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) *We assume jointly distributed random variables $X = X_1^n$ and $Y = Y_1^n$ where Y is categorical and $\mathcal{Y} = \{c_1, \ldots, c_k\}$. The X_i are observed data, the Y_i are latent or observed and the $c_j, 1 \leq j \leq k$ are called mixture components. If the distribution $P(X_1^n = x_1^n, Y_1^n = y_1^n)$ factors as*

$$P(X_1^n = x_1^n, Y_1^n = Y_1^n) = P(Y_1^n = y_1^n) \prod_{i=1}^{n} P(X_i = x_i | Y_i = y_i) \ .$$

we call this model a mixture model.

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. We want to find out what the biases of the coins are (i.e. the parameters of the binomial distributions induces by the coins) and the probability for each coin to have generated a particular sequence.

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. We will also adopt the often-used assumption that the mixture components are independent of each other. In the present case this means that the coin that generated each sequence of coin flips was chosen independently of the coins used for the other sequences.

In order to model the contribution of each coin, we introduce at latent RV C over mixture components with values c_1, c_2, c_3 representing the three coins. In the present case, each mixture component is linked to a binomial distribution, parametrized by the biases of the coins. As usual, we call our data x. This information suffices to formulate our mixture model.

$$
\begin{aligned}
(6.1) \qquad P(X = x|C_1^3 = c_1^3) = \alpha_1 P(X = x|C = c_1) \\
+ \alpha_2 P(X = x|C = c_2) + \alpha_3 P(X = x|C = c_3)
\end{aligned}
$$

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 separately. 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. In this case, doing straighforward MLE is impossible

because the data are incomplete[1].

How can we go about estimating the model parameters, i.e. the parameters of the mixture components and the mixture 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 mixture components. That is, we get the following relations:

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

Hence, we can rewrite our mixture model as

$$\begin{aligned} P(X = x) &= P(C = c_1)P(X = x|C = c_1) \\ &\quad + P(C = c_2)P(X = x|C = c_2) + P(C = c_3)P(X = x|C = c_3) \end{aligned} \tag{6.3}$$

Each summand in (6.3) is in fact a joint distribution by the chain rule so that we can again rewrite the model as

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

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

[1] In general, whenever you have complete data, analytic MLE computation is possible. The moment your data is incomplete, it becomes impossible.

(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 the latent (missing) data, 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 $X = x$. Furthermore, define Y as a random variable over latent data that can take on $|Y|$ possible values. Then the likelihood function is

$$L_x(\theta) = P(X = x|\Theta = \theta) = \sum_{i=1}^{|Y|} P(X = x, Y = y_i|\Theta = \theta) \tag{6.5}$$

where Θ ranges over the parameters of the joint distributiforon 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*.

Because 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.

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 $\theta^{(t)}$ (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 $t(y)$.

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

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 be maximum-likelihood estimation:

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

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

E-step: $t(y)^{(i+1)} = \mathbb{E}(t(C,x) \mid X=x, \Theta=\theta^{(i)})$

M-step: $\theta^{(i+1)} = \arg\max_{\theta} P(X=x, t(C,x)=t(C,x)^{(i+1)} \mid \Theta=\theta)$

to update the model parameters is called an EM algorithm.

6.3 Example of an EM Algorithm for a Mixture Model

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 representing three coins which are linked to binomials with parameters representing the biases of the coins. 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 $c=c_1^n$ where each c_i can be one of the three mixture components. We make the additional assumption that choosing a coin to generate a particular sequence is done independently of the coins chosen to generate all the other sequences. This has the effect that in our model the latent data points will be independent[2].

[2]The assumption that the mixture components in a mixture model are independent is actually quite common.

Initially we assume that the coins have biases of $0.4, 0.5$ and point 0.65, meaning that we initialize the binomial parameters of the data-generating distributions as follows:

$$\theta_{c_1}^{(0)} = 0.4 \quad \theta_{c_2}^{(0)} = 0.5 \quad \theta_{c_3}^{(0)} = 0.65 \tag{6.8}$$

We also 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 ith 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 one of three coins. In order to compute the posterior for each coin given a data point, we need the likelihood for that data point. For the first observation we get the following likelihood values.

$$\begin{aligned} P(X_1 = 6|Y_1 = c_1) &= P(X_1 = 6|\Theta = 0.4) = 0.1114767 \\ P(X_1 = 6|Y_1 = c_2) &= P(X_1 = 6|\Theta = 0.5) = 0.2050781 \\ P(X_1 = 6|Y_1 = c_3) &= P(X_1 = 6|\Theta = 0.65) = 0.2376685 \end{aligned} \tag{6.9}$$

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.

$$\begin{aligned} P(X_1 = 6, Y_1 = c_1) &= 0.25 \times P(X_1 = 6|\Theta = 0.4) = 0.02786918 \\ P(X_1 = 6, Y_1 = c_2) &= 0.5 \times P(X_1 = 6|\Theta = 0.4) = 0.1025391 \\ P(X_1 = 6, Y_1 = c_3) &= 0.25 \times P(X_1 = 6|\Theta = 0.4) = 0.05941712 \end{aligned} \tag{6.10}$$

We are ultimately 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. This is to say that $\mathbb{E}[Y|X_1 = 6, \Theta = \theta] = \mathbb{E}[t(Y, 6)|X_1 = 6, \Theta = \theta]$. The posterior given x_1 is shown below.

$$\begin{aligned} P(Y_1 = c_1|X_1 = 6) &= \frac{P(X_1 = 6, Y_1 = c_1)}{P(X_1 = 6)} = 0.146814 \\ P(Y_1 = c_2|X_1 = 6) &= \frac{P(X_1 = 6, Y_1 = c_2)}{P(X_1 = 6)} = 0.5401758 \\ P(Y_1 = c_3|X_1 = 6) &= \frac{P(X_1 = 6, Y_1 = c_{3)})}{P(X_1 = 6)} = 0.3130094 \end{aligned} \tag{6.11}$$

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.

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 $\frac{c_i}{n}$[3]. 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 their are 6 successes in 10 coin flips. Our data also contains 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:

$$\begin{aligned}\mathbb{E}(t(y_1)) =& 2 \times 0.5674795 + 3 \times 0.4568744 + 4 \times 0.3436451 + \\ & 5 \times 0.237068 + 6 \times 0.1114767 = 5.946387\end{aligned} \tag{6.12}$$

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).

[3]This 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.