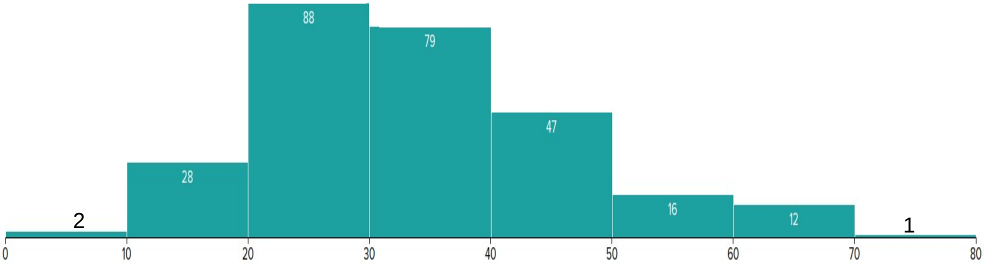
Parameter Estimation in Latent Variable Models

Piyush Rai

Introduction to Machine Learning (CS771A)

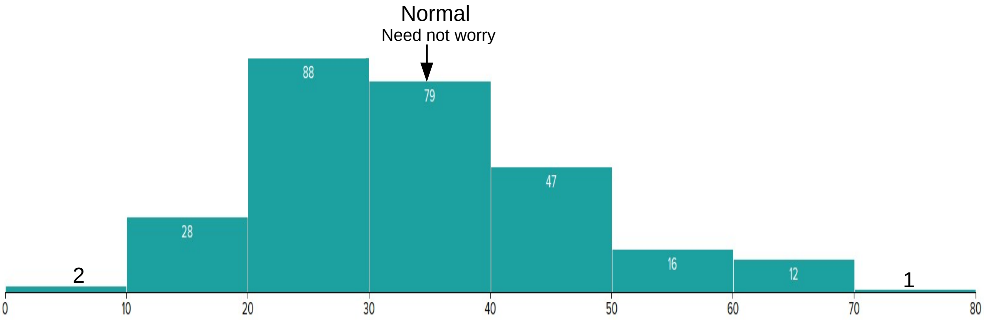
September 25, 2018

Some Mid-Sem Statistics



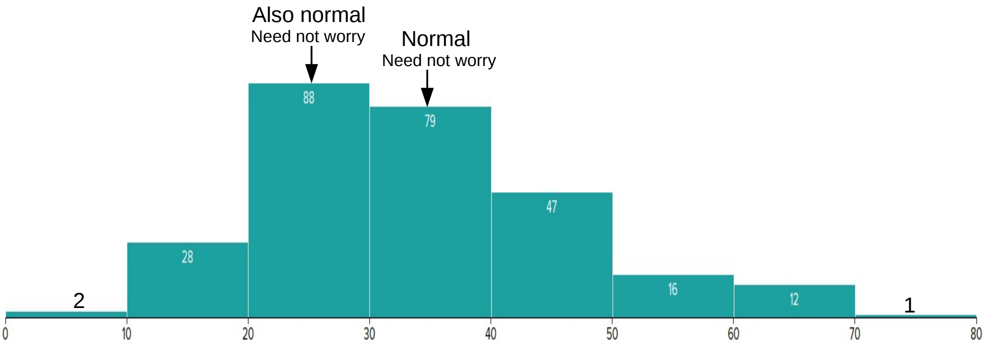
MINIMUM	MEDIAN	MAXIMUM	MEAN	STD DEV
5.5	32.0	73.5	33.46	12.65

Some Mid-Sem Statistics



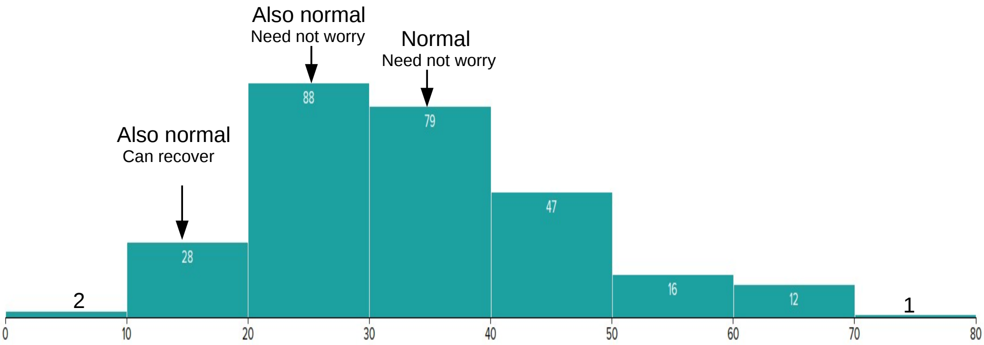
MINIMUM	MEDIAN	MAXIMUM	MEAN	STD DEV
5.5	32.0	73.5	33.46	12.65

Some Mid-Sem Statistics



MINIMUM	MEDIAN	MAXIMUM	MEAN	STD DEV
5.5	32.0	73.5	33.46	12.65

Some Mid-Sem Statistics

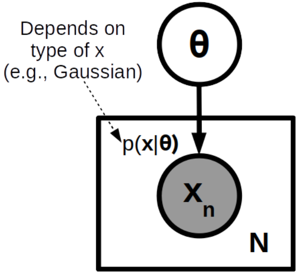


MINIMUM	MEDIAN	MAXIMUM	MEAN	STD DEV
5.5	32.0	73.5	33.46	12.65

Latent Variable Models

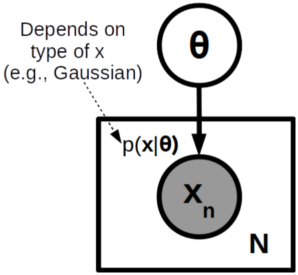
A Simple Generative Model

- All observations $\{\boldsymbol{x}_1, \ldots, \boldsymbol{x}_N\}$ generated from a distribution $p(\boldsymbol{x}|\theta)$



A Simple Generative Model

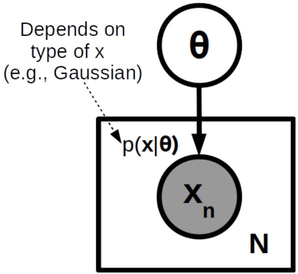
- All observations $\{\boldsymbol{x}_1, \ldots, \boldsymbol{x}_N\}$ generated from a distribution $p(\boldsymbol{x}|\theta)$



- Unknowns: Parameters θ of the assumed data distribution $p(\boldsymbol{x}|\theta)$

A Simple Generative Model

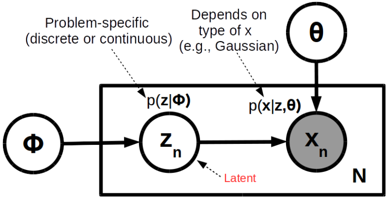
- All observations $\{\boldsymbol{x}_1, \ldots, \boldsymbol{x}_N\}$ generated from a distribution $p(\boldsymbol{x}|\theta)$



- Unknowns: Parameters θ of the assumed data distribution $p(\boldsymbol{x}|\theta)$
- Many ways to estimate the parameters (MLE, MAP, or Bayesian inference)

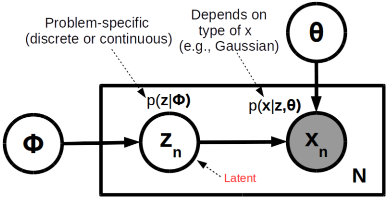
Generative Model with Latent Variables

- Assume each observation $\boldsymbol{x}_n$ to be associated with a latent variable $\boldsymbol{z}_n$



Generative Model with Latent Variables

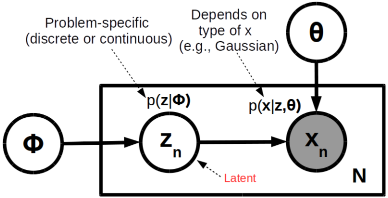
- Assume each observation $\boldsymbol{x}_n$ to be associated with a latent variable $\boldsymbol{z}_n$



- In this "latent variable model" of data, data $\boldsymbol{x}$ also depends some latent variable(s) $\boldsymbol{z}$

Generative Model with Latent Variables

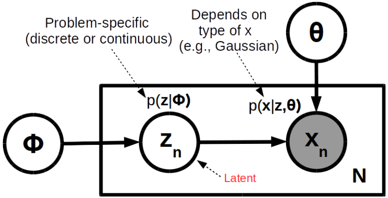
- Assume each observation $\boldsymbol{x}_n$ to be associated with a latent variable $\boldsymbol{z}_n$



- In this "latent variable model" of data, data $\boldsymbol{x}$ also depends some latent variable(s) $\boldsymbol{z}$
- $\boldsymbol{z}_n$ is akin to a latent representation or "encoding" of $\boldsymbol{x}_n$; controls what data "looks like".

Generative Model with Latent Variables

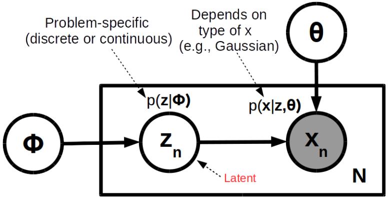
- Assume each observation $\boldsymbol{x}_n$ to be associated with a latent variable $\boldsymbol{z}_n$



- In this "latent variable model" of data, data $\boldsymbol{x}$ also depends some latent variable(s) $\boldsymbol{z}$
- $\boldsymbol{z}_n$ is akin to a latent representation or "encoding" of $\boldsymbol{x}_n$; controls what data "looks like".E.g,
  - $\boldsymbol{z}_n \in \{1, \dots, K\}$ denotes the cluster $\boldsymbol{x}_n$ belongs to

Generative Model with Latent Variables

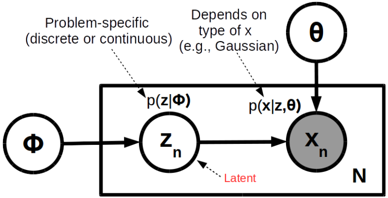
- Assume each observation $\boldsymbol{x}_n$ to be associated with a latent variable $\boldsymbol{z}_n$



- In this "latent variable model" of data, data $\boldsymbol{x}$ also depends some latent variable(s) $\boldsymbol{z}$
- $\boldsymbol{z}_n$ is akin to a latent representation or "encoding" of $\boldsymbol{x}_n$; controls what data "looks like". E.g,
  - $\boldsymbol{z}_n \in \{1, \ldots, K\}$ denotes the cluster $\boldsymbol{x}_n$ belongs to
  - $\boldsymbol{z}_n \in \mathbb{R}^K$ denotes a low-dimensional latent representation or latent "code" for $\boldsymbol{x}_n$

Generative Model with Latent Variables

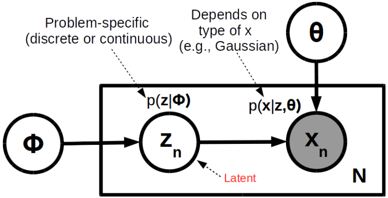
- Assume each observation $\boldsymbol{x}_n$ to be associated with a latent variable $\boldsymbol{z}_n$



- In this "latent variable model" of data, data $\boldsymbol{x}$ also depends some latent variable(s) $\boldsymbol{z}$
- $\boldsymbol{z}_n$ is akin to a latent representation or "encoding" of $\boldsymbol{x}_n$; controls what data "looks like".E.g,
  - $\boldsymbol{z}_n \in \{1, \ldots, K\}$ denotes the cluster $\boldsymbol{x}_n$ belongs to
  - $\boldsymbol{z}_n \in \mathbb{R}^K$ denotes a low-dimensional latent representation or latent "code" for $\boldsymbol{x}_n$
- Unknowns: $\{\boldsymbol{z}_1, \ldots, \boldsymbol{z}_N\}$, and (θ, ϕ).

Generative Model with Latent Variables

- Assume each observation $\boldsymbol{x}_n$ to be associated with a latent variable $\boldsymbol{z}_n$



- In this "latent variable model" of data, data $\boldsymbol{x}$ also depends some latent variable(s) $\boldsymbol{z}$
- $\boldsymbol{z}_n$ is akin to a latent representation or "encoding" of $\boldsymbol{x}_n$; controls what data "looks like".E.g,
  - $\boldsymbol{z}_n \in \{1, \ldots, K\}$ denotes the cluster $\boldsymbol{x}_n$ belongs to
  - $\boldsymbol{z}_n \in \mathbb{R}^K$ denotes a low-dimensional latent representation or latent "code" for $\boldsymbol{x}_n$
- Unknowns: $\{\boldsymbol{z}_1, \ldots, \boldsymbol{z}_N\}$, and (θ, ϕ). $\boldsymbol{z}_n$'s called "local" variables; (θ, ϕ) called "global" variables

Brief Detour/Recap:
Gaussian Parameter Estimation

MLE for Multivariate Gaussian

- Multivariate Gaussian in D dimensions

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

MLE for Multivariate Gaussian

- Multivariate Gaussian in D dimensions

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

- Goal: Given N i.i.d. observations $\{\boldsymbol{x}_n\}_{n=1}^N$ from this Gaussian, estimate parameters μ and Σ

MLE for Multivariate Gaussian

- Multivariate Gaussian in D dimensions

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

- Goal: Given N i.i.d. observations $\{\boldsymbol{x}_n\}_{n=1}^N$ from this Gaussian, estimate parameters μ and Σ
- MLE for the $D \times 1$ mean $\mu \in \mathbb{R}^D$ and $D \times D$ p.s.d. covariance matrix Σ

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

MLE for Multivariate Gaussian

- Multivariate Gaussian in D dimensions

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

- Goal: Given N i.i.d. observations $\{\boldsymbol{x}_n\}_{n=1}^N$ from this Gaussian, estimate parameters μ and Σ
- MLE for the $D \times 1$ mean $\mu \in \mathbb{R}^D$ and $D \times D$ p.s.d. covariance matrix Σ

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

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

MLE for Multivariate Gaussian

- Multivariate Gaussian in D dimensions

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

- Goal: Given N i.i.d. observations $\{\boldsymbol{x}_n\}_{n=1}^N$ from this Gaussian, estimate parameters μ and Σ
- MLE for the $D \times 1$ mean $\mu \in \mathbb{R}^D$ and $D \times D$ p.s.d. covariance matrix Σ

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

- Note: Σ depends on μ, but μ doesn't depend on Σ $\Rightarrow$ 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

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

- Goal: Given N i.i.d. observations $\{\boldsymbol{x}_n\}_{n=1}^N$ from this Gaussian, estimate parameters μ and Σ
- MLE for the $D \times 1$ mean $\mu \in \mathbb{R}^D$ and $D \times D$ p.s.d. covariance matrix Σ

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

- Note: Σ depends on μ, but μ doesn't depend on Σ $\Rightarrow$ 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

Brief Detour: Exponential Family Distributions

- An exponential family distribution is of the form

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

- θ is called the natural parameter of the family

Brief Detour: Exponential Family Distributions

- An exponential family distribution is of the form

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

- θ is called the natural parameter of the family
- $h(x)$, $\phi(x)$, and $A(\theta)$ are known functions. Note: Don't confuse ϕ with kernel mappings!

Brief Detour: Exponential Family Distributions

- An exponential family distribution is of the form

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

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

Brief Detour: Exponential Family Distributions

- An exponential family distribution is of the form

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

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

Brief Detour: Exponential Family Distributions

- An exponential family distribution is of the form

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

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

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

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

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

- θ is called the natural parameter of the family
- $h(x)$, $\phi(x)$, and $A(\theta)$ are known functions. Note: Don't confuse ϕ with kernel mappings!
- $\phi(x)$ is called the **sufficient statistics**: knowing this is sufficient to estimate θ
- 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 μ_k and covariance matrix Σ_k

MLE for Generative Classification with Gaussian Class-conditionals

- Each class k modeled using a Gaussian with mean μ_k and covariance matrix Σ_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 μ_k and covariance matrix Σ_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 $\Theta = \{\pi_k, \mu_k, \Sigma_k\}_{k=1}^K$

MLE for Generative Classification with Gaussian Class-conditionals

- Each class k modeled using a Gaussian with mean μ_k and covariance matrix Σ_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 $\{\boldsymbol{x}_n, y_n\}_{n=1}^N$, MLE for Θ will be

MLE for Generative Classification with Gaussian Class-conditionals

- Each class k modeled using a Gaussian with mean μ_k and covariance matrix Σ_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 $\{\boldsymbol{x}_n, y_n\}_{n=1}^N$, MLE for Θ 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 μ_k and covariance matrix Σ_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 $\{\boldsymbol{x}_n, y_n\}_{n=1}^N$, MLE for Θ 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 μ_k and covariance matrix Σ_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 $\{\boldsymbol{x}_n, y_n\}_{n=1}^N$, MLE for Θ will be

$$
\begin{aligned}
\hat{\pi}_k &= \frac{1}{N}\sum_{n=1}^{N} y_{nk} = \frac{N_k}{N} \\
\hat{\mu}_k &= \frac{1}{N_k}\sum_{n=1}^{N} y_{nk}\boldsymbol{x}_n
\end{aligned}
$$

MLE for Generative Classification with Gaussian Class-conditionals

- Each class k modeled using a Gaussian with mean μ_k and covariance matrix Σ_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 $\{\boldsymbol{x}_n, y_n\}_{n=1}^N$, MLE for Θ will be

$$
\begin{aligned}
\hat{\pi}_k &= \frac{1}{N}\sum_{n=1}^{N} y_{nk} = \frac{N_k}{N} \\
\hat{\mu}_k &= \frac{1}{N_k}\sum_{n=1}^{N} y_{nk}\boldsymbol{x}_n \\
\hat{\Sigma}_k &= \frac{1}{N_k}\sum_{n=1}^{N} y_{nk}(\boldsymbol{x}_n - \hat{\mu}_k)(\boldsymbol{x}_n - \hat{\mu}_k)^\top
\end{aligned}
$$

MLE for Generative Classification with Gaussian Class-conditionals

- Each class k modeled using a Gaussian with mean μ_k and covariance matrix Σ_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 $\{\boldsymbol{x}_n, y_n\}_{n=1}^N$, MLE for Θ will be

$$
\begin{aligned}
\hat{\pi}_k &= \frac{1}{N}\sum_{n=1}^{N} y_{nk} = \frac{N_k}{N} \\
\hat{\mu}_k &= \frac{1}{N_k}\sum_{n=1}^{N} y_{nk}\boldsymbol{x}_n \\
\hat{\Sigma}_k &= \frac{1}{N_k}\sum_{n=1}^{N} y_{nk}(\boldsymbol{x}_n - \hat{\mu}_k)(\boldsymbol{x}_n - \hat{\mu}_k)^\top
\end{aligned}
$$

- 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

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)

$$\hat{\Theta} = \arg\max_{\Theta} p(\mathbf{X}, \boldsymbol{y}|\Theta)$$

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)

$$\hat{\Theta} = \arg\max_{\Theta} p(\mathbf{X}, \boldsymbol{y}|\Theta) \quad = \quad \arg\max_{\Theta} \prod_{n=1}^{N} p(\boldsymbol{x}_n, y_n|\Theta) =$$

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)

$$\hat{\Theta} = \arg\max_{\Theta} p(\mathbf{X}, \boldsymbol{y}|\Theta) \quad = \quad \arg\max_{\Theta} \prod_{n=1}^{N} p(\boldsymbol{x}_n, y_n|\Theta) = \arg\max_{\Theta} \prod_{n=1}^{N} p(y_n|\Theta) p(\boldsymbol{x}_n|y_n, \Theta)$$

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)

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

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

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)

$$\begin{aligned}
\hat{\Theta} = \arg\max_{\Theta} p(\mathbf{X}, \boldsymbol{y}|\Theta) &= \arg\max_{\Theta} \prod_{n=1}^{N} p(\boldsymbol{x}_n, y_n|\Theta) = \arg\max_{\Theta} \prod_{n=1}^{N} p(y_n|\Theta)p(\boldsymbol{x}_n|y_n, \Theta) \\
&= \arg\max_{\Theta} \prod_{n=1}^{N} \prod_{k=1}^{K} [p(y_n = k|\Theta)p(\boldsymbol{x}_n|y_n = k, \Theta)]^{y_{nk}} \\
&= \arg\max_{\Theta} \log \prod_{n=1}^{N} \prod_{k=1}^{K} [p(y_n = k|\Theta)p(\boldsymbol{x}_n|y_n = k, \Theta)]^{y_{nk}}
\end{aligned}$$

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)

$$
\begin{aligned}
\hat{\Theta} = \arg\max_{\Theta} p(\mathbf{X}, \boldsymbol{y}|\Theta) &= \arg\max_{\Theta} \prod_{n=1}^{N} p(\boldsymbol{x}_n, y_n|\Theta) = \arg\max_{\Theta} \prod_{n=1}^{N} p(y_n|\Theta) p(\boldsymbol{x}_n|y_n, \Theta) \\
&= \arg\max_{\Theta} \prod_{n=1}^{N} \prod_{k=1}^{K} [p(y_n = k|\Theta) p(\boldsymbol{x}_n|y_n = k, \Theta)]^{y_{nk}} \\
&= \arg\max_{\Theta} \log \prod_{n=1}^{N} \prod_{k=1}^{K} [p(y_n = k|\Theta) p(\boldsymbol{x}_n|y_n = k, \Theta)]^{y_{nk}} \\
&= \arg\max_{\Theta} \sum_{n=1}^{N} \sum_{k=1}^{K} y_{nk} [\log p(y_n = k|\Theta) + \log p(\boldsymbol{x}_n|y_n = k, \Theta)]
\end{aligned}
$$

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)

$$
\begin{aligned}
\hat{\Theta} = \arg\max_{\Theta} p(\mathbf{X}, \boldsymbol{y}|\Theta) \quad &= \quad \arg\max_{\Theta} \prod_{n=1}^{N} p(\boldsymbol{x}_n, y_n|\Theta) = \arg\max_{\Theta} \prod_{n=1}^{N} p(y_n|\Theta) p(\boldsymbol{x}_n|y_n, \Theta) \\
&= \quad \arg\max_{\Theta} \prod_{n=1}^{N} \prod_{k=1}^{K} [p(y_n = k|\Theta) p(\boldsymbol{x}_n|y_n = k, \Theta)]^{y_{nk}} \\
&= \quad \arg\max_{\Theta} \log \prod_{n=1}^{N} \prod_{k=1}^{K} [p(y_n = k|\Theta) p(\boldsymbol{x}_n|y_n = k, \Theta)]^{y_{nk}} \\
&= \quad \arg\max_{\Theta} \sum_{n=1}^{N} \sum_{k=1}^{K} y_{nk} [\log p(y_n = k|\Theta) + \log p(\boldsymbol{x}_n|y_n = k, \Theta)] \\
&= \quad \arg\max_{\Theta} \sum_{n=1}^{N} \sum_{k=1}^{K} y_{nk} [\log \pi_k + \log \mathcal{N}(\boldsymbol{x}_n|\mu_k, \Sigma_k)]
\end{aligned}
$$

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)

$$
\begin{aligned}
\hat{\Theta} = \arg\max_{\Theta} p(\mathbf{X}, \boldsymbol{y}|\Theta) &= \arg\max_{\Theta} \prod_{n=1}^{N} p(\boldsymbol{x}_n, y_n|\Theta) = \arg\max_{\Theta} \prod_{n=1}^{N} p(y_n|\Theta) p(\boldsymbol{x}_n|y_n, \Theta) \\
&= \arg\max_{\Theta} \prod_{n=1}^{N} \prod_{k=1}^{K} [p(y_n = k|\Theta) p(\boldsymbol{x}_n|y_n = k, \Theta)]^{y_{nk}} \\
&= \arg\max_{\Theta} \log \prod_{n=1}^{N} \prod_{k=1}^{K} [p(y_n = k|\Theta) p(\boldsymbol{x}_n|y_n = k, \Theta)]^{y_{nk}} \\
&= \arg\max_{\Theta} \sum_{n=1}^{N} \sum_{k=1}^{K} y_{nk} [\log p(y_n = k|\Theta) + \log p(\boldsymbol{x}_n|y_n = k, \Theta)] \\
&= \arg\max_{\Theta} \sum_{n=1}^{N} \sum_{k=1}^{K} y_{nk} [\log \pi_k + \log \mathcal{N}(\boldsymbol{x}_n|\mu_k, \Sigma_k)]
\end{aligned}
$$

- Given $(\mathbf{X}, \boldsymbol{y})$, optimizing it w.r.t. π_k, μ_k, Σ_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

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

MLE When Labels Go Missing..

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

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

- 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

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

- 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)?

MLE When Labels Go Missing..

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

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

- 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 Θ now?

MLE When Labels Go Missing..

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

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

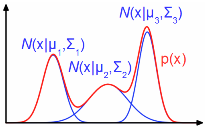
- This problem has a nice separable structure, and a straightforward solution as we saw
- What if we don't know the label y_n (i.e., don't know if y_{nk} is 0 or 1)? How to estimate Θ now?
- **When might we need to solve such a problem?**

MLE When Labels Go Missing..

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

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

- 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 Θ now?
- **When might we need to solve such a problem?**
  - Mixture density estimation: Given N inputs $\boldsymbol{x}_1, \ldots, \boldsymbol{x}_N$, model $p(\boldsymbol{x})$ as a mixture of distributions

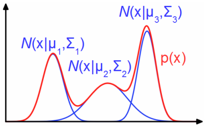


MLE When Labels Go Missing..

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

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

- 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 Θ now?
- **When might we need to solve such a problem?**
  - Mixture density estimation: Given N inputs $\boldsymbol{x}_1, \ldots, \boldsymbol{x}_N$, model $p(\boldsymbol{x})$ as a mixture of distributions



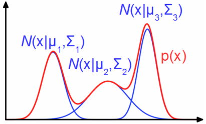
  - Probabilistic clustering: Same as density estimation; can get cluster ids once Θ is estimated

MLE When Labels Go Missing..

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

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

- 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 Θ now?
- **When might we need to solve such a problem?**
  - Mixture density estimation: Given N inputs $\boldsymbol{x}_1, \ldots, \boldsymbol{x}_N$, model $p(\boldsymbol{x})$ as a mixture of distributions



  - Probabilistic clustering: Same as density estimation; can get cluster ids once Θ is estimated
  - Semi-supervised generative classification: In training data, some y_n's are known, some not known

MLE When Labels Go Missing..

- Recall the MLE problem for Θ when the labels are known

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

MLE When Labels Go Missing..

- Recall the MLE problem for Θ when the labels are known

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

- Will estimating Θ via MLE be as easy if y_n's are unknown? We only have $\mathbf{X} = \{\boldsymbol{x}_1, \ldots, \boldsymbol{x}_N\}$

MLE When Labels Go Missing..

- Recall the MLE problem for Θ when the labels are known

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

- Will estimating Θ via MLE be as easy if y_n's are unknown? We only have $\mathbf{X} = \{\boldsymbol{x}_1, \ldots, \boldsymbol{x}_N\}$
- The MLE problem for $\Theta = \{\pi_k, \mu_k, \Sigma_k\}_{k=1}^{K}$ in this case would be (assuming i.i.d. data)

$$\hat{\Theta} = \arg\max_{\Theta} \log p(\mathbf{X}|\Theta) = \arg\max_{\Theta} \log \prod_{n=1}^{N} p(\boldsymbol{x}_n|\Theta)$$

MLE When Labels Go Missing..

- Recall the MLE problem for Θ when the labels are known

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

- Will estimating Θ via MLE be as easy if y_n's are unknown? We only have $\mathbf{X} = \{\boldsymbol{x}_1, \ldots, \boldsymbol{x}_N\}$
- The MLE problem for $\Theta = \{\pi_k, \mu_k, \Sigma_k\}_{k=1}^{K}$ in this case would be (assuming i.i.d. data)

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

MLE When Labels Go Missing..

- Recall the MLE problem for Θ when the labels are known

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

- Will estimating Θ via MLE be as easy if y_n's are unknown? We only have $\mathbf{X} = \{\boldsymbol{x}_1, \ldots, \boldsymbol{x}_N\}$
- The MLE problem for $\Theta = \{\pi_k, \mu_k, \Sigma_k\}_{k=1}^{K}$ in this case would be (assuming i.i.d. data)

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

- Computing each likelihood $p(\boldsymbol{x}_n|\Theta)$ in this case requires summing over all possible values of y_n

$$p(\boldsymbol{x}_n|\Theta) = \sum_{k=1}^{K} p(\boldsymbol{x}_n, y_n = k|\Theta)$$

MLE When Labels Go Missing..

- Recall the MLE problem for Θ when the labels are known

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

- Will estimating Θ via MLE be as easy if y_n's are unknown? We only have $\mathbf{X} = \{\boldsymbol{x}_1, \ldots, \boldsymbol{x}_N\}$
- The MLE problem for $\Theta = \{\pi_k, \mu_k, \Sigma_k\}_{k=1}^{K}$ in this case would be (assuming i.i.d. data)

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

- Computing each likelihood $p(\boldsymbol{x}_n|\Theta)$ in this case requires summing over all possible values of y_n

$$p(\boldsymbol{x}_n|\Theta) = \sum_{k=1}^{K} p(\boldsymbol{x}_n, y_n = k|\Theta) = \sum_{k=1}^{K} p(y_n = k|\Theta) p(\boldsymbol{x}_n|y_n = k, \Theta)$$

MLE When Labels Go Missing..

- Recall the MLE problem for Θ when the labels are known

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

- Will estimating Θ via MLE be as easy if y_n's are unknown? We only have $\mathbf{X} = \{\boldsymbol{x}_1, \ldots, \boldsymbol{x}_N\}$
- The MLE problem for $\Theta = \{\pi_k, \mu_k, \Sigma_k\}_{k=1}^{K}$ in this case would be (assuming i.i.d. data)

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

- Computing each likelihood $p(\boldsymbol{x}_n|\Theta)$ in this case requires summing over all possible values of y_n

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

MLE When Labels Go Missing..

- Recall the MLE problem for Θ when the labels are known

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

- Will estimating Θ via MLE be as easy if y_n's are unknown? We only have $\mathbf{X} = \{\boldsymbol{x}_1, \ldots, \boldsymbol{x}_N\}$
- The MLE problem for $\Theta = \{\pi_k, \mu_k, \Sigma_k\}_{k=1}^{K}$ in this case would be (assuming i.i.d. data)

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

- Computing each likelihood $p(\boldsymbol{x}_n|\Theta)$ in this case requires summing over all possible values of y_n

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

- The MLE problem for Θ when the labels are unknown

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

MLE When Labels Go Missing..

- So we saw that the MLE problem for Θ when the labels are unknown

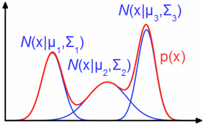
$$\hat{\Theta} = \arg\max_{\Theta} \sum_{n=1}^{N} \log \sum_{k=1}^{K} \pi_k \mathcal{N}(\boldsymbol{x}_n | \mu_k, \Sigma_k)$$

MLE When Labels Go Missing..

- So we saw that the MLE problem for Θ when the labels are unknown

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

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

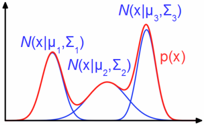


MLE When Labels Go Missing..

- So we saw that the MLE problem for Θ when the labels are unknown

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

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



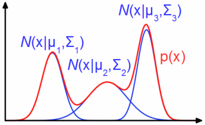
- Note: The Gaussian can be replaced by other distributions too (e.g., Poisson mixture model)

MLE When Labels Go Missing..

- So we saw that the MLE problem for Θ when the labels are unknown

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

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



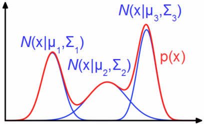
- Note: The Gaussian can be replaced by other distributions too (e.g., Poisson mixture model)
- A small issue now: Log can't go inside the summation. Expressions won't be simple anymore

MLE When Labels Go Missing..

- So we saw that the MLE problem for Θ when the labels are unknown

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

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



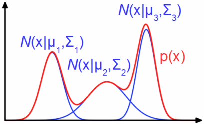
- Note: The Gaussian can be replaced by other distributions too (e.g., Poisson mixture model)
- A small issue now: Log can't go inside the summation. Expressions won't be simple anymore
- Note: Can still take (partial) derivatives and do GG/SGD etc. but these are iterative methods

MLE When Labels Go Missing..

- So we saw that the MLE problem for Θ when the labels are unknown

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

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



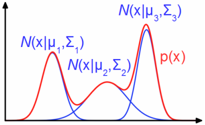
- Note: The Gaussian can be replaced by other distributions too (e.g., Poisson mixture model)
- A small issue now: Log can't go inside the summation. Expressions won't be simple anymore
- Note: Can still take (partial) derivatives and do GG/SGD etc. but these are iterative methods
  - Recall that we 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 Θ when the labels are unknown

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

- 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

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

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 Θ as $\hat{\Theta}$

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

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

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

$$
\begin{aligned}
\hat{z}_n &= \arg\max_{k \in \{1,\ldots,K\}} p(\boldsymbol{x}_n, z_n = k | \hat{\Theta}) \\
&= \arg\max_{k \in \{1,\ldots,K\}} p(z_n = k | \boldsymbol{x}_n, \hat{\Theta})
\end{aligned}
$$

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

$$\begin{aligned} \hat{z}_n &= \arg\max_{k \in \{1,\ldots,K\}} p(\boldsymbol{x}_n, z_n = k | \hat{\Theta}) \\ &= \arg\max_{k \in \{1,\ldots,K\}} p(z_n = k | \boldsymbol{x}_n, \hat{\Theta}) \end{aligned}$$

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

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

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

$$\begin{aligned}\hat{z}_n &= \arg\max_{k\in\{1,\ldots,K\}} p(\boldsymbol{x}_n, z_n = k|\hat{\Theta}) \\ &= \arg\max_{k\in\{1,\ldots,K\}} p(z_n = k|\boldsymbol{x}_n, \hat{\Theta})\end{aligned}$$

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

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

4. Go to step 2 if not yet converged

Is ALT-OPT Doing The Correct Thing?

- Our original problem was

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

Is ALT-OPT Doing The Correct Thing?

- Our original problem was

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

- What ALT-OPT did was the following

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

Is ALT-OPT Doing The Correct Thing?

- Our original problem was

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

- What ALT-OPT did was the following

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

- We clearly aren't solving the original problem!

$$\arg\max_{\Theta} \log p(\mathbf{X}|\Theta) \quad \text{vs} \quad \arg\max_{\Theta} \log p(\mathbf{X}, \hat{\mathbf{Z}}|\Theta)$$

Is ALT-OPT Doing The Correct Thing?

- Our original problem was

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

- What ALT-OPT did was the following

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

- We clearly aren't solving the original problem!

$$\arg\max_{\Theta} \log p(\mathbf{X}|\Theta) \quad \text{vs} \quad \arg\max_{\Theta} \log p(\mathbf{X}, \hat{\mathbf{Z}}|\Theta)$$

- Also, we updated $\hat{z}_n$ as follows

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

Is ALT-OPT Doing The Correct Thing?

- Our original problem was

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

- What ALT-OPT did was the following

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

- We clearly aren't solving the original problem!

$$\arg\max_{\Theta} \log p(\mathbf{X}|\Theta) \quad \text{vs} \quad \arg\max_{\Theta} \log p(\mathbf{X}, \hat{\mathbf{Z}}|\Theta)$$

- Also, we updated $\hat{z}_n$ as follows

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

- 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

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

- What ALT-OPT did was the following

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

- We clearly aren't solving the original problem!

$$\arg\max_{\Theta} \log p(\mathbf{X}|\Theta) \quad \text{vs} \quad \arg\max_{\Theta} \log p(\mathbf{X}, \hat{\mathbf{Z}}|\Theta)$$

- Also, we updated $\hat{z}_n$ as follows

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

- 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)

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

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)

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

- The above is true for any choice of the distribution $q(\mathbf{Z})$

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)

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

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

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

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)

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

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

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

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

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)

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

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

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

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

EM for GMM

- The EM algorithm for GMM does the following

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

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

- Here $\mathbb{E}[z_{nk}]$ is the expectation of z_{nk} w.r.t. posterior $p(\boldsymbol{z}_n|\boldsymbol{x}_n)$ and is given by

$$\mathbb{E}[z_{nk}] = 0 \times p(z_{nk} = 0|\boldsymbol{x}_n) + 1 \times p(z_{nk} = 1|\boldsymbol{x}_n)$$

EM for GMM

- The EM algorithm for GMM does the following

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

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

- Here $\mathbb{E}[z_{nk}]$ is the expectation of z_{nk} w.r.t. posterior $p(\boldsymbol{z}_n|\boldsymbol{x}_n)$ and is given by

$$\begin{aligned} \mathbb{E}[z_{nk}] &= 0 \times p(z_{nk} = 0|\boldsymbol{x}_n) + 1 \times p(z_{nk} = 1|\boldsymbol{x}_n) \\ &= p(z_{nk} = 1|\boldsymbol{x}_n) \end{aligned}$$

EM for GMM

- The EM algorithm for GMM does the following

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

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

- Here $\mathbb{E}[z_{nk}]$ is the expectation of z_{nk} w.r.t. posterior $p(\boldsymbol{z}_n|\boldsymbol{x}_n)$ and is given by

$$\begin{aligned}
\mathbb{E}[z_{nk}] &= 0 \times p(z_{nk} = 0|\boldsymbol{x}_n) + 1 \times p(z_{nk} = 1|\boldsymbol{x}_n) \\
&= p(z_{nk} = 1|\boldsymbol{x}_n) \\
&\propto p(z_{nk} = 1)p(\boldsymbol{x}_n|z_{nk} = 1) \qquad \text{(from Bayes Rule)}
\end{aligned}$$

EM for GMM

- The EM algorithm for GMM does the following

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

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

- Here $\mathbb{E}[z_{nk}]$ is the expectation of z_{nk} w.r.t. posterior $p(\boldsymbol{z}_n|\boldsymbol{x}_n)$ and is given by

$$\begin{aligned}
\mathbb{E}[z_{nk}] &= 0 \times p(z_{nk} = 0|\boldsymbol{x}_n) + 1 \times p(z_{nk} = 1|\boldsymbol{x}_n) \\
&= p(z_{nk} = 1|\boldsymbol{x}_n) \\
&\propto p(z_{nk} = 1)p(\boldsymbol{x}_n|z_{nk} = 1) \qquad \text{(from Bayes Rule)} \\
\text{Thus} \quad \mathbb{E}[z_{nk}] &\propto \pi_k \mathcal{N}(\boldsymbol{x}_n|\boldsymbol{\mu}_k, \boldsymbol{\Sigma}_k) \qquad \text{(Posterior prob. that } \boldsymbol{x}_n \text{ is generated by } k\text{-th Gaussian)}
\end{aligned}$$

EM for GMM

- The EM algorithm for GMM does the following

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

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

- Here $\mathbb{E}[z_{nk}]$ is the expectation of z_{nk} w.r.t. posterior $p(\boldsymbol{z}_n|\boldsymbol{x}_n)$ and is given by

$$\begin{aligned}
\mathbb{E}[z_{nk}] &= 0 \times p(z_{nk} = 0|\boldsymbol{x}_n) + 1 \times p(z_{nk} = 1|\boldsymbol{x}_n) \\
&= p(z_{nk} = 1|\boldsymbol{x}_n) \\
&\propto p(z_{nk} = 1)p(\boldsymbol{x}_n|z_{nk} = 1) \qquad \text{(from Bayes Rule)} \\
\text{Thus} \quad \mathbb{E}[z_{nk}] &\propto \pi_k \mathcal{N}(\boldsymbol{x}_n|\boldsymbol{\mu}_k, \boldsymbol{\Sigma}_k) \qquad \text{(Posterior prob. that } \boldsymbol{x}_n \text{ is generated by } k\text{-th Gaussian)}
\end{aligned}$$

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