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. The marginal probabilities $P(Y_i =$

$y_i)$ *for* $1 \leq i \leq n$ *are called 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. The modelling of the data under that view is done by the conditional distribution induced by the mixture component. This technique 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 associated with 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 one coin only. This is a bad assumption, because we know that each sequence may 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.

In order to model the contribution of each coin, we introduce a latent RV Y 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. We will also adopt the often-used assumption that the mixture components are independent of each other. This means that the coin that generated each sequence of coin flips was chosen independently of the coins used for the other sequences. As usual, we call our data x. This information suffices to formulate our mixture model.

$$
\begin{aligned}
(6.1) \qquad & P(X = x_1^n, Y_1^n = y_1^n \mid \Theta = \theta) \\
& = \prod_{i=1}^{n} P(Y_i = y_i | \Theta = \theta) P(X_i = x_i | Y_i = y_i, \Theta = \theta)
\end{aligned}
$$

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 is incomplete **Chris:** It seems to me that *incomplete* is not defined formally enough. What do we mean by it? **Philip:** I extended the footnote with an explanation. Does that work?[1].

How can we go about estimating the model parameters, i.e. the parameters of the mixture components and the mixture weights? Each factor in (6.1) is in fact a joint distribution (by the chain rule). We exploit this to rewrite the model as a model of the observed data only.

$$P(X = x_1^n | \Theta = \theta) = \prod_{i=1}^{n} \sum_{j=1}^{3} P(X_i = x_i, Y_i = c_j | \Theta = \theta) \tag{6.2}$$

Philip: Compare this to Equation (6.1). There we constructed a probability distribution over the joint outcomes X_1^n and Y_1^n. Because we assumed independence between mixture components, the distribution could be factorised over pairs $(X_i, Y_i$ for $1 \leq i \leq n$. In Equation (6.2) we then inferred the marginal distribution for X_1^n. Recall that in order to compute the marginal of X_1^n we need to sum over all possible Y_1^n. Because their joint distribution factorises nicely, we can perform the marginalisation per $(X_i, Y_i$ pair. This is exactly what is happening in Equation (6.2). Please pay attention to a crucial notational difference between (6.1) and (6.2): In the former we wrote $Y_i = y_i$ whilst in the latter we wrote $Y_i = c_j$. Why did we do this? Equation (6.1) is a joint distribution and thus Y_i needs to take on a *specific* value $y_i \in \{c_1, c_2, c_3\}$. To get the marginal distribution of X_i, however, we need to sum over *all* values that Y_i can take on. We do this by writing $Y_i = c_j$ which provides us with a means of summing over the possible outcomes c_1, c_2, c_3. **Chris:** this is going too fast! I don't understand it, you also switched from y_1^n to c_j somehow... where did you use (6.1) here?? **Philip:** Sorry, the conditioning was wrong in 6.1. Please check whether the above explanation helps you to understand what's going on now.

Philip: We thus see that the sums are marginalisation steps. Now that we have related mixture models to the probability of the observed data, we can hope to estimate their parameters using the maximum likelihood principle.

[1]Incomplete data is another name for a collection of random variables of which some have unobservable (latent) outcomes. Data modelled with a mixture model is incomplete because it contains an observed part (the number of heads in our example) and a latent part (the mixture components which in our example are the coins). In general, whenever you have complete data, analytic MLE computation is possible. The moment your data is incomplete, it becomes impossible.

There is one remaining 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.

6.2 The EM Algorithm

In order to estimate the parameters of mixture models, we can employ a classical algorithm of **unsupervised learning**[2], 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.

Another example of latent 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. Thus you have a data set with some missing data that you have to fill in somehow.

[2]Whenever the feature of our data that we want to predict (such as the coin which generated a sequence) is observed, we speak of supervised learning. If the target feature is latent, we speak of unsupervised learning.

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 a data point x_i. 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 x_i.

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: first, compute the expected number of occurrences of the missing data values (the mixture components). Then treat these expectations as observed and do maximum likelihood estimation as usual[3]. 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

$$L_x(\theta) = P(X = x|\Theta = \theta) = \sum_{j=1}^{|Y|} P(X = x, Y = y_j|\Theta = \theta) \tag{6.3}$$

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*[4].

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

[3]Notice that expectations can be fractional **Chris:** what does "fractional" mean here? **Philip:** i.e. take on non-integer values and thus when treating them as observations we are handling a dataset with fractional observations. While this may be conceptually awkward it does not change the mathematics of maximum likelihood estimation.

[4] 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.

can be set arbitrarily). Based on this parameter estimate we then compute the expected sufficient statistics of our model which we will call $t(y, x)$.

$$t(y,x)^{(l+1)} = \mathbb{E}(t(Y,x)|X = x, \Theta = \theta^{(l)}) \tag{6.4}$$

Equation (6.4) 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:

$$\theta^{(l+1)} = \arg\max_{\theta} P(X = x, t(Y,X) = t(y,x)^{(l+1)}|\theta) \tag{6.5}$$

Definition 6.2 (EM algorithm) *We assume a data set x and postulate that there is unobserved data y. We also assume a probabilistic model $P(X = x, Y = y|\Theta = \theta)$ whose parameters are realisations of a RV Θ. Let $t(x, y)$ be the sufficient statistics for that model. Then any iterative algorithm with m iterations that performs the following steps for $0 \leq l \leq m - 1$,*

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

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

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 $y = y_1^n$ where each y_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[5].

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

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

$$\theta_{c_1}^{(0)} = 0.4 \quad \theta_{c_2}^{(0)} = 0.5 \quad \theta_{c_3}^{(0)} = 0.65 \tag{6.6}$$

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.

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

Then for each x_i we check how likely each coin is to have generated that point. 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.

$$\begin{aligned} &P(X_1 = 6|Y_1 = c_1) = P(X_1 = 6|Y_1 = c_1, \Theta = 0.4) = 0.1114767 \\ &P(X_1 = 6|Y_1 = c_2) = P(X_1 = 6|Y_2 = c_2, \Theta = 0.5) = 0.2050781 \\ &P(X_1 = 6|Y_1 = c_3) = P(X_1 = 6|Y_3 = c_3, \Theta = 0.65) = 0.2376685 \end{aligned} \tag{6.7}$$

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.

$$\begin{aligned} &P(X_1 = 6, Y_1 = c_1) = 0.25 \times P(X_1 = 6|Y_1 = c_1, \Theta = 0.4) = 0.02786918 \\ &P(X_1 = 6, Y_1 = c_2) = 0.5 \times P(X_1 = 6|Y_2 = c_2, \Theta = 0.5) = 0.1025391 \\ &P(X_1 = 6, Y_1 = c_3) = 0.25 \times P(X_1 = 6|Y_3 = c_3, \Theta = 0.65) = 0.05941712 \end{aligned} \tag{6.8}$$

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 $\mathbb{E}[Y = c_j|X_1 = 6, \Theta = \theta^0] = P(Y_1 = c_j|X_1 = 6, \Theta = \theta^0)$. The posterior given x_1 is shown below.

$$\begin{aligned} P(Y_1 = c_1|X_1 = 6, \Theta = \theta^0) &= \frac{P(X_1 = 6, Y_1 = c_1|\Theta = \theta^0)}{P(X_1 = 6|\Theta = \theta^0)} = 0.146814 \\ P(Y_1 = c_2|X_1 = 6, \Theta = \theta^0) &= \frac{P(X_1 = 6, Y_1 = c_2|\Theta = \theta^0)}{P(X_1 = 6|\Theta = \theta^0)} = 0.5401758 \\ P(Y_1 = c_3|X_1 = 6, \Theta = \theta^0) &= \frac{P(X_1 = 6, Y_1 = c_3|\Theta = \theta^0)}{P(X_1 = 6|\Theta = \theta^0)} = 0.3130094 \end{aligned} \tag{6.9}$$

outcome	#occurrences	c_1	c_2	c_3
2	4	0.5674795	0.4124300	0.0200905
3	2	0.4568744	0.4980674	0.0450583
4	5	0.3436451	0.5619435	0.0944114
5	5	0.2370680	0.5814960	0.1814361
6	4	0.1468149	0.5401758	0.3130094

Table 6.1: Posteriors per outcome for the mixture components of the coin flip data set from our EM example.

We compute these expectations over mixture components for each data point and add them up. The reason we can simply add the expectation is that the latent variables are independent and consequently their expectations are independent. If this was not the case, simple adding would not be possible. How exactly the expectations can be accumulated depends on how the model's distribution over latent variables factorises. This has to be handled on a case-by-case basis. For our examples the expectations for each outcome can be found in Table 6.1.

Once the expectations have been added, we need to compute the sufficient statistics for our model. First of all notice that we are dealing with a mixture model where the latent variables are always categorical. Thus, in order to update the parameters of P_Y we need to find the sufficient statistics for a categorical distribution. Recall that these sufficient statistics are simply the counts per outcome observed in the data. Thus the expected sufficient statistics for a categorical are the expected counts per outcome. But these are simply the posterior probabilities! Thus we need to multiply the posteriors per of each observed outcome with the number of times this outcome was seen in the data and sum over all observed outcomes. For mixture component c_1 (the first coin with parameter $\theta^0_{c_1} = 0.4$) this gives

$$\begin{aligned}(6.10)\qquad &\mathbb{E}(Y = c_1|X = x, \Theta = \theta^0) = 2 \times 0.5674795 + 3 \times 0.4568744 \\ &+4 \times 0.3436451 + 5 \times 0.237068 + 6 \times 0.1114767 = 5.946387\ .\end{aligned}$$

By parallel calculations we get $\mathbb{E}(Y = c_2 \mid X = x, \Theta = \theta^0) = 10.7153$ and $\mathbb{E}(Y = c_3 \mid X = x, \Theta = \theta^0) = 3.338238$. These are our expected sufficient statistics. Importantly, we get $\mathbb{E}[Y = c_1 \mid X = x, \Theta = \theta^0] + \mathbb{E}[Y = c_2 \mid X = x, \Theta = \theta^0] + \mathbb{E}[Y = c_3 \mid X = x, \Theta = \theta^0] \approx 20$ (there is some slight numerical imprecision caused by our computer). Notice that this is a useful debugging technique when implementing the algorithm: if the expected number of mixture components does not add up to the number of latent variables in your model, then you almost certainly have a bug in your code!

Next we turn to the sufficient statistics for the binomial distributions linked to the mixture components. These are the counts of the observed outcomes. However, since the binomials are conditional distributions (they

are conditioned on the identity of the coins), the counts have to be taken with respect to their conditioning contexts. In other words: we cannot count the observed outcomes independently but we have to count pairs of observed and latent variables. Formally this means that for one binomially distributed observation x_i that $\mathbb{E}[X = x_i, Y_i = c_j|\Theta = \theta^0] = P(Y_i = c_j|X_i = x_i, \Theta = \theta^0)$. This is again just the posterior that we find in Table 6.1.

To get the expectations for the mixture components, we summed the columns in Table 6.1, effectively conflating all outcomes of X. We did this because we did not care about X at this stage, only about the expectations of Y. When computing the posteriors for the outcomes given the mixture components, we have to sum the posteriors differently. Now we actually need to discriminate between observed outcomes. We only sum over observations that had the same outcome. Working with Table 6.1 this means that we multiply each cell, which contains the posterior for one observation, by the number of times each outcome has been observed in the data set. In other words, we are summing the posteriors over the observations that were equal to the given outcome. For the pairs $(X = 2, Y = c_1)$ and $(X = 4, Y = c_3)$ this gives:

$$\mathbb{E}[X = 2, Y = c_1|\Theta = \theta^0_{c^1}] = 4 \times 0.5674795 = 2.269918 \tag{6.11}$$

$$\mathbb{E}[X = 4, Y = c_3|\Theta = \theta^0_{c^3}] = 5 \times 0.0944114 = 0.472057 \tag{6.12}$$

The expected number of times each outcome has been drawn from each coin can be found in Table 6.2.

Recall that the sufficient statistic for the binomial is $\sum_{i=1}^{n} x_i$ where x_i is the i^{th} Bernoulli trial of that binomial. When dealing with m i.i.d. binomial draws, this becomes $\sum_{i=1}^{nm} x_i$. Notice that we do not know m because we don't actually know how many draws stem from each coin. Instead, we use the expected number of times that each coin (mixture component) was used. We have already computed these expectations and noted them down in Table 6.1. Thus, for coin c_1 the expected sufficient binomial statistic is

$$\mathbb{E}\left[\sum_{i=1}^{nm} x_i \mid c_1\right] = 2 \times 2.2699180 + 3 \times 0.9137487 + 4 \times 1.7182254 + 5 \times 1.1853398 + 6 \times 0.5872594 = 23.60424 \tag{6.13}$$

The expected sufficient statistics for all coins are given in Table 6.3.

That we have computed all sufficient statistics for our model means that we have completed the E-step!

M-step As pointed out before, the M-step is rather trivial once we have all the necessary expectations. Recall that 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_j of a

outcome	c_1	c_2	c_3
2	2.2699180	1.6497199	0.0803621
3	0.9137487	0.9961348	0.0901165
4	1.7182254	2.8097177	0.4720569
5	1.1853398	2.9074798	0.9071804
6	0.5872594	2.1607030	1.2520376

Table 6.2: Posterior expectations of the observed outcomes in the context of each mixture component.

c_1	c_2	c_3
23.60424	45.02833	14.36743

Table 6.3: Expected sufficient statistics for each or the three coins.

categorical is $\frac{\#c_j}{n}$[6]. In our case $n = 10$. Since we have not observed the latent variables, we simply use their expected sufficient statistics (their expected counts) instead. Thus we set $\theta_j^{(1)} = \frac{\mathbb{E}[Y=c_j|X=x,\Theta=\theta^0]}{20}$[7]. The updated mixture weights then are

$$
\begin{aligned}
P(Y = c_1|\Theta = \theta^1) &= 0.3337246 \\
P(Y = c_2|\Theta = \theta^1) &= 0.5261878 \\
P(Y = c_3|\Theta = \theta^1) &= 0.1400877
\end{aligned} \tag{6.14}
$$

We already know the maximum likelihood estimate for binomial distributions from Section**??**. The only novelty when it comes to EM for mixture models is that the conditioning contexts are not observed anymore. Again, we fall back on the expected context counts. The maximum likelihood estimate for an outcome x conditioned on mixture component c_j is

$$
\theta^1_{c_j} = \frac{\mathbb{E}[X = x|Y = c_j, \Theta = \theta^0]}{\sum_{x\in\mathcal{X}} \mathbb{E}[X = x|Y_i = c_j, \Theta = \theta^0]} \ . \tag{6.15}
$$

For example, if we want to find the conditional distribution for c_1, we need to compute the binomial MLE *only for the data associated with* c_1! We

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

[7] When implementing the algorithm, numerical imprecisions occur due the limited memory of our computers. To ensure that the distribution obtained from the M-step really sums to 1, you should use the sum $\sum_{j=1}^{3} \frac{\mathbb{E}[Y=c_j|X=x,\Theta=\theta^0]}{20}$. This sum should be equal to 20 up to at least the third decimal but may not be exactly equal to 20. Having a distribution that does not exactly sum to 1 may not be to bad after 1 M-step. However, since the M-step gets repeated after every iteration, the error introduced by numerical imprecision would grow exponentially, if we always divided by 20 instead of the above sum.

c_1	c_2	c_3
0.3536485	0.4278732	0.5128013

Table 6.4: New parameter values for the binomials associated with each mixture component.

already know from Equation (6.10) that the total expected occurrence of c_1 in our data is 5.946387. You can verify that this is exactly the sum of the first column in Table 6.2. We also know that the MLE for repeated binomial trials is $\frac{\sum_{i=1}^{nm} x_i}{nm}$ where n is the number of Bernoulli trials in each binomial draw (10 in our case) and m is the number of of binomial draws. Since we do not observe the number of draws from c_1 our best guess is the posterior expectation. Thus we set $m = 5.946387$. We also do not directly observe the number of success per draw from c_1 and instead compute the number of expected successes. The number of expected successes happens to be the expected sufficient statistic for the binomial. We have already computed those sufficient statistics in Table 6.3. For coin c_1 the binomial MLE is thus

$$\theta_{c_j}^{(1)} = \frac{\mathbb{E}\left[\sum_{i=1}^{nm} x_i \mid c_1\right]}{n\mathbb{E}\left[c_j\right]} = \frac{23.60424}{10 \times 5.946387} = \tag{6.16}$$

The updated binomial distributions are given in Table 6.4. This completes our parameter updates and thus the M-step. We can start the next round of estimation by doing the E-step using the updated parameters θ^1. We will stop iterating when the parameters do not change anymore or after a pre-specified number of iterations.

Further reading

A great tutorial on EM is provided by Michael Collins who uses Naïve Bayes as a running example. If you want to delve deeper into the theory behind EM and see how it can be interpreted from an Information-theoretic perspective, have a look at this classic paper. The daring ones amongst you may also want to consult the original EM paper. Be cautious of he notation they use which is very different from the notation used in this script.