

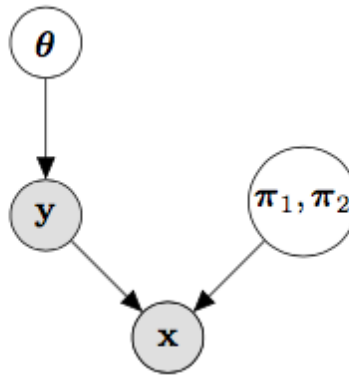
# CS 181 Spring 2018 Section 8

## Mixture Modeling and EM

### 1 Graphical Models

We can represent a large class of probabilistic models within the framework of *directed graphical models*. A directed graphical model is a visual representation that shows the conditional relationships between parameters, latent variables, and observed variables, and provides a language for factorizing a joint distribution.

**Exercise 1.** Use the following graphical model to factorize the joint distribution  $p(\mathbf{x}, \mathbf{y} | \theta, \pi_1, \pi_2)$ . Which of the models we have studied in this class does this graphical model represent?



*Solution.*

## 2 Mixture Models

A *mixture model* is a type of probabilistic model for unsupervised learning. The basic idea is the following: we assume that our data is generated by first sampling a category from some predefined set, and then sampling an observation within that category. That is, we sample a latent variable  $\mathbf{z}$  according to a categorical distribution  $p(\mathbf{z} = C_k; \boldsymbol{\theta}) = \theta_k$ , and then sample an observation  $\mathbf{x}$  from some distribution  $p(\mathbf{x}|\mathbf{z})$ .

Mixture models are very useful for modeling *latent variables*, i.e., variables that aren't directly observed at training time, but are still part of the data generation process. For example, we can cluster data by fitting a mixture model, and then determining the most likely latent class for each data point. Unfortunately, training mixture models cannot be done simply in closed form. Instead we need to use a form of approximate maximum-likelihood estimation called *expectation maximization*.

## 3 Expectation Maximization

Expectation maximization is a general technique for maximum-likelihood estimation used primarily for models with latent variables. While we talk primarily here about mixture models, EM is often used for a variety of other models.

Consider a generative mixture model consisting of a latent variable  $\mathbf{z}$  from a distribution  $p(\mathbf{z}; \mathbf{w})$  and an observed variable  $\mathbf{x}$ , such that we draw  $\mathbf{x}$  from a distribution  $p(\mathbf{x}|\mathbf{z}; \mathbf{w})$  for some vector of parameters  $\mathbf{w}$ .

We have 2 goals: first, to compute the MLE for  $\mathbf{w}$ , i.e. the value of  $\mathbf{w}$  that maximizes  $p(\mathbf{x}; \mathbf{w})$ , and second, to estimate the latent variable  $\mathbf{z}$  corresponding to a particular  $\mathbf{x}$ , which in this case means maximize the distribution  $p(\mathbf{z}|\mathbf{x}; \mathbf{w})$ . The latter goal is easy once we have an estimate of the MLE for  $\boldsymbol{\theta}$ , because we can apply Bayes rule:

$$p(\mathbf{z}|\mathbf{x}; \mathbf{w}) \propto p(\mathbf{x}|\mathbf{z}; \mathbf{w})p(\mathbf{z}; \mathbf{w}) \quad (1)$$

So all we need to do is build a model for the generative process ( $p(\mathbf{x}|\mathbf{z}; \mathbf{w})$  and  $p(\mathbf{z}; \mathbf{w})$ ), and determine out how to calculate the MLE.

### 3.1 Why EM?

Unfortunately calculating the MLE is often computationally intractable, because the log-likelihood is:

$$\log p(\mathbf{x}; \mathbf{w}) = \log \sum_{\mathbf{z} \in Z} p(\mathbf{x}, \mathbf{z}; \mathbf{w}) \quad (2)$$

We know the form of the model  $p(\mathbf{x}, \mathbf{z}; \mathbf{w})$ , but in general we don't know the likelihood  $p(\mathbf{x}; \mathbf{w})$  in closed form. There is no closed form expression for the MLE because it is the log of a sum of expressions, which makes simplifying difficult. Note that we've assumed that our latent variable  $\mathbf{z}$  takes on values in a discrete space  $Z$  (below, that space will be a set of class labels or clusters, and thus  $Z = \{C_k\}_{k=1}^c$ , and  $\mathbf{z}$  is one-hot)—things get even more difficult in the continuous case (the sum becomes an integral, etc.).

### 3.2 Algorithm

Since finding the MLE directly is difficult, we will use an approximate iterative approach. This takes the following form: we approximate the MLE, approximate the distribution of the latent variables  $\mathbf{z}$ , and repeat until the approximation is very good. This makes the problem simpler, since we usually have good ways to find the MLE given the value of the  $\mathbf{z}$ , and to find the distribution of  $\mathbf{z}$  given the MLE. In particular, if we know the value of  $\mathbf{w}$ , then the distribution of  $\mathbf{z}$  is  $p(\mathbf{z}|\mathbf{x}; \mathbf{w})$ . Additionally if we know the distribution of  $\mathbf{z}$ , then the expected complete data log-likelihood is tractable, since we are calculating:

$$\mathbb{E}_{\mathbf{z}|\mathbf{x}; \mathbf{w}} \left[ \sum_{i=1}^n \log p(\mathbf{x}_i, \mathbf{z}_i; \mathbf{w}) \right] = \mathbb{E}_{\mathbf{z}|\mathbf{x}; \mathbf{w}} \left[ \sum_{i=1}^n (\log p(\mathbf{x}_i|\mathbf{z}_i; \mathbf{w}) + \log p(\mathbf{z}_i; \mathbf{w})) \right] \quad (3)$$

$$= \sum_{\mathbf{z} \in Z} p(\mathbf{z}|\mathbf{x}; \mathbf{w}) \left[ \sum_{i=1}^n (\log p(\mathbf{x}_i|\mathbf{z}_i; \mathbf{w}) + \log p(\mathbf{z}_i; \mathbf{w})) \right] \quad (4)$$

$$= \sum_{\mathbf{z} \in Z} q(\mathbf{z}) \left[ \sum_{i=1}^n (\log p(\mathbf{x}_i|\mathbf{z}_i; \mathbf{w}) + \log p(\mathbf{z}_i; \mathbf{w})) \right] \quad (5)$$

where  $p(\mathbf{z}|\mathbf{x}; \mathbf{w}) = q(\mathbf{z})$ . So the steps of the algorithm are:

1. Initialize a  $\mathbf{w}^{(0)}$  randomly (what are we trying to avoid in this initialization?).
2. Calculate the distribution  $\mathbf{q}$  over  $\mathbf{z}$ :

$$q_{i,k} = p(\mathbf{z}_i = C_k|\mathbf{x}_i; \mathbf{w}^{(i)}) \propto p(\mathbf{x}_i|\mathbf{z}_i; \mathbf{w}^{(i)})p(\mathbf{z}_i; \mathbf{w}^{(i)}) \quad (6)$$

3. Choose the value of  $\mathbf{w}^{(i+1)}$  that maximizes the expected complete data log likelihood (where the expectation is over the distribution calculated above):

$$\mathbf{w}^{(i+1)} = \underset{\mathbf{w}}{\operatorname{argmax}} \mathbb{E}_{\mathbf{q}} [\log p(\mathbf{x}, \mathbf{z}; \mathbf{w})] \quad (7)$$

4. Go back to step 2 until the log likelihood estimate converges.

### 3.3 Example 1 - Coins

Consider a setup where we have 2 biased coins. We generate data by picking between the two coins with another biased coin, and then flip the chosen coin to generate a new data point  $x_i$ . We wish to do inference on the parameters of the coins, but the only data we're given is the outcomes of the flips.

Here, we have a very clear choice of  $\mathbf{z}_i$ : a value that indicates whether the  $i$ th coin flip was from the first or second coins. To keep parity with the lecture, which essentially uses the same model for the mixture of multinomials, we'll let the variable  $\mathbf{z}_i$  be a one-hot vector (of size 2) indicating which coin it came from. Also, we'll denote the vector of probabilities for the coin used to choose between coins as  $\boldsymbol{\theta} \in \mathbb{R}^2$ , where  $\theta_1$  is the probability we'll pick the first coin, and  $\theta_2$  the second. Finally, we'll use  $\boldsymbol{\pi}_1, \boldsymbol{\pi}_2 \in \mathbb{R}^2$  to denote the biases for each coin, where  $\boldsymbol{\pi}_1$  is the vector of probabilities for coin 1, etc. This is exactly the same setup as in class for a mixture of multinomials except we only have two multinomials here (coins 1 and 2) and they have 2 outcomes (heads or tails). We let  $\mathbf{w} := \{\boldsymbol{\theta}, \boldsymbol{\pi}\}$ .

**Exercise 2.** Draw a graphical model that represents this generative process.

*Solution.*

□

First we note that we can calculate  $\mathbf{q}_i$  from  $\mathbf{w}^{(i)}$  by writing:

$$\mathbf{q}_i = \begin{bmatrix} p(\mathbf{z}_i = C_1 | \mathbf{x}_i; \mathbf{w}^{(i)}) \\ p(\mathbf{z}_i = C_2 | \mathbf{x}_i; \mathbf{w}^{(i)}) \end{bmatrix} \quad (8)$$

$$\propto \begin{bmatrix} p(\mathbf{x}_i | \mathbf{z}_i = C_1; \mathbf{w}^{(i)}) p(\mathbf{z}_i = C_1; \mathbf{w}^{(i)}) \\ p(\mathbf{x}_i | \mathbf{z}_i = C_2; \mathbf{w}^{(i)}) p(\mathbf{z}_i = C_2; \mathbf{w}^{(i)}) \end{bmatrix} \quad (9)$$

$$\propto \begin{bmatrix} (\pi_{11})^{x_1} (\pi_{12})^{x_2} \theta_1 \\ (\pi_{21})^{x_1} (\pi_{22})^{x_2} \theta_2 \end{bmatrix} \quad (10)$$

We also have the data log-likelihood as

$$\log p(\mathbf{x}_i, \mathbf{z}_i; \mathbf{w}) = \log p(\mathbf{x}_i | \mathbf{z}_i; \mathbf{w}) p(\mathbf{z}_i; \mathbf{w}) \quad (11)$$

$$= \log \prod_{k=1}^{c=2} \left( \theta_k \prod_{j=1}^2 \pi_{kj}^{x_{ij}} \right)^{z_{ik}} \quad (12)$$

$$= z_{i1} (\log \theta_1 + x_{i1} \log \pi_{11} + x_{i2} \log \pi_{12}) \quad (13)$$

$$+ z_{i2} (\log \theta_2 + x_{i1} \log \pi_{21} + x_{i2} \log \pi_{22}) \quad (14)$$

$$\log p(\mathbf{x}, \mathbf{z}; \mathbf{w}) = \sum_{i=1}^n \log p(\mathbf{x}_i, \mathbf{z}_i; \mathbf{w}) \quad (15)$$

Finally, if we have an estimate for  $\mathbf{q}$ , we also have

$$\mathcal{L}_c = \mathbb{E}_{\mathbf{z} | \mathbf{x}, \mathbf{w}} \left[ \sum_{i=1}^n \log p(\mathbf{x}_i, \mathbf{z}_i; \mathbf{w}) \right] \quad (16)$$

$$= \mathbb{E}_{\mathbf{z} | \mathbf{x}, \mathbf{w}} \left[ \sum_{i=1}^n \log p(\mathbf{z}_i; \mathbf{w}) + \log p(\mathbf{x}_i | \mathbf{z}_i; \mathbf{w}) \right] \quad (17)$$

$$= \sum_{i=1}^n \mathbb{E}_{\mathbf{z} | \mathbf{x}, \mathbf{w}} [\log p(\mathbf{z}_i; \mathbf{w}) + \log p(\mathbf{x}_i | \mathbf{z}_i; \mathbf{w})] \quad (18)$$

$$= \sum_{i=1}^n \sum_{k=1}^c q_{ik} \left( \log \theta_k + \sum_{j=1}^2 x_{ij} \log \pi_{kj} \right) \quad (19)$$

$$= \sum_{i=1}^n q_{i1} (\log \theta_1 + x_{i1} \log \pi_{11} + x_{i2} \log \pi_{12}) + q_{i2} (\log \theta_2 + x_{i1} \log \pi_{21} + x_{i2} \log \pi_{22}) \quad (20)$$

since  $\mathbf{q}$  is just a pair of probabilities for each example. This gives us everything we need to use expectation-maximization. Let's walk through the steps:

1. Initialize  $\mathbf{w}^{(0)}$  randomly.
2. Use  $\mathbf{w}^{(i)}$  to calculate the vector of probabilities  $\mathbf{q}_i$  for the distribution of each  $\mathbf{z}_i$  (eqs. 8-10).

3. Calculate the approximate expected likelihood using  $\mathbf{q}_i$  and  $\mathbf{w}^{(i)}$  (eqs. 11-15). This step is not strictly necessary for calculating updates, but can be helpful for a variety of purposes, including debugging and testing convergence. Note that we need *both*  $\mathbf{q}$  and  $\mathbf{w}^{(i)}$  to get a value here.
4. Use  $\mathbf{q}$  to calculate an updated set of parameters  $\mathbf{w}^{(i+1)}$  by maximizing the expected likelihood as a function of  $\mathbf{w}$  (eqs. 16-20). Note that here we do *not* use  $\mathbf{w}^{(i)}$ .

During optimization we need to enforce that  $\sum_k \theta_k = 1$  and that  $\sum_j \pi_{kj} = 1$ , so that the distributions parameterized by  $\boldsymbol{\theta}$  and  $\boldsymbol{\pi}$  are valid. In general we want to use Lagrange multipliers, but in this case we can substitute  $\theta_2 = 1 - \theta_1$  and  $\pi_{k2} = 1 - \pi_{k1}$ :

$$\mathcal{L}_c = \sum_{i=1}^n q_{i1} (\log \theta_1 + x_{i1} \log \pi_{11} + x_{i2} \log(1 - \pi_{11})) \quad (21)$$

$$+ q_{i2} (\log(1 - \theta_1) + x_{i1} \log \pi_{21} + x_{i2} \log(1 - \pi_{21})) \quad (22)$$

And then optimize w.r.t.  $\theta_1, \pi_{11}, \pi_{21}$ :

$$\frac{\partial \mathcal{L}_c}{\partial \theta_1} = \sum_{i=1}^n \left( \frac{q_{i1}}{\theta_1} - \frac{q_{i2}}{1 - \theta_1} \right) = 0 \quad (23)$$

$$\frac{\partial \mathcal{L}_c}{\partial \pi_{11}} = \sum_{i=1}^n q_{i1} \left( \frac{x_{i1}}{\pi_{11}} - \frac{x_{i2}}{1 - \pi_{11}} \right) = 0 \quad (24)$$

$$\frac{\partial \mathcal{L}_c}{\partial \pi_{21}} = \sum_{i=1}^n q_{i2} \left( \frac{x_{i1}}{\pi_{21}} - \frac{x_{i2}}{1 - \pi_{21}} \right) = 0 \quad (25)$$

From here we can solve for the optimal value of  $\mathbf{w}$  (i.e.  $\theta_1, \pi_{11}, \pi_{22}$ ), and set  $\mathbf{w}^{(i+1)} = \operatorname{argmax}_{\mathbf{w}} \mathbb{E}_{\mathbf{z}|\mathbf{x};\mathbf{w}} \mathcal{L}_c$ .

*Note:* Above we show the derivation of all steps of the algorithm, but in practice  $\mathbf{w}^{(i+1)}$  has a closed form, so the steps of the algorithm are really just initialization, calculate the distribution  $\mathbf{q}_i$  from  $\mathbf{w}^{(i)}$ , and then calculate  $\mathbf{w}^{(i+1)}$  from  $\mathbf{q}$ . All the difficult work is in deriving the update equations. In more complicated models it can happen that  $\mathbf{w}^{(i+1)}$  does not have a closed form, so instead we can do gradient descent to calculate the optimal value.

**Exercise 3.** Derive the closed form updates for  $\boldsymbol{\theta}^{(i)}, \boldsymbol{\pi}^{(i)}$  from the steps above.

*Solution.*

□

Once we have an estimate for the MLE, we can use it to do prediction of hidden states for a new incoming coin flip, using step 2 from above. So, given a new coin flip, we can predict whether it can from the first or the second coin. As you can imagine, this is a terrible process, since we get a single bit of information, and in particular it is impossible to tell the difference between having one coin with high probability at 0.5 (and another picked almost never with bias = 0.1 e.g.) and two equally likely coins with biases 0.4 and 0.6. This problem is due more to the data setup rather than the method, so let's try an easier problem:

**Exercise 4.** Consider the following data generation process: the setup is the same as above, but instead of flipping the chosen coin once, we flip it 10 times before choosing a new coin.

1. Find an appropriate choice of latent variables  $\mathbf{z}_i$  and calculate the distribution of  $\mathbf{z}_i$  given the data  $\mathbf{x}_{i,j}$  (where  $i$  iterates over the sets of 10 coin flips, and  $j \in [1, 10]$ ) and an estimate for  $\boldsymbol{\theta}$ .
2. Find the expression for the expected complete data log-likelihood
3. Find the closed form update equations for  $\boldsymbol{\theta}^{(i)}$ , and compare them to the result from Exercise 1.

Solution.

□

### 3.4 Example 2 - Gaussian Mixture Modeling

We covered this in class, but it's a good idea to go over some of the points in the derivation from class. Remember that the setup is that we have data  $\mathbf{x}_i \in \mathbb{R}^m$  and a latent variable  $\mathbf{z}_i$  (corresponding to the cluster that the point is drawn from) such that  $\mathbf{x} \sim p(\mathbf{x}|\mathbf{z} = C_k) = \mathcal{N}(\mathbf{x}; \boldsymbol{\mu}_k, \boldsymbol{\Sigma}_k)$ , where  $\boldsymbol{\mu}_k, \boldsymbol{\Sigma}_k$  are the mean and covariance of the  $k$ th cluster. The choice of cluster is drawn from a categorical distribution with probabilities  $\boldsymbol{\theta} \in \mathbb{R}^c$ . We are able to observe the data  $\mathbf{x}_i$  and want to find the cluster centers and their covariances.

Following the same format as above, the steps of EM inference applied to this problem are:

1. Randomly initialize  $\boldsymbol{\theta}, \{\boldsymbol{\mu}_k, \boldsymbol{\Sigma}_k\}_k$ .
2. Next, calculate the new distribution of each  $\mathbf{z}_i$ :

$$q_{ik} = p(z_i = C_k | \mathbf{x}_i) \propto \theta_k \mathcal{N}(\mathbf{x}_i; \boldsymbol{\mu}_k, \boldsymbol{\Sigma}_k) \quad (26)$$

This is our new estimate of the distribution of  $\mathbf{z}_i$  given the data and our estimate for  $\boldsymbol{\theta}, \{\boldsymbol{\mu}_k, \boldsymbol{\Sigma}_k\}_k$ .

3. Find the expected complete data log-likelihood:

$$\mathbb{E}_{\mathbf{Z}} [\mathcal{L}] = \mathbb{E}_{\mathbf{Z}} \left[ \sum_{i=1}^n \ln(p(\mathbf{x}_i, \mathbf{z}_i; \boldsymbol{\theta}, \{\boldsymbol{\mu}_k, \boldsymbol{\Sigma}_k\}_k)) \right] \quad (27)$$

$$= \sum_{i=1}^n \sum_{k=1}^c q_{ik} \ln \theta_k + q_{ik} \ln \mathcal{N}(\mathbf{x}_i; \boldsymbol{\mu}_k, \boldsymbol{\Sigma}_k) \quad (28)$$

and then optimize it for each of the parameters  $\boldsymbol{\theta}, \{\boldsymbol{\mu}_k, \boldsymbol{\Sigma}_k\}_k$ . However, we need to be careful to remember constraints: since  $\sum_k \theta_k = 1$ , we must use Lagrange multipliers to optimize the parameters. We get the following update equations:

$$\theta_k^{(i+1)} = \frac{\sum_{i=1}^n q_{ik}}{n} \quad (29)$$

$$\boldsymbol{\mu}_k^{(i+1)} = \frac{\sum_{i=1}^n q_{ik} \mathbf{x}_i}{\sum_{i=1}^n q_{ik}} \quad (30)$$

$$\boldsymbol{\Sigma}_k^{(i+1)} = \frac{\sum_{i=1}^n q_{ik} (\mathbf{x}_i - \boldsymbol{\mu}_k^{(i+1)}) (\mathbf{x}_i - \boldsymbol{\mu}_k^{(i+1)})^\top}{\sum_{i=1}^n q_{ik}} \quad (31)$$