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 to 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 **mixture model**.

Definition 6.1 (Mixture Model) *Given any set of probability distributions $P_{X1}, \ldots, P_{Xn}$ that are defined over the same random variable X we define a mixture model as $P = \sum_{i=1}^{n} \alpha_i P_{Xi}$ where we require that $\alpha_i \geq 0$ and $1 \leq i \leq n$ and $\sum_{i=1}^{n} \alpha_i = 1$. We call the distributions $P_{X1}, \ldots, P_{Xn}$ **mixture components** and their weights $\alpha_i, 1 \leq i \leq n$ **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 will use for the rest of this section.

Example of a mixture model Assume we observe 10 sequences of coin tosses. Each sequence contains 10 flips. We also know that there are 3 coins from which these sequences could possibly be 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 use a multinomial distribution over coins to model our data. We would then employ maximum likelihood estimation and pick the coin that has the highest likelihood. However, this model might actually turn out to be pretty bad because we are committing to picking only one coin, although 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 10 sequences, we assume that all three coins have contributed to generating the 10 sequences. However, their contributions may be unequal. This is exactly what the mixture weights capture. We will, as usual, call our data x. Also, each mixture component will be a binomial distribution, parametrized by the parameters of the coins. This is enough to formulate our mixture model.

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

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 the opposite problem with mixture models. 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. This means that we can interpret them as a probability distribution. In particular, this will be a distribution over parameters of the mixture components (assuming that all mixture components have the same parametric form). That is, we get the following equivalences:

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

This means that we can rewrite our mixture model as

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

Notice that in our examples we assume that we have only three possible binomial parameters, namely $\theta_1 = 0.4, \theta_2 = 0.5, \theta_3 = 0.65$. Notice further that each summand in (6.3) is in fact a joint distribution by the chain rule

so that we can again rewrite the model to

$$P(X = x) = P(X = x, \Theta_1 = 0.4) + P(X = x, \Theta_2 = 0.5) + P(X = x, \Theta_3 = 0.65) \tag{6.4}$$

$$= \sum_{i=1}^{3} P(X = x, \Theta = \theta_{c_i}) \tag{6.5}$$

where line (6.5) 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 are definable 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 $\alpha_i P_i(X = x) = P(X = x)$ if α_i are mixture weights as defined in 6.1.*

Notice that we have shown above that the formulation of mixture models as purely probabilistic model 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 us to estimate them using the maximum likelihood principle. This 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 their priors, the mixture weights, are unknown. In other words, we can not simply apply calculus as we have been doing up to now. For this reason, we will turn to the **EM algorithm**, that allows us to at least find a local maximum of the likelihood function.

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 us to find a local maximum of the likelihood function of mixture models or, more generally, models with missing data.

Let us quickly introduce the idea of missing data: Assume you run a website the 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 for you 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 they are lacking a rating in one category. Thus you have a data set with some missing data that you have to fill in somehow.

In mixture models the missing data can be thought of as annotations that tell you which mixture component generated a given data point. If you had this information, you could simply do maximum likelihood estimation. However, since you do not know which mixture component generated your data point, you can only assume that all mixture components may have done so with a probability that is given by the posterior distribution 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 simply 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 mixture components. In the more general case of missing data, Y is called **latent** or **hidden variable** because it is not observed. Then the likelihood function is

$$L_x(\theta) = P(X = x|\Theta = \theta) = \sum_{i=1}^{m} P(X = x, Y = y|\Theta = \theta) \tag{6.6}$$

where Θ ranges over the parameters of the joint distribution P_{XY}. Let $t(y)$ be the statistic that allows for the computation of the MLE. For all distributions that we have seen so far and in fact for all distributions in the exponential family, this statistic is the number of times we observe a particular outcome y. The formulation that we give of the EM algorithm is specific for exponential family distributions and can be made more specific. 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.

Given some estimate $\overline{\theta}^{(i)}$ for the parameters of the joint distribution compute the expected value of the statistic $t(y)$.

$$t(y)^{(i+1)} = \mathbb{E}(t(y)|X = x, \overline{\theta}^{(i+1)}) \tag{6.7}$$

Observe that we introduced superscripts. The EM algorithm is an iterative algorithm, meaning we repeat its steps several times. We use the the superscript to indicate the number of the repetition, thus $i \in \mathbb{N},\ 0 \leq i \leq m$. We percolate information from previous iterations by conditioning on their parameter estimates; $\overline{\theta}^{(0)}$ can be set arbitrarily.

Equation (6.7) is know as the **E(xpectation)-step** of the EM algorithm. To make the algorithm complete, we are still lacking a **M(aximization)-step**. But this one is simple. We pretend that the expected statistics of the latent data were actually observed. If our statistics are counts, then we simply pretend that we observed outcome y exactly $t(y)^{(i+1)}$ times. Notice that since $t(y)^{(i+1)}$ is an expectation, it is a real and not a natural number. Once we pretend to observe the expected statistics, the maximization step can done using be maximum likelihood estimation.

$$\overline{\theta}^{(i+1)} = \arg\max_{\theta} P(X = x, t(Y) = t(y)|\theta) \tag{6.8}$$

Definition 6.3 (EM algorithm) *We assume a data set $x = x_1^n$ and postulate that there is unobserved data $y = y_1^m$ that belongs to that data set. We also assume a probabilistic model whose parameters are realisations of a RV Θ. Take any statistic $t(y)$ of the latent data. Then any iterative algorithm with k iterations $i \leq k; i, k \in \mathbb{N}$ that performs the steps*

$$\begin{array}{lll} \textbf{\textit{E-step}} & t(y)^{(i+1)} & = \mathbb{E}(t(y)|X = x, \Theta = \overline{\theta}^{(i+1)}) \\ \textbf{\textit{M-step}} & \overline{\theta}^{(i+1)} & = arg\ \underset{\theta}{max}\ \ P(X = x, t(Y) = t(y)|\Theta = \theta) \end{array}$$

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 $x = x_1^{10}$ where each x_i is the number of heads that we observed in a sequence of ten 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 $supp(Y) = \{0.4, 0.5, 0.65\}$. Both the observed and latent variables are assumed i.i.d. We will 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). Now let us take a closer look a our data. To shorten notation, we write it as a list where the ith entry is the value of x_i.

$$[6, 4, 7, 5, 6, 3, 5, 5, 7, 3]$$

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.

$$
\begin{aligned}
P(X_1 = 6|\Theta = 0.4) &= 0.1114767 \\
P(X_1 = 6|\Theta = 0.5) &= 0.2050781 \\
P(X_1 = 6|\Theta = 0.65) &= 0.2376685
\end{aligned} \tag{6.9}
$$

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.

$$
\begin{aligned}
P(X_1 = 6, \Theta = 0.4) &= 0.02786918 \\
P(X_1 = 6, \Theta = 0.5) &= 0.1025391 \\
P(X_1 = 6, \Theta = 0.65) &= 0.05941712
\end{aligned} \tag{6.10}
$$

We are interested in the expected number of times that each coin has generated x_1. Since x_1 has been generated by exactly one coin, the highest expected count is 1. To see this, simply imagine that each sequence of tosses is padded with one more number indicating the coin used to produce the sequence. Then clearly, this coin is observed once as a sequence generator for every sequence that it is padded to.

In order to compute an expectation we first need a distribution over the mixture components. This distribution is simply the posterior. The posterior given x_1 is shown below.

$$
\begin{aligned}
P(\Theta = 0.4|X_1 = 6) &= 0.146814 \\
P(\Theta = 0.5|X_1 = 6) &= 0.5401758 \\
P(\Theta = 0.65|X_1 = 6) &= 0.3130094
\end{aligned} \tag{6.11}
$$

Now we can compute how often we expect each value of $supp(Y)$ to occur. We will use an indicator function ($I_a(x) = 1$ if $a = x$; 0 otherwise) for this. The probability of observing the latent value together with x_1 follows the posterior distribution in (6.11). This means that $\mathbb{E}[I_{0.4}(\theta)] = P(\Theta = 0.4|X_1 = 6)$ and likewise for the other two mixture components.

We compute these expectations for each data point and add them up. This gives us the expectations over the whole data set x and 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 multinomially distributed and thus in the M-step we want to find the MLE of that multinomial. In general, the MLE for θ_i of a multinomial is $\frac{c_i}{n}$[1]. In

[1]This 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.

our case $n = 10$. Thus we set $\theta_i^1 = \frac{c_i}{n}$ and complete the M-step. With this as our new parameter estimates, we can now proceed to the second iteration of EM.

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 basic will often be 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 will surprise 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 will not occur very often, meaning that if we sample from the RV repeatedly, we will not be surprised very often. This happens when a lot of probability mass is concentrated on only one or very few outcomes.

On the other hand, we will quite often be surprised if we cannot predict what the outcome of our next draw from the RV might be. This is exactly the case 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 of an outcome $x \in supp(X)$ of some RV X is*

$$-\log_2(P(X = x)) \ .$$

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 of a RV X is $H(X)$ where $H : X \to \mathbb{R}_{0+}$ is defined as*

$$H(X) := \mathbb{E}[-\log_2(P(X = x))] \ .$$

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 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 is it also easy to see that entropy is never negative. This is true for all entropy functions, because they are just expectations 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 will 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 this is when the entropy is highest. 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 $\frac{1}{|supp(X)|}$. Clearly, as we increase $|supp(X)|$, we will decrease the probabilities. By decreasing the probabilities, we increase their negative logarithms, and hence their surprisal. Let us make this more formal.

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

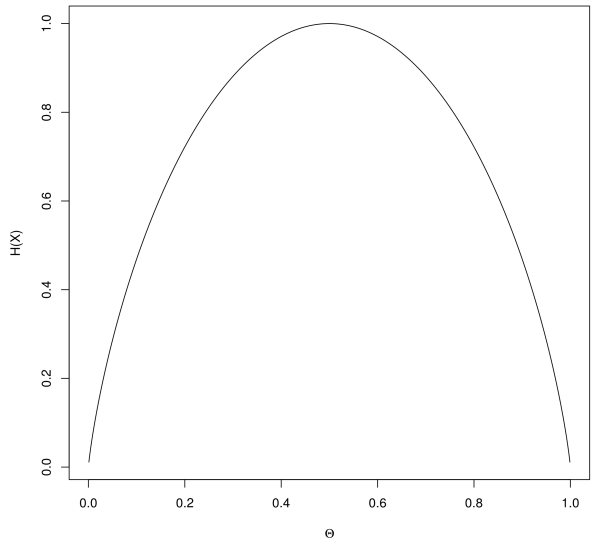

Figure 6.1: Binary entropy function.

Proof:

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

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

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 the centre 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.

Definition 6.8 (Conditional Entropy) *For two jointly distributed RVs X, Y the conditional entropy of X given that Y is defined as*

$$H(X|Y) := \mathbb{E}[-\log_2(P(X = x|Y = y))] \ .$$

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

(6.14)
$$H(X,Y) = \sum_{x \in supp(X), y \in supp(Y)} -\log(P(X = x, Y = y)) \times P(X = x, Y = y)$$

(6.15)
$$= \sum_{x \in supp(X), y \in supp(Y)} -\log(P(X = x|Y = y)) - \sum_{y \in supp(Y)} \log(P(Y = y)) \times P(Y = y)$$

(6.16)
$$= H(X|Y) + H(Y)$$

Exercise 6.9 *Proof 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 also is an information-theoretic interpretation of EM. This interpretation will also help us to get a better intuition for the algorithm. To formulate that interpretation we will need one more concept, however.

Definition 6.10 (Relative Entropy) *The relative entropy of RVs X, Y with $supp(X) \subseteq supp(Y)$ is defined as*

$$D(X||Y) = \mathbb{E}_{P_X}\left[\frac{P(X = x)}{P(Y = x)}\right] \ .$$

If $supp(X) \not\subseteq supp(Y)$ we define $D(X||Y) = \infty$.

The relative entropy is also 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. The relative entropy measure the uncertainty that you have about P_X if you know P_Y. This uncertainty will be 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 Christan 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 .