CS 181 Spring 2021 Section 8
Mixture Models and Expectation Maximization (EM)

1 Mixture Models

1.1 Review of notation

Vectors are denoted using **bold letters**. In the statement,

"Assume you have observed data $\{\mathbf{x}_n\}_{n=1}^N$."

This means there are some constant N distinct observations. Each observation $\mathbf{x}_n$ is a vector, where each component of the vector represents each feature.

When used to define distributions, the **semicolon** means that you can plug in the deterministic value of the variable after the semicolon into your expression for the distribution. Typically the variables that appear after the semicolon are unknown parameters for which you have some fixed estimate of what they may be.

For example, say you are given that random variable $x \sim N(0, \sigma)$, where σ is unknown. You believe that $\sigma = 2$. Then

$$p(x; \sigma) \sim N(0, 2)$$

When reading mathematical expressions, pay close attention to which variables are *random* (random variables can be *observed* or *unobserved*), and which variables are *deterministic constants*.

1.2 Motivation

Textbook sections 9.1, 9.2.

A *mixture model* is a type of probabilistic model for unsupervised learning.

Suppose you have some observed data $\{\mathbf{x}_n\}_{n=1}^N$.

Mixture models are used when you have reason to believe that each individual observation has a discrete *latent variable* $\mathbf{z}_n$ that determines the data generating process. A latent variable is some piece of data that is unknown, but influences the observed data.

Say there are K possible values for each $\mathbf{z}_n$, denoted $\{C_k\}_{k=1}^K$ where each C_k is a one-hot encoded vector of length K.

Consider the following data-generating process for each data point $\mathbf{x}_n$:

- Sample latent class $\mathbf{z}_n$ from $\boldsymbol{\theta}$, the categorical distribution over $\{C_k\}_{k=1}^K$ s.t. $p(\mathbf{z} = C_k; \boldsymbol{\theta}) = \theta_k$. Call this sampled latent class C_S.
- Given that $\mathbf{z}_n = C_S$, sample $\mathbf{x}_n$ from the distribution

$$p(\mathbf{x}|\mathbf{z} = C_S; \mathbf{w})$$

  This conditional distribution is a modeling assumption (which means we will give it to you in this class), and is specified using unknown parameters $\mathbf{w}$.

  For example, we may assume 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 unknown mean and covariance of the k-th cluster. (See Section 2.4 for more about Gaussian mixture models!)

Example: Say you have a dataset containing weights from a random sample of animals in a pet store. Each x_n is the animal's weight. The latent variables z_n represent what kind of animal is being weighed, so the possible values $\{C_1, C_2, \ldots, C_K\}$ may represent the categories cat, dog, bird, etc. In your model, you also use the assumption that $p(x|z = C_k; \mathbf{w}) \sim N(\mu_k, \sigma_k)$.

Exercise 1. *In this example, can you give an intuitive explanation of what vector $\boldsymbol{\theta}$ represents? What does it mean that $p(x|z = C_k; \mathbf{w}) \sim N(\mu_k, \sigma_k)$?*

2 Expectation Maximization

Textbook sections 9.3, 9.4.

Expectation maximization is a general technique for maximum-likelihood estimation used primarily for models with latent variables. Here we will show how to use EM to train a mixture model, but EM is also used for a variety of other models!

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

We have 2 goals:

1. To compute the MLE for $\mathbf{w}$ and $\boldsymbol{\theta}$, i.e. the values of $\mathbf{w}$, $\boldsymbol{\theta}$ that maximize $p(\mathbf{x}; \mathbf{w}, \boldsymbol{\theta})$.
2. To estimate the latent variable $\mathbf{z}_n$ corresponding to a particular $\mathbf{x}_n$, which in this case means maximize the distribution $p(\mathbf{z}_n|\mathbf{x}_n; \mathbf{w}, \boldsymbol{\theta})$.

Goal 2 is easy once we have an estimate of the MLE for $\mathbf{w}, \boldsymbol{\theta}$, because we can apply Bayes' rule:

$$p(\mathbf{z}|\mathbf{x}; \mathbf{w}, \boldsymbol{\theta}) \propto p(\mathbf{x}|\mathbf{z}; \mathbf{w}, \boldsymbol{\theta}) p(\mathbf{z}; \mathbf{w}, \boldsymbol{\theta})$$

$$p(\mathbf{z}|\mathbf{x}; \mathbf{w}, \boldsymbol{\theta}) \propto p(\mathbf{x}|\mathbf{z}; \mathbf{w}) p(\mathbf{z}; \boldsymbol{\theta}) \tag{1}$$

2.1 Why EM?

The likelihood of the data can be written as

$$p(\mathbf{x}; \mathbf{w}, \boldsymbol{\theta}) = \sum_{\mathbf{z} \in Z} p(\mathbf{x}, \mathbf{z}; \mathbf{w}, \boldsymbol{\theta})$$

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

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

There is no closed form for the MLE of the log-likelihood because it is the log of a sum of expressions. We know the form of the model $p(\mathbf{x}, \mathbf{z}; \mathbf{w}, \boldsymbol{\theta})$, but in general we cannot solve for the $(\mathbf{w}, \boldsymbol{\theta})$ which maximize the likelihood $p(\mathbf{x}; \mathbf{w}, \boldsymbol{\theta})$ in closed form.

2.2 The EM Algorithm

Since finding the MLE directly is difficult, we will use expectation maximization: an approximate iterative approach. The steps of the algorithm are:

1. Initialize $\mathbf{w}^{(0)}, \boldsymbol{\theta}^{(0)}$ randomly.
2. *(E-step)* Use the parameters to predict the distribution $\mathbf{q}$ for each example. The vector $\mathbf{q}_n$ represents how likely it is that the latent variable $\mathbf{z}_n$ comes from each class, given our current setting for the model parameters:

$$q_{n,k} := p(\mathbf{z}_n = C_k | \mathbf{x}_n; \mathbf{w}^{(t)}, \boldsymbol{\theta}^{(t)}) \propto p(\mathbf{x}_n | \mathbf{z}_n = C_k; \mathbf{w}^{(t)}) p(\mathbf{z}_n = C_k; \boldsymbol{\theta}^{(t)}) \tag{3}$$

3. *(M-step)* Update parameters: Choose the value of $\mathbf{w}^{(t+1)}, \boldsymbol{\theta}^{(t+1)}$ that maximizes the expected complete data log likelihood (where the expectation is over the distribution calculated above):

$$\mathbf{w}^{(t+1)}, \boldsymbol{\theta}^{(t+1)} = \underset{\mathbf{w}, \boldsymbol{\theta}}{\operatorname{argmax}} \, \mathbb{E}_{\mathbf{Z}|\mathbf{X}} \left[\sum_{n=1}^{N} \log p(\mathbf{x}, \mathbf{z}; \mathbf{w}, \boldsymbol{\theta}) \right] \tag{4}$$

4. Go back to step 2 until the log-likelihood estimate in step 3 converges.

2.3 Example: Gaussian Mixture Modeling

Lecture 14 and textbook section 9.5.

Recall from lecture the following setup:

We have data $\mathbf{x}_n \in \mathbb{R}^D$ and a latent variable $\mathbf{z}_n$ (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 $\boldsymbol{\pi} \in [0,1]^K$. We are able to observe the data $\mathbf{x}_n$ and want to find the cluster centers and their covariances.

The steps of EM inference applied to this problem are:

1. Randomly initialize $\boldsymbol{\pi}, \{\boldsymbol{\mu}_k, \boldsymbol{\Sigma}_k\}_{k=1}^K$.
2. Next, calculate the new distribution of each $\mathbf{z}_n$:

$$q_{n,k} = p(z_n = C_k | \mathbf{x}_n) \propto \pi_k \mathcal{N}(\mathbf{x}_n; \boldsymbol{\mu}_k, \boldsymbol{\Sigma}_k) \tag{5}$$

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

3. Find the expected complete data log-likelihood:

$$\mathbb{E}_{\mathbf{Z}|\mathbf{X}}[\mathcal{L}] = \mathbb{E}_{\mathbf{Z}|\mathbf{X}}[\sum_{n=1}^{N} \ln(p(\mathbf{x}_n, \mathbf{z}_n; \boldsymbol{\pi}, \{\boldsymbol{\mu}_k, \boldsymbol{\Sigma}_k\}_k))] \tag{6}$$

$$= \sum_{n=1}^{N}\sum_{k=1}^{K} q_{n,k} \ln \pi_k + q_{n,k} \ln \mathcal{N}(\mathbf{x}_n; \boldsymbol{\mu}_k, \boldsymbol{\Sigma}_k) \tag{7}$$

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

$$\pi_k^{(t+1)} = \frac{\sum_{n=1}^{N} q_{n,k}}{N} \tag{8}$$

$$\boldsymbol{\mu}_k^{(t+1)} = \frac{\sum_{n=1}^{N} q_{n,k}\mathbf{x}_n}{\sum_{n=1}^{N} q_{n,k}} \tag{9}$$

$$\boldsymbol{\Sigma}_k^{(t+1)} = \frac{\sum_{n=1}^{N} q_{n,k}(\mathbf{x}_n - \boldsymbol{\mu}_k^{(t+1)})(\mathbf{x}_n - \boldsymbol{\mu}_k^{(t+1)})^\top}{\sum_{n=1}^{N} q_{n,k}} \tag{10}$$

2.4 Example: Modeling Biased Coins with a Binomial Mixture Model

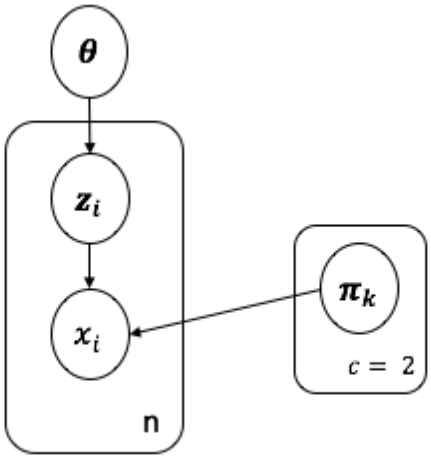
We've seen one case of mixtures of Gaussians, but we can consider mixtures of any distribution. In this example, we'll take a look at EM for a binomial mixture model. To get started, we consider a mixture of Bernoulli model, where $x_n \sim p(x_n|z_n = C_k) = \text{Bern}(x_n; p_k)$.

Consider a setup where we have 2 biased coins C_1 and C_2, where $Pr(C_1 = 1) = \pi_1$ and $Pr(C_2 = 1) = \pi_2$.

Data points x_n are generated by:

- First, flip another biased coin C_z.
- If C_z is heads, then x_n is the outcome of flipping C_1.
- Otherwise, if C_z is tails, then x_n is the outcome of flipping C_2.

We can visualize this setup with the following diagram:



We wish to do inference to learn the unknown parameters of the coins (π_1, π_2), but the only data we're given is the outcomes of the flips (the x_n's).[1]

Exercise 2. *In this example, what is a reasonable choice for the latent variables z_n?*

To be consistent with Textbook Example 9.4.5, which uses the same model for the mixture of multinomials, we'll let $\mathbf{x}_n$ be a one-hot vector s.t. $\mathbf{x}_{n,1} = 1$ if the result of coin flip n was heads; $\mathbf{x}_{n,2} = 1$ otherwise. $\mathbf{z}_n$ is a one-hot vector (of size 2) indicating which coin was flipped to generate $\mathbf{x}_n$.

We'll denote the vector of probabilities for C_z used to choose between coins as $\boldsymbol{\theta} \in [0,1]^2$, where θ_1 is the probability we'll pick C_1, and θ_2 for C_2. Finally, we'll use $\boldsymbol{\pi}_1, \boldsymbol{\pi}_2 \in [0,1]^2$ to denote the biases for each coin, where $\boldsymbol{\pi}_1$ is the vector of probabilities for C_1, etc. Our model is a mixture of binomials where we have two binomials (coins 1 and 2), each with 2 outcomes (heads or tails). We let $\mathbf{w} := \{\boldsymbol{\theta}, \boldsymbol{\pi}\}$.

Now that we have the problem set up, let's use expectation maximization to learn parameters $\mathbf{w} := \{\boldsymbol{\theta}, \boldsymbol{\pi}\}$!

[1] In fact, when we only get 1 coin flip per example, so that each x_n is just a single head or tail, and this is a mixture-of-Bernoulli model, we won't be able to usefully identify the parameters. Consider the case of trying to tell between two coins with π_1 = 0.3 and π_2 =0.7 and $\theta_1 = \theta_2 = 0.5$ and two coins with $\pi_1 = \pi_2 = 0.5$ and $\theta_1 = \theta_2 = 0.5$. These two parameterizations put the same likelihood on any data set. Still, the work we do in this context extends to the case where x_n represents multiple coin tosses per example and we have a mixture-of-Binomial model. There we can usefully estimate the parameters of a mixture model. We get to this in Exercise 4.

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

$$\mathbf{q}_n = \begin{bmatrix} p(\mathbf{z}_n = C_1 | \mathbf{x}_n; \mathbf{w}^{(t)}) \\ p(\mathbf{z}_n = C_2 | \mathbf{x}_n; \mathbf{w}^{(t)}) \end{bmatrix} \tag{11}$$

$$\propto \begin{bmatrix} p(\mathbf{x}_n | \mathbf{z}_n = C_1; \mathbf{w}^{(t)}) p(\mathbf{z}_n = C_1; \mathbf{w}^{(t)}) \\ p(\mathbf{x}_n | \mathbf{z}_n = C_2; \mathbf{w}^{(t)}) p(\mathbf{z}_n = C_2; \mathbf{w}^{(t)}) \end{bmatrix} \tag{12}$$

$$\propto \begin{bmatrix} (\pi_{11})^{x_{n,1}} (\pi_{12})^{x_{n,2}} \theta_1 \\ (\pi_{21})^{x_{n,1}} (\pi_{22})^{x_{n,2}} \theta_2 \end{bmatrix} \tag{13}$$

We also have the complete data log-likelihood:

$$\log p(\mathbf{x}_n, \mathbf{z}_n; \mathbf{w}) = \log p(\mathbf{x}_n | \mathbf{z}_n; \mathbf{w}) p(\mathbf{z}_n; \mathbf{w}) \tag{14}$$

$$= \log \prod_{k=1}^{2} \left(\theta_k \prod_{j=1}^{2} \pi_{kj}^{x_{n,j}} \right)^{z_{n,k}} \tag{15}$$

$$= z_{n,1} \left(\log \theta_1 + x_{n,1} \log \pi_{11} + x_{n,2} \log \pi_{12} \right) + z_{n,2} \left(\log \theta_2 + x_{n,1} \log \pi_{21} + x_{n,2} \log \pi_{22} \right) \tag{16}$$

$$\log p(\mathbf{X}, \mathbf{Z}; \mathbf{w}) = \sum_{n=1}^{N} \log p(\mathbf{x}_n, \mathbf{z}_n; \mathbf{w}) \tag{17}$$

Now expand the expected complete data log-likelihood:

$$\mathcal{L}_c = \mathbb{E}_{\mathbf{Z}|\mathbf{X};\mathbf{w}} \left[\sum_{n=1}^{N} \log p(\mathbf{x}_n, \mathbf{z}_n; \mathbf{w}) \right] \tag{18}$$

$$= \mathbb{E}_{\mathbf{Z}|\mathbf{X};\mathbf{w}} \left[\sum_{n=1}^{N} \log p(\mathbf{z}_n; \mathbf{w}) + \log p(\mathbf{x}_n | \mathbf{z}_n; \mathbf{w}) \right] \tag{19}$$

$$= \sum_{n=1}^{N} \mathbb{E}_{\mathbf{Z}|\mathbf{X};\mathbf{w}} \left[\log p(\mathbf{z}_n; \mathbf{w}) + \log p(\mathbf{x}_n | \mathbf{z}_n; \mathbf{w}) \right] \tag{20}$$

$$= \sum_{n=1}^{N} \sum_{k=1}^{K} q_{n,k} \left(\log \theta_k + \sum_{j=1}^{2} x_{n,j} \log \pi_{kj} \right) \tag{21}$$

$$= \sum_{n=1}^{N} q_{n,1} \left(\log \theta_1 + x_{n,1} \log \pi_{11} + x_{n,2} \log \pi_{12} \right) + q_{n,2} \left(\log \theta_2 + x_{n,1} \log \pi_{21} + x_{n,2} \log \pi_{22} \right) \tag{22}$$

Now we can use these derivations to do expectation maximization!:

1. Initialize $\mathbf{w}^{(0)}$ randomly.
2. Use $\mathbf{w}^{(t)}$ to calculate the vector of probabilities $\mathbf{q}_n$ for the distribution of each $\mathbf{z}_n$ (eq. 13).

3. Calculate the current expected likelihood using $\mathbf{q}_n$ and $\mathbf{w}^{(t)}$ (eq. 22).

   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 $\mathbf{w}^{(t)}$ to get a value here.

4. Use $\mathbf{q}$ to calculate an updated set of parameters $\mathbf{w}^{(t+1)}$ by maximizing the expected likelihood as a function of $\mathbf{w}$ (eq. 22). Note that here we do *not* use $\mathbf{w}^{(t)}$.

   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 can enforce this constraint using Lagrange multipliers. Here, in the 2-dimensional case, we don't need to use Lagrangian methods and can instead substitute $\theta_2 = 1 - \theta_1$ and $\pi_{k2} = 1 - \pi_{k1}$:

$$\begin{aligned}\mathcal{L}_c = \sum_{n=1}^{N} \; & q_{n,1} \left(\log \theta_1 + x_{n,1} \log \pi_{11} + x_{n,2} \log(1 - \pi_{11})\right) \\ & + q_{n,2} \left(\log(1 - \theta_1) + x_{n,1} \log \pi_{21} + x_{n,2} \log(1 - \pi_{21})\right)\end{aligned} \tag{23}$$

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

$$\frac{\partial \mathcal{L}_c}{\partial \theta_1} = \sum_{n=1}^{N} \left(\frac{q_{n,1}}{\theta_1} - \frac{q_{n,2}}{1 - \theta_1}\right) = 0 \tag{24}$$

$$\frac{\partial \mathcal{L}_c}{\partial \pi_{11}} = \sum_{n=1}^{N} q_{n,1} \left(\frac{x_{n,1}}{\pi_{11}} - \frac{x_{n,2}}{1 - \pi_{11}}\right) = 0 \tag{25}$$

$$\frac{\partial \mathcal{L}_c}{\partial \pi_{21}} = \sum_{n=1}^{N} q_{n,2} \left(\frac{x_{n,1}}{\pi_{21}} - \frac{x_{n,2}}{1 - \pi_{21}}\right) = 0 \tag{26}$$

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

Note: Above we show the derivation of all steps of the algorithm, but once you know the closed form expression for $\mathbf{w}^{(t+1)}$, the steps of the algorithm are really just initialization, calculate the distribution $\mathbf{q}_n$ from $\mathbf{w}^{(t)}$, and then calculate $\mathbf{w}^{(t+1)}$ from $\mathbf{q}$. All the difficult work is in deriving the update equations.

In more complicated models, the optimal $\mathbf{w}^{(t+1)}$ may not have a closed form solution; in these cases, instead we can do gradient descent to calculate the optimal value.

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

Once we have an estimate for the MLE $\mathbf{w}$, 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 came from the first or the second coin.

Our model may not be very good, since in particular it is impossible to tell the difference between having one coin chosen with high probability with $\pi_1 = 0.5$ (and another picked almost never with $\pi_2 = 0.1$) and two equally likely coins with biases 0.4 and 0.6. In this case, as discussed above, with only one observation for each coin we cannot successfully estimate the parameters of the mixture model. This problem is due to the data setup: we need a mixture of binomials, with multiple observations per coin.

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 $\mathbf{z}_n$ and calculate the distribution of $\mathbf{z}_n$ given the data $\mathbf{x}_{n,j}$ (where n iterates over each set 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 $\boldsymbol{\theta}^{(t)}$, and compare them to the result from Exercise 3.*