have that

$$L(\boldsymbol{\theta} : D) = \left(\begin{array}{c} \theta_B^{M[b^1]} \cdot (1-\theta_B)^{M[b^0]} \cdot \\ \theta_E^{M[e^1]} \cdot (1-\theta_E)^{M[e^0]} \cdot \\ \theta_{A|b^0,e^0}^{M[b^0,e^0,a^1]} \cdot (1-\theta_{A|b^0,e^0})^{M[b^0,e^0,a^0]} \cdot \\ \theta_{A|b^0,e^1}^{M[b^0,e^1,a^1]} \cdot (1-\theta_{A|b^0,e^1})^{M[b^0,e^1,a^0]} \cdot \\ \theta_{A|b^1,e^0}^{M[b^1,e^0,a^1]} \cdot (1-\theta_{A|b^1,e^0})^{M[b^1,e^0,a^0]} \cdot \\ \theta_{A|b^1,e^1}^{M[b^1,e^1,a^1]} \cdot (1-\theta_{A|b^1,e^1})^{M[b^1,e^1,a^0]} \cdot \\ \theta_{C|a^0}^{M[a^0,c^1]} \cdot (1-\theta_{C|a^0})^{M[a^0,c^0]} \end{array}\right)$$

We can do this analysis more generally. A BN $\mathcal{B}$ is parameterized by a set of Dirichlet distributions $\boldsymbol{\theta}_{X|\mathbf{u}}$, one for each node X and each instantiation $\mathbf{u}$ of its parents. Each such Multinomial distribution has a vector of parameters $\theta_{x^1|\mathbf{u}}, \ldots, \theta_{x^k|\mathbf{u}}$, where k is the number of values on $Val(X)$ and $\theta_{x^1|\mathbf{u}}$ is the parameter for the jth value x^j of X (given the assignment $\mathbf{u}$ to X's parents). An analysis very similar to what we just did shows:

Theorem 3.2: *Let G be a BN structure over $X_1, \ldots, X_n$, and let D be a dataset where each entry $\mathbf{X}[m]$ is an assignment of values to $X_1, \ldots, X_n$. Let $\boldsymbol{\theta}$ be a possible parameter vector for G. Then*

$$L(\boldsymbol{\theta} : D) = \prod_{i=1}^{n} \prod_{\mathbf{u} \in Val(\mathbf{Pa_{X_i}})} \prod_{x_j \in Val(X_i)} \theta_{x_j|\mathbf{u}}^{M[x_j,\mathbf{u}]}.$$

This decomposition property is crucial. As one immediate consequence, it shows us how to do MLE for BNs. To see this more clearly, consider the log-likelihood function:

$$\ell(\boldsymbol{\theta} : D) = \sum_{i=1}^{n} \sum_{\mathbf{u} \in Val(\mathbf{Pa_{X_i}})} [\sum_{x_j \in Val(X_i)} M[x_j, \mathbf{u}] \log \theta_{x_j|\mathbf{u}}].$$

This is a summation of terms, each of which involves a different parameter. The ones inside the parentheses all belong to the same Multinomial distribution, so they are constrained to sum to 1. For example, in our decomposition above, we would have such an expression

$$M[a^0, c^1] \log \theta_{C|a^0} + M[a^0, c^0] \log(1-\theta_{C|a^0})$$

The other terms don't involve $\theta_{C|a^0}$, so we can optimize this expression individually. But we already know the value of $\theta_{C|a^0}$ that maximizes this expression:

$$\hat{\theta}_{C|a^0} = \frac{M[a^0,c^1]}{M[a^0,c^1] + M[a^0,c^0]} = \frac{M[a^0,c^1]}{M[a^0]}$$

The same thing applies to other parameters. In general, we can optimize the parameters for each multinomial $\boldsymbol{\theta}_{X|\mathbf{u}}$ separately. Each of these maximizations is exactly a separate "coin": we have a parameter $\theta_{x_j|\mathbf{u}}$ for each j, and we have counts (sufficient statistics) for each of them. Hence, our maximum likelihood estimate is:

$$\hat{\theta}_{x_j|\mathbf{u}} = \frac{M[x_j, \mathbf{u}]}{\sum_{\ell=1}^{k} M[x_\ell, \mathbf{u}]}$$

Returning to our simplified Alarm network, we have eight parameters, as shown in Eq. (1). Each parameter can be treated as a separate coin, and estimated independently using a maximum likelihood computation. The decomposability property guarantees that if we maximize the likelihood associated with each such parameter, we have maximized the total likelihood function. The maximization of each parameter separately gives the obvious result; for example,

$$\hat{\theta}_{A|b^1,e^0} = \frac{M[a^1, b^1, e^0]}{M[b^1, e^0]},$$

i.e., the fraction, among those data cases with b^1, e^0, of those that have a^1.

What can we say about the answer provided by our maximum likelihood estimate? Is it "correct" in some sense? To answer this question, we need to provide some formal foundations. First, we assume that the data is sampled as IID instances from some true (but unknown) underlying distribution $P^*(X_1, \ldots, X_n)$. The question we would like to answer is: How close is our MLE distribution $\hat{P}$ (P with the maximum likelihood parameters $\hat{\boldsymbol{\theta}}$) to P^*. Clearly, for a finite amount of data, we can get an unrepresentative sample of P^*. However, we can get asymptotic guarantees as the amount of available data grows to infinity.

Definition 3.3: Let $P(\mathbf{X} \mid \boldsymbol{\theta})$ be a parametric family, and let $P^*(\mathbf{X})$ be the distribution that generates the data (as IID cases). Let $\pi(D)$ be an estimator that takes a data set and returns a parameter estimate $\boldsymbol{\theta}$. We say that $\pi(D)$ is *consistent* if, as $|D| \to \infty$, we have that

$$\pi(D) \to \hat{\boldsymbol{\theta}}^*$$

where

$$\hat{\boldsymbol{\theta}}^* = \text{argmin}_{\boldsymbol{\theta}} \mathbb{D}_{KL}(P^*(\mathbf{X}); P(\mathbf{X} \mid \boldsymbol{\theta}))$$

∎

In other words, in the limit, the estimator $\pi(D)$ returns the distribution, within the parametric family, that is closest to the true distribution P^*. If P^* is exactly representable within the parametric family, then consistency implies that, at the limit, the estimator converges to the true underlying distribution.

Theorem 3.4: *The maximum likelihood estimator is consistent.*

To understand the implications of this theorem, consider a parametric family which is specified as a BN structure $\mathcal{B}$ with each CPD represented as a set of multinomials, one for each parent. If our true distribution P^* can be represented as a Bayesian network over $\mathcal{B}$ (i.e., if P^* factorizes over $\mathcal{B}$), then this theorem implies that the network defined by the MLE parameters will converge to P^*. If the true underlying distribution does not factorize over $\mathcal{B}$, then consistency tells us that we will converge to the distribution $\hat{P}$ over $\mathcal{B}$ which is closest to P^* in KL-distance, in this case, the distribution $\hat{P}$ that has $\hat{P}(X_i \mid \mathbf{Pa}_{X_i}) = \mathbf{P}^*(\mathbf{X_i} \mid \mathbf{Pa}_{\mathbf{X_i}})$, satisfies certain independence assumptions that P^* does not.

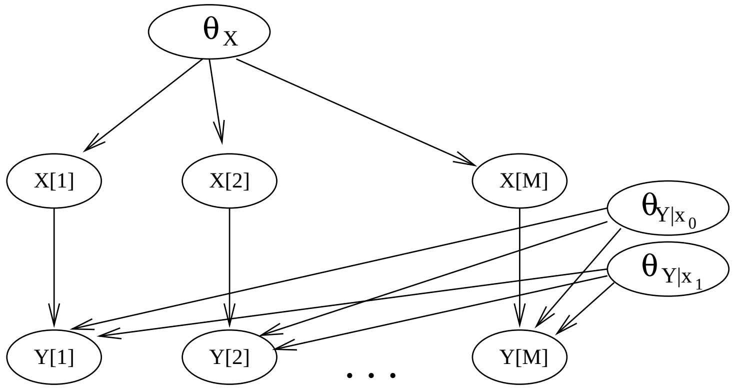

Figure 5: The Bayesian network for parameter estimation for a simple two-node BN.

3.3 The Bayesian approach

The same type of result also applies to the Bayesian case. As in the single parameter case, we can now use a Dirichlet distribution for each parameter. As in the single parameter case, we can understand the joint distribution over parameters and data as a Bayesian network. Figure 5 shows such a Bayesian network for the BN structure $X \rightarrow Y$.

We can read several things from this Bayesian network. As above, the instances are independent given the unknown parameters. Also, it encodes an assumption that the priors for the individual parameters are independent. I.e., we believe that knowing the value of one parameter tells us nothing about another. This assumption is nontrivial. For example, if we believed that the alarm node has a noisy or distribution, knowledge of $\theta_{A|b^0,e^1}$ would tell us a lot about $\theta_{A|b^1,e^1}$. In this case, however, a standard parameterization that has a separate parameter for each assignment to the parents is inappropriate, and a different model should be used. In practice, this assumption of *parameter independence* is made almost universally. From this assumption, we obtain an important conclusion. Complete data d-separates the parameters for different families. Thus, if they were independent a priori, they are also independent a posteriori. Less obviously, we also get independence of the parameter posteriors within a family: $\theta_{Y|x^0}$ is independent of $\theta_{Y|x^1}$ given the evidence. We can't read that off the network structure. So how do we know it's true? It's an instance of CSI! If $X[m] = x^1$, $Y[m]$ depends only on $\theta_{Y|x^1}$; conversely, if $X[m] = x^0$, $Y[m]$ depends only on $\theta_{Y|x^0}$. This type of dependence is the same as the dependence we had in our driving-time example, where we had that *T101* is independent of *T280* given *Road* and *Time*. As we showed there, $I(\mathit{T101}; \mathit{T280} \mid \mathit{Road}, \mathit{Time})$. Here, each $Y[m]$ plays the role of *Time*, $X[m]$ plays the role of *Road*, and the two parameters play the role of the *T101* and *T280* variables. Hence, we have that $I(\theta_{Y|x^0}; \theta_{Y|x^1} \mid X[m], Y[m])$, as desired. We can apply this same analysis to all the examples, concluding that: $I(\theta_{Y|x^0}; \theta_{Y|x^1} \mid D)$.

As a consequence, our *posterior* distribution over the parameters in the BN can be viewed as a product of separate posteriors, one for each Dirichlet distribution involved. This is important for several reasons. First, each of these posteriors is a nice unimodal Dirichlet distribution (has a single peak); hence, the same is true for the joint posterior over parameters. Second, it shows that we can compute the posteriors separately; i.e., we can compute $P(\boldsymbol{\theta} \mid D)$ by separately computing, e.g., $P(\theta_{C|a^0} \mid D)$. The only part of the data $\theta_{C|a^0}$ depends on is the value of C in those data cases for which $A[m] = a^0$. In fact, this turns out to be identical to looking at $\theta_{C|a^0}$ as an individual coin. If the prior distribution for this parameter was $\mathit{Beta}(\alpha_0, \alpha_1)$, its posterior would be $Beta(\alpha_0 + M[c^0, a^0], \alpha_1 + M[c^1, a^0])$. More generally, after seeing a data set D, the posterior distribution at a node X with parents $\mathbf{U}$ will be parameterized by a set of Dirichlet distributions

$$Dir(\alpha_{x_1|\mathbf{u}} + M[x_1, \mathbf{u}], \ldots, \alpha_{x_k|\mathbf{u}} + M[x_k, \mathbf{u}])$$

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

$$P(x_i \mid \mathbf{u}, D) = \frac{\alpha_{x_i|\mathbf{u}} + M[x_i, \mathbf{u}]}{\sum_i \alpha_{x_i|\mathbf{u}} + M[x_i, \mathbf{u}]}$$

In other words, for the next prediction, we can use a single network parameterized as usual, via a set of multinomials, but ones computed as in this equation.

To understand some of the advantages of using a prior, let's consider what happens when we are estimating the parameters at a node. At that point, we have to split up our data cases into the mutually exclusive and exhaustive cases for the parents. For example, in order to estimate the parameter of the C node (with its parent A) in our Alarm network, we have to split up the data into the a^0 cases — used to estimate $P(C \mid a^0)$, and the a^1 cases — used to estimate $P(C \mid a^1)$. Thus, each estimation process has fewer data cases to utilize, making it more prone to errors. For example, if C also had E as a parent (the neighbor is less likely to call in case of an earthquake), we would have to fragment our data into four separate groups.

In a network where nodes have a lot of parents, we may have parent contexts for which we get little or no data. What do we do then? For example, what if a particular parent context, e.g., e^1, a^0 only has 3 data cases, and in all of them, there was no call? Do we conclude that $P(c^1 \mid e^1, a^0) = 0$? This is the conclusion that we would get from maximum likelihood. A Bayesian approach helps smooth out the parameters in cases where we do not have enough data to jump to conclusions.

Thus, for small amounts of data, the Bayesian approach is different from MLE. It smoothes out the estimates, and makes them more robust. But asymptotically, as we get more data, the effects of the prior are going to become vanishingly small relative to the counts. Hence, asymptotically, the Bayesian prediction converges to the MLE prediction. As a consequence, we have that the Bayesian estimator is also consistent.

It remains only to address the question of assessing the set of priors required for a Bayesian network. In a general BN, each node X_i has a set of multinomial distribution $\boldsymbol{\theta}_{X_i|\mathbf{u}}$, one for each instantiation $\mathbf{u}$ of X_i's parents $\mathbf{U}$. Each of these parameters will have a separate Dirichlet prior, governed by hyperparameters $\alpha_{X_i|\mathbf{u}} = (\alpha_{x_i^1}, \ldots, \alpha_{x_i^k})$ (where k is the number of values of X_i). The most common approach to assigning priors, and the most correct from a philosophical perspective, is to use a prior joint distribution $P_0(X_1, \ldots, X_n)$ over all of the BN variables, and an equivalent sample size α. We then estimate $\alpha_{x_i|\mathbf{u}}$ as $\alpha \cdot P_0(x_i, \mathbf{u})$. Intuitively, α is the total number of previous samples that the knowledge engineer has seen and P_0 is some rough estimate of the distribution seen in these samples. Note that P_0 does not have to be structured in the same way as our BN. (Although it can be.) It is, in fact, quite common to define P_0 as a set of independent marginals over the X_i's.

4 Applications

We note that simple parameter learning can be very useful. In many cases, the structure (or a good approximation to it) can be elicited quite easily, whereas parameters are much harder to elicit.

For example, one of the most commonly used techniques for classification in text domains is the Naive Bayes model. For text, the hypothesis variable is the category (topic) of the document. There are two ways of defining the features.

In the *multivariate Binomial model*, each attribute (feature) X_i is a binary variable representing whether a given word w_i in some dictionary $\mathcal{D}$ appeared in the document or not. This model makes two assumptions. First, it assumes that we care only about the presence of a word, not about how many times it appeared. Second, it assumes that the appearance of one word in the document is independent of the appearance of another.

In the *multinomial model*, X_i corresponds to the ith word in the document; we have that $Val(X_i) = \mathcal{D}$, i.e., each X_i can take on many values, one for each possible word. Here, we assume that the choice of word in position i is uncorrelated with the choice of word in position j. Furthermore, we assume that the probability that a particular word is used in position i does not depend on i; i.e., the probability that $X_i = w$ (given the class) is the same as the probability that $X_j = w$. However, if a word appears in several different positions in the document, it will be counted accordingly. (Note that, here, documents of different lengths are associated with a different number of random variables.

In both models, the parameters are estimated from data, and the resulting model used for classifying new documents. Usually, Bayesian parameter estimation is used to avoid overfitting, which is a real problem in a domain such as this where data is sparse relative to the number of parameters we want to estimate. This approach, despite its simplicity (or perhaps because of it), is remarkably successful. Empirical evidence suggests that the multinomial model is, in general, more successful than the multivariate binomial.

5 Summary: Parameter estimation

- Estimation relies on sufficient statistics; for multinomials and binomials these have the form $M[x, \mathbf{u}]$.
- Parameter estimation for $\theta_{x|\mathbf{u}}$ is $M[x, \mathbf{u}]/M[\mathbf{u}]$ in the MLE case, and $Dir(\alpha_{x_1}+M[x_1, \mathbf{u}], \ldots, \alpha_{x_k}+M[x_k, \mathbf{u}])$ in the Bayesian case.
- If the data was actually generated from the given network structure, then both methods converge asymptotically to the correct parameter setting. If not, then they converge to the distribution with the given structure which is “closest” to the distribution from which the data was generated.
- Both estimation methods can be implemented online by accumulating sufficient statistics.