

Chapter 6

The EM algorithm and Information Theory

6.1 Mixture Models

In the previous chapter we have mentioned that it may happen that a likelihood function has multiple maxima and that sometimes it may be hard or impossible to find the global maximum (i.e. the maximum with the overall highest likelihood value). Such a situation occurs whenever the probabilistic model that we use to model our observations is a **latent** or **hidden variable model**. Latent variable models are models that besides modelling observed data also model a portion of unobserved data. For example, if we look at the income distribution of a population we may want to further differentiate between age groups. If the age of all or some members of the population is not provided in the data, we can still model it as a latent variable. The difficulty is that we will have to make inferences about the age of an individual based on other information that we have about it (e.g. the income).

While it may in general be quite hard to formulate latent variable models, there are certain standard latent variable models that have wide-spread applications. One such class are **mixture models**.

Definition 6.1 (Mixture Model) *Given any set of probability distributions P_{X_1}, \dots, P_{X_n} we define a mixture model as $P_X = \sum_{i=1}^n \alpha_i P_{X_i}$ where we require that all $\alpha_i \geq 0$ and $\sum_{i=1}^n \alpha_i = 1$. We call the distributions P_{X_1}, \dots, P_{X_n} **mixture components** and their weights α_i **mixture weights**.*

Mixture models are extremely useful whenever we have different ways to think about our data. Each way of conceptualising our data can be encoded by one of the mixture components of the mixture model. This can help

us to build a better overall model of our data. Let us introduce a running example that we use for the rest of this section.

Example of a mixture model Assume we observe 20 sequences of coin tosses. Each sequence contains 10 flips. We also know that there are 3 coins with which these sequences could possibly have been generated and for each sequence a different coin may have been used. Coin 1 is unbiased, coin 2 has parameter $\theta = 0.4$ and coin 3 has parameter $\theta = 0.65$.

We could assume that the entire data set was generated by exactly one coin. We would then employ maximum likelihood estimation to find the parameter of that coin. However, this model might actually turn out to be pretty bad because we are committing to picking only one coin. This is a bad assumption because we said in the beginning that each sequence may possibly have been generated by a different coin.

A mixture model comes to the rescue. Instead of assuming that only one coin has generated all 20 sequences, we assume that all three coins have contributed to generating the 20 sequences. However, their contributions may not be equal. This inequality is exactly what the mixture weights capture. As usual, we call our data x . Also, each mixture component is a binomial distribution, parametrized by the parameters of the coins. This information suffices to formulate our mixture model.

$$(6.1) \quad \begin{aligned} P(X = x | \Theta_1^3 = \theta_1^3, A = \alpha_1^3) &= \alpha_1 P(X = x | \Theta_1 = 0.4) \\ &+ \alpha_2 P(X = x | \Theta_2 = 0.5) + \alpha_3 P(X = x | \Theta_3 = 0.65) \end{aligned}$$

Notice that if we were given the mixture weights, estimating the parameters of the mixture components would be easy: we would simply find the MLE for each mixture component. The mixture model could then easily be constructed because the mixture weights are known.

Usually, we face a more difficult problem when working with mixture models. Neither the mixture weights nor the parameters of the mixture components are known. For ease of exposition, we will focus on the medium hard case where we know the mixture components, but do not know the mixture weights.

How can we go about estimating the weights? First, observe the constraints on mixture weights in Definition 6.1. All mixture weights have to be non-negative and they have to sum to 1. These constraints mean that we can interpret the weights as a probability distribution. In particular, the weights form a distribution over parameters of the mixture components (assuming that all mixture components have the same parametric form). That is, we get the following relations:

$$(6.2) \quad \alpha_1 = P(\Theta_1 = \theta_1) \quad \alpha_2 = P(\Theta_2 = \theta_1) \quad \alpha_3 = P(\Theta_3 = \theta_1)$$

Hence, we can rewrite our mixture model as

$$(6.3) \quad \begin{aligned} P(X = x) &= P(\Theta_1 = 0.4)P(X = x|\Theta_1 = 0.4) \\ &\quad + P(\Theta_2 = 0.5)P(X = x|\Theta_2 = 0.5) + P(\Theta_3 = 0.65)P(X = x|\Theta_3 = 0.65) \end{aligned}$$

In our examples, we assume that we have only three possible binomial parameters, namely $\theta_1 = 0.4, \theta_2 = 0.5, \theta_3 = 0.65$. Each summand in (6.3) is in fact a joint distribution by the chain rule so that we can again rewrite the model as

$$(6.4) \quad \begin{aligned} P(X = x) &= P(X = x, \Theta_1 = 0.4) + P(X = x, \Theta_2 = 0.5) \\ &\quad + P(X = x, \Theta_3 = 0.65) \\ &= \sum_{i=1}^3 P(X = x, \Theta = \theta_i) \end{aligned}$$

where (6.4) is a simple marginalisation step. Notice that we have just accomplished something impressive: we have given a probabilistic justification for why mixture models do indeed model our data. Each mixture-component/weight pair gives rise to a joint distribution over parameters and data but by summing over the possible parameters we get the probability of the data. We can easily show that mixture models can be defined for any number of mixture components.

Exercise 6.2 Show that a mixture model of size n is a model of the data, i.e. show that $\sum_{i=1}^n \alpha_i P(X_i = x) = P(X = x)$ if α_i are mixture weights as defined in Definition 6.1.

Notice that we have shown above that the formulation of mixture models as purely probabilistic models and as linear combinations of probability distributions are equivalent. So why did we even bother to give a purely probabilistic justification? On the one hand, it is mathematically satisfying to trace back new concepts to concepts that we are already familiar with (like joint distributions and the chain rule). More importantly, however, the probabilistic interpretation of mixture weights allows to estimate them using the maximum-likelihood principle. This estimation is something we could not have done if we had regarded them solely as scale factors for the mixture components.

There is one additional problem, however: if the mixture weights are unknown, there is no closed-form solution for estimating the likelihood function. This is because the likelihood depends on the parameters of the mixture components whose distribution, given by the mixture weights, is unknown. This means that while we can compute the likelihood for each individual mixture component, we cannot compute the likelihood of the entire

mixture model because we do not know how much each component contributes to the overall likelihood term. As a consequence, we can not simply apply calculus as we have been doing up to now. For this reason, we turn to the **EM algorithm**, that allows to at least find a local maximum of the likelihood function.

One final note: At the beginning we introduced latent variable models (and hence mixture models) as models of *latent data*. We have cast the problem of inferring the mixture weights as inferring a distribution over mixture components, however. One may argue that those are not data, neither latent nor observed. This is a fair criticism but there is an easy way out. Simply imagine that each sequence of coin flips was annotated with a pointer to the mixture component (the coin) that generated it. This annotation is clearly part of the data. Since in our actual data, these annotations are missing, we treat them as latent data. This strategy of defining additional latent random variables that serve as pointers or stand-ins for mixture components is often employed in the literature. Obviously, seeing mixture weights as probabilities of mixture components or probabilities of pointers to mixture components is fully equivalent.

6.2 The EM algorithm

In order to estimate the parameters of mixture models, we can employ a classical algorithm of **unsupervised learning**, namely the **expectation-maximisation (EM) algorithm**. This algorithm allows to find a local maximum of the likelihood function of mixture models or, more generally, models with missing data.

To give you some more intuition for what latent data is we provide a further example that has spawned a lot of research. Assume you run a website that recommends movies based on a user's preferences. In order to make statistical predictions about what type of user likes what kind of movie, you ask your users to rate movies according to different categories. Say you ask your users to rate the movies for entertainment value, action and fun. What may happen is that some of your users only rate a movie in one or two of these three categories. However, these ratings are still valuable to you and you do not want to throw them away, just because the rating is incomplete in one category. Thus you have a data set with some missing data that you have to fill in somehow.

In mixture models, annotations that tell you which mixture component generated a given data point can be thought of as missing data. As we have seen, such annotations are usually missing and thus we cannot do maximum likelihood estimation. The idea of EM is to make an educated guess at the probability with which each mixture component could *potentially* have generated each data point. What we do know is our observed data and some

initial guess of the mixture weights which act as prior probabilities for the mixture components (this guess may be arbitrary). Our educated guess is then simply based on Bayes' rule. It is the posterior probability of each mixture component given the data point.

Using the posterior over mixture components, the EM algorithm allows us to probabilistically fill in the missing data and find good mixture weights (where you should understand *good* in the maximum-likelihood sense). The idea behind the algorithm is simple: compute the expected number of occurrences of the missing data values (the mixture components) and then do maximum likelihood estimation on those expectations. Repeat the procedure until the likelihood does not increase any further. Notice that this procedure requires to fix the number of mixture components in advance.

More formally, assume a data set x_1^n . Furthermore, define Y as a random variable over m possible mixture components (we normally choose $m \ll n$). Then the likelihood function is

$$(6.5) \quad L_x(\theta) = P(X = x | \Theta = \theta) = \sum_{y=1}^m P(X = x, Y = y | \Theta = \theta)$$

where Θ ranges over the parameters of the joint distribution on P_{XY} . Recall that EM probabilistically fills in missing data. However, since we are doing maximum likelihood estimation, we are not so much interested in the missing data itself but rather in the sufficient statistics of that data (see Section ??). Since we cannot directly obtain the sufficient statistics of missing data, we will instead compute the *expected sufficient statistics* where the expectation is taken per data point with respect to the posterior distribution over mixture components.

Since we are referring to sufficient statistics our exposition of EM is specific to distributions in the exponential family. The EM algorithm can also be made more general. However, since virtually all distributions that are of interest in practice do belong to the exponential family, we will not make this kind of generalisation.

It is important to notice that the distribution over mixture components is always a categorical distribution for which each mixture component constitutes an outcome. Since the sufficient statistics for the categorical are simply the counts of the outcomes, we have already identified an important set of sufficient statistics that we will need in any EM algorithm. Be aware, however, that the sufficient statistics for the mixture components (if we wish to estimate their parameters as well) may be different from the categorical sufficient statistics.

The EM algorithm is an iterative algorithm, meaning we repeat its steps several times. We use superscripts to indicate the number i of the repetition, where $0 \leq i \leq k$. To formalize the EM algorithm, assume we are at iteration i for which we have some parameter estimate $\theta^{(i)}$ (the initial estimate $\theta^{(0)}$)

can be set arbitrarily). Based on this parameter estimate we then compute the expected sufficient statistics of the missing data which we call $t(y)$.

$$(6.6) \quad t(y)^{(i+1)} = \mathbb{E}(t(Y) | X = x, \Theta = \theta^{(i)})$$

Equation (6.6) is known as the **E(xpectation)-step** of the EM algorithm. This name comes from the fact that in this step we compute the expected sufficient statistics. To make the algorithm complete, we still miss a **M(aximization)-step**. But that step is simple. We pretend that the expected sufficient statistics of the latent data were actually observed. Once we pretend to observe the expected statistics, the maximization step can be performed using maximum-likelihood estimation:

$$(6.7) \quad \theta^{(i+1)} = \arg \max_{\theta} P(X = x, t(Y) = t(y)^{(i+1)} | \theta)$$

Definition 6.3 (EM algorithm) *We assume a data set $x = x_1^n$ and postulate that there is unobserved data $y = y_1^m$ which is a realisation of RV Y . We also assume a probabilistic model whose parameters are realisations of a RV Θ . Let $t(y)$ be the sufficient statistics of a realization y . Then any iterative algorithm with k iterations that performs the following steps for $0 \leq i \leq k-1$,*

$$\textbf{E-step:} \quad t(y)^{(i+1)} = \mathbb{E}(t(Y) | X = x, \Theta = \theta^{(i)})$$

$$\textbf{M-step:} \quad \theta^{(i+1)} = \arg \max_{\theta} P(X = x, t(Y) = t(y)^{(i+1)} | \Theta = \theta)$$

to update the model parameters is called an EM algorithm.

Example of an EM algorithm Assume as in Section 6.1 that our data is $x = x_1^{20}$ where each x_i is the number of heads that we observed in a sequence of a 10 coin tosses. Again we also assume mixture components that are binomials with parameters 0.4, 0.5, 0.65. The latent data in this case is an annotation that for each observed sequence x_i reveals the coin that has been used to generate that sequence. Thus we have latent data $y = y_1^n$ with $y_i \in \text{supp}(Y) = \{0.4, 0.5, 0.65\}$. Both the observed and latent variables are assumed i.i.d. We assume that the fair coin is more likely to be used and hence set its initial mixture weight to 0.5 and the mixture weights of the other two coins to 0.25 (any other choice would also be fine). Let us take a closer look at our data. To shorten notation, we write it as a list where the i^{th} entry is the value of x_i .

[6, 5, 4, 2, 6, 6, 6, 5, 4, 2, 5, 5, 3, 4, 6, 4, 5, 6, 3, 3]

Then for each x_i we assume that it was generated by each of the three coins. For the first observation we get the following likelihood values.

$$(6.8) \quad \begin{aligned} P(X_1 = 6 | \Theta = 0.4) &= 0.1114767 \\ P(X_1 = 6 | \Theta = 0.5) &= 0.2050781 \\ P(X_1 = 6 | \Theta = 0.65) &= 0.2376685 \end{aligned}$$

Recall that the mixture weights are nothing else than priors over mixture components. Hence, in order to get the joint distribution over observed and latent data, we multiply the likelihoods by the mixture weights.

$$(6.9) \quad \begin{aligned} P(X_1 = 6, \Theta = 0.4) &= 0.25 \times P(X_1 = 6 | \Theta = 0.4) = 0.02786918 \\ P(X_1 = 6, \Theta = 0.5) &= 0.5 \times P(X_1 = 6 | \Theta = 0.4) = 0.1025391 \\ P(X_1 = 6, \Theta = 0.65) &= 0.25 \times P(X_1 = 6 | \Theta = 0.4) = 0.05941712 \end{aligned}$$

We are interested in the posterior over mixture components. Because we are dealing with a categorical distribution here, the posterior probability is exactly the expected number of times that each coin has generated data point x_1 . The posterior given x_1 is shown below.

$$(6.10) \quad \begin{aligned} P(\Theta = 0.4 | X_1 = 6) &= \frac{P(X_1 = 6, \Theta = 0.4)}{P(X_1 = 6)} = 0.146814 \\ P(\Theta = 0.5 | X_1 = 6) &= \frac{P(X_1 = 6, \Theta = 0.5)}{P(X_1 = 6)} = 0.5401758 \\ P(\Theta = 0.65 | X_1 = 6) &= \frac{P(X_1 = 6, \Theta = 0.65)}{P(X_1 = 6)} = 0.3130094 \end{aligned}$$

We compute these expectations for each data point and add them up. The added expectations give us the expected sufficient statistics of the categorical over mixture components for this data set. This completes the E-step.

In the M-step we assume that these expected values are the actual counts of how often we have observed each latent value. Let us call the counts $c_{0.4}, c_{0.5}, c_{0.65}$ where the index points to the corresponding mixture component. According to our model, the mixture components are categorically distributed and thus in the M-step we want to find the MLE of that categorical. In general, the MLE for θ_i of a categorical is $\frac{c_i}{n}$ ¹. In our case $n = 20$. Thus we set $\theta_i^{(1)} = \frac{c_i}{20}$ and complete the M-step. With our new parameter estimates, we can proceed to the second iteration of EM.

For our concrete example, we have already computed the posterior distribution when there are 6 successes in 10 coin flips. Our data also contains

¹This fact can easily be seen by letting c_i be the number of successes in the realisation of a binomial RV and the sum of all c_j , $j \neq i$ be the number of failures. Then clearly $\frac{c_i}{n}$ is the MLE.

Outcome	# occurrences	θ_1 (0.4)	θ_2 (0.5)	θ_4 (0.65)
2	2	0.5674795	0.41243	0.02009052
3	3	0.4568744	0.4980674	0.04505826
4	4	0.3436451	0.5619435	0.09441138
5	5	0.237068	0.581496	0.1814361
6	6	0.146814	0.5401758	0.3130094

Table 6.1: Posteriors of the coin flip data set from our EM example.

the outcomes 2,3,4 and 5. All posteriors are summarized in Table (6.1). The table also contains the number of occurrences of each outcomes in our data set. What is left is to add up the posteriors per data point. Since all outcomes occur multiple times, we can simply multiply the posterior probabilities for each outcome with its number of occurrences. For θ_1 this gives:

$$(6.11) \quad \mathbb{E}(t(y_1)) = 2 \times 0.5674795 + 3 \times 0.4568744 + 4 \times 0.3436451 + 5 \times 0.237068 + 6 \times 0.1114767 = 5.946387$$

By parallel calculations we get $\mathbb{E}(t(y_2)) = 10.7153$ and $\mathbb{E}(t(y_3)) = 3.338238$. These are our expected sufficient statistics. Importantly, we get $\mathbb{E}(t(y_1)) + \mathbb{E}(t(y_2)) + \mathbb{E}(t(y_3)) \approx 20$ (there is some slight numerical imprecision caused by our computer).

6.3 Basics of Information Theory

When we talk about *information*, we often use the term in qualitative sense. We say things like *This is valuable information* or *We have a lack of information*. We can also make statements about some information being more helpful than other. For a long time, however, people have been unable to quantify information. The person who succeeded in this endeavour was [Claude E. Shannon](#) who with his famous 1948 article *A Mathematical Theory of Communication* single-handedly created a new discipline: Information Theory! He also revolutionised digital communication and can be seen as one of the main contributors to our modern communication systems like the telephone, the internet etc.

The beauty about information theory is that it is based on probability theory and many results from probability theory seamlessly carry over to information theory. In this chapter, we are going to discuss the bare basics of information theory. These basics are often enough to understand many information theoretic arguments that researchers make in fields like computer science, psychology and linguistics.

Shannon's idea of information is as simple as it is compelling. Intuitively, if we are observing a realisation of a random variable, this realisation is

surprising if it is unlikely to occur according to the distribution of that random variable. However, if the probability for the realisation is very low, than on average it does not occur very often, meaning that if we sample from the RV repeatedly, we are not surprised very often. We are not surprised when the probability mass of the distribution is concentrated on only a small subset of its support.

On the other hand, we quite often are surprised, if we cannot predict what the outcome of our next draw from the RV might be. We are surprised when the distribution over values of the RV is (close to) uniform. Thus, we are going to be most surprised on average if we are observing realisations of a uniformly distributed RV.

Shannon's idea was that observing RVs that cause a lot of surprises is informative because we cannot predict the outcomes and with each new outcome we have effectively learned something (namely that the i^{th} outcome took on the value that it did). Observing RVs with very concentrated distributions is not very informative under this conception because by just choosing the most probable outcome we can correctly predict most actually observed outcomes. Obviously, if I manage to predict an outcome beforehand, it's occurrence is not teaching me anything.

The goal of Shannon was to find a function that captures this intuitive idea. He eventually found it and showed that it is the only function to have properties that encompass the intuition. This function is called the **entropy** of a RV and it is simply the expected **surprisal** value.

Definition 6.4 (Surprisal) *The surprisal (value) of an outcome $x \in \text{supp}(X)$ of some RV X is defined as $-\log_2(P(X = x))$.*

Notice that we are using the logarithm of base 2 here. This is because surprisal and entropy are standardly measured in bits. Intuitively, the surprisal measures how many bits one needs to encode an observed outcome given that one knows the distribution underlying that outcome. The entropy measures how many bits one will need on average to encode an outcome that is generated by the distribution P_X .

Definition 6.5 (Entropy) *The entropy $H(P_X)$ of a RV X with distribution P_X is defined as*

$$H(P_X) := \mathbb{E}[-\log_2(P(X = x))] = - \sum_{x \in \text{supp}(X)} P(X = x) \log_2(P(X = x)).$$

For the ease of notation, we often write $H(X)$ instead of $H(P_X)$.

Figure 6.1 shows the entropy of the Bernoulli distribution as a function of the parameter θ . The entropy function of the Bernoulli is often called the **binary entropy**. It measures the information of a binary decision, like a

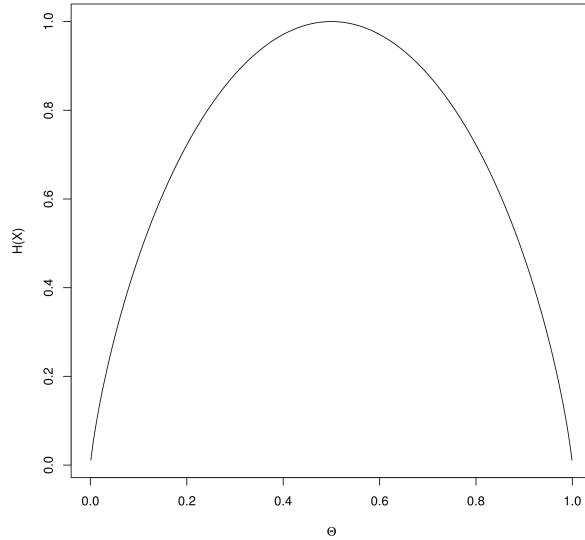


Figure 6.1: Binary entropy function.

coin flip or an answer to a yes/no-question. The entropy of the Bernoulli is 1 bit when the distribution is uniform, i.e. when both choices are equally probable.

From the plot it is also easy to see that entropy is never negative. It holds in general that entropy is non-negative, because entropy is defined as expectation of surprisal and surprisal is the negative logarithm of probabilities. Because $\log(x) \leq 0$ for $x \in (0, 1]$, it is clear that $-\log(x) \geq 0$ for x in the same interval. Notice that from here on we drop the subscript and by convention let $\log = \log_2$.

A standard interpretation of the entropy is that it quantifies uncertainty. As we have pointed out before, a uniform distribution means that you are most uncertain and indeed the uniform distribution maximizes the entropy. However, the more choices you have to pick from, the more uncertain you are going to be. The entropy function also captures this intuition. Notice that if a discrete distribution is uniform, all probabilities are $\frac{1}{|\text{supp}(X)|}$. Clearly, as we increase $|\text{supp}(X)|$, we decrease the probabilities. By decreasing the probabilities, we increase their negative logarithms, and hence their surprisal. Let us make this intuition more formal.

Theorem 6.6 *A discrete RV X with uniform distribution and support of size n has entropy $H(X) = \log(n)$.*

Proof:

$$(6.12) \quad H(X) = \sum_{x \in \text{supp}(X)} -\log(P(X = x))P(X = x)$$

$$(6.13) \quad = \sum_{x \in \text{supp}(X)} \log(n)P(X = x) = \log(n). \quad \square$$

Exercise 6.7 *You are trying to learn chess and you start by studying where chess grandmasters move their king when it is positioned in one of the middle fields of the board. The king can move to any of the adjoining 8 fields. Since you do not know a thing about chess yet, you assume that each move is equally probable. In this situation, what is the entropy of moving the king?*

At the outset of this section we promised you that you could easily transfer results from probability theory to information theory. We will not be able to show any kind of linearity for entropy because it contains log-terms and the logarithm is not linear. We can however find alternative expressions for joint entropy (where the joint entropy is simply the entropy of a joint RV). Before we do so, let us also define the notion of conditional entropy. We have seen in Section ?? that $P_{X|Y=y}$ is a valid probability distribution for any $y \in \text{supp}(Y)$ such that $P(Y = y) > 0$. Hence, we can also define its conditional entropy.

Definition 6.8 (Conditional Entropy) *For two jointly distributed RVs X, Y and $y \in \text{supp}(Y)$ such that $P(Y = y) > 0$, the conditional entropy of X given that $Y = y$ is defined as*

$$\begin{aligned} H(X|Y = y) &:= \mathbb{E}_X[-\log_2(P(X = x|Y = y))] \\ &= - \sum_{x \in \text{supp}(X)} P(X = x|Y = y) \log_2(P(X = x|Y = y)). \end{aligned}$$

The conditional entropy of X given Y is defined as

$$H(X|Y) := \mathbb{E}_Y[H(X|Y)] = \sum_{y \in \text{supp}(Y)} P(Y = y)H(X|Y = y).$$

With this definition at hand we show that the joint entropy decomposes according to the chain rule.

$$\begin{aligned}
H(X, Y) &= \sum_{\substack{x \in \text{supp}(X) \\ y \in \text{supp}(Y)}} -\log(P(X = x, Y = y)) \times P(X = x, Y = y) \\
&= \sum_{\substack{x \in \text{supp}(X) \\ y \in \text{supp}(Y)}} -\log(P(X = x|Y = y)) \times P(X = x, Y = y) \\
&\quad - \sum_{y \in \text{supp}(Y)} \log(P(Y = y)) \times \sum_{x \in \text{supp}(X)} P(X = x, Y = y) \\
&= \sum_{y \in \text{supp}(Y)} P(Y = y) \times \sum_{x \in \text{supp}(X)} -\log(P(X = x|Y = y)) \times P(X = x|Y = y) \\
&\quad - \sum_{y \in \text{supp}(Y)} \log(P(Y = y)) \times P(Y = y) \\
&= H(X|Y) + H(Y)
\end{aligned}$$

Exercise 6.9 Prove that $H(X, Y|Z) = H(X|Z) + H(Y|Z)$ if $X \perp Y|Z$.

Now that we have seen some information-theoretic concepts, you may be happy to hear that there is an information-theoretic interpretation of EM. This interpretation helps us to get a better intuition for the algorithm. To formulate that interpretation we need one more concept, however.

Definition 6.10 (Relative Entropy) The relative entropy of RVs X, Y with distributions P_X, P_Y and $\text{supp}(X) \subseteq \text{supp}(Y)$ is defined as

$$D(P_X||P_Y) := \sum_{x \in \text{supp}(X)} P(X = x) \log \frac{P(X = x)}{P(Y = x)}.$$

If $P(X = y) = 0$ for any $y \in \text{supp}(Y)$ we define $D(P_X||P_Y) = \infty$. As with entropy, we often abbreviate $D(P_X||P_Y)$ with $D(X||Y)$.

The relative entropy is commonly known as **Kullback-Leibler (KL)** divergence. It measures the entropy of X as scaled to Y . Intuitively, it gives a measure of how “far away” P_X is from P_Y . To understand “far away”, recall that entropy is a measure of uncertainty. This uncertainty is low if both distributions place most of their mass on the same outcomes. Since $\log(1) = 0$ the relative entropy is 0 if $P_X = P_Y$.

It is worthwhile to point out the difference between relative and conditional entropy. Conditional entropy is the average entropy of X given that you know what value Y takes on. In the case of relative entropy you do not know the value of Y , only its distribution.

Exercise 6.11 *Show that $D(X, Y||Y) = H(X|Y)$. Furthermore show that $D(X, Y||Y) = H(X)$ if $X \perp Y$.*

Further Material

At the ILLC, the best place to learn more about information theory is [Christian Schaffner's course](#) that is taught every year. David MacKay also offers [a free book on the subject](#). Finally, Coursera also offers [an online course on information theory](#).

To get a better understanding of EM and the other concepts discussed in this script along with some more examples, consult [Micheal Collins' lecture notes](#) .