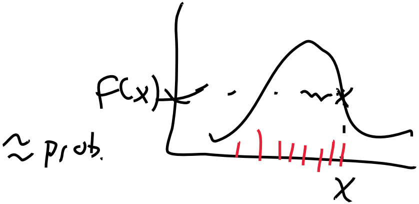
CSCI 4022 Spring 2021
EM Algorithm



Example: The *joint density* of a collection of independent and identically distribution Gaussian random variables $\vec{x} = [x_1, x_2, \ldots, x_n]$ is

$$f(\vec{x}|\mu, \sigma^2) = \boxed{\prod_{i=1}^{n}} \boxed{\frac{1}{\sqrt{2\pi\sigma^2}} e^{\frac{-(x_i-\mu)^2}{2\sigma^2}}}$$

density of ONE normal $N(\mu, \sigma^2)$

What is the *maximum likelihood estimate* for μ?

→ product → joint density of n Normals, iid $N(\mu, \sigma^2)$.

Announcements and To-Dos

Z's OH:

M 3-5p

Tu 3-5p

W-F 3-4p .

Announcements:

1. HW 2 due Tuesday not tonight!

Min form roundup:

Gaussian Mixture Model (GMM) Recap

Goal: describe a data set by a combination or *mixture* of Gaussian/bell curves. **The whole model:** $X = \underbrace{(1-\Delta)}_{\Delta=0 \Rightarrow \text{Red}} \cdot X_1 + \underbrace{\Delta}_{\Delta=1 \Rightarrow \text{Blue}} \cdot X_2$ We need to estimate:

$$\Theta = (\boxed{\pi}, \boxed{\mu_1, \sigma_1^2}, \boxed{\mu_2, \sigma_2^2})$$

If we denote $\phi(x|\mu, \sigma^2)$ as the normal with mean μ and variance σ^2, the model is now

$$\underline{f(x|\Theta)} = (1-\pi)\phi(x|\mu_1, \sigma_1^2) + \pi \underbrace{\phi(x|\mu_2, \sigma_2^2)}_{\text{density of normal: } \frac{1}{\sqrt{2\pi\sigma_2^2}} e^{\frac{-(x-\mu_2)^2}{2\sigma_2^2}}}$$

1. The GMM is a *generative* model, since it specifies the probabilities for new data points.
2. We have to estimate the μ, σ/Σ values for each Gaussian *component* and the π *mixture* proportions.
3. We can have any number M of mixtures! Instead of one probability π, we'd give each Gaussian a *weight* w_i, and ensure that $\sum_{m=1}^{M} w_m = 1$. Our pdf is now:

$$f(x|\Theta) = \underbrace{w_1\phi(x|\mu_1, \sigma_1^2)} + \underbrace{w_2\phi(x|\mu_2, \sigma_2^2)} + \cdots + \underbrace{w_M\phi(x|\mu_M, \sigma_M^2)}$$

← (can approximate anything smooth with enough normals.

Likelihoods

$X=(x_1,x_2)$

Our goal in the GMM problem is to estimate *parameters* - for the Gaussian that's a mean, variance, and possibly cov. → Σ

Covariance: $E[(X_1 - E[X_1])(X_2 - E[X_2])]$ center μ

dim 1&2

σ^2 spread

We think about a pdf as a function that measures the probability of the data **given** the parameters. If we want to "flip" those things, Bayes' theorem comes to the rescue!

density: data GIVEN parameters

$$P(\theta|x) = \frac{\boxed{P(x|\theta)}P\cancel{(}\theta)}{P(\cancel{x})}$$

$\underset{\theta}{\operatorname{argmax}}\, x$

likelihood.

$P(\theta)=$ constant

...but $\frac{P(\theta)}{P(x)}$ are just constants, so we can often take the *joint density* of a data set - the list of **and** multiplications that represent each data point as a separate *actualization* of the pdf - and perform our optimization on that function!

Definition: The *likelihood function* of random variable with pdf $f(x;\Theta)$ and parameters Θ with a set of observations $\vec{x} = [x_1, x_2, \ldots x_n]$ is the probability of the parameters given the data; $L(\theta|\vec{x})$.

The Gaussian Likelihood

Time to get to the real payoff: how do we estimate parameters if our data is **Gaussian**?

The setup to maximum likelihood is always the same: we look at all n of our data points $x_1, x_2, \ldots x_n$ and ask about the probability of $f(x = x_i|\mu, \sigma^2)$ for each one, then multiply them all together as an "and" or *joint* probability. Then we hit it with a logarithm to make maximization easier.

$\log(a \cdot b) = \log(a) + \log(b)$

$$\underset{\theta = \mu, \sigma^2}{\text{argmax}} \; \log f(\vec{x}|\mu, \sigma^2) = \log \prod_{i=1}^{n} \frac{1}{\sqrt{2\pi\sigma^2}} e^{\frac{-(x_i-\mu)^2}{2\sigma^2}}$$

$$= \sum_{i=1}^{n} \ln\left(\frac{1}{\sqrt{2\pi\sigma^2}}\right) + \cancel{\ln\left(e\right)}^{\frac{-(x_i-\mu)^2}{2\sigma^2}}$$

Log-likelihood.

$$= \sum_{i=1}^{n} -\frac{1}{2}\ln\left(2\pi\sigma^2\right) - \frac{1}{2\sigma^2}\left(x_i - \mu\right)^2$$

The Gaussian Likelihood

Time to get to the real payoff: how do we estimate parameters if our data is **Gaussian**?

The setup to maximum likelihood is always the same: we look at all n of our data points $x_1, x_2, \ldots x_n$ and ask about the probability of $f(x = x_i|\mu, \sigma^2)$ for each one, then multiply them all together as an "and" or *joint* probability. Then we hit it with a logarithm to make maximization easier.

$$
\begin{aligned}
f(\vec{x}|\mu,\sigma^2) &= \prod_{i=1}^{n} \frac{1}{\sqrt{2\pi\sigma^2}} e^{\frac{-(x_i-\mu)^2}{2\sigma^2}} \\
LL(\mu,\sigma^2|x) &= \log \prod_{i=1}^{n} \frac{1}{\sqrt{2\pi\sigma^2}} e^{\frac{-(x_i-\mu)^2}{2\sigma^2}} \\
&= \frac{-n}{2} \log\left(2\pi\sigma^2\right) - \frac{1}{2\sigma^2}\sum_{i=1}^{n}(x_i-\mu)^2
\end{aligned}
$$

The Gaussian Likelihood

Finally, we have to take two derivatives and set both $\frac{d}{d\mu}LL$ and $\frac{d}{d\sigma^2}LL$ equal to zero.

outside sum

$$LL(\mu, \sigma^2|x) = \frac{-n}{2}\log\left(2\pi\sigma^2\right) - \frac{1}{2\sigma^2}\sum_{i=1}^{n}(x_i - \mu)^2$$

(partial)

$$\frac{d}{d\mu}LL = \frac{d}{d\boxed{\mu}}\left(\frac{-n}{2}\log\left(2\pi\sigma^2\right) - \frac{1}{2\sigma^2}\sum_{i=1}^{n}(x_i - \boxed{\mu})^2\right)$$

NO μ

$$\frac{d}{dx}(a-x)^2$$

$$\frac{d}{dx} = 2\cdot(a-x)(-1)$$

The Gaussian Likelihood

<u>Different</u> than $\bar{X}$ is $\arg\min_c \sum (x_i - c)^2$

Finally, we have to take two derivatives and set both $\frac{d}{d\mu}LL$ and $\frac{d}{d\sigma^2}LL$ equal to zero.

$$LL(\mu,\sigma^2|x) = \frac{-n}{2}\log\left(2\pi\sigma^2\right) - \frac{1}{2\sigma^2}\sum_{i=1}^{n}(x_i-\mu)^2$$

$$\frac{d}{d\mu}LL = \frac{d}{d\mu}\left(\frac{-n}{2}\underset{0}{\cancel{\log}}\left(2\pi\sigma^2\right) - \frac{1}{2\sigma^2}\sum_{i=1}^{n}(x_i-\mu)^2\right)$$

$$= -\frac{1}{2\sigma^2}\sum_{i=1}^{n} -2\underline{(x_i-\mu)}$$

set; remove $\frac{1}{\sigma^2}$

$$0 = \sum_{i=1}^{n}(x_i-\mu) = \sum_{i=1}^{n}(x_i) \boxed{-\sum_{i=1}^{n}\mu}$$

$$\implies n\mu = \sum_{i=1}^{n}(x_i) \implies \mu = \bar{X} = \frac{\sum x_i}{n}$$

our best choice to estimate $\mu = \bar{X}$

The Gaussian Likelihood

What does that even mean? It means that the "best guess" of the mean μ of the *probability density function* giving rise to our data was the sample mean $\bar{x}$. This is one of the measures as to why the sample mean is a **great measure** of population mean.

as $n \to \infty$, $\bar{x} \to \mu$, $\bar{x} \sim N(\mu, \sigma^2)$

There are others, including the central limit theorem and the result that the sample mean is closest point in terms of sums of squared deviations (Euclidean distance) to each other point.

What about σ^2?

The Gaussian Likelihood: Variance

Now, for $\frac{d}{d\sigma^2}LL$, then setting equal to zero.

$$LL(\mu, \sigma^2|x) = \frac{-n}{2}\log\left(2\pi\sigma^2\right) - \frac{1}{2\sigma^2}\sum_{i=1}^{n}(x_i-\mu)^2$$

don't usually, now σ^2

$$\frac{d}{d\sigma^2}LL = \boxed{\frac{d}{d\sigma^2}}\left(\frac{-n}{2}\log\left(2\pi\sigma^2\right) - \frac{1}{2\sigma^2}\sum_{i=1}^{n}(x_i-\mu)^2\right)$$

$$= \frac{-n}{2\sigma^2} + \frac{1}{2(\sigma^2)^2}\sum_{i=1}^{n}(x_i-\mu)^2$$

$$\implies 0 = -n\sigma^2 + \sum_{i=1}^{n}(x_i-\mu)^2$$

$$\implies \boxed{\sigma^2 = \frac{\sum_{i=1}^{n}(x_i-\mu)^2}{n}}$$

...which looks a lot like the classical estimate for *sample variance*, $\boxed{s^2 = \frac{\sum_{i=1}^{n}(x_i-\mu)^2}{n-1}}$

MLE: Big Picture

The **maximum likelihood estimate** (MLEs) for μ, σ^2 of a Gaussian is $\overset{\mu}{\uparrow}$ $\bar{x}$, $\overset{\sigma^2}{\uparrow}$ $\frac{\sum(x_i-\bar{x})^2}{n}$... but it turns out MLEs are often slightly worse - especially for small samples - than the estimates you learned in your intro class. They're just a way to arrive at similar results: if you want to estimate the mean and variance of a normal, you use *sample* mean and *sample* variance.

And you can use *sample* covariance for covariances! (/n versions).

The takeaways from MLE theory:

1. Hitting a probability density function with calculus gives us a method to find "best" values of a distributions' underlying parameters.
2. Since *joint density* functions and *likelihood* functions are large products, we often use their logarithms for ARGMAX calculations instead.

   np.mean np.var np.cov
3. For a Gaussian, the classical **sample** estimates for means, variance, covariances are good for estimating **population** parameters. These MLEs have denominator of n, not $n-1$.

The GMM Likelihood

Recall: our model is more than just a single Gaussian. It's multiple, added up!

$$\underline{f(x|\Theta) = (1-\pi)\phi(x|\mu_1, \sigma_1^2) + \pi\phi(x|\mu_2, \sigma_2^2)}$$

2 components
1-D

In order to estimate this and create a log-likelihood, we want to break down the pdf into two steps. Rather than a single instance of π, we define the **latent variable** Δ_i:

$$\begin{cases} \Delta_i = 0 & \text{if } x_i \text{ is from component 1} \\ \Delta_i = 1 & \text{if } x_i \text{ is from component 2} \end{cases}$$

How does this help? Recall how the **Bernoulli** random variable works:
$P(X_i = \Delta_i) = \pi^{\Delta_i}(1-\pi)^{1-\Delta_i}$; this returns π or $1-\pi$ as we input 0 or 1 for Δ_i.

$f(x) = p^x (1-p)^{1-x} \quad x = 0, 1$

The GMM Likelihood

$$\Delta_{im} = \begin{cases} 1 & \text{if } x_i \text{ is from mixture } m \\ 0 & \text{if } x_i \text{ is from another.} \end{cases}$$

By using an individual Δ_i for each data point, we can combine terms like $P(\Delta_i) = \pi^{\Delta_i}(1-\pi)^{1-\Delta_i}$ with a rewritten pdf of the form

$$f(x_i|\Delta_i) = \phi(x|\mu_1,\sigma_1^2)^{\boxed{1-\Delta_i}} \underbrace{\phi(x|\mu_2,\sigma_2^2)^{\boxed{\Delta_i}}}_{\text{if } \Delta_i = 0 \text{ becomes } (\ )^0 = 1.}$$

if $\Delta_i = 0$

survives

The GMM Likelihood

By using an individual Δ_i for each data point, we can combine terms like $P(\Delta_i) = \pi^{\Delta_i}(1-\pi)^{1-\Delta_i}$ with a rewritten pdf of the form

$$f(x_i|\Delta_i) = \phi(x|\mu_1, \sigma_1^2)^{1-\Delta_i}\phi(x|\mu_2, \sigma_2^2)^{\Delta_i}$$

which we can combine and then hit with a logarithm to get a *log-likelihood* function for the GME:

$$L(\theta|x) = \sum_{i=1}^{N}\left[(1-\Delta_i)\underbrace{\log\phi(x|\mu_1,\sigma_1^2)}_{\text{pdf}} + \Delta_i \underbrace{\log\phi(x|\mu_2,\sigma_2^2)]}_{\text{pdf}}\right.$$

$$+\sum_{i=1}^{N}[(1-\Delta_i)\log(1-\underbrace{\pi) + \Delta_i \log}_{\text{Bernoulli.}}\pi]$$

Argmax:

μ_1 μ_2

σ_1^2 σ_2^2

π

The GMM Likelihood

Of course, we don't actually know the 0-or-1 value of Δ_i for each data point. So we have to estimate it along the way! Part of the GMM solution is to regularly estimate which component/mixture each data point came from, because where it came from is part of the pdf!

Definition: We estimate the Δ_i by using the *responsibility* of each component, in light of the current estimates of the parameters Θ and the complete data set x.

For our two-component model, we would write $\gamma_i(\Theta) :=$ the probability that data point x_i came from component 2, given our current estimates of Θ and the overall x:

$$\gamma_i(\Theta) = P(\Delta_i = 1 | \Theta, x)$$

The GMM Likelihood

Of course, we don't actually know the 0-or-1 value of Δ_i for each data point. So we have to estimate it along the way! Part of the GMM solution is to regularly estimate which component/mixture each data point came from, because where it came from is part of the pdf!

Definition: We estimate the Δ_i by using the *responsibility* of each component, in light of the current estimates of the parameters Θ and the complete data set x.

For our two-component model, we would write $\gamma_i(\Theta) :=$ the probability that data point x_i came from component 2, given our current estimates of Θ and the overall x:

$$\gamma_i(\Theta) = P(\Delta_i = 1|\Theta, x)$$

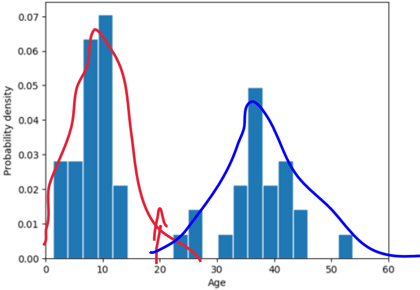
This suggest an important difference between GMMs and k-means

- k-means: we declare which cluster each point belongs to: **hard** clustering.
- GMM: we define a probability for each cluster for each point: **soft** clustering.

The Gaussian Mixture Model (GMM): 1D Example

We want probabilities that *any given* data point was generated by *any given* cluster of our Gaussian Mixture.

Example: In our Chuck E. Cheese data, we have a patron who is 20 years old. Is it more likely to belong to the child component of $\mu_1 = 10$ or to the adult component of $\mu_2 = 38$?



What about if we consider that the adult component had more variance?

The Gaussian Mixture Model (GMM): m-Dim

Let's add some generality to the GMM.

Define $d :=$ the # of dimensions

Denote $i = 1, 2, \dots N$ be the # of data points, $\underline{x_i \in \mathbb{R}^d}$ Denote $m = 1, 2, \dots M$ be the # of mixtures/components, so we have means $\mu_m \in \mathbb{R}^d$ and covariance matrices $\Sigma_m \in \mathbb{M}_{\underline{d \times d}}$

Goal: represent a probability that point x_i came from cluster m.

$$p_{mi} := \boxed{P(C = m|x_i)}$$

The Gaussian Mixture Model (GMM): m-Dim

Let's add some generality to the GMM.

Define $d :=$ the # of dimensions

Denote $i = 1, 2, \dots N$ be the # of data points, $x_i \in \mathbb{R}^d$ Denote $m = 1, 2, \dots M$ be the # of mixtures/components, so we have means $\mu_m \in \mathbb{R}^d$ and covariance matrices $\Sigma_m \in \mathbb{M}_{d \times d}$

Goal: represent a probability that point x_i came from cluster m.

$$p_{mi} := P(\underset{\sim}{C} = m|x_i)$$

We know probabilities of x's *given* the right cluster! Bayes:

$$p_{mi} := P(C = m|x_i) = \frac{\overset{\text{data given cluster... pdf}}{P(x_i|C = m)}\overset{\text{mixing}}{\boxed{P(C = m)}}}{P(x_i)}$$

The Gaussian Mixture Model (GMM): Piece by Piece

$$\boxed{\overset{0}{p_{mi}}} := P(C = m|x_i) = \frac{\overset{1}{P(x_i|C = m)}\overset{2}{P(C = m)}}{\underset{3}{P(x_i)}}$$

So what are all of these terms?

1. $\boxed{\underline{p_{mi}}} = P(\underline{C = m}|x_i)$. This is the actual **goal** of a GMM: which cluster/classification does each point belong to? $\sim \Delta_i$.

2. $\boxed{P(x_i|C = m)}$. Probability of observing point x_i from *known* component m. Same as evaluating the pdf $f(x)$ of a normal distribution with mean μ_m and covariance Σ_m at the value $x = x_i$. stats.norm.pdf
   Recall that in 1-D, the matrix Σ_m is just a scalar σ_m^2.

The Gaussian Mixture Model (GMM): Piece by Piece

$$p_{mi} := P(C = m | x_i) = \frac{P(x_i | C = m)\boxed{P(C = m)}}{P(x_i)}$$

So what are all of these terms?

3. $\boxed{P(C = m)}$. The *weight* of component m. In our Chuck E. Cheese model this was the weights $w_1 = 1 - \pi$, $w_2 = \pi$. w_i estimate $P(\Delta$ is from ith cluster$)$.
   In general, we can estimate this as

$$w_m = \frac{n_m}{N} = \frac{\#\text{ of data points in cluster } m}{\text{Total }\#\text{ of data points}} \quad \frac{\checkmark}{\text{total}}$$

   ... but we have "fractional" or probabilities of points-in-clusters, so we actually have to use $n_m = \sum_{i=1}^{N} p_{mi}$

4. $\boxed{P(x_i)}$. How about the denominator?

The Gaussian Mixture Model (GMM): Piece by Piece

$$\boxed{p_{mi}} = P(C = m|x_i) = \frac{\overset{\text{(pdf)}}{P(x_i|C = m)}\overset{(\pi\text{s/weights})}{P(C = m)}}{\boxed{P(x_i)}}$$

There are always two options on a Bayes' theorem denominator.

- **Option 1:** use the Law of Total Probability to compute the full joint pdf $P(x_i)$ by *summing* over all possible outcomes. $P(x_i) = \underline{\sum_{i=1}^{M} P(x_i|C = k)P(C = k)}$
- **Option 2:** Be a little bit lazy and realize that we're *already* going to calculate $P(x_i|C = k)P(C = k)$ (the numerator!) for each data point and cluster. Since we're calculating a set of probabilities, they have to sum up to 1:

$$\sum_{m=1}^{M} p_{mi} = \sum_{m=1}^{M} P(C = m|x_i) = 1$$

The Gaussian Mixture Model (GMM): Piece by Piece

Prob point i belongs to cluster m

$$\boxed{p_{mi}} := P(C=m|x_i) = \frac{P(x_i|C=m)P(C=m)}{P(x_i)}$$

Denominator Implementation: Denote $\frac{1}{P(x_i)} = \alpha$.

Then we have $p_{mi} := P(C=m|x_i) = \alpha \boxed{P(x_i|C=m)} P(C=m)$, and we *know* $\sum_{m=1}^{M} p_{mi} = 1$.

- Calculate an un-normalized $\underline{\tilde{p}_{mi}} = \frac{\tilde{p}_{mi}}{\alpha} = \underbrace{P(x_i|C=m)P(C=m)}_{\text{numerator (.M numerators)}}$ for each point-cluster pair.
- Then add them all up. $\sum_{m=1}^{M} \tilde{p}_{mi} = \frac{\sum_{m=1}^{M} p_{mi}}{\alpha} = \frac{1}{\alpha}$, so
- $\alpha = \boxed{\frac{1}{\sum_{m=1}^{M} p_{mi}}}$.

e.g. if we had $\begin{bmatrix} 1 \\ 3 \end{bmatrix} \rightarrow \begin{bmatrix} 1/4 \\ 3/4 \end{bmatrix}$ as probs.

We've computed all 3 terms on the right side! We can get probabilities of clusters *given* the data values... but also given the parameters of each mixture.

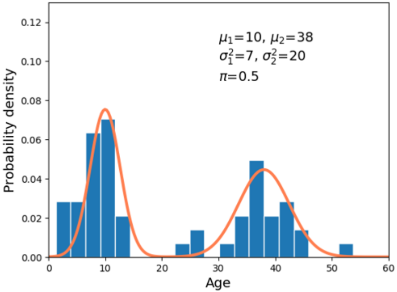
The Gaussian Mixture Model (GMM): 1D Example

Example: What cluster do we assign a 20-year old patron to, given the parameters?

1. $=$

.

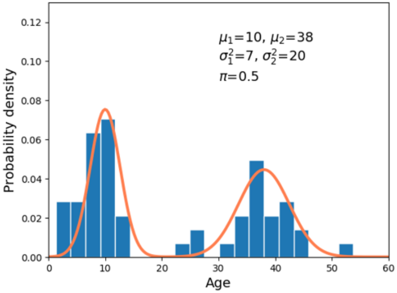
2.



The Gaussian Mixture Model (GMM): 1D Example

Example: What cluster do we assign a 20-year old patron to, given the parameters?

Solution: We need to compute $\tilde{p}_{mi}$ for each group: the probability that a 20-year would come from each group.



1. $\tilde{p}_{1,i} = P(20\text{ y.o. kid}) = P(20\text{ y.o. }\textbf{given}\text{ kid})P(\text{kid}) =$

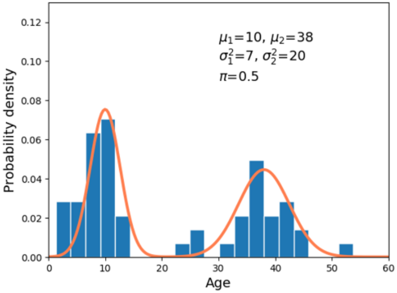
   .

2.

The Gaussian Mixture Model (GMM): 1D Example

Example: What cluster do we assign a 20-year old patron to, given the parameters?

Solution: We need to compute $\tilde{p}_{mi}$ for each group: the probability that a 20-year would come from each group.



1. $\tilde{p}_{1,i} = P(\text{20 y.o. kid}) = P(\text{20 y.o. } \textbf{given} \text{ kid})P(\text{kid}) =$
   $P(20|C=1)P(C=1) = \underbrace{\phi(20|\mu_1 = 10,\, \sigma_1^2 = 7)}_{Normal} \underbrace{\cdot \frac{1}{2}}_{\pi}$

   .

2.

The Gaussian Mixture Model (GMM): 1D Example

Example: What cluster do we assign a 20-year old patron to, given the parameters?

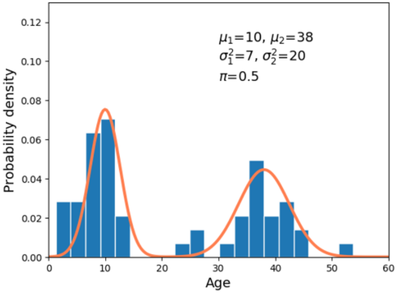
Solution: We need to compute $\tilde{p}_{mi}$ for each group: the probability that a 20-year would come from each group.

1. $\tilde{p}_{1,i} = P(\text{20 y.o. kid}) = P(\text{20 y.o. } \textbf{given} \text{ kid})P(\text{kid}) =$

   $$P(20|C=1)P(C=1) = \underbrace{\phi(20|\mu_1 = 10, \sigma_1^2 = 7)}_{Normal} \cdot \underbrace{\frac{1}{2}}_{\pi}$$

   $$= 5.96 \times 10^{-5}.$$

2.



The Gaussian Mixture Model (GMM): 1D Example

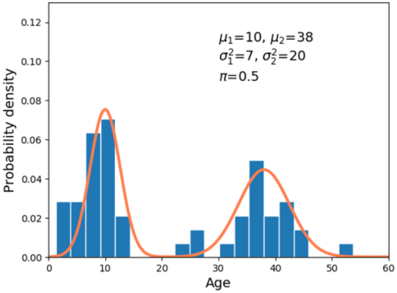
Example: What cluster do we assign a 20-year old patron to, given the parameters?

Solution: We need to compute $\tilde{p}_{mi}$ for each group: the probability that a 20-year would come from each group.

1. $\tilde{p}_{1,i} = P(\text{20 y.o. kid}) = P(\text{20 y.o. } \textbf{given} \text{ kid})P(\text{kid}) = P(20|C=1)P(C=1) = \underbrace{\phi(20|\mu_1 = 10,\, \sigma_1^2 = 7)}_{Normal} \cdot \underbrace{\frac{1}{2}}_{\pi}$

   $= 5.96 \times 10^{-5}$.

2. $\tilde{p}_{2,i} = P(\text{20 y.o. adult}) = P(\text{20 y.o. } \textbf{given} \text{ adult})P(\text{adult}) = P(20|C=2)P(C=2) = \phi(20|\mu_1 = 38\, \sigma_2^2 = 20) \cdot \frac{1}{2} = 1.35 \times 10^{-5}$



The Gaussian Mixture Model (GMM): 1D Example

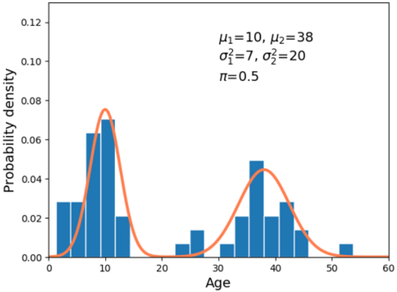
Example: What cluster do we assign a 20-year old patron to, given the parameters?

1. $\tilde{p}_{1,i} = P(\text{20 y.o. kid}) = P(\text{20 y.o. } \textbf{given} \text{ kid})P(\text{kid}) =$

   $= 5.96 \times 10^{-5}.$

2. $\tilde{p}_{2,i} = P(\text{20 y.o. adult})$

$$= 1.35 \times 10^{-5}$$



Intuition on these results: in general, the probability of a 20-year old kid is 4x higher than the probability of that of a 20-year-old adult.

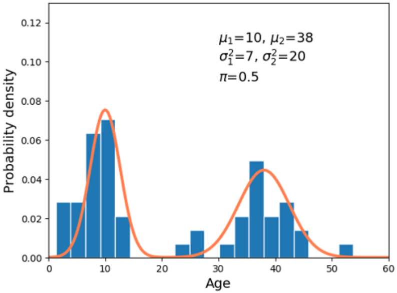
The Gaussian Mixture Model (GMM): P(cluster)

Example: What cluster do we assign a 20-year old patron to, given the parameters? We have estimates of: $\tilde{p}_{1,i} = 5.96 \times 10^{-5}$ and $\tilde{p}_{2,i} = 1.35 \times 10^{-5}$.

If we just wanted to **classify** (or *hard cluster*), we could say $\tilde{p}_{1,i}$ is larger than the other cluster, so it is more likely the $x_i = 20$ observation same from cluster 1... but we really want these to be probabilities!

1. So we find $\alpha = \tilde{p}_{1,i} + \tilde{p}_{2,i} = 7.31 \times 10^{-5}$...
2. And we compute $p_{mi} = \tilde{p}_{mi}/\alpha$:

$$p_{1,i} = \frac{5.96 \times 10^{-5}}{7.31 \times 10^{-5}} = 0.81; \quad p_{2,i} = \frac{1.35 \times 10^{-5}}{7.31 \times 10^{-5}} = 0.19$$



Back to the MLE

That example was how we might perform estimations of clusters **given** model parameters. But we still don't know how to get the values of μ, Σ, and π! The whole log-likelihood from earlier is below:

Pm: approximate Δ_i's Probability

$$LL(\theta|x) = \sum_{i=1}^{N} \left[(1 - \Delta_i) \log \phi(x|\mu_1, \sigma_1^2) + \Delta_i \log \phi(x|\mu_2, \sigma_2^2) \right]$$

$$+ \sum_{i=1}^{N} \left[(1 - \Delta_i) \log(1 - \pi) + \Delta_i \log \pi \right]$$

done!

What if we know Δ_i? How does this maximum look?

Back to the MLE

That example was how we might perform estimations of clusters **given** model parameters. But we still don't know how to get the values of μ, Σ, and π! The whole log-likelihood from earlier is below:

$$LL(\theta|x) = \sum_{i=1}^{N} \left[(1-\Delta_i) \log \phi(x|\boxed{\mu_1, \sigma_1^2}) + \Delta_i \log \phi(x|\boxed{\mu_2, \sigma_2^2}) \right]$$

$$+ \sum_{i=1}^{N} \left[(1-\Delta_i) \log(1-\boxed{\pi}) + \Delta_i \log \boxed{\pi} \right]$$

What if we know Δ_i? How does this maximum look?

Each piece is now a completely disconnected, **seperate** part of the sum. We're still trying to ARGMAX the Log-likelihood, but the values for π; μ_1 **and** σ_1^2; μ_2 **and** σ_2^2 are all in completely different terms!

Back to the MLE

That example was how we might perform estimations of clusters **given** model parameters. But we still don't know how to get the values of μ, Σ, and π! The whole log-likelihood from earlier is below:

$$LL(\theta|x) = \sum_{i=1}^{N} \left[(1 - \Delta_i) \log \phi(x|\mu_1, \sigma_1^2) + \Delta_i \log \phi(x|\mu_2, \sigma_2^2) \right] + \sum_{i=1}^{N} \left[(1 - \Delta_i) \log(1 - \pi) + \Delta_i \log \pi \right]$$

What we just did was describe a scheme for estimating the probabilities of Δ_i **given** the other terms. This is a common practice in *iterative* optimization: estimate one thing, use those values to estimate another thing, and then loop.

Payoff: Once we "know" Δ_i, we just have to estimate properties of Gaussians... and we get to do so one at a time!

The EM Algorithm

The real algorithm for estimating the full model is going to use our cluster-given-parameters, but then *iteratively* alternate between

1. Estimating the responsibilities of each point to each cluster:
   $\underbrace{\gamma_{mi}(\Theta) = P(\Delta_i = m|\Theta, x)}_{true\,mixing\,proportion}$ estimated with $\underbrace{p_{mi} = \frac{\tilde{p}_{mi}}{\alpha}}_{pointwise\text{ ``}mixing\text{''}}$
2. Estimate the parameter estimates μ_m, Σ_m, w_m **given** those updated p_{mi} estimates.

This is the *Expectation-Maximization Algorithm*, and is used extensively in data science whenever problems can be turned into maximum likelihood.

The EM Algorithm: Initialize

Step 0: Initialize all parts of the model.

- Initialize the Σ_m values. A reasonable choice is set each and every Σ_m matrix to be equal to the covariance matrix of the entire data set ($\textsc{np.cov}$).
- Initialize the μ_m values:
  1. Like k-means, could do it randomly...
  2. or pick M data points to start as centers but enforce that they're spread out...
  3. or use a small subset of the data, use k-means or hierarchical clustering and pick those centroids/clustroids
- Unless we have a good reason not to, start with $w_m = \frac{1}{M}$, or all clusters equally weighted/likely.

The EM Algorithm: Expectation

Step 1: *Expectation Step.* Intuition: what do we *expect* the probabilities/assignments to be? We soft assign each datum a probability to each and every component, based on the relative density functions of that datum for each of the component training models.

- *For each* data point i **and** *for each* component m, compute

$$\tilde{p}_{mi} = P(x_i|cluster = m)P(cluster = m) = \phi(x_i|\hat{\mu}_m, \hat{\Sigma}_m)\hat{w}_m$$

and then consolidate into the probabilities using $1/\alpha_i = \sum_m \tilde{p}_{mi}$:

$$\hat{p}_{mi} = \frac{\tilde{p}_{mi}}{\sum_m \tilde{p}_{mi}}$$

normalization

The EM Algorithm: Expectation

Step 1: *Expectation Step.* Intuition: what do we *expect* the probabilities/assignments to be? We soft assign each datum a probability to each and every component, based on the relative density functions of that datum for each of the component training models.

- *For each* data point i **and** *for each* component m, compute

$$\tilde{p}_{mi} = P(x_i|cluster = m)P(cluster = m) = \phi(x_i|\hat{\mu}_m, \hat{\Sigma}_m)\hat{w}_m$$

  and then consolidate into the probabilities using $1/\alpha_i = \sum_m \tilde{p}_{mi}$:

$$\hat{p}_{mi} = \frac{\tilde{p}_{mi}}{\sum_m \tilde{p}_{mi}}$$

NB: you may have seen the "hat" notation in your intro class. In statistical estimation, we often use hats to denote our *estimates* of the underlying true population values. Since we're only estimating probabilities of points-within-components using estimates of those components' means and variances, everything has a hat!

The EM Algorithm: Maximiation

Step 2: *Maximization Step.* Intuition: how do we find the means and variances of a component given the points within it? The goal here is to maximize the likelihood (or more generally any objective function). **Sample means and sample variances** are best estimates (like MLEs!) for population means/variances.

- *For each* component m, count the number of points in it $\hat{n_m}$. Note that this will be fractional, not a whole number!
- *For each* component m, estimate its mean μ_m by using a *weighted* sample mean: since a point could be only e.g. 10% associated with a component, it might only count for 0.1 of a point on the mean calculation
- *For each* component m, estimate its variance-covariance matrix Σ_m by using a *weighted* sample covariance

The EM Algorithm: Maximiation

Step 2: *Maximization Step. For each* component m:

- Count the number of points in it $\hat{n_m}$. Then get proportions/weights:

$$\hat{w}_m = \frac{\hat{n_m}}{N} = \frac{\sum_{i=1}^{N} \hat{p}_{mi}}{N}$$

- Estimate its mean μ_m by using a *weighted* sample mean:

$$\hat{\mu}_m = \frac{1}{\hat{n_m}} \sum_{i=1}^{N} \hat{p}_{mi} \cdot x_i$$

- Estimate its variance-covariance Σ_m by *weighted* sample covariance:

$$\hat{\Sigma}_m = \frac{1}{\hat{n_m}} \sum_{i=1}^{N} \hat{p}_{mi} \cdot (x_i - \hat{\mu}_m)\,(x_i - \hat{\mu}_m)^T$$

The EM Algorithm: All together

And we're done! **Step 0:** Initialize clusters and their means, variances, equal proportions.

Step 1: *Expectation*. For each data point x_i and for each each component m

1. $\tilde{p}_{mi} = \phi(x_i | \hat{\mu}_m, \hat{\Sigma}_m)\hat{w}_m$ and then consolidate into the probabilities:
2. $\hat{p}_{mi} = \dfrac{\tilde{p}_{mi}}{\sum_m \tilde{p}_{mi}}$

Step 2: *Maximization*. For each component m,

1. $\hat{w}_m = \dfrac{\hat{n_m}}{N} = \dfrac{\sum_{i=1}^N \hat{p}_{mi}}{N}$
2. $\hat{\mu}_m = \dfrac{1}{\hat{n_m}} \displaystyle\sum_{i=1}^N \hat{p}_{mi} \cdot x_i$
3. $\hat{\Sigma}_m = \dfrac{1}{\hat{n_m}} \displaystyle\sum_{i=1}^N \hat{p}_{mi} \cdot (x_i - \hat{\mu}_m)(x_i - \hat{\mu}_m)^T$

Step 3: Convergence check! Are things changing?

We will do a worked example on our EM algorithm notebook Friday.

The EM Algorithm: More than just Gauss

You may note here that we could have fit *any* probability density into components and soft-clusters here, not just normals/Gaussians! We would have to replace the parameters μ, Σ of the normal to the alternative parameters Θ of the underlying distributions. We'd still fit M components and weights $\hat{w}_m$.

How would this change the EM algorithm?

The EM Algorithm: More than just Gauss

You may note here that we could have fit *any* probability density into components and soft-clusters here, not just normals/Gaussians! We would have to replace the parameters μ, Σ of the normal to the alternative parameters Θ of the underlying distributions. We'd still fit M components and weights $\hat{w}_m$.

How would this change the EM algorithm? **Solution:** Not that much!

Step 1: *Expectation*. For each data point x_i and for each each component m

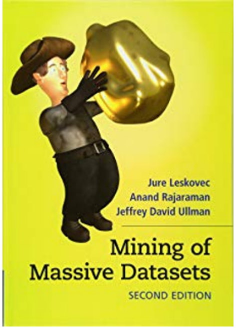
1. $\tilde{p}_{mi} = \underbrace{f(x_i|\Theta)}_{any\, pdf} \hat{w}_m$ and then consolidate into the probabilities:
2. $\hat{p}_{mi} = \dfrac{\tilde{p}_{mi}}{\sum_m \tilde{p}_{mi}}$

Step 2: *Maximization*. For each component m,

1. $\hat{w}_m = \dfrac{\hat{n_m}}{N} = \dfrac{\sum_{i=1}^{N} \hat{p}_{mi}}{N}$
2. Estimate whatever is inside Θ... somehow (MLEs? OPTIM? Bootstrapping?)

Acknowledgments

Some material is adapted/adopted from Mining of Massive Data Sets, by Jure Leskovec, Anand Rajaraman, Jeff Ullman (Stanford University) `http://www.mmds.org`



Special thanks to Tony Wong for sharing his original adaptation and adoption of slide material.