Latent Variable Models

CS771: Introduction to Machine Learning

Dimensionality Reduction: Out-of-sample Embedding

- Some dim-red methods can only compute the embedding of the training data
- Given N training samples $\{x_1, x_2, \dots, x_N\}$ they will give their embedding $\{z_1, z_2, \dots, z_N\}$
- However, given a new point x_* (not in the training samples), they can't produce its embedding z_* easily
  - Thus no easy way of getting "out-of-sample" embedding
- Some of the nonlinear dim-red methods like LLE, SNE, KPCA, etc have this limitation
  - Reason: They don't learn an explicit encoder and directly optimize for $\{z_n\}_{n=1}^N$ given $\{x_n\}_{n=1}^N$
  - To get "out-of-sample" embeddings, these methods require some modifications*
- But many other methods do explicitly learn a mapping $z = f(x)$ in form of an "encoder" f that can give z_* for any new x_* as well (such methods are more useful)
  - For PCA, the $D \times K$ projection matrix W_K is this encoder function and $z_* = W_K^\top x_*$
  - Neural network based autoencoders can also do this (will see them later)

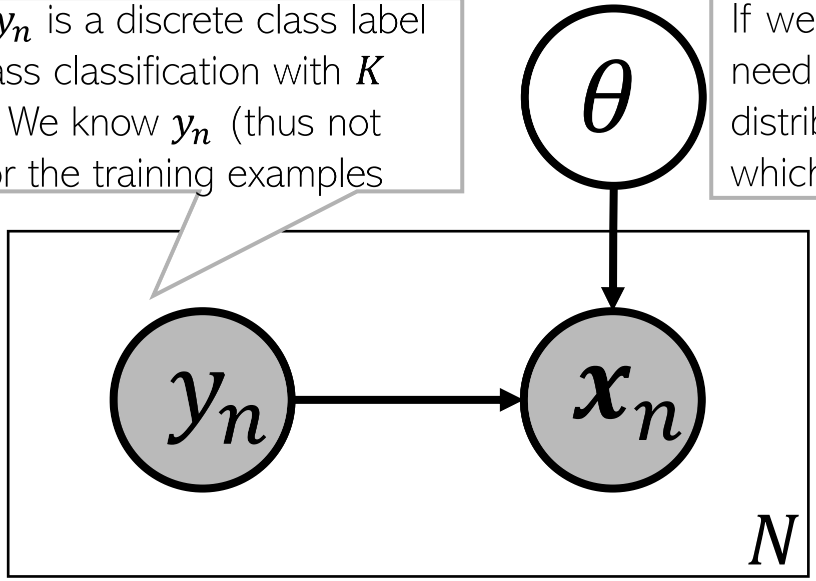
*Out-of-Sample Extensions for LLE, Isomap, MDS, Eigenmaps, and Spectral Clustering (Bengio et al, 2003)

Latent Variable Models

Example: Generative Models with Latent Variables

- Two generative models of inputs $\boldsymbol{x}_n$ without (left) and with (right) latent variables

Assume y_n is a discrete class label (multi-class classification with K classes). We know y_n (thus not latent) for the training examples

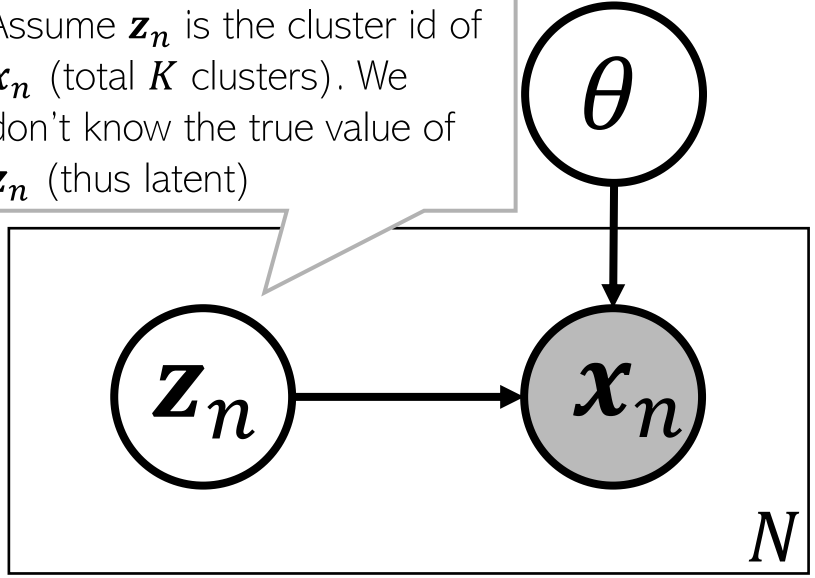


If we knew the value of z_n, we won't need to do MLE/MAP of the mixture distribution but only of $p(\boldsymbol{x}_n|z_n = k, \theta)$ which is much easier

When z_n is not known, we can do MLE/MAP for $\boldsymbol{\theta}$ using an alternating estimating procedure: Estimate $\boldsymbol{z}_n$ given $\boldsymbol{\theta}$ and estimate $\boldsymbol{\theta}$ given $\boldsymbol{z}_n$

This procedure of estimating $\boldsymbol{\theta}$ also gives $\boldsymbol{z}_n$ as a by-product

Assume $\boldsymbol{z}_n$ is the cluster id of $\boldsymbol{x}_n$ (total K clusters). We don't know the true value of $\boldsymbol{z}_n$ (thus latent)



- Suppose we wish to estimate (e.g., using MLE/MAP) params $\boldsymbol{\theta}$ of distribution of $\boldsymbol{x}_n$
- For case 1, the distribution is $p(\boldsymbol{x}_n|y_n, \boldsymbol{\theta})$ and MLE/MAP of $\boldsymbol{\theta}$ easy since y_n is known
- For case 2, distribution is more complex because true $\boldsymbol{z}_n$ is not known

Reason: The functional form of mixture can be messy

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

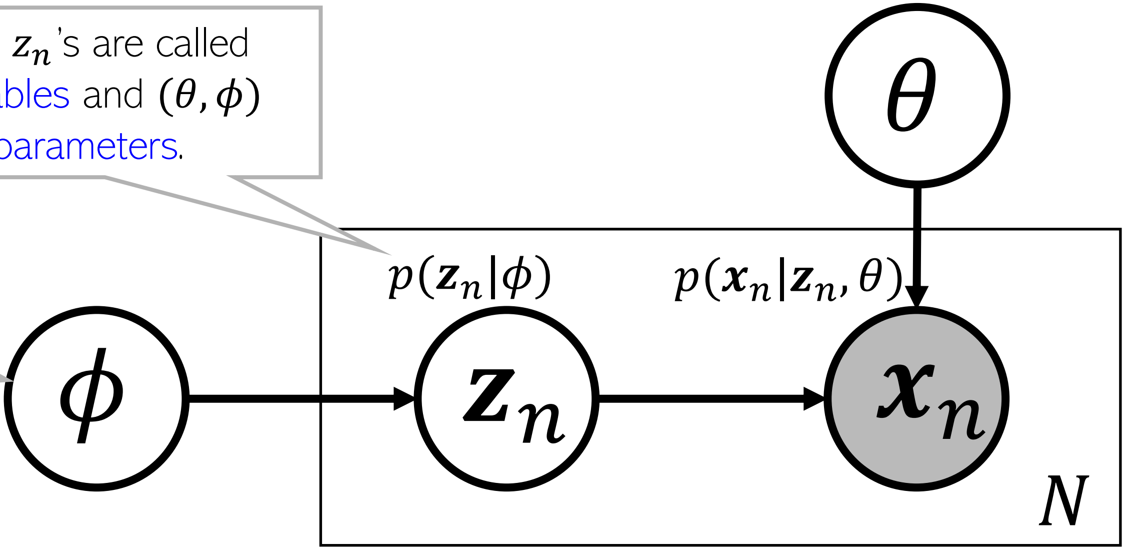
MLE/MAP a bit difficult for this more complex "mixture" of distributions

Components of an LVM

- Recall that the goal is to estimate $\boldsymbol{\theta}$ (and $\boldsymbol{z}_n$ is also unknown)
- In LVM, we treat $\boldsymbol{z}_n$ as a random variable and assume a prior distribution $p(\boldsymbol{z}_n|\phi)$

In an LVM, z_n's are called latent variables and (θ, ϕ) are called parameters.

Will also need to estimate $\boldsymbol{\phi}$ in addition to $\boldsymbol{\theta}$



This prior tells us what the value of $\boldsymbol{z}_n$ is before we have seen the input $\boldsymbol{x}_n$

Ultimately, we will compute the distribution of $\boldsymbol{z}_n$ conditioned on the input $\boldsymbol{x}_n$

For example, a D-dimensional Gaussian if $\boldsymbol{x}_n \in \mathbb{R}^D$

- We will also assume a suitable conditional distribution $p(\boldsymbol{x}_n|\boldsymbol{z}_n, \theta)$ for $\boldsymbol{x}_n$
- The form of $p(\boldsymbol{z}_n|\phi)$ will depend on the nature of $\boldsymbol{z}_n$, e.g.,
  - If $\boldsymbol{z}_n$ is discrete with K possible values, $p(\boldsymbol{z}_n|\phi) = \textbf{multinoulli}(\boldsymbol{z}_n|\boldsymbol{\pi})$
  - If $\boldsymbol{z}_n \in \mathbb{R}^K$, $p(\boldsymbol{z}_n|\phi) = \mathcal{N}(\boldsymbol{z}_n|\boldsymbol{\mu}, \boldsymbol{\Sigma})$, a K-dim Gaussian

Why Direct MLE/MAP is Hard for LVMs?

- Direct MLE/MAP of parameters $(\theta, \phi) = \Theta$ without estimating $\boldsymbol{z}_n$ is hard
- Reason: Given N observations $\boldsymbol{x}_n, n = 1,2, \dots, N$, the MLE problem for Θ will be

Also note that $p(\boldsymbol{x}_n, \boldsymbol{z}_n|\Theta) = p(\boldsymbol{z}_n|\phi)p(\boldsymbol{x}_n|\boldsymbol{z}_n, \theta)$

$$\underset{\Theta}{\operatorname{argmax}} \sum_{n=1}^{N} \log p(\boldsymbol{x}_n|\Theta) = \underset{\Theta}{\operatorname{argmax}} \sum_{n=1}^{N} \log \sum_{\boldsymbol{z}_n} p(\boldsymbol{x}_n, \boldsymbol{z}_n|\Theta)$$

Summing over all possible values $\boldsymbol{z}_n$ can take (would be an integral instead of sum if $\boldsymbol{z}_n$ is continuous

Gaussian Mixture Model (GMM).

- For a mixture of K Gaussians, $p(\boldsymbol{x}_n|\Theta)$ will be

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

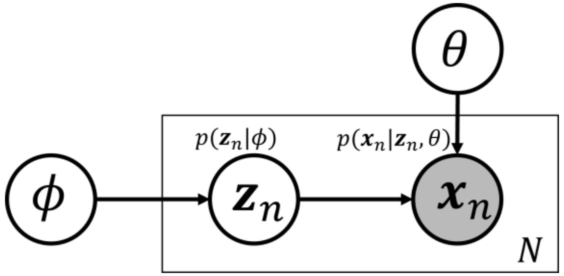
- The MLE problem for GMM would be

ALT-OPT or EM makes it simpler by using hard/soft guesses of $\boldsymbol{z}_n$'s

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

The log of sum doesn't give us a simple expression; MLE can still be done using gradient based methods but updates will be complicated.

How to Guess z_n in an LVM?



- Note that $\boldsymbol{z}_n$ is a random variable with prior distribution $p(\boldsymbol{z}_n|\phi)$
- Can compute its conditional posterior (CP) distribution as

Called conditional posterior because it is conditioned on data as well as Θ (assuming we have already estimated Θ)

$(\theta, \phi) = \Theta$

$$p(\boldsymbol{z}_n|\boldsymbol{x}_n,\Theta) = \frac{p(\boldsymbol{z}_n|\Theta)p(\boldsymbol{x}_n|\boldsymbol{z}_n,\Theta)}{p(\boldsymbol{x}_n|\Theta)} = \frac{p(\boldsymbol{z}_n|\phi)p(\boldsymbol{x}_n|\boldsymbol{z}_n,\theta)}{p(\boldsymbol{x}_n|\Theta)}$$

- If we just want the single best (hard) guess of $\boldsymbol{z}_n$ then that can be computed as

Used in ALT-OPT for LVMs

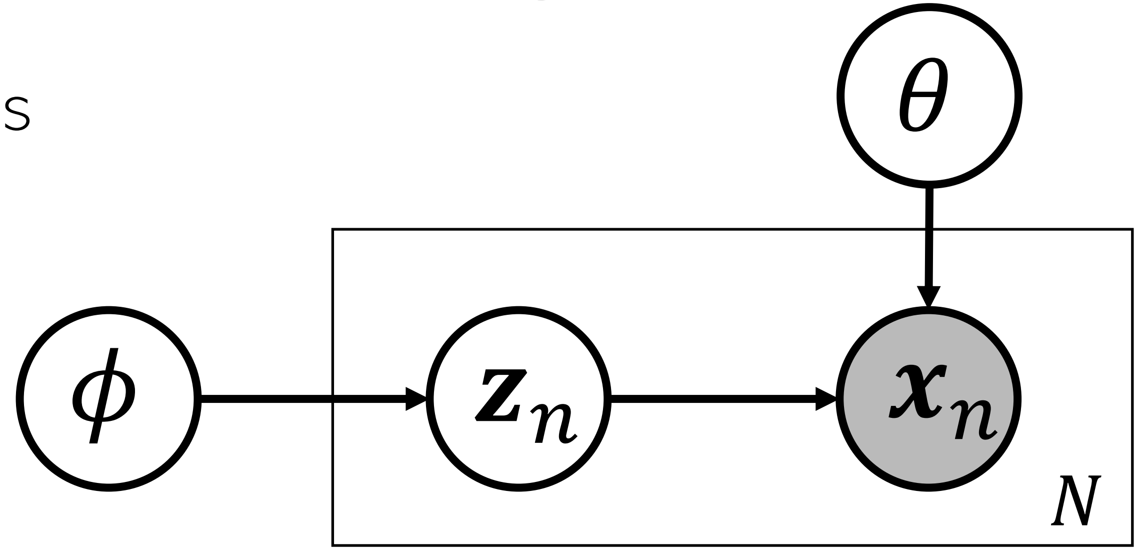
$$\hat{z}_n = \text{argmax}_{z_n}\, p(\boldsymbol{z}_n|\boldsymbol{x}_n,\Theta) = \text{argmax}_{z_n}\, p(\boldsymbol{z}_n|\phi)p(\boldsymbol{x}_n|\boldsymbol{z}_n,\theta)$$

Used in Expectation-Maximization (EM) algo for LVMs

- Otherwise, we can compute and use CP $p(\boldsymbol{z}_n|\boldsymbol{x}_n,\Theta)$ to get a soft/probabilistic guess
  - Using the CP $p(\boldsymbol{z}_n|\boldsymbol{x}_n,\Theta)$ we can compute quantities such as expectation of $\boldsymbol{z}_n$
  - If $p(\boldsymbol{z}_n|\phi)$ and $p(\boldsymbol{x}_n|\boldsymbol{z}_n,\theta)$ are conjugate to each other then CP $p(\boldsymbol{z}_n|\boldsymbol{x}_n,\Theta)$ is easy to compute
- Computing hard guess is usually easier but ignores the uncertainty in $\boldsymbol{z}_n$

LVMs: Incomplete vs Complete Data Log Likelihood

- We can define two types of likelihoods for LVMs
  - Incomplete data log likelihood (ILL) $\log p(\boldsymbol{X}|\Theta)$
  - Complete data log likelihood (CLL) $\log p(\boldsymbol{X}, \boldsymbol{Z}|\Theta)$



- Named so because we can think of latent $\boldsymbol{Z}$ "completing" the observed data $\boldsymbol{X}$
- Since $\boldsymbol{Z}$ is never observed (is latent), to estimate Θ we must maximize the ILL

$$\underset{\Theta}{\operatorname{argmax}} \log p(\boldsymbol{X}|\Theta) = \underset{\Theta}{\operatorname{argmax}} \log \sum_{\boldsymbol{Z}} p(\boldsymbol{X}, \boldsymbol{Z}|\Theta)$$

- But since ILL maximization is hard (log of sum/integral over the unknown $\boldsymbol{Z}$), we instead maximize the CLL $p(\boldsymbol{X}, \boldsymbol{Z}|\Theta)$ using hard/soft guesses of $\boldsymbol{Z}$

MLE for LVM

Also, we can use this idea to find MAP solution of Θ if we want. Assume a prior $p(\Theta)$ and simply add a $\log p(\Theta)$ term to these objectives

Note that we aren't solving the original MLE problem $\underset{\Theta}{\operatorname{argmax}} \log p(\boldsymbol{X}|\Theta)$ anymore.

However, what we are solving now is still justifiable theoretically (will see later)

- If using a hard guess

$$\Theta_{MLE} = \underset{\Theta}{\operatorname{argmax}} \log p(\boldsymbol{X}, \hat{\boldsymbol{Z}} \,|\Theta)$$

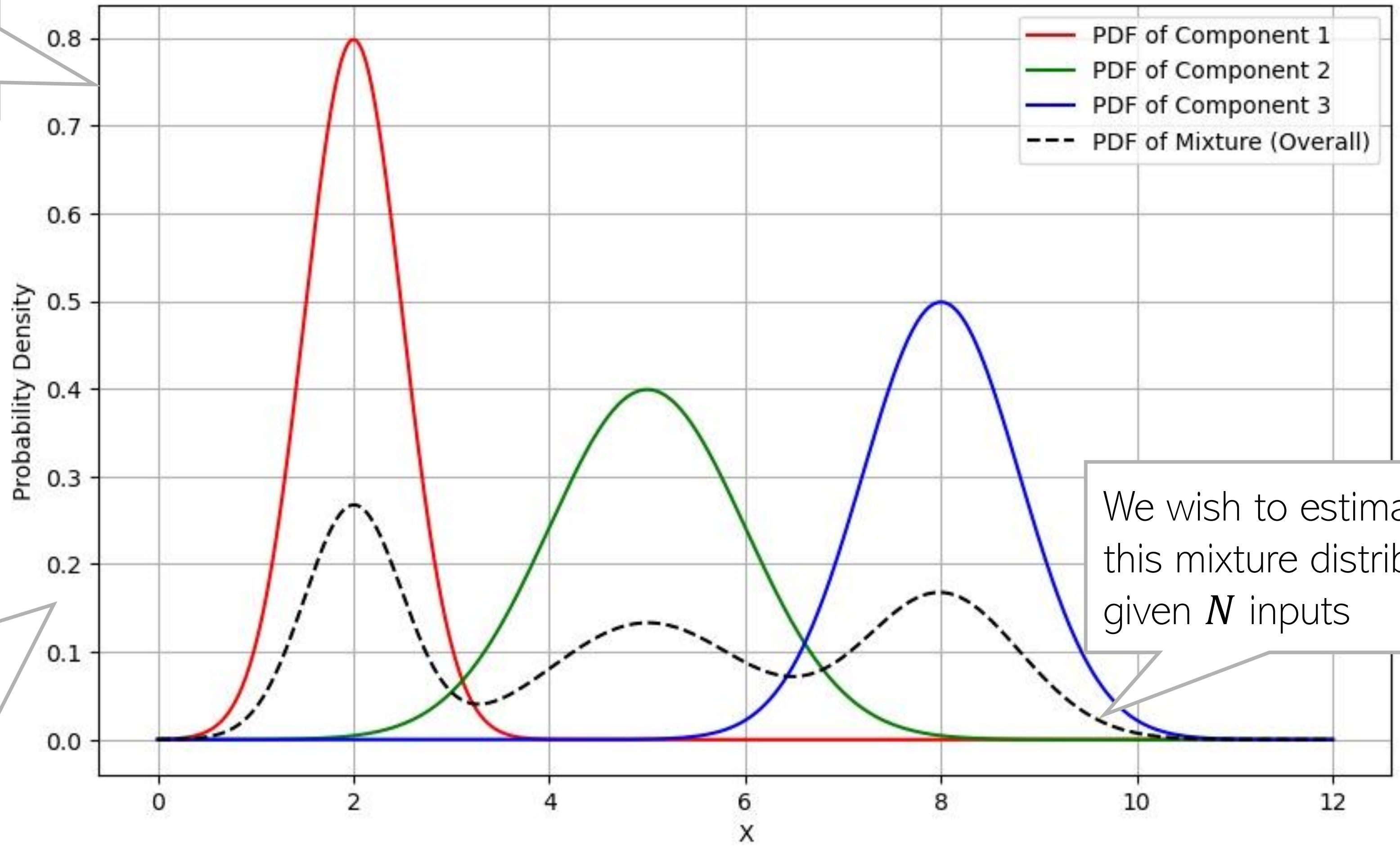
- If using a soft (probabilistic) guess

$$\Theta_{MLE} = \underset{\Theta}{\operatorname{argmax}} \mathbb{E}[\log p(\boldsymbol{X}, \boldsymbol{Z}|\Theta)]$$

- In LVMs, hard and soft guesses of $\boldsymbol{Z}$ would depend on Θ (since $\boldsymbol{Z}$ and Θ are coupled)
- Thus we need a procedure which alternates between estimating $\boldsymbol{Z}$ and estimating Θ

An LVM: Gaussian Mixture Model

Inputs are assumed generated from a mixture of Gaussians. But we don't know which input was generated by which Gaussian



If we knew which input came from which Gaussian (akin to knowing their true labels), the problem is easy – simply estimate each Gaussian using the inputs that came from that Gaussian (just like generative classification)

We wish to estimate this mixture distribution given N inputs

Detour: MLE for Generative Classification

- Assume a K class generative classification model with Gaussian class-conditionals
- Assume class $k = 1,2, \ldots, K$ is modeled by a Gaussian with mean μ_k and cov matrix Σ_k
- Can assume label z_n to be one-hot and then $z_{nk} = 1$ if $z_n = k$, and $z_{nk} = 0$, o/w
  - Note: For each label, using notation z_n instead of y_n
- Assuming class marginal $p(z_n = k) = \pi_k$, the model's params $\Theta = \{\pi_k, \mu_k, \Sigma_k\}_{k=1}^{K}$
- The MLE objective $\log p(\boldsymbol{X}, \boldsymbol{Z}|\Theta)$ is (will provide a note for the proof)

$$\Theta_{MLE} = \operatorname{argmax}_{\{\pi_k, \mu_k, \Sigma_k\}_{k=1}^{K}} \sum_{n=1}^{N} \sum_{k=1}^{K} z_{nk}[\log \pi_k + \log \mathcal{N}(\boldsymbol{x}_n|\mu_k, \Sigma_k)]$$

$$\hat{\pi}_k = \frac{1}{N} \sum_{n=1}^{N} z_{nk} \qquad \hat{\mu}_k = \frac{1}{N_k} \sum_{n=1}^{N} z_{nk} \boldsymbol{x}_n \qquad \hat{\Sigma}_k = \frac{1}{N_k} \sum_{n=1}^{N} z_{nk} (\boldsymbol{x}_n - \hat{\mu}_k)(\boldsymbol{x}_n - \hat{\mu}_k)^\top$$

Same as $\frac{N_k}{N}$

Same as $\frac{1}{N_k} \sum_{n:z_n=k}^{N} \boldsymbol{x}_n$

Same as $\frac{1}{N_k} \sum_{n:z_n=k}^{N} (\boldsymbol{x}_n - \hat{\mu}_k)(\boldsymbol{x}_n - \hat{\mu}_k)^\top$

MLE for GMM: Using Guesses of z_n

Will have the exact same form for the expression of MLE objective as generative classification with Gaussian class-conditionals (except z_n is unknown)

- Using a hard guess $\hat{\boldsymbol{z}}_n = \text{argmax}_{z_n}\, p(\boldsymbol{z}_n|\boldsymbol{x}_n,\Theta)$, the MLE problem for GMM

Log likelihood of Θ w.r.t. data $\boldsymbol{X}$ and hard guesses $\hat{\boldsymbol{Z}}$ of cluster ids

Assuming x_n given z_n and Θ are i.i.d.

$$\Theta_{MLE} = \underset{\Theta}{\text{argmax}} \log p(\boldsymbol{X}, \hat{\boldsymbol{Z}}\,|\Theta) = \text{argmax}_{\Theta} \sum_{n=1}^{N}\sum_{k=1}^{K} \hat{z}_{nk}[\log \pi_k + \log \mathcal{N}(\boldsymbol{x}_n|\mu_k,\Sigma_k)]$$

- Using a soft guess $\mathbb{E}[\boldsymbol{z}_n]$, the MLE problem for GMM

Expected log likelihood of Θ w.r.t. data $\boldsymbol{X}$ and $\boldsymbol{Z}$

z_{nk} appears at only one place in the log likelihood expression so easily replaced by expectation of z_{nk} w.r.t the CP $p(\boldsymbol{z}_n|\boldsymbol{x}_n,\Theta)$

$$\Theta_{MLE} = \underset{\Theta}{\text{argmax}}\, \mathbb{E}[\log p(\boldsymbol{X},\boldsymbol{Z}|\Theta)] = \text{argmax}_{\Theta} \sum_{n=1}^{N}\sum_{k=1}^{K} \mathbb{E}[z_{nk}][\log \pi_k + \log \mathcal{N}(\boldsymbol{x}_n|\mu_k,\Sigma_k)]$$

- In both cases, the MLE solution for $\Theta = \{\pi_k,\mu_k,\Sigma_k\}_{k=1}^{K}$ will be identical to that of generative classification with Gaussian class cond with z_{nk} replaced by $\hat{z}_{nk}$ or $\mathbb{E}[z_{nk}]$
  - Case 1 solved using ALT-OPT alternating b/w estimating Θ_{MLE} and $\hat{\boldsymbol{Z}}$
  - Case 2 solved using Expectation Maximization (EM) alternating b/w estimating Θ_{MLE} and $\mathbb{E}[\boldsymbol{Z}]$

ALT-OPT for GMM

- We will assume we have a "hard" (most probable) guess of z_n, say $\hat{z}_n$
- ALT-OPT which maximizes $\log p(\boldsymbol{X}, \hat{\boldsymbol{Z}} | \Theta)$ would look like this
  - Initialize $\Theta = \{\pi_k, \mu_k, \Sigma_k\}_{k=1}^{K}$ as $\hat{\Theta}$
  - Repeat the following until convergence
    - For each n, compute most probable value (our best guess) of z_n as

$$\hat{z}_n = \text{argmax}_{k=1,2,\ldots,K}\, p(z_n = k | \hat{\Theta}, \boldsymbol{x}_n)$$

Proportional to prior prob times likelihood, i.e., $p(z_n = k|\hat{\Theta})\, p(x_n|z_n = k, \hat{\Theta}) = \hat{\pi}_k \mathcal{N}(x_n|\hat{\mu}_k, \hat{\Sigma}_k)$

Posterior probability of point x_n belonging to cluster k, given current Θ

- Solve MLE problem for Θ using most probable z_n's

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

$$\hat{\pi}_k = \frac{1}{N} \sum_{n=1}^{N} \hat{z}_{nk} \qquad \hat{\mu}_k = \frac{1}{N_k} \sum_{n=1}^{N} \hat{z}_{nk} \boldsymbol{x}_n$$

N_k : Effective number of points in cluster k

$$\hat{\Sigma}_k = \frac{1}{N_k} \sum_{n=1}^{N} \hat{z}_{nk} (\boldsymbol{x}_n - \hat{\mu}_k)(\boldsymbol{x}_n - \hat{\mu}_k)^\top$$

Expectation-Maximization (EM) for GMM

- EM finds Θ_{MLE} by maximizing $\mathbb{E}[\log p(\boldsymbol{X}, \boldsymbol{Z}|\Theta)]$

Why w.r.t. this distribution? Will see justification in a bit

- Note: Expectation will be w.r.t. the CP of $\boldsymbol{Z}$, i.e., $p(\boldsymbol{Z}|\boldsymbol{X}, \Theta)$
- The EM algorithm for GMM operates as follows
  - Initialize $\Theta = \{\pi_k, \mu_k, \Sigma_k\}_{k=1}^{K}$ as $\hat{\Theta}$
  - Repeat until convergence
    - Compute CP $p(\boldsymbol{Z}|\boldsymbol{X}, \hat{\Theta})$ using current estimate of Θ. Since obs are i.i.d, compute for each n (and for $k = 1,2, .. K$)

Note that EM for GMM also gives a soft clustering $z_n = [\gamma_{n1}, \gamma_{n2}, \dots, \gamma_{nK}]$ for each input x_n

Same as $p(z_{nk} = 1|\,\boldsymbol{x}_n, \hat{\Theta})$, just a different notation

$$p\left(\boldsymbol{z}_n = k|\boldsymbol{x}_n, \hat{\Theta}\right) \propto p\left(\boldsymbol{z}_n = k|\hat{\Theta}\right) p\left(\boldsymbol{x}_n|\boldsymbol{z}_n = k, \hat{\Theta}\right) = \hat{\pi}_k \mathcal{N}\left(x_n|\hat{\mu}_k, \hat{\Sigma}_k\right)$$

  - Update Θ by maximizing $\mathbb{E}[\log p(\boldsymbol{X}, \boldsymbol{Z}|\Theta)]$

$$\frac{\hat{\pi}_k \mathcal{N}(x_n|\hat{\mu}_k, \hat{\Sigma}_k)}{\sum_{\ell=1}^{K} \hat{\pi}_\ell \mathcal{N}(x_n|\hat{\mu}_\ell, \hat{\Sigma}_\ell)}$$

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

Solution has a similar form as ALT-OPT (or gen. class.), except we now have the expectation of z_{nk} being used

$$\hat{\pi}_k = \frac{1}{N} \sum_{n=1}^{N} \mathbb{E}[z_{nk}] \qquad \hat{\mu}_k = \frac{1}{N_k} \sum_{n=1}^{N} \mathbb{E}[z_{nk}] \boldsymbol{x}_n$$

$$\hat{\Sigma}_k = \frac{1}{N_k} \sum_{n=1}^{N} \mathbb{E}[z_{nk}] (\boldsymbol{x}_n - \hat{\mu}_k)(\boldsymbol{x}_n - \hat{\mu}_k)^\top$$

$$\mathbb{E}[z_{nk}] = \gamma_{nk} = 0 \times p(z_{nk} = 0|x_n, \hat{\Theta}) + 1 \times p(z_{nk} = 1|x_n, \hat{\Theta})$$

$$= p(z_{nk} = 1|x_n, \hat{\Theta})$$

$$\propto \hat{\pi}_k \mathcal{N}\left(x_n|\hat{\mu}_k, \hat{\Sigma}_k\right)$$

Posterior probability of x_n belonging to k^{th} cluster

EM for GMM (Contd)

EM for Gaussian Mixture Model

1. Initialize $\Theta = \{\pi_k, \mu_k, \Sigma_k\}_{k=1}^K$ as $\Theta^{(0)}$, set $t = 1$
2. E step: compute the expectation of each $\boldsymbol{z}_n$ (we need it in M step)

Accounts for fraction of points in each cluster

Accounts for cluster shapes (since each cluster is a Gaussian

Soft K-means, which are more of a heuristic to get soft-clustering, also gave us probabilities but didn't account for cluster shapes or fraction of points in each cluster

$$\mathbb{E}[z_{nk}^{(t)}] = \gamma_{nk}^{(t)} \quad = \quad \frac{\pi_k^{(t-1)} \mathcal{N}(\boldsymbol{x}_n | \boldsymbol{\mu}_k^{(t-1)}, \boldsymbol{\Sigma}_k^{(t-1)})}{\sum_{\ell=1}^K \pi_\ell^{(t-1)} \mathcal{N}(\boldsymbol{x}_n | \boldsymbol{\mu}_\ell^{(t-1)}, \boldsymbol{\Sigma}_\ell^{(t-1)})} \quad \forall n, k$$

3. Given "responsibilities" $\gamma_{nk} = \mathbb{E}[z_{nk}]$, and $N_k = \sum_{n=1}^N \gamma_{nk}$, re-estimate Θ via MLE

Effective number of points in the k^{th} cluster

M-step:

$$\boldsymbol{\mu}_k^{(t)} = \frac{1}{N_k} \sum_{n=1}^N \gamma_{nk}^{(t)} \boldsymbol{x}_n$$

$$\boldsymbol{\Sigma}_k^{(t)} = \frac{1}{N_k} \sum_{n=1}^N \gamma_{nk}^{(t)} (\boldsymbol{x}_n - \boldsymbol{\mu}_k^{(t)})(\boldsymbol{x}_n - \boldsymbol{\mu}_k^{(t)})^\top$$

$$\pi_k^{(t)} = \frac{N_k}{N}$$

4. Set $t = t + 1$ and go to step 2 if not yet converged