# Clustering, Mixtures, and the EM Algorithm

## Machine Learning: Module 1

Sean Norton; Simon Hoellerbauer

September 24, 2018

# The Problem: Finding Groups

In social science, we often believe our observations have some sort of group structure.

- Regime types
- Types of voters
- Types of legislators

However, our data doesn't (generally) come with these groupings conveniently pre-labeled.

# E.g. Regime Types

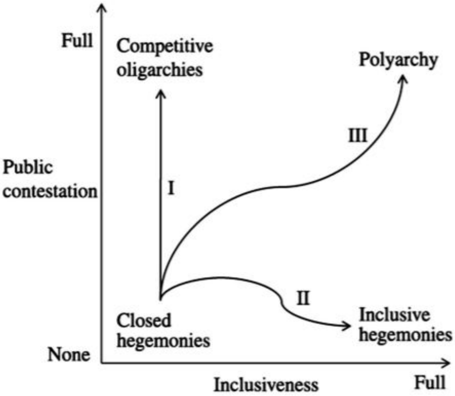


Figure: Dahl: Regime Types

# The Problem Continued

The general approach to dealing with this problem has been to hand-label cases. This is problematic because:

- It's time-consuming
- Humans aren't able to consider all available data at once
- It relies on researcher discretion
- For large datasets (e.g. all countries in the world), this is impossible.

What if there was a way to find latent groupings between our cases quickly and with as little researcher discretion as possible?

# Enter Cluster Analysis

This is exactly what cluster analysis is intended to do!
Given:

- Data
- Number of clusters
- Variables
- Similarity measure

A cluster analysis algorithm finds groupings, or clusters, that maximize the similarity between observations within a cluster.

# K-Means and Notation

One of the most common similarity measures is the squared distance between the center (mean) of a cluster.
This is known as *k-means* or *k-nearest neighbor* clustering.
Before we dive into the math, some notation:

- $k$: total number of clusters
- $r_{nk}$: indicator vector of cluster membership for observation $x_n$
- $\mu_k$: the centroid of cluster $k$

# K-Means: The Math

Cluster analysis relies on a measure of similarity, which in k-means is:

$$J = \sum_{n=1}^{N} \sum_{k=1}^{K} r_{nk} \|x_n - \mu_k\|^2$$

This number $J$ is also known as a *distortion measure*.
What (hopefully) familiar thing does this measure look like?

# The Math cont.

**Q**: But how do we choose values of $\mu_k$ given that we don't actually know the cluster assignments?
**A**: We don't!
We can find $k_m$ through a version of the *expectation-maximization algorithm*:

1. Initialize some random values for $\mu_k$
2. Minimize $J$ w.r.t. $r_{nk}$; i.e. assign cluster memberships in order to minimize distortion
3. Using the previous $r_{nk}$, minimize J w.r.t to $\mu_k$; i.e., assign new means that minimize distortion
4. Repeat until convergence ($J$ does not change, or the change falls below some threshold)

# The Math: Visualized

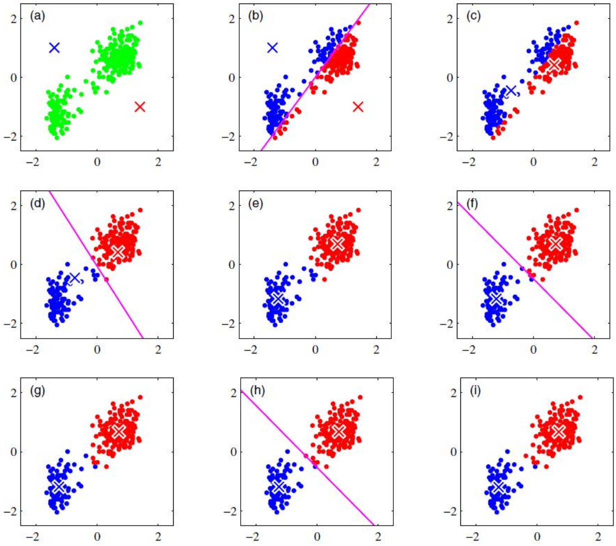


Figure: Optimizing Clusters

# Problems with Clustering

- Choosing $k$: this requires trying a lot of different $k$ w.r.t. some similarity measure
    - Plot the WCSS against the number of clusters, and look for a "bend" in the plot; this is known as the elbow method
    - Use average silhouette width: the silhouette is a measure how similar an observation is to its own cluster (consistency) and how dissimilar it is to other clusters (separation)
    - Gap statistics: compare multiple values of $k$ to a simulated reference distribution of datasets with clusters varying from $k = 1$ to $k = max$
- Sensitivity to outliers; luckily, there are clustering methods other than $k$-means
    - Partitioning around medoids (PAM): uses median instead of mean
    - Hierarchical clustering: creates a tree-based representation of the data without specifying $k$; clusters are created by "cutting" the tree.
- Fundamentally descriptive; clusters will not necessarily be the same given different data

# Problems cont.

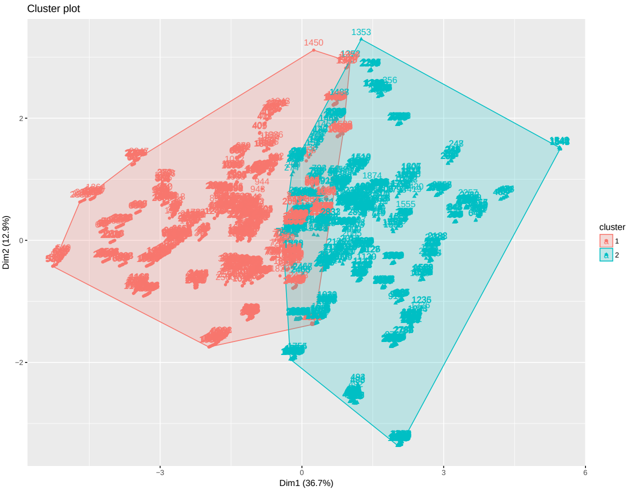


Figure: My Cluster Plot

# Mixture Models: Motivation

What if instead of “hard" cluster assignments, we could assign a probability that cases belong to a particular group?

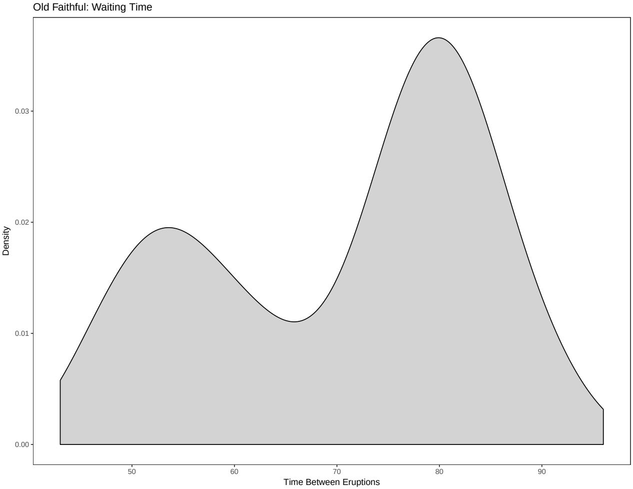
That is precisely what *mixture models* are intended to do!

Mixture models allow observations to belong to different distributions - either different parameterizations of the same distribution, or different distributions entirely.

This allows us to:

- Sort cases into groups, much like cluster models, but do so in a way that quantifies uncertainty
- Estimate a model in the same step as the grouping, which most out-of-the-box clustering packages do not do
- Work with data that has a multimodal distribution without having to discard a substantial amount of variation.

# Motivating Example: Old Faithful



# Motivating Example: Old Faithful

Clearly, this data appears to be normally distributed, but not in a single normal distribution.
Instead of fitting a single Gaussian, we can fit a *mixture* of Gaussians of the following form:

$$p(x) = \sum_{k=1}^{K} \pi_k \mathcal{N}(x|\mu_k, \Sigma_k)$$

Where $\pi_k$ is the probability that an observation belongs to cluster $k$.
This is a Gaussian mixture model (GMM)

# Deriving the Model

To be able to estimate $\pi_k$, let us define a $K$-dimensional vector $\mathbf{z}$, which is 1 if an observation is in group $k$ and 0 otherwise. $\mathbf{z}$ is a latent variable - deriving the model this way will be of use later.

$$p(z_k = 1) = \pi_k$$

Or alternatively, since this is a 1-of-K vector:

$$p(\mathbf{z}) = \prod_{k=1}^{K} \pi_k^{z_k}$$

What must be true of $\pi$

# Deriving the Model

The conditional distribution $(\mathbf{x}|\mathbf{z})$ is then:

$$p(x|z) = \prod_{k=1}^{K} \mathcal{N}(\mathbf{x}|\mu_k, \Sigma_k)^{z_k}$$

Marginalizing the distribution over $\mathbf{z}$ returns the original formulation:

$$p(x) = \sum_{k=1}^{K} \pi_k \mathcal{N}(x|\mu_k, \Sigma_k)$$

While this seems like a useless exercise, we can now work with the joint distribution of our observations and the latent variable, $p(x|z)$, which will make estimation considerably easier.

# More for Later

Another important quantity for later will be $\gamma(z_k)$.
For now, think of $\pi_k$ as the *prior* probability of group membership. We are also interested in the posterior probability ($p(z_k = 1|x)$), which can be calculated using Bayes' Rule:

$$\frac{p(z_k = 1)p(x|z_k = 1)}{\sum_{j=1}^{k} p(z_j = 1)p(x|z_j = 1)}$$

Do this terms look familiar? How would you put this in terms of the joint PDF we just derived?

# Attempting Vanilla MLE

The log-likelihood of the GMM is given by:

$$ln\ p(x|\pi, \mu, \Sigma) = \sum_{n=1}^{N} ln\{\sum_{k=1}^{k} (x_n|\mu_k, \Sigma_k)\}$$

What do you notice is strange about this log-likelihood compared to ones we've seen before?

# Attempting Vanilla MLE

In the log likelihood of the GMM, the logarithm does not act directly on the Gaussian distribution. When we set the derivatives of likelihood to 0 in order to maximize the parameters, this results in there not being a closed form solution.

$$0 = -\sum_{n=1}^{N} \gamma(z_{nk}) \Sigma_k (x_n - \mu_k)$$

Where the mean of group $k$ is the mean of each observation in the group, weighted by the posterior probability of its membership in that group:

$$\mu_k = \frac{1}{N_k} \sum_{n=1}^{N} \gamma(z_{nk}) x_n$$

$$N_k = \sum_{n=1}^{N} \gamma(z_{nk}) = \# \text{ of points in cluster k}$$

# Attempting Vanilla MLE

Similarly, the covariance is given by:

$$\Sigma_k = \frac{1}{N_k} \sum_{n=1}^{N} \gamma(z_{nk})(x_n - \mu_k)(x_n - \mu_k)^\intercal$$

And the mixing coefficients are (intuitively):

$$\pi_k = \frac{N_k}{N}$$

# Problems with MLE

- As can be seen from the previous partial derivatives, the mean and variance of the GMM estimates depend on the posterior probabilities, which themselves depend on the mean and variance. Therefore, there is not a closed form solution.
- Because we now have multiple components, a situation can arise where a component collapses on a single point when an observation is equal to the component mean, contributing infinitely to the log likelihood:

$$\mathcal{N}(x_n|\mu_k = x_n, \Sigma_j) = \frac{1}{2\pi^{\frac{1}{2}}}\frac{1}{\sigma_j}$$

$$\lim_{\sigma_j \to \infty} = \infty$$

# Problems with MLE

- MLE also suffers from an *identifiability* issue: in a $K$ component mixture, there are $K!$ possible solutions for assigning $K$ parameters to $K$ components; for any chosen parameter in the $\pi_k$ parameter space there are $k-1!$ points that produce the exact same distribution
- This creates a multimodal distribution of the log likelihood in which there is no unique solution; more generally for all latent variable models, this is known as *label nonidentifiability*

# The Expectation-Maximization Algorithm

- EM Algorithm allows us to approach MLE in a much easier way
- However, as we will come to see later, the EM algorithm—or a certain interpretation of it—can also be used in Bayesian analysis, either to retrieve the MAP parameter estimates, or to retrieve the entire posterior, with variational inference
- In general, we use it when we have latent variables (the collection of which, in line with terminology used with mixture models, is usually termed $\mathbf{Z}$)
- We will exploit the existence of these latent variables to make finding parameter and latent variables estimates easier

## The Basics: Our Starting Point

Given the former:

- We have a probabilistic model where we term all observed variables as $\mathbf{X}$, all latent variables as $\mathbf{Z}$, and all parameters as $\boldsymbol{\theta}$
- We want to find the maximum likelihood estimates for

$$p(\mathbf{X}|\boldsymbol{\theta}) = \sum_{\mathbf{Z}} p(\mathbf{X}, \mathbf{Z}|\boldsymbol{\theta}) \tag{1}$$

  Note: here we assume $\mathbf{Z}$ is discrete, but it doesn't have to be. What would change if it were continuous?

- Let us assume that $p(\mathbf{X}|\boldsymbol{\theta})$ cannot be maximized as is, and that maximizing the **complete data likelihood** $p(\mathbf{X}, \mathbf{Z}|\boldsymbol{\theta})$ is more straightforward. Note that, as Sean said, we will not be able to maximize even this function directly, because of the interdependence of parameters and latent variables.

# The Basics: Oh Goodness, This Isn't So Basic Anymore

Then let there be a distribution $q(\mathbf{Z})$ defined over the latent variables. For any choice of this distribution, it can be shown that

$$\ln p(\mathbf{X}|\boldsymbol{\theta}) = \mathcal{L}(q, \boldsymbol{\theta}) + \mathrm{KL}(q\|p) \tag{2}$$

where

$$\mathcal{L}(q, \boldsymbol{\theta}) = \sum_{\mathbf{Z}} q(\mathbf{Z}) \ln \left\{ \frac{p(\mathbf{X}, \mathbf{Z}|\boldsymbol{\theta})}{q(\mathbf{Z})} \right\} \tag{3}$$

and

$$\mathrm{KL}(q\|p) = -\sum_{\mathbf{Z}} q(\mathbf{Z}) \ln \left\{ \frac{p(\mathbf{Z}|\mathbf{X}, \boldsymbol{\theta})}{q(\mathbf{Z})} \right\} \tag{4}$$

# The Basics: Functionals + Lower Bound of Marginal Likelihood

- Note that w.r.t the distribution $q(\mathbf{Z})$, $\mathcal{L}(q, \boldsymbol{\theta})$ is a *functional*, that is, "an operator that takes a function .... and returns and output value" (Bishop 2006, 703). The "value" that maximizes a functional is a function. This idea is especially important in variational inference, which is built on the calculus of variations. W.r.t to $\theta$, $\mathcal{L}(q, \boldsymbol{\theta})$ is a function.
- Note also that $\text{KL}(q\|p)$ is the Kullback-Leibler divergence between our distribution $q(\mathbf{Z})$ and the posterior distribution $p(\mathbf{Z}|\mathbf{X}, \boldsymbol{\theta})$.
- Because $\text{KL}(q\|p) \geq 0$, equalizing 0 iff $q(\mathbf{Z}) = p(\mathbf{Z}|\mathbf{X}, \boldsymbol{\theta})$, this implies that $\mathcal{L}(q, \boldsymbol{\theta}) \leq \ln p(\mathbf{X}|\boldsymbol{\theta})$
- $\mathcal{L}(q, \boldsymbol{\theta})$ turns out to be the lower bound of $\ln p(\mathbf{X}|\boldsymbol{\theta})$.

# Decomposition in Picture Form

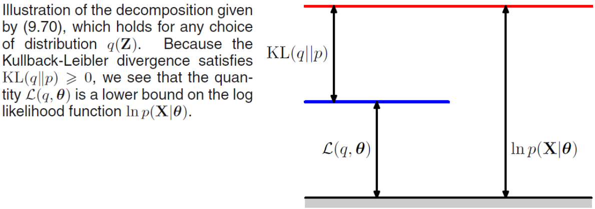


Illustration of the decomposition given by (9.70), which holds for any choice of distribution $q(\mathbf{Z})$. Because the Kullback-Leibler divergence satisfies $\mathrm{KL}(q\|p) \geqslant 0$, we see that the quantity $\mathcal{L}(q, \boldsymbol{\theta})$ is a lower bound on the log likelihood function $\ln p(\mathbf{X}|\boldsymbol{\theta})$.

$\mathrm{KL}(q\|p)$
$\mathcal{L}(q, \boldsymbol{\theta})$
$\ln p(\mathbf{X}|\boldsymbol{\theta})$

Figure: Illustration of Decomposition of $\ln p(\mathbf{X}|\boldsymbol{\theta})$

*Source:* Bishop (2006), pg. 451

# The Basics: The Steps

The EM algorithm is an algorithm with four steps, two of which are iterative:

1. We pick initial values for $\boldsymbol{\theta}^{old}$
2. In the E Step, using these initial values for $\theta$, we maximize the lower bound $\mathcal{L}(q, \boldsymbol{\theta}^{old})$ described in the previous slide, with respect to the function $q(\mathbf{Z})$. That is to say, we hold the $\theta$'s fixed and allow only the distributional form of $q(\mathbf{Z})$ to vary. This produces a new distribution $\mathcal{L}(q, \theta)$.
3. In the M Step, we maximize this function with respect to the parameters $\boldsymbol{\theta}$, while holding $q(\mathbf{Z})$ constant. This produces new parameters $\boldsymbol{\theta}^{new}$.
4. We repeat Steps 2 and 3 (the E and M Steps) until some convergence criteria is satisfied. In other words, $\boldsymbol{\theta}^{new}$ becomes the $\theta$ that are fixed in Step 2, and so on.

# The Particulars: Something to Note

- The EM algorithm is often described as having just two steps, but it is important to realize that it must be initialized (the choice of initial values can have consequences) and that you have to have some convergence tolerance, otherwise the algorithm could run forever. In addition, there are various convergence criteria that could be used.

# The Particulars: Where Does the Expectation Come In?

- Although E stands for expectation, this is something of a misnomer. You are really finding the distribution $q(\mathbf{Z})$ that maximizes the lower bound of the true "likelihood" (and in turn reduces the KL divergence between $q(\mathbf{Z})$ and $p(\mathbf{Z}|\mathbf{X}, \boldsymbol{\theta}^{old})$ to 0) when the parameters are fixed at a certain value. This really means that you are finding the parameters that make up this function.
- It turns out that, because it drives the KL divergence to 0, this distribution $q(\mathbf{Z}) = p(\mathbf{Z}|\mathbf{X}, \boldsymbol{\theta}^{old})$.
- If we substitute this into 3, we see that after the E step

$$\mathcal{L}(q, \boldsymbol{\theta}) = \sum_{\mathbf{Z}} p(\mathbf{Z}|\mathbf{X}, \boldsymbol{\theta}^{old}) \ln p(\mathbf{Z}, \mathbf{X}|\boldsymbol{\theta}) - \sum_{\mathbf{Z}} p(\mathbf{Z}|\mathbf{X}, \boldsymbol{\theta}^{old}) \ln p(\mathbf{Z}, \mathbf{X}|\boldsymbol{\theta}^{old})$$

When it comes to taking the M step, we are optimizing this function w.r.t $\boldsymbol{\theta}$. As there are no $\boldsymbol{\theta}$'s in the second half of the function, this part is treated as a constant.

# The Particulars: Where Does the Expectation Come In?

- We then note that $\sum_{\mathbf{Z}} p(\mathbf{Z}|\mathbf{X}, \boldsymbol{\theta}^{old})\ln p(\mathbf{Z}, \mathbf{X}|\boldsymbol{\theta})$ is just the expectation of the complete data log-likelihood.
- This function is termed, in alternative formulations of the EM algorithm, $\mathcal{Q}(\boldsymbol{\theta}, \boldsymbol{\theta}^{old})$
- If we think through this, we can see that the E step, if we take a few shortcuts, just requires us to calculate the expectation of the complete data log-likelihood, which we then maximize in the M step. This explains the names—but we should recognize that there is a lot more going on in the background!

# E-Step in Picture Form

Illustration of the E step of the EM algorithm. The $q$ distribution is set equal to the posterior distribution for the current parameter values $\boldsymbol{\theta}^{\text{old}}$, causing the lower bound to move up to the same value as the log likelihood function, with the KL divergence vanishing.

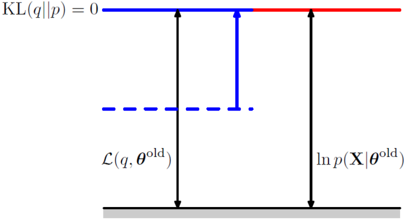

$\mathrm{KL}(q||p) = 0$
$\mathcal{L}(q, \boldsymbol{\theta}^{\text{old}})$
$\ln p(\mathbf{X}|\boldsymbol{\theta}^{\text{old}})$

Figure: Visualization of E Step

*Source:* Bishop (2006), pg. 452

# M-Step in Picture Form

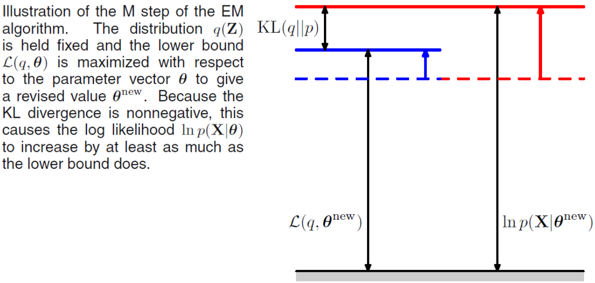


Illustration of the M step of the EM algorithm. The distribution $q(\mathbf{Z})$ is held fixed and the lower bound $\mathcal{L}(q, \boldsymbol{\theta})$ is maximized with respect to the parameter vector $\boldsymbol{\theta}$ to give a revised value $\boldsymbol{\theta}^{\text{new}}$. Because the KL divergence is nonnegative, this causes the log likelihood $\ln p(\mathbf{X}|\boldsymbol{\theta})$ to increase by at least as much as the lower bound does.

$\mathrm{KL}(q||p)$
$\mathcal{L}(q, \boldsymbol{\theta}^{\text{new}})$
$\ln p(\mathbf{X}|\boldsymbol{\theta}^{\text{new}})$

Figure: Visualization of M Step

*Source:* Bishop (2006), pg. 452

# Putting it All Together in Picture Form

The EM algorithm involves alternately computing a lower bound on the log likelihood for the current parameter values and then maximizing this bound to obtain the new parameter values. See the text for a full discussion.

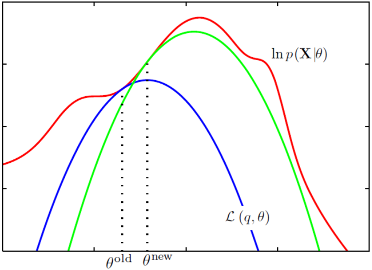


Figure: Visualization of EM Algorithm Process

*Source:* Bishop (2006), pg. 453

# The Particulars: The Intuition Behind Why The EM Monotonically Increases The Observed Data Log-Likelihood

As was shown in the previous slides, due to fact that $\mathcal{L}(q, \boldsymbol{\theta})$ is a lower bound on the , as we iterate the following string of inequalities becomes apparent:

$$\ln p(\mathbf{X}|\boldsymbol{\theta}^t) = \mathcal{L}(q, \boldsymbol{\theta}^t) \leq \ln p(\mathbf{X}|\boldsymbol{\theta}^{t+1}) = \mathcal{L}(q, \boldsymbol{\theta}^{t+1}) \leq \ln p(\mathbf{X}|\boldsymbol{\theta}^{t+2}) \quad (5)$$

In short, in each E step, we bring the lower bound as close as possible to the log likelihood. In each M step, we poke the log likelihood a bit further up the hill.

This means that we are steadily increasing the log likelihood, and we keep doing so until the increases get so small that they are indistinguishable from zero—and that's what we call convergence, folks! **Calculating the log likelihood is a good way to catch errors and bugs: if it decreases, something has gone wrong.**

# Issues with the MLE Version of EM Algorithm

- As it has been explained here, we are still just doing MLE, really, as we are still just climbing a hill
- MLE can overfit
- MLE can fall into local maxima trap and is not guaranteed to find global maxima
    - The EM algorithm needs to be initialized, so need to be careful with initial values, might have to start over various times
- In particular, when it comes to Gaussian Mixture Models, MLE EM can find singularities, which is when a component has zero variance and a mean equal to one of the data points

# Not Just MLE

- This brings us to the important realization that EM is just an algorithm
- There are various ways of approaching it as a whole and the two steps individually
- For example, in order to help with the issue identified in the previous slide, can take a more Bayesian approach and use MAP.
    - Simply add the log prior to the complete data log-likelihood
    - This does not change the E step and only impacts the M step
- It is also possible to go “fully Bayesian,” which will lead us to variational inference!

# EM for Gaussian Mixture Models

We can apply the above discussion to find relatively easy parameter estimates for Gaussian Mixture Models in the following way:

1. Choose initial values for means $\boldsymbol{\mu}_k$, covariances $\boldsymbol{\Sigma}_k$, and mixing coefficients $\pi_k$
2. E Step: Using these parameter values, determine the posterior probability of membership for each observation (also known as *responsibilities*, as mentioned earlier), with the following equation:

$$\gamma(z_{nk}) = \frac{\pi_k \mathcal{N}(x_n|\boldsymbol{\mu}_k, \boldsymbol{\Sigma}_k)}{\sum_{j=1}^{K} \pi_j \mathcal{N}(x_n|\boldsymbol{\mu}_j, \boldsymbol{\Sigma}_j)} \tag{6}$$

This equation can be derived from the expected complete data log-likelihood. See page 351 of Murphy (2012).

# EM for Gaussian Mixture Models

3. M Step: Using the responsibilities calculated in the E-Step, re-estimate the parameters. This will involve maximizing the expected complete data log-likelihood w.r.t each of the parameters in turn. The common thread that runs through the maximizing values is the responsibilities:

$$\pi_k^{new} = \frac{N_k}{N} \tag{7}$$

$$\boldsymbol{\mu}_k^{new} = \frac{1}{N_k} \sum_{n=1}^{N} \gamma(z_{nk}) x_n \tag{8}$$

$$\Sigma_k = \frac{1}{N_k} \sum_{n=1} N\gamma(z_{nk})(x_n - \mu_k^{new})(x_n - \mu_k^{new})^\intercal \tag{9}$$

With

$$N_k = \sum_{n=1} \gamma(z_{nk}) = \# \text{ of points in cluster k} \tag{10}$$

You've seen these already; we discussed them when we first talked about GMM.

# EM for Gaussian Mixture Models

4. Evaluate the log likelihood in order to be able to check convergence and for errors and bugs in the code. You can also check for convergence within the parameters themselves. If convergence has not been reached, return to Step 2 (the E Step).

# Non-Gaussian Mixtures

Mixture models can be generalized to fit any distribution:

$$p(x) = \sum_{k=1}^{K} \pi_k p(x|\theta)$$

Or even to mix *different* distributions:

$$p(x) = \sum_{k=1}^{K} \pi_k p(x|z_i = k, \theta)$$

Or even mixtures of non-parametric distributions - the mixing coefficient places no restrictions on the form of the distribution after it.

# Mixtures of Bernoullis

Another common mixture is a mixture of Bernoulli distributions, also called *latent class analysis*.

As a reminder, the Bernoulli distribution is:

$$p(x|\mu) = \prod_{n=1}^{N} \mu_i^{x_i}(1-\mu_i)^{x_i}$$

And a mixture of Bernoullis is:

$$p(x|\mu, \pi) = \sum_{k=1}^{K} \pi_k p(x|\mu_k)$$

# Mixtures of Bernoullis

This yields the following expressions of the mean and variance:

$$\mathbb{E}(x) = \sum_{k=1}^{K} \pi_k \mu_k$$

$$cov[x] = \sum_{k=1}^{K} \pi_k (\Sigma_k + \mu_k \mu_k^{\intercal}) - \mathbb{E}(x)\mathbb{E}(x)^{\intercal}$$

Where $\Sigma_k$ is the variance of a single Bernoulli component.
What do you notice about the variance? How is this an improvement on a single Bernoulli?

# Mixtures of Bernoullis

The log likelihood is then:

$$lnp(\mathbf{X}|\mu, \pi) = \sum_{n=1}^{N} ln(\sum_{k=1}^{K} \pi_k p(x_n|\mu_k))$$

Does this produce a closed form solution?
Like GMM, does this have the potential to produce a singularity? Why or why not?

# EM with LCA

To use the EM algorithm on a mixture of Bernoulli's, we again introduce $z$ as latent mixture assignment. The conditional distribution $p(x|z,\mu)$ is then:

$$p(x|z,\mu) = \sum_{k=1}^{K} p(x|\mu_k)^{z_k}$$

And the prior for $z$:

$$p(z|\mu) = \sum_{k=1}^{K} \pi_k^{z_k}$$

Just like with the GMM, if we take the product of the conditional and the prior then marginalize out $z$, we get the original formulation back.

# EM with LCA

This then yields the complete-data log likelihood:

$$lnp(\mathbf{X},\mathbf{Z}|\mu,\pi) = \sum_{n=1}^{N}\sum_{k=1}^{K} z_{nk}\{ln\pi_k + \sum_{i=1}^{D}[x_{ni}ln\mu_{ki} + (1-x_{ni})ln(1-\mu_{ki})]\}$$

How do we derive the E step from the complete data likelihood?

# EM with LCA

The E step evaluates the expectation for the latent variables w.r.t to the posterior distribution of **Z**:

$$\mathbb{E}_{\mathbf{Z}}(.) = \gamma(z_{nk})\{ln\pi_k + \sum_{i=1}^{D}[x_{ni}ln\mu_{ki} + (1 - x_{ni})ln(1 - \mu_{ki})]\}$$

Where, just like with the GMM, the posterior is calculated using Bayes' rule:

$$\gamma(z_{nk}) = \frac{\pi_k p(x_n|\mu_k)}{\sum_{j=1}^{K} \pi_j p(x_n|\mu_j)}$$

What does the M step do after this?

# Putting It All Together

This suggests a generalized way to put together any mixture model for estimation with EM:

1. Create a PDF, $p(x)$, that includes a mixing coefficient $\pi$
2. Re-conceptualize that PDF in terms of a latent variable, $z$
3. Find the complete data log-likelihood
4. Take the expectation of step 3 w.r.t. to $Z$
5. Calculate the expression of the posterior of $Z$ using Bayes' rule.
6. Run the EM algorithm, and hopefully get interesting results!

Extra credit question: if you write your own EM algorithm based on the above, how can you check that it's working properly?