Parameter Estimation in Latent Variable Models

Piyush Rai

Introduction to Machine Learning (CS771A)

September 25, 2018
Some Mid-Sem Statistics

- Minimum: 5.5
- Median: 32.0
- Maximum: 73.5
- Mean: 33.46
- Std Dev: 12.65
Some Mid-Sem Statistics

MINIMUM 5.5
MEDIAN 32.0
MAXIMUM 73.5
MEAN 33.46
STD DEV 12.65

Can recover
Need not worry
too much
Also normal
Need not worry
Normal
Need not worry
1
Some Mid-Sem Statistics

Also normal
Need not worry

Normal
Need not worry

MINIMUM  5.5
MEDIAN  32.0
MAXIMUM  73.5
MEAN  33.46
STD DEV  12.65
Some Mid-Sem Statistics

![Bar chart showing distribution of scores and statistical summary]

- **Minimum**: 5.5
- **Median**: 32.0
- **Maximum**: 73.5
- **Mean**: 33.46
- **Std Dev**: 12.65
Latent Variable Models
A Simple Generative Model

- All observations \( \{x_1, \ldots, x_N\} \) generated from a distribution \( p(x|\theta) \)

\[ \text{Depends on type of } x \text{ (e.g., Gaussian)} \]
A Simple Generative Model

- All observations \( \{x_1, \ldots, x_N\} \) generated from a distribution \( p(x|\theta) \)

- Unknowns: Parameters \( \theta \) of the assumed data distribution \( p(x|\theta) \)
A Simple Generative Model

- All observations \( \{x_1, \ldots, x_N\} \) generated from a distribution \( p(x|\theta) \)

- Unknowns: Parameters \( \theta \) of the assumed data distribution \( p(x|\theta) \)
- Many ways to estimate the parameters (MLE, MAP, or Bayesian inference)
Assume each observation $x_n$ to be associated with a latent variable $z_n$. In this "latent variable model" of data, data $x_n$ also depends on some latent variable(s) $z_n$. $z_n$ is akin to a latent representation or "encoding" of $x_n$; controls what data "looks like". E.g., $z_n \in \{1, \ldots, K\}$ denotes the cluster $x_n$ belongs to. $z_n \in \mathbb{R}^K$ denotes a low-dimensional latent representation or latent "code" for $x_n$. Unknowns: \{ $z_1, \ldots, z_N$ \}, and (\theta, \phi). $z_n$'s called "local" variables; (\theta, \phi) called "global" variables.
Generative Model with Latent Variables

- Assume each observation $x_n$ to be associated with a latent variable $z_n$

- In this "latent variable model" of data, data $x$ also depends on some latent variable(s) $z$
Generative Model with Latent Variables

- Assume each observation $x_n$ to be associated with a latent variable $z_n$

In this “latent variable model” of data, data $x$ also depends some latent variable(s) $z$

- $z_n$ is akin to a latent representation or “encoding” of $x_n$; controls what data “looks like”.

[Diagram of generative model with latent variables]
Generative Model with Latent Variables

- Assume each observation $x_n$ to be associated with a latent variable $z_n$

In this “latent variable model” of data, data $x$ also depends some latent variable(s) $z$

- $z_n$ is akin to a latent representation or “encoding” of $x_n$; controls what data “looks like”. E.g,
  - $z_n \in \{1, \ldots, K\}$ denotes the cluster $x_n$ belongs to
Generative Model with Latent Variables

- Assume each observation $x_n$ to be associated with a latent variable $z_n$

- In this “latent variable model” of data, data $x$ also depends some latent variable(s) $z$

- $z_n$ is akin to a latent representation or “encoding” of $x_n$; controls what data “looks like”. E.g,
  - $z_n \in \{1, \ldots, K\}$ denotes the cluster $x_n$ belongs to
  - $z_n \in \mathbb{R}^K$ denotes a low-dimensional latent representation or latent “code” for $x_n$
Generative Model with Latent Variables

- Assume each observation $x_n$ to be associated with a latent variable $z_n$

- In this “latent variable model” of data, data $x$ also depends some latent variable(s) $z$

- $z_n$ is akin to a latent representation or “encoding” of $x_n$; controls what data “looks like”. E.g,
  - $z_n \in \{1, \ldots, K\}$ denotes the cluster $x_n$ belongs to
  - $z_n \in \mathbb{R}^K$ denotes a low-dimensional latent representation or latent “code” for $x_n$

- Unknowns: $\{z_1, \ldots, z_N\}$, and $(\theta, \phi)$. 
Generative Model with Latent Variables

- Assume each observation $x_n$ to be associated with a latent variable $z_n$

- In this “latent variable model” of data, data $x$ also depends some latent variable(s) $z$

- $z_n$ is akin to a latent representation or “encoding” of $x_n$; controls what data “looks like”. E.g,
  - $z_n \in \{1, \ldots, K\}$ denotes the cluster $x_n$ belongs to
  - $z_n \in \mathbb{R}^K$ denotes a low-dimensional latent representation or latent “code” for $x_n$

- Unknowns: $\{z_1, \ldots, z_N\}$, and $(\theta, \phi)$. $z_n$’s called “local” variables; $(\theta, \phi)$ called “global” variables
Brief Detour/Recap: Gaussian Parameter Estimation
MLE for Multivariate Gaussian

- Multivariate Gaussian in $D$ dimensions

$$p(x|\mu, \Sigma) = \frac{1}{(2\pi)^{D/2}|\Sigma|^{1/2}} \exp \left( -\frac{1}{2} (x - \mu)^\top \Sigma^{-1} (x - \mu) \right)$$

Goal: Given $N$ i.i.d. observations $\{x_n\}_{n=1}^N$ from this Gaussian, estimate parameters $\mu$ and $\Sigma$

$$\hat{\mu} = \frac{1}{N} \sum_{n=1}^N x_n$$
$$\hat{\Sigma} = \frac{1}{N} \sum_{n=1}^N (x_n - \hat{\mu})(x_n - \hat{\mu})^\top$$

Note: $\Sigma$ depends on $\mu$, but $\mu$ doesn't depend on $\Sigma$ ⇒ no need for alternating opt.

Note: log works nicely with exp of the Gaussian. Simplifies MLE expressions in this case

In general, when the distribution is an exponential family distribution, MLE is usually very easy
MLE for Multivariate Gaussian

- Multivariate Gaussian in $D$ dimensions

$$p(x|\mu, \Sigma) = \frac{1}{(2\pi)^{D/2}|\Sigma|^{1/2}} \exp \left( -\frac{1}{2} (x - \mu)^\top \Sigma^{-1} (x - \mu) \right)$$

- Goal: Given $N$ i.i.d. observations $\{x_n\}_{n=1}^N$ from this Gaussian, estimate parameters $\mu$ and $\Sigma$
MLE for Multivariate Gaussian

- Multivariate Gaussian in $D$ dimensions

$$p(x | \mu, \Sigma) = \frac{1}{(2\pi)^{D/2}|\Sigma|^{1/2}} \exp \left( -\frac{1}{2} (x - \mu)^\top \Sigma^{-1} (x - \mu) \right)$$

- Goal: Given $N$ i.i.d. observations $\{x_n\}_{n=1}^N$ from this Gaussian, estimate parameters $\mu$ and $\Sigma$

- MLE for the $D \times 1$ mean $\mu \in \mathbb{R}^D$ and $D \times D$ p.s.d. covariance matrix $\Sigma$

$$\hat{\mu} = \frac{1}{N} \sum_{n=1}^N x_n \quad \text{and} \quad \hat{\Sigma} = \frac{1}{N} \sum_{n=1}^N (x_n - \hat{\mu})(x_n - \hat{\mu})^\top$$

Note: $\Sigma$ depends on $\mu$, but $\mu$ doesn't depend on $\Sigma$ $\Rightarrow$ no need for alternating opt.

Note: log works nicely with exp of the Gaussian. Simplifies MLE expressions in this case

In general, when the distribution is an exponential family distribution, MLE is usually very easy
MLE for Multivariate Gaussian

- Multivariate Gaussian in $D$ dimensions
  
  $p(x|\mu, \Sigma) = \frac{1}{(2\pi)^{D/2}|\Sigma|^{1/2}} \exp \left( -\frac{1}{2} (x - \mu)^\top \Sigma^{-1} (x - \mu) \right)$

- Goal: Given $N$ i.i.d. observations $\{x_n\}_{n=1}^N$ from this Gaussian, estimate parameters $\mu$ and $\Sigma$

- MLE for the $D \times 1$ mean $\mu \in \mathbb{R}^D$ and $D \times D$ p.s.d. covariance matrix $\Sigma$

  $\hat{\mu} = \frac{1}{N} \sum_{n=1}^N x_n$ and $\hat{\Sigma} = \frac{1}{N} \sum_{n=1}^N (x_n - \hat{\mu})(x_n - \hat{\mu})^\top$

- Note: $\Sigma$ depends on $\mu$, but $\mu$ doesn't depend on $\Sigma \Rightarrow$ no need for alternating opt.
MLE for Multivariate Gaussian

- Multivariate Gaussian in $D$ dimensions

$$
p(x|\mu, \Sigma) = \frac{1}{(2\pi)^{D/2}|\Sigma|^{1/2}} \exp \left( -\frac{1}{2} (x - \mu)^\top \Sigma^{-1} (x - \mu) \right)
$$

- Goal: Given $N$ i.i.d. observations $\{x_n\}_{n=1}^N$ from this Gaussian, estimate parameters $\mu$ and $\Sigma$

- MLE for the $D \times 1$ mean $\mu \in \mathbb{R}^D$ and $D \times D$ p.s.d. covariance matrix $\Sigma$

$$
\hat{\mu} = \frac{1}{N} \sum_{n=1}^N x_n \quad \text{and} \quad \hat{\Sigma} = \frac{1}{N} \sum_{n=1}^N (x_n - \hat{\mu})(x_n - \hat{\mu})^\top
$$

- Note: $\Sigma$ depends on $\mu$, but $\mu$ doesn't depend on $\Sigma \Rightarrow$ no need for alternating opt.

- Note: log works nicely with exp of the Gaussian. Simplifies MLE expressions in this case
MLE for Multivariate Gaussian

- Multivariate Gaussian in $D$ dimensions

$$p(x|\mu, \Sigma) = \frac{1}{(2\pi)^{D/2}|\Sigma|^{1/2}} \exp \left( -\frac{1}{2} (x - \mu)^\top \Sigma^{-1} (x - \mu) \right)$$

- Goal: Given $N$ i.i.d. observations $\{x_n\}_{n=1}^N$ from this Gaussian, estimate parameters $\mu$ and $\Sigma$

- MLE for the $D \times 1$ mean $\mu \in \mathbb{R}^D$ and $D \times D$ p.s.d. covariance matrix $\Sigma$

$$\hat{\mu} = \frac{1}{N} \sum_{n=1}^{N} x_n \quad \text{and} \quad \hat{\Sigma} = \frac{1}{N} \sum_{n=1}^{N} (x_n - \hat{\mu})(x_n - \hat{\mu})^\top$$

- Note: $\Sigma$ depends on $\mu$, but $\mu$ doesn't depend on $\Sigma$ $\Rightarrow$ no need for alternating opt.

- Note: log works nicely with exp of the Gaussian. Simplifies MLE expressions in this case

- In general, when the distribution is an exponential family distribution, MLE is usually very easy
A Brief Detour: Exponential Family Distributions

- An exponential family distribution is of the form

\[ p(x|\theta) = h(x) \exp[\theta^T \phi(x) - A(\theta)] \]

- \( \theta \) is called the natural parameter of the family

Many well-known distributions (Bernoulli, Binomial, Multinoulli, Beta, Gamma, Gaussian, etc.) are exponential family distributions.

https://en.wikipedia.org/wiki/Exponential_family
An exponential family distribution is of the form

\[ p(x|\theta) = h(x) \exp[\theta^\top \phi(x) - A(\theta)] \]

\( \theta \) is called the natural parameter of the family

\( h(x), \phi(x), \) and \( A(\theta) \) are known functions. Note: Don’t confuse \( \phi \) with kernel mappings!
Brief Detour: Exponential Family Distributions

- An exponential family distribution is of the form

\[ p(x|\theta) = h(x) \exp[\theta^T \phi(x) - A(\theta)] \]

- \( \theta \) is called the natural parameter of the family
- \( h(x), \phi(x), \) and \( A(\theta) \) are known functions. Note: Don’t confuse \( \phi \) with kernel mappings!
- \( \phi(x) \) is called the sufficient statistics: knowing this is sufficient to estimate \( \theta \)

[https://en.wikipedia.org/wiki/Exponential_family]
Brief Detour: Exponential Family Distributions

- An exponential family distribution is of the form

\[ p(x|\theta) = h(x) \exp[\theta^\top \phi(x) - A(\theta)] \]

- \( \theta \) is called the natural parameter of the family
- \( h(x), \phi(x), \) and \( A(\theta) \) are known functions. Note: Don’t confuse \( \phi \) with kernel mappings!
- \( \phi(x) \) is called the sufficient statistics: knowing this is sufficient to estimate \( \theta \)
- Every exp. family distribution also has a conjugate distribution (often also in exp. family)
An exponential family distribution is of the form

\[ p(x|\theta) = h(x) \exp[\theta^\top \phi(x) - A(\theta)] \]

\( \theta \) is called the natural parameter of the family

\( h(x), \phi(x), \) and \( A(\theta) \) are known functions. Note: Don’t confuse \( \phi \) with kernel mappings!

\( \phi(x) \) is called the sufficient statistics: knowing this is sufficient to estimate \( \theta \)

Every exp. family distribution also has a conjugate distribution (often also in exp. family)

Many other nice properties (especially useful in Bayesian inference)
Brief Detour: Exponential Family Distributions

- An exponential family distribution is of the form

  \[ p(x|\theta) = h(x) \exp[\theta^\top \phi(x) - A(\theta)] \]

- \( \theta \) is called the **natural parameter** of the family

- \( h(x), \phi(x), \text{ and } A(\theta) \) are known functions. Note: Don’t confuse \( \phi \) with kernel mappings!

- \( \phi(x) \) is called the **sufficient statistics**: knowing this is sufficient to estimate \( \theta \)

- Every exp. family distribution also has a conjugate distribution (often also in exp. family)

- Many other nice properties (especially useful in Bayesian inference)

- Also, MLE/MAP is usually quite simple (note that \( \log p(x|\theta) \) will typically have a simple form)
Brief Detour: Exponential Family Distributions

- An exponential family distribution is of the form

\[ p(x|\theta) = h(x) \exp[\theta^\top \phi(x) - A(\theta)] \]

- \( \theta \) is called the **natural parameter** of the family
- \( h(x), \phi(x), \) and \( A(\theta) \) are known functions. Note: Don’t confuse \( \phi \) with kernel mappings!
- \( \phi(x) \) is called the **sufficient statistics**: knowing this is sufficient to estimate \( \theta \)
- Every exp. family distribution also has a conjugate distribution (often also in exp. family)
- Many other nice properties (especially useful in Bayesian inference)
- Also, MLE/MAP is usually quite simple (note that \( \log p(x|\theta) \) will typically have a simple form)

Many well-known distribution (Bernoulli, Binomial, multinoulli, beta, gamma, Gaussian, etc.) are exponential family distributions

https://en.wikipedia.org/wiki/Exponential_family
MLE for Generative Classification with Gaussian Class-conditionals

- Each class $k$ modeled using a Gaussian with mean $\mu_k$ and covariance matrix $\Sigma_k$
MLE for Generative Classification with Gaussian Class-conditionals

- Each class $k$ modeled using a Gaussian with mean $\mu_k$ and covariance matrix $\Sigma_k$
- Note: Can assume label $y_n$ to be one-hot and then $y_{nk} = 1$ if $y_n = k$, and $y_{nk} = 0$, otherwise
MLE for Generative Classification with Gaussian Class-conditionals

- Each class $k$ modeled using a Gaussian with mean $\mu_k$ and covariance matrix $\Sigma_k$
- Note: Can assume label $y_n$ to be one-hot and then $y_{nk} = 1$ if $y_n = k$, and $y_{nk} = 0$, otherwise
- Assuming $p(y_n = k) = \pi_k$, this model has parameters $\Theta = \{\pi_k, \mu_k, \Sigma_k\}_{k=1}^K$

Basically estimating $K$ Gaussians instead of just 1 (each using data only from that class)
MLE for Generative Classification with Gaussian Class-Conditionals

- Each class $k$ modeled using a Gaussian with mean $\mu_k$ and covariance matrix $\Sigma_k$
- Note: Can assume label $y_n$ to be one-hot and then $y_{nk} = 1$ if $y_n = k$, and $y_{nk} = 0$, otherwise
- Assuming $p(y_n = k) = \pi_k$, this model has parameters $\Theta = \{\pi_k, \mu_k, \Sigma_k\}_{k=1}^K$
- (We have done this before) Given $\{x_n, y_n\}_{n=1}^N$, MLE for $\Theta$ will be
MLE for Generative Classification with Gaussian Class-conditionals

- Each class $k$ modeled using a Gaussian with mean $\mu_k$ and covariance matrix $\Sigma_k$
- Note: Can assume label $y_n$ to be one-hot and then $y_{nk} = 1$ if $y_n = k$, and $y_{nk} = 0$, otherwise
- Assuming $p(y_n = k) = \pi_k$, this model has parameters $\Theta = \{\pi_k, \mu_k, \Sigma_k\}_{k=1}^K$
- (We have done this before) Given $\{x_n, y_n\}_{n=1}^N$, MLE for $\Theta$ will be

$$\hat{\pi}_k = \frac{1}{N} \sum_{n=1}^N y_{nk}$$
MLE for Generative Classification with Gaussian Class-conditionals

- Each class $k$ modeled using a Gaussian with mean $\mu_k$ and covariance matrix $\Sigma_k$
- Note: Can assume label $y_n$ to be one-hot and then $y_{nk} = 1$ if $y_n = k$, and $y_{nk} = 0$, otherwise
- Assuming $p(y_n = k) = \pi_k$, this model has parameters $\Theta = \{\pi_k, \mu_k, \Sigma_k\}_{k=1}^K$
- (We have done this before) Given $\{x_n, y_n\}_{n=1}^N$, MLE for $\Theta$ will be

\[
\hat{\pi}_k = \frac{1}{N} \sum_{n=1}^N y_{nk} = \frac{N_k}{N}
\]
MLE for Generative Classification with Gaussian Class-conditionals

- Each class $k$ modeled using a Gaussian with mean $\mu_k$ and covariance matrix $\Sigma_k$
- Note: Can assume label $y_n$ to be one-hot and then $y_{nk} = 1$ if $y_n = k$, and $y_{nk} = 0$, otherwise
- Assuming $p(y_n = k) = \pi_k$, this model has parameters $\Theta = \{\pi_k, \mu_k, \Sigma_k\}_{k=1}^K$
- (We have done this before) Given $\{x_n, y_n\}_{n=1}^N$, MLE for $\Theta$ will be

$$\hat{\pi}_k = \frac{1}{N} \sum_{n=1}^N y_{nk} = \frac{N_k}{N}$$

$$\hat{\mu}_k = \frac{1}{N_k} \sum_{n=1}^N y_{nk} x_n$$
MLE for Generative Classification with Gaussian Class-conditionals

- Each class \( k \) modeled using a Gaussian with mean \( \mu_k \) and covariance matrix \( \Sigma_k \)
- Note: Can assume label \( y_n \) to be one-hot and then \( y_{nk} = 1 \) if \( y_n = k \), and \( y_{nk} = 0 \), otherwise
- Assuming \( p(y_n = k) = \pi_k \), this model has parameters \( \Theta = \{\pi_k, \mu_k, \Sigma_k\}_{k=1}^K \)
- (We have done this before) Given \( \{x_n, y_n\}_{n=1}^N \), MLE for \( \Theta \) will be

\[
\hat{\pi}_k = \frac{1}{N} \sum_{n=1}^N y_{nk} = \frac{N_k}{N}
\]

\[
\hat{\mu}_k = \frac{1}{N_k} \sum_{n=1}^N y_{nk} x_n
\]

\[
\hat{\Sigma}_k = \frac{1}{N_k} \sum_{n=1}^N y_{nk} (x_n - \hat{\mu}_k) (x_n - \hat{\mu}_k)^\top
\]

Basically estimating \( K \) Gaussians instead of just 1 (each using data only from that class)
MLE for Generative Classification with Gaussian Class-conditionals

- Each class \( k \) modeled using a Gaussian with mean \( \mu_k \) and covariance matrix \( \Sigma_k \)
- Note: Can assume label \( y_n \) to be one-hot and then \( y_{nk} = 1 \) if \( y_n = k \), and \( y_{nk} = 0 \), otherwise
- Assuming \( p(y_n = k) = \pi_k \), this model has parameters \( \Theta = \{\pi_k, \mu_k, \Sigma_k\}_{k=1}^K \)
- (We have done this before) Given \( \{x_n, y_n\}_{n=1}^N \), MLE for \( \Theta \) will be
  \[
  \hat{\pi}_k = \frac{1}{N} \sum_{n=1}^N y_{nk} = \frac{N_k}{N} \\
  \hat{\mu}_k = \frac{1}{N_k} \sum_{n=1}^N y_{nk} x_n \\
  \hat{\Sigma}_k = \frac{1}{N_k} \sum_{n=1}^N y_{nk} (x_n - \hat{\mu}_k)(x_n - \hat{\mu}_k)^T
  \]
- Basically estimating \( K \) Gaussians instead of just 1 (each using data only from that class)
MLE for Generative Classification with Gaussian Class-conditionals

Let’s look at the “formal” procedure of deriving MLE in this case

\[ \hat{\Theta} = \arg \max_{\Theta} p(X, y|\Theta) = \arg \max_{\Theta} N \prod_{n=1}^{N} p(x_n, y_n|\Theta) = \arg \max_{\Theta} N \prod_{n=1}^{N} K \prod_{k=1}^{K} p(y_n=k|\Theta)p(x_n|y_n=k,\Theta) \]

\[ = \arg \max_{\Theta} N \sum_{n=1}^{N} K \sum_{k=1}^{K} y_{nk} \left[ \log p(y_n=k|\Theta) + \log p(x_n|y_n=k,\Theta) \right] \]

Given \((X, y)\), optimizing it w.r.t. \(\pi_k, \mu_k, \Sigma_k\) will give us the solution we saw on the previous slide.
MLE for Generative Classification with Gaussian Class-conditionals

Let’s look at the “formal” procedure of deriving MLE in this case

MLE for $\Theta = \{\pi_k, \mu_k, \Sigma_k\}_{k=1}^K$ in this case can be written as (assuming i.i.d. data)

$$\hat{\Theta} = \arg \max_{\Theta} p(X, y|\Theta)$$
Let’s look at the “formal” procedure of deriving MLE in this case

MLE for $\Theta = \{\pi_k, \mu_k, \Sigma_k\}_{k=1}^K$ in this case can be written as (assuming i.i.d. data)

$$\hat{\Theta} = \arg \max_{\Theta} p(X, y|\Theta) = \arg \max_{\Theta} \prod_{n=1}^N p(x_n, y_n|\Theta) =$$
Let’s look at the “formal” procedure of deriving MLE in this case

MLE for \( \Theta = \{\pi_k, \mu_k, \Sigma_k\}_{k=1}^K \) in this case can be written as (assuming i.i.d. data)

\[
\hat{\Theta} = \arg \max_{\Theta} p(X, y | \Theta) = \arg \max_{\Theta} \prod_{n=1}^N p(x_n, y_n | \Theta) = \arg \max_{\Theta} \prod_{n=1}^N p(y_n | \Theta) p(x_n | y_n, \Theta)
\]
Let’s look at the “formal” procedure of deriving MLE in this case.

MLE for $\Theta = \{\pi_k, \mu_k, \Sigma_k\}_{k=1}^K$ in this case can be written as (assuming i.i.d. data)

$$\hat{\Theta} = \arg\max_{\Theta} p(X, y | \Theta) = \arg\max_{\Theta} \prod_{n=1}^N p(x_n, y_n | \Theta) = \arg\max_{\Theta} \prod_{n=1}^N p(y_n | \Theta)p(x_n | y_n, \Theta)$$

$$= \arg\max_{\Theta} \prod_{n=1}^N \prod_{k=1}^K [p(y_n = k | \Theta)p(x_n | y_n = k, \Theta)]^{y_{nk}}$$
MLE for Generative Classification with Gaussian Class-conditionals

- Let’s look at the “formal” procedure of deriving MLE in this case
- MLE for $\Theta = \{\pi_k, \mu_k, \Sigma_k\}_{k=1}^{K}$ in this case can be written as (assuming i.i.d. data)

$$\hat{\Theta} = \arg \max_{\Theta} \log \prod_{n=1}^{N} \left[ \prod_{k=1}^{K} \left[ p(y_n = k|\Theta) p(x_n|y_n = k, \Theta) \right] \right]^{y_{nk}}$$
Let’s look at the “formal” procedure of deriving MLE in this case

MLE for \( \Theta = \{ \pi_k, \mu_k, \Sigma_k \}_{k=1}^{K} \) in this case can be written as (assuming i.i.d. data)

\[
\hat{\Theta} = \arg \max_{\Theta} p(\mathbf{X}, \mathbf{y} | \Theta) = \arg \max_{\Theta} \prod_{n=1}^{N} p(x_n, y_n | \Theta) = \arg \max_{\Theta} \prod_{n=1}^{N} p(y_n | \Theta) p(x_n | y_n, \Theta)
\]

\[
= \arg \max_{\Theta} \prod_{n=1}^{N} \prod_{k=1}^{K} [p(y_n = k | \Theta) p(x_n | y_n = k, \Theta)]^{y_{nk}}
\]

\[
= \arg \max_{\Theta} \log \prod_{n=1}^{N} \prod_{k=1}^{K} [p(y_n = k | \Theta) p(x_n | y_n = k, \Theta)]^{y_{nk}}
\]

\[
= \arg \max_{\Theta} \sum_{n=1}^{N} \sum_{k=1}^{K} y_{nk} [\log p(y_n = k | \Theta) + \log p(x_n | y_n = k, \Theta)]
\]
MLE for Generative Classification with Gaussian Class-conditionals

- Let’s look at the “formal” procedure of deriving MLE in this case
- MLE for \( \Theta = \{\pi_k, \mu_k, \Sigma_k\}_{k=1}^K \) in this case can be written as (assuming i.i.d. data)

\[
\hat{\Theta} = \arg \max_{\Theta} p(\mathbf{X}, y | \Theta) = \arg \max_{\Theta} \prod_{n=1}^{N} p(x_n, y_n | \Theta) = \arg \max_{\Theta} \prod_{n=1}^{N} p(y_n | \Theta)p(x_n | y_n, \Theta)
\]

\[
= \arg \max_{\Theta} \prod_{n=1}^{N} \prod_{k=1}^{K} [p(y_n = k | \Theta)p(x_n | y_n = k, \Theta)]^{y_{nk}}
\]

\[
= \arg \max_{\Theta} \log \prod_{n=1}^{N} \prod_{k=1}^{K} [p(y_n = k | \Theta)p(x_n | y_n = k, \Theta)]^{y_{nk}}
\]

\[
= \arg \max_{\Theta} \sum_{n=1}^{N} \sum_{k=1}^{K} y_{nk} \left[ \log p(y_n = k | \Theta) + \log p(x_n | y_n = k, \Theta) \right]
\]

\[
= \arg \max_{\Theta} \sum_{n=1}^{N} \sum_{k=1}^{K} y_{nk} \left[ \log \pi_k + \log \mathcal{N}(x_n | \mu_k, \Sigma_k) \right]
\]
MLE for Generative Classification with Gaussian Class-conditionals

- Let's look at the "formal" procedure of deriving MLE in this case
- MLE for \( \Theta = \{\pi_k, \mu_k, \Sigma_k\}_{k=1}^{K} \) in this case can be written as (assuming i.i.d. data)

\[
\hat{\Theta} = \text{arg max}_{\Theta} p(X, y|\Theta) = \text{arg max}_{\Theta} \prod_{n=1}^{N} p(x_n, y_n|\Theta) = \text{arg max}_{\Theta} \prod_{n=1}^{N} p(y_n|\Theta)p(x_n|y_n, \Theta)
\]

\[
= \text{arg max}_{\Theta} \prod_{n=1}^{N} \prod_{k=1}^{K} [p(y_n = k|\Theta)p(x_n|y_n = k, \Theta)]^{y_{nk}}
\]

\[
= \text{arg max}_{\Theta} \log \prod_{n=1}^{N} \prod_{k=1}^{K} [p(y_n = k|\Theta)p(x_n|y_n = k, \Theta)]^{y_{nk}}
\]

\[
= \text{arg max}_{\Theta} \sum_{n=1}^{N} \sum_{k=1}^{K} y_{nk} [\log p(y_n = k|\Theta) + \log p(x_n|y_n = k, \Theta)]
\]

\[
= \text{arg max}_{\Theta} \sum_{n=1}^{N} \sum_{k=1}^{K} y_{nk} [\log \pi_k + \log \mathcal{N}(x_n|\mu_k, \Sigma_k)]
\]

- Given \((X, y)\), optimizing it w.r.t. \(\pi_k, \mu_k, \Sigma_k\) will give us the solution we saw on the previous slide
MLE When Labels Go Missing..

So the MLE problem for generative classification with Gaussian class-conditionals was

\[
\hat{\Theta} = \arg \max_{\Theta} \log p(X, y | \Theta) = \arg \max_{\Theta} \sum_{n=1}^{N} \sum_{k=1}^{K} y_{nk} \left[ \log \pi_k + \log \mathcal{N}(x_n | \mu_k, \Sigma_k) \right]
\]
MLE When Labels Go Missing.

So the MLE problem for generative classification with Gaussian class-conditionals was

\[ \hat{\Theta} = \arg \max_{\Theta} \log p(X, y | \Theta) = \arg \max_{\Theta} \sum_{n=1}^{N} \sum_{k=1}^{K} y_{nk} [\log \pi_k + \log \mathcal{N}(x_n | \mu_k, \Sigma_k)] \]

This problem has a nice separable structure, and a straightforward solution as we saw.
MLE When Labels Go Missing..

- So the MLE problem for generative classification with Gaussian class-conditionals was
  \[
  \hat{\Theta} = \arg \max_{\Theta} \log p(X, y|\Theta) = \arg \max_{\Theta} \sum_{n=1}^{N} \sum_{k=1}^{K} y_{nk} \left[ \log \pi_k + \log N(x_n|\mu_k, \Sigma_k) \right]
  \]

- This problem has a nice separable structure, and a straightforward solution as we saw

- What if we don’t know the label \( y_n \) (i.e., don’t know if \( y_{nk} \) is 0 or 1)?
So the MLE problem for generative classification with Gaussian class-conditionals was

\[ \hat{\Theta} = \arg \max_{\Theta} \log p(X, y|\Theta) = \arg \max_{\Theta} \sum_{n=1}^{N} \sum_{k=1}^{K} y_{nk} [\log \pi_k + \log \mathcal{N}(x_n|\mu_k, \Sigma_k)] \]

This problem has a nice separable structure, and a straightforward solution as we saw.

What if we don’t know the label \( y_n \) (i.e., don’t know if \( y_{nk} \) is 0 or 1)? How to estimate \( \Theta \) now?
MLE When Labels Go Missing..

So the MLE problem for generative classification with Gaussian class-conditionals was

$$\hat{\Theta} = \arg \max_\Theta \log p(X, y|\Theta) = \arg \max_\Theta \sum_{n=1}^N \sum_{k=1}^K y_{nk} [\log \pi_k + \log N(x_n|\mu_k, \Sigma_k)]$$

This problem has a nice separable structure, and a straightforward solution as we saw

What if we don’t know the label $y_n$ (i.e., don’t know if $y_{nk}$ is 0 or 1)? How to estimate $\Theta$ now?

When might we need to solve such a problem?
MLE When Labels Go Missing..

- So the MLE problem for generative classification with Gaussian class-conditionals was

\[
\hat{\Theta} = \arg \max_{\Theta} \log p(X, y|\Theta) = \arg \max_{\Theta} \sum_{n=1}^{N} \sum_{k=1}^{K} y_{nk} [\log \pi_k + \log \mathcal{N}(x_n|\mu_k, \Sigma_k)]
\]

- This problem has a nice separable structure, and a straightforward solution as we saw

- What if we don’t know the label \(y_n\) (i.e., don’t know if \(y_{nk}\) is 0 or 1)? How to estimate \(\Theta\) now?

- **When might we need to solve such a problem?**
  - **Mixture density estimation:** Given \(N\) inputs \(x_1, \ldots, x_N\), model \(p(x)\) as a mixture of distributions
MLE When Labels Go Missing..

- So the MLE problem for generative classification with Gaussian class-conditionals was
  \[
  \hat{\Theta} = \arg \max_{\Theta} \log p(X, y|\Theta) = \arg \max_{\Theta} \sum_{n=1}^{N} \sum_{k=1}^{K} y_{nk} \left[ \log \pi_k + \log \mathcal{N}(x_n|\mu_k, \Sigma_k) \right]
  \]

- This problem has a nice separable structure, and a straightforward solution as we saw

- What if we don’t know the label \(y_n\) (i.e., don’t know if \(y_{nk}\) is 0 or 1)? How to estimate \(\Theta\) now?

- **When might we need to solve such a problem?**
  - **Mixture density estimation:** Given \(N\) inputs \(x_1, \ldots, x_N\), model \(p(x)\) as a mixture of distributions

  ![Mixture Density Estimation Diagram]

  - **Probabilistic clustering:** Same as density estimation; can get cluster ids once \(\Theta\) is estimated
So the MLE problem for generative classification with Gaussian class-conditionals was:

\[ \hat{\Theta} = \arg \max_{\Theta} \log p(X, y|\Theta) = \arg \max_{\Theta} \sum_{n=1}^{N} \sum_{k=1}^{K} y_{nk} \left[ \log \pi_k + \log \mathcal{N}(x_n|\mu_k, \Sigma_k) \right] \]

This problem has a nice separable structure, and a straightforward solution as we saw.

What if we don’t know the label \( y_n \) (i.e., don’t know if \( y_{nk} \) is 0 or 1)? How to estimate \( \Theta \) now?

**When might we need to solve such a problem?**

- **Mixture density estimation**: Given \( N \) inputs \( x_1, \ldots, x_N \), model \( p(x) \) as a mixture of distributions.

- **Probabilistic clustering**: Same as density estimation; can get cluster ids once \( \Theta \) is estimated.

- **Semi-supervised generative classification**: In training data, some \( y_n \)'s are known, some not known.
Recall the MLE problem for $\Theta$ when the labels are known

$$\hat{\Theta} = \arg \max_{\Theta} \sum_{n=1}^{N} \sum_{k=1}^{K} y_{nk} \left[ \log \pi_k + \log \mathcal{N}(x_n | \mu_k, \Sigma_k) \right]$$
MLE When Labels Go Missing..

- Recall the MLE problem for $\Theta$ when the labels are known

$$\hat{\Theta} = \arg\max_{\Theta} \sum_{n=1}^{N} \sum_{k=1}^{K} y_{nk} [\log \pi_k + \log \mathcal{N}(x_n | \mu_k, \Sigma_k)]$$

- Will estimating $\Theta$ via MLE be as easy if $y_n$’s are unknown? We only have $X = \{x_1, \ldots, x_N\}$
MLE When Labels Go Missing..

- Recall the MLE problem for $\Theta$ when the labels are known

$$\hat{\Theta} = \arg \max_{\Theta} \sum_{n=1}^{N} \sum_{k=1}^{K} y_{nk} \left[ \log \pi_k + \log \mathcal{N}(x_n | \mu_k, \Sigma_k) \right]$$

- Will estimating $\Theta$ via MLE be as easy if $y_n$’s are unknown? We only have $X = \{x_1, \ldots, x_N\}$

- The MLE problem for $\Theta = \{\pi_k, \mu_k, \Sigma_k\}_{k=1}^{K}$ in this case would be (assuming i.i.d. data)

$$\hat{\Theta} = \arg \max_{\Theta} \log p(X|\Theta) = \arg \max_{\Theta} \log \prod_{n=1}^{N} p(x_n|\Theta)$$
MLE When Labels Go Missing..

- Recall the MLE problem for $\Theta$ when the labels are known

$$\hat{\Theta} = \arg \max_{\Theta} \sum_{n=1}^{N} \sum_{k=1}^{K} y_{nk} [\log \pi_k + \log \mathcal{N}(x_n|\mu_k, \Sigma_k)]$$

- Will estimating $\Theta$ via MLE be as easy if $y_n$'s are unknown? We only have $X = \{x_1, \ldots, x_N\}$

- The MLE problem for $\Theta = \{\pi_k, \mu_k, \Sigma_k\}_{k=1}^{K}$ in this case would be (assuming i.i.d. data)

$$\hat{\Theta} = \arg \max_{\Theta} \log p(X|\Theta) = \arg \max_{\Theta} \log \prod_{n=1}^{N} p(x_n|\Theta) = \arg \max_{\Theta} \sum_{n=1}^{N} \log p(x_n|\Theta)$$
MLE When Labels Go Missing..

Recall the MLE problem for $\Theta$ when the labels are known

$$\hat{\Theta} = \arg \max_{\Theta} \sum_{n=1}^{N} \sum_{k=1}^{K} y_{nk} \left[ \log \pi_k + \log \mathcal{N}(x_n | \mu_k, \Sigma_k) \right]$$

Will estimating $\Theta$ via MLE be as easy if $y_n$'s are unknown? We only have $X = \{x_1, \ldots, x_N\}$

The MLE problem for $\Theta = \{\pi_k, \mu_k, \Sigma_k\}_{k=1}^{K}$ in this case would be (assuming i.i.d. data)

$$\hat{\Theta} = \arg \max_{\Theta} \log p(X | \Theta) = \arg \max_{\Theta} \log \prod_{n=1}^{N} p(x_n | \Theta) = \arg \max_{\Theta} \sum_{n=1}^{N} \log p(x_n | \Theta)$$

Computing each likelihood $p(x_n | \Theta)$ in this case requires summing over all possible values of $y_n$

$$p(x_n | \Theta) = \sum_{k=1}^{K} p(x_n, y_n = k | \Theta)$$
MLE When Labels Go Missing..

- Recall the MLE problem for $\Theta$ when the labels are known

$$\hat{\Theta} = \arg \max_{\Theta} \sum_{n=1}^{N} \sum_{k=1}^{K} y_{nk} \left[ \log \pi_k + \log \mathcal{N}(x_n | \mu_k, \Sigma_k) \right]$$

- Will estimating $\Theta$ via MLE be as easy if $y_n$’s are unknown? We only have $X = \{x_1, \ldots, x_N\}$

- The MLE problem for $\Theta = \{\pi_k, \mu_k, \Sigma_k\}_{k=1}^{K}$ in this case would be (assuming i.i.d. data)

$$\hat{\Theta} = \arg \max_{\Theta} \log p(X | \Theta) = \arg \max_{\Theta} \log \prod_{n=1}^{N} p(x_n | \Theta) = \arg \max_{\Theta} \sum_{n=1}^{N} \log p(x_n | \Theta)$$

- Computing each likelihood $p(x_n | \Theta)$ in this case requires summing over all possible values of $y_n$

$$p(x_n | \Theta) = \sum_{k=1}^{K} p(x_n, y_n = k | \Theta) = \sum_{k=1}^{K} p(y_n = k | \Theta) p(x_n | y_n = k, \Theta)$$
MLE When Labels Go Missing..

- Recall the MLE problem for $\Theta$ when the labels are known

$$\hat{\Theta} = \arg \max_{\Theta} \sum_{n=1}^{N} \sum_{k=1}^{K} y_{nk} [\log \pi_k + \log \mathcal{N}(x_n | \mu_k, \Sigma_k)]$$

- Will estimating $\Theta$ via MLE be as easy if $y_n$'s are unknown? We only have $X = \{x_1, \ldots, x_N\}$

- The MLE problem for $\Theta = \{\pi_k, \mu_k, \Sigma_k\}_{k=1}^{K}$ in this case would be (assuming i.i.d. data)

$$\hat{\Theta} = \arg \max_{\Theta} \log p(X | \Theta) = \arg \max_{\Theta} \log \prod_{n=1}^{N} p(x_n | \Theta) = \arg \max_{\Theta} \sum_{n=1}^{N} \log p(x_n | \Theta)$$

- Computing each likelihood $p(x_n | \Theta)$ in this case requires summing over all possible values of $y_n$

$$p(x_n | \Theta) = \sum_{k=1}^{K} p(x_n, y_n = k | \Theta) = \sum_{k=1}^{K} p(y_n = k | \Theta)p(x_n | y_n = k, \Theta) = \sum_{k=1}^{K} \pi_k \mathcal{N}(x_n | \mu_k, \Sigma_k)$$
MLE When Labels Go Missing..

- Recall the MLE problem for $\Theta$ when the labels are known

\[
\hat{\Theta} = \arg \max_{\Theta} \sum_{n=1}^{N} \sum_{k=1}^{K} y_{nk} \left[ \log \pi_k + \log \mathcal{N}(x_n | \mu_k, \Sigma_k) \right]
\]

- Will estimating $\Theta$ via MLE be as easy if $y_n$’s are unknown? We only have $X = \{x_1, \ldots, x_N\}$

- The MLE problem for $\Theta = \{\pi_k, \mu_k, \Sigma_k\}_{k=1}^{K}$ in this case would be (assuming i.i.d. data)

\[
\hat{\Theta} = \arg \max_{\Theta} \log p(X|\Theta) = \arg \max_{\Theta} \log \prod_{n=1}^{N} p(x_n|\Theta) = \arg \max_{\Theta} \sum_{n=1}^{N} \log p(x_n|\Theta)
\]

- Computing each likelihood $p(x_n|\Theta)$ in this case requires summing over all possible values of $y_n$

\[
p(x_n|\Theta) = \sum_{k=1}^{K} p(x_n, y_n = k|\Theta) = \sum_{k=1}^{K} p(y_n = k|\Theta) p(x_n|y_n = k, \Theta) = \sum_{k=1}^{K} \pi_k \mathcal{N}(x_n | \mu_k, \Sigma_k)
\]

- The MLE problem for $\Theta$ when the labels are unknown

\[
\hat{\Theta} = \arg \max_{\Theta} \sum_{n=1}^{N} \sum_{k=1}^{K} \log \pi_k \mathcal{N}(x_n | \mu_k, \Sigma_k)
\]
MLE When Labels Go Missing...

So we saw that the MLE problem for $\Theta$ when the labels are unknown

$$\hat{\Theta} = \arg \max_{\Theta} \sum_{n=1}^{N} \log \sum_{k=1}^{K} \pi_k \mathcal{N}(x_n | \mu_k, \Sigma_k)$$

Solving this would enable us to learn a Gaussian Mixture Model (GMM)

Note: The Gaussian can be replaced by other distributions too (e.g., Poisson mixture model)

A small issue now: Log can’t go inside the summation. Expressions won’t be simple anymore

Note: Can still take (partial) derivatives and do GD/SGD etc. but these are iterative methods

Recall that we didn’t need GD/SGD etc when doing MLE with fully observed $y_n$’s

One workaround: Can try doing alternating optimization
MLE When Labels Go Missing..

- So we saw that the MLE problem for $\Theta$ when the labels are unknown

$$\hat{\Theta} = \arg \max_{\Theta} \sum_{n=1}^{N} \log \sum_{k=1}^{K} \pi_k \mathcal{N}(x_n | \mu_k, \Sigma_k)$$

- Solving this would enable us to learn a Gaussian Mixture Model (GMM)
MLE When Labels Go Missing..

- So we saw that the MLE problem for $\Theta$ when the labels are unknown

$$\hat{\Theta} = \arg \max_{\Theta} \sum_{n=1}^{N} \log \sum_{k=1}^{K} \pi_k N(x_n | \mu_k, \Sigma_k)$$

- Solving this would enable us to learn a Gaussian Mixture Model (GMM)

- Note: The Gaussian can be replaced by other distributions too (e.g., Poisson mixture model)

Note: Log can't go inside the summation. Expressions won't be simple anymore

One workaround: Can try doing alternating optimization

Intro to Machine Learning (CS771A)  Parameter Estimation in Latent Variable Models 13
MLE When Labels Go Missing..

- So we saw that the MLE problem for $\Theta$ when the labels are unknown

$$\hat{\Theta} = \arg \max_{\Theta} \sum_{n=1}^{N} \log \sum_{k=1}^{K} \pi_k \mathcal{N}(x_n | \mu_k, \Sigma_k)$$

- Solving this would enable us to learn a Gaussian Mixture Model (GMM)

- Note: The Gaussian can be replaced by other distributions too (e.g., Poisson mixture model)

- A small issue now: Log can’t go inside the summation. Expressions won’t be simple anymore
MLE When Labels Go Missing..

So we saw that the MLE problem for \( \Theta \) when the labels are unknown

\[
\hat{\Theta} = \arg \max_{\Theta} \sum_{n=1}^{N} \log \sum_{k=1}^{K} \pi_k N(x_n | \mu_k, \Sigma_k)
\]

Solving this would enable us to learn a Gaussian Mixture Model (GMM)

Note: The Gaussian can be replaced by other distributions too (e.g., Poisson mixture model)

A small issue now: Log can’t go inside the summation. Expressions won’t be simple anymore

Note: Can still take (partial) derivatives and do GG/SGD etc. but these are iterative methods
So we saw that the MLE problem for $\Theta$ when the labels are unknown

$$\hat{\Theta} = \arg \max_{\Theta} \sum_{n=1}^{N} \log \sum_{k=1}^{K} \pi_k \mathcal{N}(x_n | \mu_k, \Sigma_k)$$

Solving this would enable us to learn a Gaussian Mixture Model (GMM)

Note: The Gaussian can be replaced by other distributions too (e.g., Poisson mixture model)

A small issue now: Log can’t go inside the summation. Expressions won’t be simple anymore

Note: Can still take (partial) derivatives and do GG/SGD etc. but these are iterative methods

Recall that we didn’t need GD/SGD etc when doing MLE with fully observed $y_n$’s
MLE When Labels Go Missing..

- So we saw that the MLE problem for $\Theta$ when the labels are unknown

$$\hat{\Theta} = \arg \max_{\Theta} \sum_{n=1}^{N} \log \sum_{k=1}^{K} \pi_k \mathcal{N}(x_n | \mu_k, \Sigma_k)$$

- Solving this would enable us to learn a Gaussian Mixture Model (GMM)

- Note: The Gaussian can be replaced by other distributions too (e.g., Poisson mixture model)

- A small issue now: Log can’t go inside the summation. Expressions won’t be simple anymore

- Note: Can still take (partial) derivatives and do GG/SGD etc. but these are iterative methods
  
  - Recall that we didn’t need GD/SGD etc when doing MLE with fully observed $y_n$’s

- One workaround: Can try doing alternating optimization
MLE for Gaussian Mixture Model using ALT-OPT

- Based on the fact that MLE is simple when labels are known

1. Initialize $\Theta$ as $\hat{\Theta}$
2. For $n = 1, \ldots, N$, find the best $z_n$
   
   $\hat{z}_n = \arg \max_{k \in \{1, \ldots, K\}} p(x_n, z_n = k | \hat{\Theta})$

3. Given $\hat{Z} = \{\hat{z}_1, \ldots, \hat{z}_N\}$, re-estimate $\Theta$ using MLE
   
   $\hat{\Theta} = \arg \max_{\Theta} \frac{1}{N} \sum_{n=1}^{N} \sum_{k=1}^{K} \hat{z}_{nk} \left[ \log \pi_k + \log N(x_n, \mu_k, \Sigma_k) \right]$

4. Go to step 2 if not yet converged
MLE for Gaussian Mixture Model using ALT-OPT

- Based on the fact that MLE is simple when labels are known
- **Notation change:** We will now use $z_n$ instead of $y_n$ and $z_{nk}$ instead of $y_{nk}$

1. Initialize $\Theta$ as $\hat{\Theta}$
2. For $n = 1, \ldots, N$, find the best $z_n$: 
   
   $\hat{z}_n = \arg \max_{k \in \{1, \ldots, K\}} p(z_n = k | x_n, \hat{\Theta})$

3. Given $\hat{Z} = \{\hat{z}_1, \ldots, \hat{z}_N\}$, re-estimate $\Theta$ using MLE: 
   
   $\hat{\Theta} = \arg \max_{\Theta} \frac{1}{N} \sum_{n=1}^{N} \sum_{k=1}^{K} \hat{z}_{nk} \left[ \log \pi_k + \log N(x_n | \mu_k, \Sigma_k) \right]$

4. Go to step 2 if not yet converged
MLE for Gaussian Mixture Model using ALT-OPT

- Based on the fact that MLE is simple when labels are known
- **Notation change:** We will now use $z_n$ instead of $y_n$ and $z_{nk}$ instead of $y_{nk}$
MLE for Gaussian Mixture Model using ALT-OPT

- Based on the fact that MLE is simple when labels are known
- **Notation change:** We will now use $z_n$ instead of $y_n$ and $z_{nk}$ instead of $y_{nk}$

MLE for Gaussian Mixture Model using ALT-OPT

1. Initialize $\Theta$ as $\hat{\Theta}$

2. For $n = 1, \ldots, N$, find the best $z_n$:
   
   $\hat{z}_n = \arg \max_{k \in \{1, \ldots, K\}} p(x_n | z_n = k, \hat{\Theta})$

3. Given $\hat{Z} = \{\hat{z}_1, \ldots, \hat{z}_N\}$, re-estimate $\Theta$ using MLE:
   
   $\hat{\Theta} = \arg \max_{\Theta} \frac{1}{N} \sum_{n=1}^{N} \sum_{k=1}^{K} \hat{z}_{nk} \left[ \log \pi_k + \log \mathcal{N}(x_n | \mu_k, \Sigma_k) \right]$

4. Go to step 2 if not yet converged
MLE for Gaussian Mixture Model using ALT-OPT

- Based on the fact that MLE is simple when labels are known
- **Notation change:** We will now use \( z_n \) instead of \( y_n \) and \( z_{nk} \) instead of \( y_{nk} \)

MLE for Gaussian Mixture Model using ALT-OPT

1. Initialize \( \Theta \) as \( \hat{\Theta} \)
2. For \( n = 1, \ldots, N \), find the best \( z_n \)
   \[
   \hat{z}_n = \arg \max_{k \in \{1, \ldots, K\}} p(x_n, z_n = k | \hat{\Theta})
   \]
3. Given \( \hat{Z} = \{\hat{z}_1, \ldots, \hat{z}_N\} \), re-estimate \( \Theta \) using MLE
   \[
   \hat{\Theta} = \arg \max_{\Theta} \sum_{n=1}^{N} \sum_{k=1}^{K} \hat{z}_{nk} \left[ \log \pi_k + \log N(x_n | \mu_k, \Sigma_k) \right]
   \]
4. Go to step 2 if not yet converged
MLE for Gaussian Mixture Model using ALT-OPT

- Based on the fact that MLE is simple when labels are known
- Notation change: We will now use $z_n$ instead of $y_n$ and $z_{nk}$ instead of $y_{nk}$

MLE for Gaussian Mixture Model using ALT-OPT

1. Initialize $\Theta$ as $\hat{\Theta}$
2. For $n = 1, \ldots, N$, find the best $z_n$

\[
\hat{z}_n = \arg \max_{k \in \{1, \ldots, K\}} p(x_n, z_n = k | \hat{\Theta}) \\
= \arg \max_{k \in \{1, \ldots, K\}} p(z_n = k | x_n, \hat{\Theta})
\]
MLE for Gaussian Mixture Model using ALT-OPT

- Based on the fact that MLE is simple when labels are known

- **Notation change:** We will now use $z_n$ instead of $y_n$ and $z_{nk}$ instead of $y_{nk}$

---

**MLE for Gaussian Mixture Model using ALT-OPT**

1. Initialize $\Theta$ as $\hat{\Theta}$
2. For $n = 1, \ldots, N$, find the best $z_n$
   
   $$
   \hat{z}_n = \arg \max_{k \in \{1, \ldots, K\}} p(x_n, z_n = k|\hat{\Theta})
   $$
   
   $$
   = \arg \max_{k \in \{1, \ldots, K\}} p(z_n = k|x_n, \hat{\Theta})
   $$

3. Given $\hat{Z} = \{\hat{z}_1, \ldots, \hat{z}_N\}$, re-estimate $\Theta$ using MLE

   $$
   \hat{\Theta} = \arg \max_{\Theta} \sum_{n=1}^{N} \sum_{k=1}^{K} \hat{z}_{nk} [\log \pi_k + \log \mathcal{N}(x_n | \mu_k, \Sigma_k)]
   $$
MLE for Gaussian Mixture Model using ALT-OPT

- Based on the fact that MLE is simple when labels are known
- **Notation change:** We will now use $z_n$ instead of $y_n$ and $z_{nk}$ instead of $y_{nk}$

**MLE for Gaussian Mixture Model using ALT-OPT**

1. Initialize $\Theta$ as $\hat{\Theta}$
2. For $n = 1, \ldots, N$, find the best $z_n$

   $$\hat{z}_n = \arg \max_{k \in \{1, \ldots, K\}} p(x_n, z_n = k | \hat{\Theta})$$

   $$= \arg \max_{k \in \{1, \ldots, K\}} p(z_n = k | x_n, \hat{\Theta})$$

3. Given $\hat{Z} = \{\hat{z}_1, \ldots, \hat{z}_N\}$, re-estimate $\Theta$ using MLE

   $$\hat{\Theta} = \arg \max_{\Theta} \sum_{n=1}^{N} \sum_{k=1}^{K} \hat{z}_{nk} [\log \pi_k + \log \mathcal{N}(x_n | \mu_k, \Sigma_k)]$$

4. Go to step 2 if not yet converged
Is ALT-OPT Doing The Correct Thing?

- Our original problem was

\[
\hat{\Theta} = \arg \max_{\Theta} \sum_{n=1}^{N} \log \sum_{k=1}^{K} \pi_k N(x_n | \mu_k, \Sigma_k)
\]

What ALT-OPT did was the following

\[
\hat{\Theta} = \arg \max_{\Theta} \sum_{n=1}^{N} \sum_{k=1}^{K} \hat{\mathbf{z}}_{nk} \left[ \log \pi_k + \log N(x_n | \mu_k, \Sigma_k) \right]
\]

We clearly aren't solving the original problem!

\[
\arg \max_{\Theta} \log p(X | \Theta) \text{ vs } \arg \max_{\Theta} \log p(X, \hat{Z} | \Theta)
\]

Also, we updated \(\hat{\mathbf{z}}_{n}\) as follows

\[
\hat{\mathbf{z}}_{n} = \arg \max_{k \in \{1, \ldots, K\}} p(z_n = k | x_n, \hat{\Theta})
\]

Why choose \(\hat{\mathbf{z}}_{n}\) to be this (makes intuitive sense, but is there a formal justification)?

It turns out (as we will see), this ALT-OPT is an approximation of the Expectation Maximization (EM) algorithm for GMM.
Is ALT-OPT Doing The Correct Thing?

- Our original problem was

\[ \hat{\Theta} = \arg \max_\Theta \sum_{n=1}^N \log \sum_{k=1}^K \pi_k \mathcal{N}(x_n | \mu_k, \Sigma_k) \]

- What ALT-OPT did was the following

\[ \hat{\Theta} = \arg \max_\Theta \sum_{n=1}^N \sum_{k=1}^K \hat{z}_{nk} [\log \pi_k + \log \mathcal{N}(x_n | \mu_k, \Sigma_k)] \]
Is ALT-OPT Doing The Correct Thing?

- Our original problem was
  \[ \hat{\Theta} = \arg \max_{\Theta} \sum_{n=1}^{N} \log \sum_{k=1}^{K} \pi_k N(x_n | \mu_k, \Sigma_k) \]

- What ALT-OPT did was the following
  \[ \hat{\Theta} = \arg \max_{\Theta} \sum_{n=1}^{N} \sum_{k=1}^{K} \hat{z}_{nk} [\log \pi_k + \log N(x_n | \mu_k, \Sigma_k)] \]

- We clearly aren’t solving the original problem!
  \[ \arg \max_{\Theta} \log p(X|\Theta) \quad \text{vs} \quad \arg \max_{\Theta} \log p(X, \hat{Z}|\Theta) \]
Is ALT-OPT Doing The Correct Thing?

- Our original problem was
  \[ \hat{\Theta} = \text{arg max}_{\Theta} \sum_{n=1}^{N} \log \sum_{k=1}^{K} \pi_{k} \mathcal{N}(x_n | \mu_k, \Sigma_k) \]

- What ALT-OPT did was the following
  \[ \hat{\Theta} = \text{arg max}_{\Theta} \sum_{n=1}^{N} \sum_{k=1}^{K} \hat{z}_{nk} \left[ \log \pi_k + \log \mathcal{N}(x_n | \mu_k, \Sigma_k) \right] \]

- We clearly aren’t solving the original problem!
  \[ \text{arg max}_{\Theta} \log p(X | \Theta) \quad \text{vs} \quad \text{arg max}_{\Theta} \log p(X, \hat{Z} | \Theta) \]

- Also, we updated \( \hat{z}_n \) as follows
  \[ \hat{z}_n = \text{arg max}_{k \in \{1, \ldots, K\}} p(z_n = k | x_n, \hat{\Theta}) \]
Is ALT-OPT Doing The Correct Thing?

Our original problem was
\[
\hat{\Theta} = \arg \max_{\Theta} \sum_{n=1}^{N} \log \sum_{k=1}^{K} \pi_k \mathcal{N}(x_n | \mu_k, \Sigma_k)
\]

What ALT-OPT did was the following
\[
\hat{\Theta} = \arg \max_{\Theta} \sum_{n=1}^{N} \sum_{k=1}^{K} \hat{z}_{nk} [\log \pi_k + \log \mathcal{N}(x_n | \mu_k, \Sigma_k)]
\]

We clearly aren’t solving the original problem!
\[
\arg \max_{\Theta} \log p(X | \Theta) \text{ vs } \arg \max_{\Theta} \log p(X, \hat{Z} | \Theta)
\]

Also, we updated \( \hat{z}_n \) as follows
\[
\hat{z}_n = \arg \max_{k \in \{1, \ldots, K\}} p(z_n = k | x_n, \hat{\Theta})
\]

Why choose \( \hat{z}_n \) to be this (makes intuitive sense, but is there a formal justification)?
Is ALT-OPT Doing The Correct Thing?

- Our original problem was
  \[ \hat{\Theta} = \arg \max_{\Theta} \sum_{n=1}^{N} \log \sum_{k=1}^{K} \pi_k \mathcal{N}(x_n | \mu_k, \Sigma_k) \]

- What ALT-OPT did was the following
  \[ \hat{\Theta} = \arg \max_{\Theta} \sum_{n=1}^{N} \sum_{k=1}^{K} \hat{z}_{nk} [\log \pi_k + \log \mathcal{N}(x_n | \mu_k, \Sigma_k)] \]

- We clearly aren't solving the original problem!
  \[ \arg \max_{\Theta} \log p(X | \Theta) \quad vs \quad \arg \max_{\Theta} \log p(X, \hat{Z} | \Theta) \]

- Also, we updated \( \hat{z}_n \) as follows
  \[ \hat{z}_n = \arg \max_{k \in \{1, \ldots, K\}} p(z_n = k | x_n, \hat{\Theta}) \]

- Why choose \( \hat{z}_n \) to be this (makes intuitive sense, but is there a formal justification)?

- It turns out (as we will see), this ALT-OPT is an approximation of the Expectation Maximization (EM) algorithm for GMM.
Expectation Maximization (EM)

- A very popular algorithm for parameter estimation in latent variable models
Expectation Maximization (EM)

- A very popular algorithm for parameter estimation in latent variable models
- The EM algorithm is based on the following identity (exercise: verify)

\[
\log p(X|\Theta) = \mathbb{E}_{q(Z)} \left[ \log \frac{p(X, Z|\Theta)}{q(Z)} \right] + \text{KL}[q(Z)||p(Z|X, \Theta)]
\]
Expectation Maximization (EM)

- A very popular algorithm for parameter estimation in latent variable models
- The EM algorithm is based on the following identity (exercise: verify)

\[
\log p(X|\Theta) = \mathbb{E}_{q(Z)} \left[ \log \frac{p(X, Z|\Theta)}{q(Z)} \right] + \text{KL}[q(Z)||p(Z|X, \Theta)]
\]

- The above is true for any choice of the distribution \( q(Z) \)
Expectation Maximization (EM)

- A very popular algorithm for parameter estimation in latent variable models
- The EM algorithm is based on the following identity (exercise: verify)

$$\log p(X|\Theta) = \mathbb{E}_{q(Z)} \left[ \log \frac{p(X,Z|\Theta)}{q(Z)} \right] + \text{KL}[q(Z)||p(Z|X,\Theta)]$$

- The above is true for any choice of the distribution $q(Z)$
- Since KL divergence is non-negative, we must have

$$\log p(X|\Theta) \geq \mathbb{E}_{q(Z)} \left[ \log \frac{p(X,Z|\Theta)}{q(Z)} \right]$$
Expectation Maximization (EM)

- A very popular algorithm for parameter estimation in latent variable models
- The EM algorithm is based on the following identity (exercise: verify)

\[
\log p(X|\Theta) = \mathbb{E}_{q(Z)} \left[ \log \frac{p(X, Z|\Theta)}{q(Z)} \right] + \text{KL}[q(Z)||p(Z|X, \Theta)]
\]

- The above is true for any choice of the distribution \(q(Z)\)
- Since KL divergence is non-negative, we must have

\[
\log p(X|\Theta) \geq \mathbb{E}_{q(Z)} \left[ \log \frac{p(X, Z|\Theta)}{q(Z)} \right]
\]

- So \(\mathcal{L}(\Theta) = \mathbb{E}_{q(Z)} \left[ \log \frac{p(X, Z|\Theta)}{q(Z)} \right]\) is a lower bound on what we want to maximize, i.e., \(\log p(X|\Theta)\)
Expectation Maximization (EM)

- A very popular algorithm for parameter estimation in latent variable models
- The EM algorithm is based on the following identity (exercise: verify)

\[
\log p(X|\Theta) = \mathbb{E}_{q(Z)} \left[ \log \frac{p(X,Z|\Theta)}{q(Z)} \right] + \text{KL}[q(Z)||p(Z|X,\Theta)]
\]

- The above is true for any choice of the distribution \(q(Z)\)
- Since KL divergence is non-negative, we must have

\[
\log p(X|\Theta) \geq \mathbb{E}_{q(Z)} \left[ \log \frac{p(X,Z|\Theta)}{q(Z)} \right]
\]

- So \(\mathcal{L}(\Theta) = \mathbb{E}_{q(Z)} \left[ \log \frac{p(X,Z|\Theta)}{q(Z)} \right]\) is a lower bound on what we want to maximize, i.e., \(\log p(X|\Theta)\)
- Also, if we choose \(q(Z) = p(Z|X,\Theta)\), then \(\log p(X|\Theta) = \mathbb{E}_{q(Z)} \left[ \log \frac{p(X,Z|\Theta)}{q(Z)} \right]\)
The EM algorithm for GMM does the following

\[ \hat{\Theta}_{new} = \arg \max_{\Theta} \sum_{n=1}^{N} \sum_{k=1}^{K} \mathbb{E}[z_{nk}] \left[ \log \pi_k + \log \mathcal{N}(x_n | \mu_k, \Sigma_k) \right] \]

.. which is nothing but maximizing \( \mathbb{E}_{q(Z)}[\log p(X, Z|\Theta)] \) with \( q(Z) = p(Z|X, \hat{\Theta}_{old}) \)

Here \( \mathbb{E}[z_{nk}] \) is the expectation of \( z_{nk} \) w.r.t. posterior \( p(z_n|x_n) \) and is given by

\[
\mathbb{E}[z_{nk}] = 0 \times p(z_{nk} = 0|x_n) + 1 \times p(z_{nk} = 1|x_n)
\]
The EM algorithm for GMM does the following:

\[
\hat{\Theta}_{new} = \arg \max_{\Theta} \sum_{n=1}^{N} \sum_{k=1}^{K} \mathbb{E}[z_{nk}] [\log \pi_k + \log \mathcal{N}(x_n | \mu_k, \Sigma_k)]
\]

.. which is nothing but maximizing \( \mathbb{E}_q(Z) [\log p(X, Z|\Theta)] \) with \( q(Z) = p(Z|X, \hat{\Theta}_{old}) \)

Here \( \mathbb{E}[z_{nk}] \) is the expectation of \( z_{nk} \) w.r.t. posterior \( p(z_n|x_n) \) and is given by:

\[
\mathbb{E}[z_{nk}] = 0 \times p(z_{nk} = 0|x_n) + 1 \times p(z_{nk} = 1|x_n) = p(z_{nk} = 1|x_n)
\]
The EM algorithm for GMM does the following:

\[
\hat{\Theta}_{new} = \arg \max_{\Theta} \sum_{n=1}^{N} \sum_{k=1}^{K} E[z_{nk}][\log \pi_k + \log \mathcal{N}(x_n|\mu_k, \Sigma_k)]
\]

.. which is nothing but maximizing \( E_q(Z)[\log p(X, Z|\Theta)] \) with \( q(Z) = p(Z|X, \hat{\Theta}_{old}) \)

Here \( E[z_{nk}] \) is the expectation of \( z_{nk} \) w.r.t. posterior \( p(z_n|x_n) \) and is given by

\[
E[z_{nk}] = 0 \times p(z_{nk} = 0|x_n) + 1 \times p(z_{nk} = 1|x_n)
\]

\[
= p(z_{nk} = 1|x_n)
\]

\[
\propto p(z_{nk} = 1)p(x_n|z_{nk} = 1) \quad \text{(from Bayes Rule)}
\]
EM for GMM

- The EM algorithm for GMM does the following

\[
\hat{\Theta}_{new} = \arg \max_{\Theta} \sum_{n=1}^{N} \sum_{k=1}^{K} \mathbb{E}[z_{nk}] \left[ \log \pi_k + \log \mathcal{N}(x_n|\mu_k, \Sigma_k) \right]
\]

.. which is nothing but maximizing \( \mathbb{E}_{q(Z)}[\log p(X, Z|\Theta)] \) with \( q(Z) = p(Z|X, \hat{\Theta}_{old}) \)

- Here \( \mathbb{E}[z_{nk}] \) is the expectation of \( z_{nk} \) w.r.t. posterior \( p(z_n|x_n) \) and is given by

\[
\mathbb{E}[z_{nk}] = 0 \times p(z_{nk} = 0|x_n) + 1 \times p(z_{nk} = 1|x_n) \\
= p(z_{nk} = 1|x_n) \propto p(z_{nk} = 1)p(x_n|z_{nk} = 1) \quad (\text{from Bayes Rule})
\]

Thus \( \mathbb{E}[z_{nk}] \propto \pi_k \mathcal{N}(x_n|\mu_k, \Sigma_k) \) (Posterior prob. that \( x_n \) is generated by \( k \)-th Gaussian)
EM for GMM

- The EM algorithm for GMM does the following

\[ \hat{\Theta}_{\text{new}} = \arg \max_{\Theta} \sum_{n=1}^{N} \sum_{k=1}^{K} \mathbb{E}[z_{nk}] \left[ \log \pi_k + \log \mathcal{N}(x_n | \mu_k, \Sigma_k) \right] \]

.. which is nothing but maximizing \( \mathbb{E}_q(Z) [\log p(X, Z | \Theta)] \) with \( q(Z) = p(Z|X, \hat{\Theta}_{\text{old}}) \)

- Here \( \mathbb{E}[z_{nk}] \) is the expectation of \( z_{nk} \) w.r.t. posterior \( p(z_n | x_n) \) and is given by

\[
\mathbb{E}[z_{nk}] = 0 \times p(z_{nk} = 0 | x_n) + 1 \times p(z_{nk} = 1 | x_n) \\
= p(z_{nk} = 1 | x_n) \\
\propto p(z_{nk} = 1) p(x_n | z_{nk} = 1) \quad \text{(from Bayes Rule)}
\]

Thus \( \mathbb{E}[z_{nk}] \propto \pi_k \mathcal{N}(x_n | \mu_k, \Sigma_k) \) (Posterior prob. that \( x_n \) is generated by \( k \)-th Gaussian)

- Next class: Details of EM for GMM, special cases, and the general EM algorithm and its properties