10-701 Introduction to Machine Learning

The EM Algorithm

Thanks!

Spring 2019

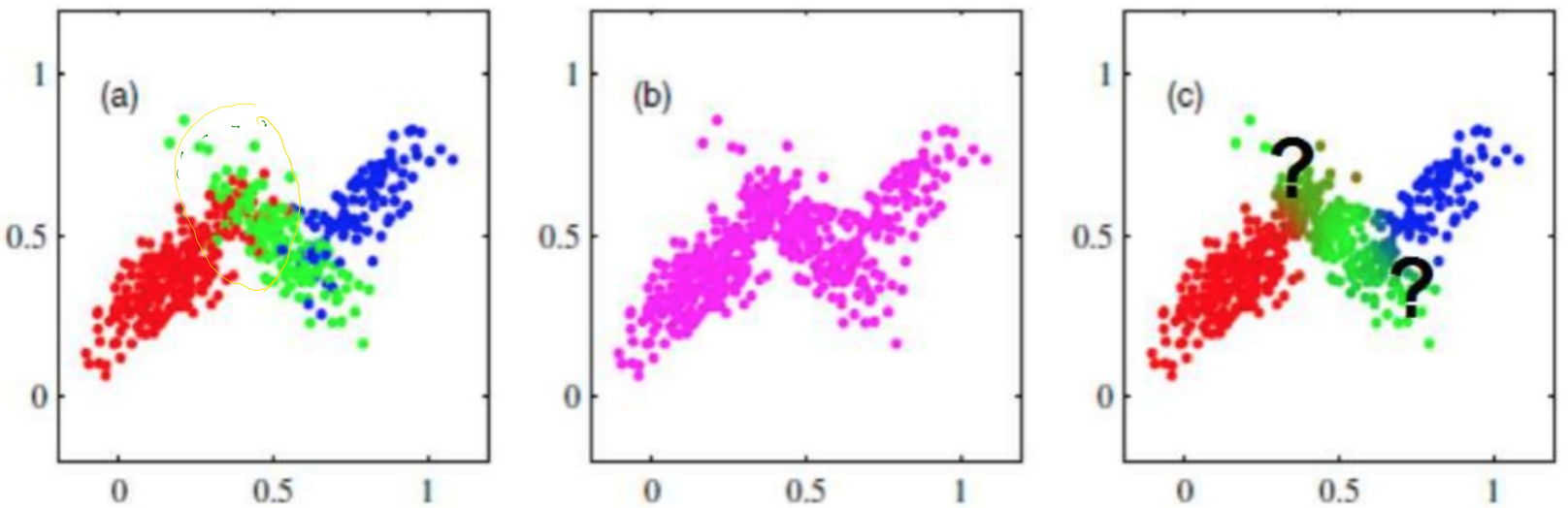
Ameet Talwalkar
(slide credit: Virginia Smith)

Gaussian mixture models

Potential issue with k-means . . .

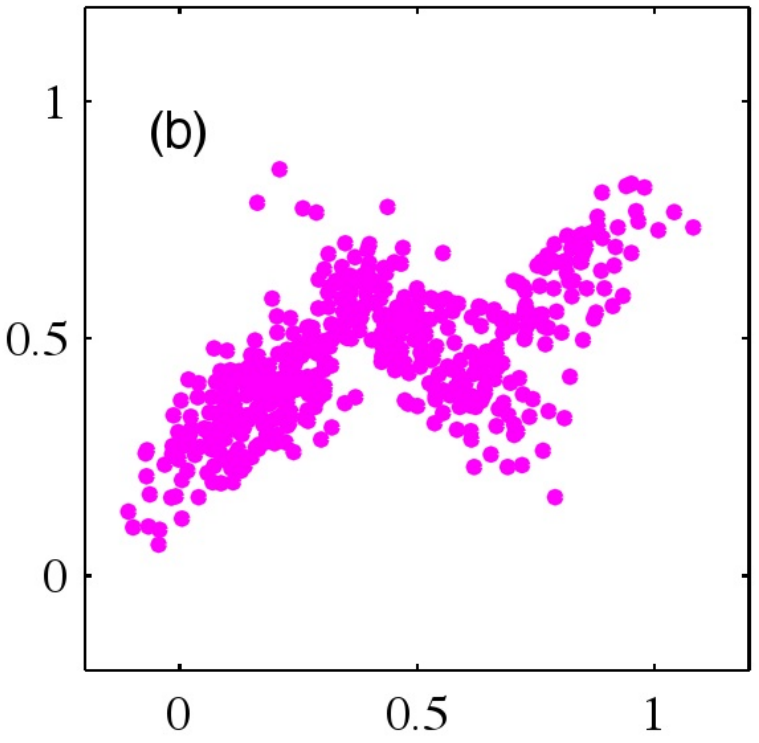
Data points are assigned *deterministically* to one (and only one) cluster

In reality, clusters may overlap, and it may be better to identify the *probability* that a point belongs to each cluster



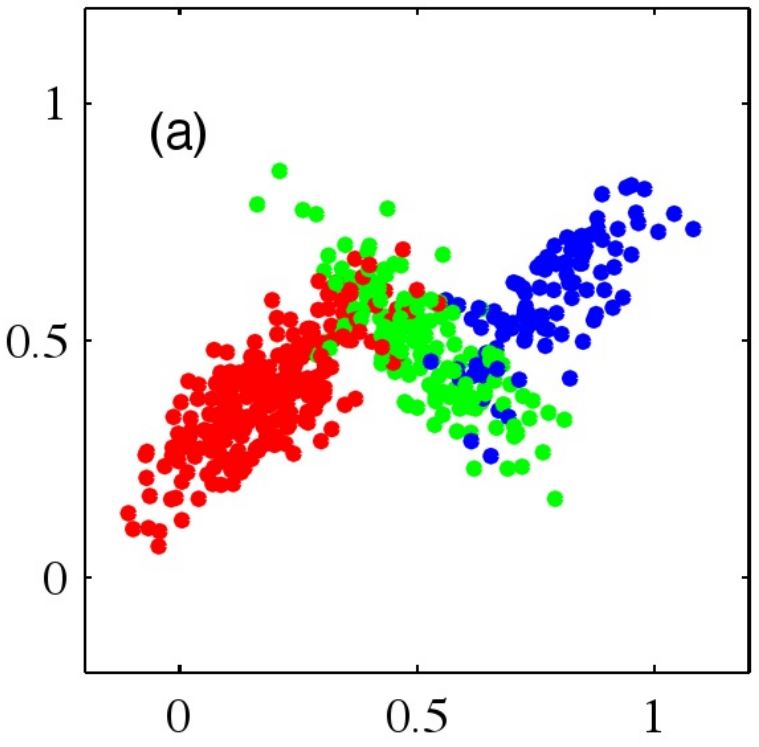
Probabilistic interpretation of clustering?

How can we model $p(\boldsymbol{x})$ to reflect our intuition that points stay close to their cluster centers?



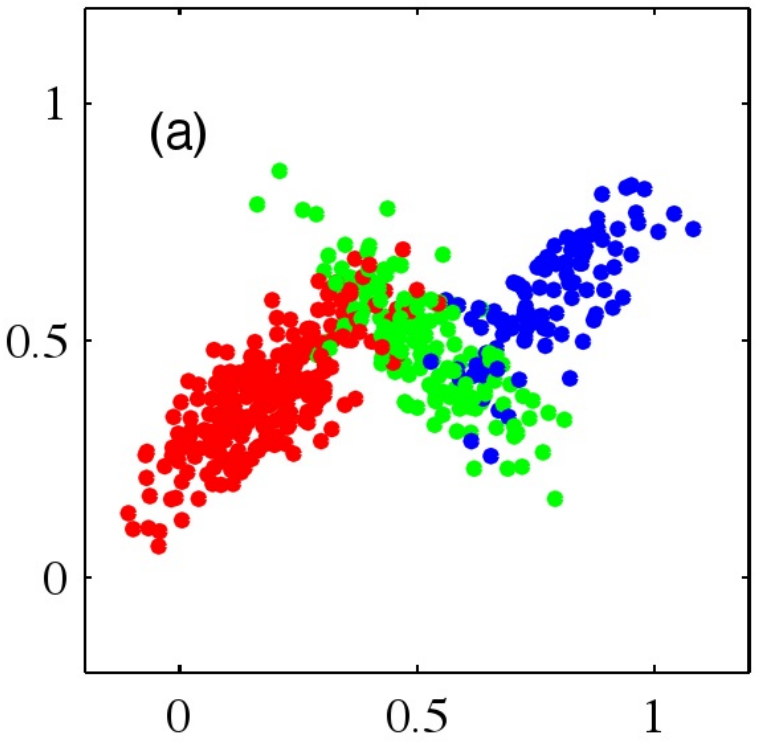
- Points seem to form 3 clusters
- We cannot model $p(\boldsymbol{x})$ with simple and known distributions
- E.g., the data is not a Gaussian b/c we have 3 distinct concentrated regions

Gaussian mixture models: intuition



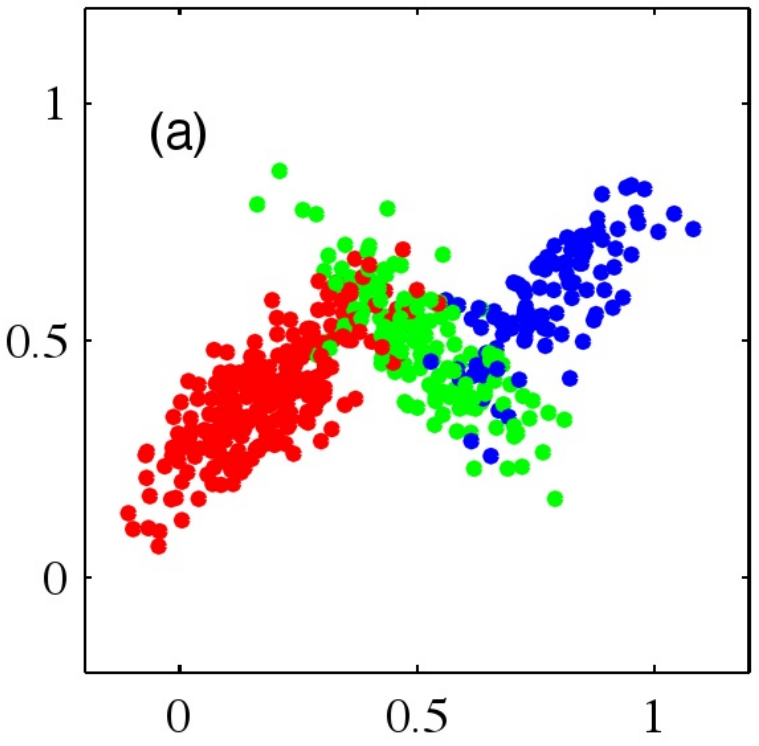
- **Key idea:** Model *each* region with a distinct distribution

Gaussian mixture models: intuition



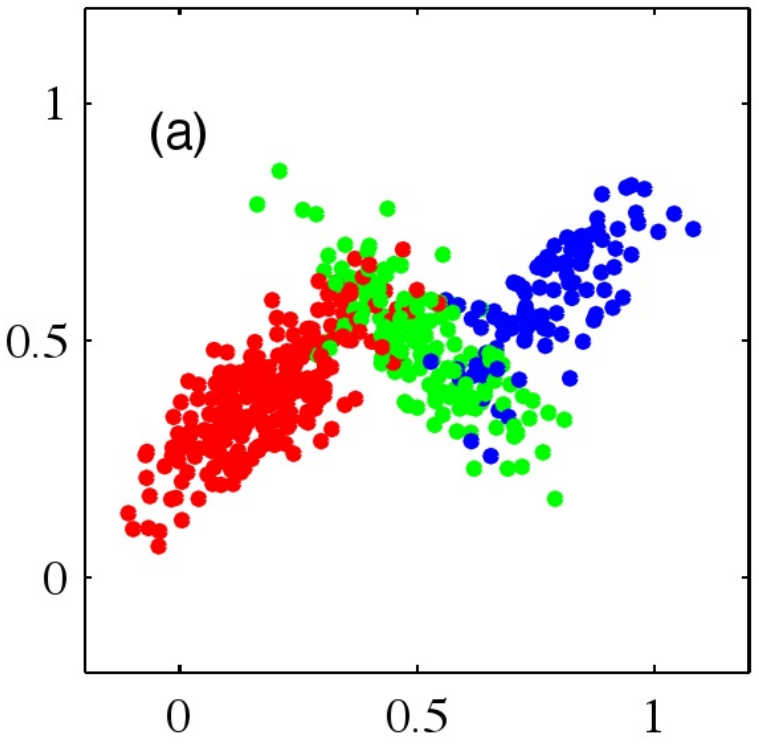
- **Key idea:** Model *each* region with a distinct distribution
- Can use Gaussians — Gaussian mixture models (GMMs)

Gaussian mixture models: intuition



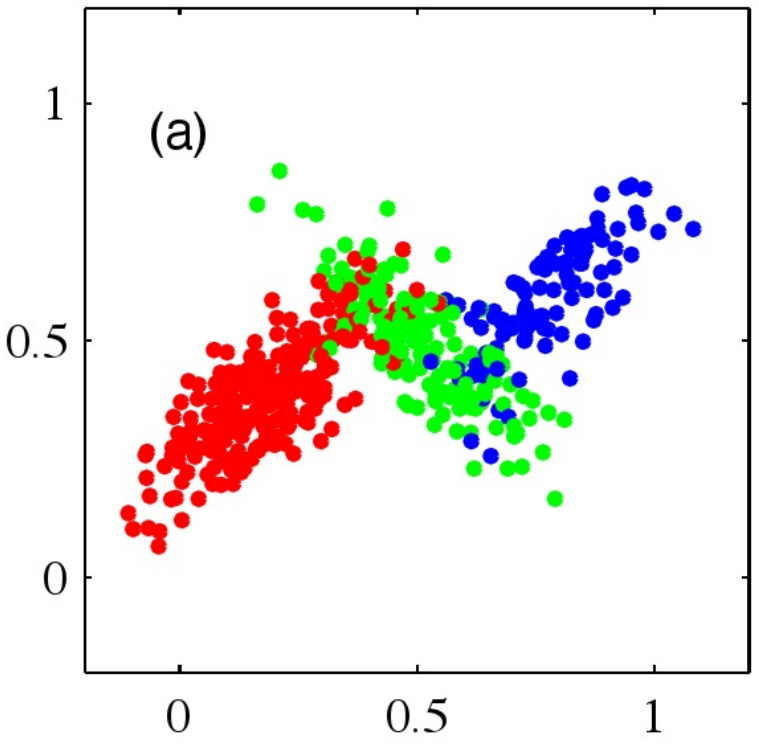
- **Key idea:** Model *each* region with a distinct distribution
- Can use Gaussians — Gaussian mixture models (GMMs)

Gaussian mixture models: intuition



- **Key idea:** Model *each* region with a distinct distribution
- Can use Gaussians — Gaussian mixture models (GMMs)
- *However*, we don't know *cluster assignments* (label), *parameters* of Gaussians, or *mixture components*!

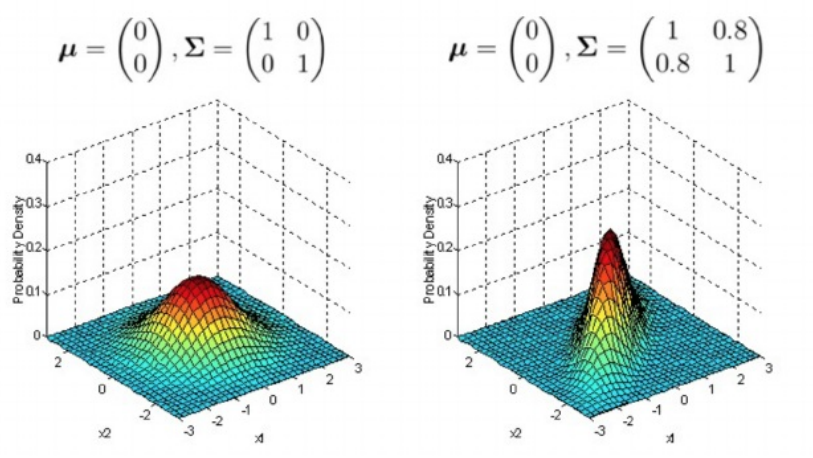
Gaussian mixture models: intuition



- **Key idea:** Model *each* region with a distinct distribution
- Can use Gaussians — Gaussian mixture models (GMMs)
- *However*, we don't know *cluster assignments* (label), *parameters* of Gaussians, or *mixture components*!
- Must learn from *unlabeled* data $\mathcal{D} = \{\boldsymbol{x}_n\}_{n=1}^{N}$

Recall: Gaussian (normal) distributions

$$\mathbf{x} \sim N(\boldsymbol{\mu}, \boldsymbol{\Sigma})$$



Gaussian mixture models: formal definition

GMM has the following density function for $\boldsymbol{x}$

$$p(\boldsymbol{x}) = \sum_{k=1}^{K} \omega_k N(\boldsymbol{x}|\boldsymbol{\mu}_k, \boldsymbol{\Sigma}_k)$$

- K: number of Gaussians — they are called mixture components

Gaussian mixture models: formal definition

GMM has the following density function for $\boldsymbol{x}$

$$p(\boldsymbol{x}) = \sum_{k=1}^{K} \omega_k N(\boldsymbol{x}|\boldsymbol{\mu}_k, \boldsymbol{\Sigma}_k)$$

- K: number of Gaussians — they are called mixture components
- $\boldsymbol{\mu}_k$ and $\boldsymbol{\Sigma}_k$: mean and covariance matrix of k-th component

Gaussian mixture models: formal definition

GMM has the following density function for $\boldsymbol{x}$

$$p(\boldsymbol{x}) = \sum_{k=1}^{K} \omega_k N(\boldsymbol{x}|\boldsymbol{\mu}_k, \boldsymbol{\Sigma}_k)$$

- K: number of Gaussians — they are called mixture components
- $\boldsymbol{\mu}_k$ and $\boldsymbol{\Sigma}_k$: mean and covariance matrix of k-th component
- ω_k: mixture weights (or priors) represent how much each component contributes to final distribution. They satisfy 2 properties:

$$\forall\ k,\ \omega_k > 0, \quad \text{and} \quad \sum_k \omega_k = 1$$

These properties ensure $p(\boldsymbol{x})$ is in fact a probability density function

GMM as the marginal distribution of a joint distribution

Consider the following joint distribution

$$p(\boldsymbol{x}, z) = p(z)p(\boldsymbol{x}|z)$$

where z is a discrete random variable taking values between 1 and K.

GMM as the marginal distribution of a joint distribution

Consider the following joint distribution

$$p(\boldsymbol{x}, z) = p(z)p(\boldsymbol{x}|z)$$

where z is a discrete random variable taking values between 1 and K.

Denote

$$\omega_k = p(z = k)$$

Now, assume the conditional distributions are Gaussian distributions

$$p(\boldsymbol{x}|z = k) = N(\boldsymbol{x}|\boldsymbol{\mu}_k, \boldsymbol{\Sigma}_k)$$

GMM as the marginal distribution of a joint distribution

Consider the following joint distribution

$$p(\boldsymbol{x}, z) = p(z)p(\boldsymbol{x}|z) \quad \Rightarrow \quad p(\boldsymbol{x}) = \sum_{z=1}^{K} p(\boldsymbol{x}, z)$$

where z is a discrete random variable taking values between 1 and K.

Denote

$$\omega_k = p(z = k)$$

Now, assume the conditional distributions are Gaussian distributions

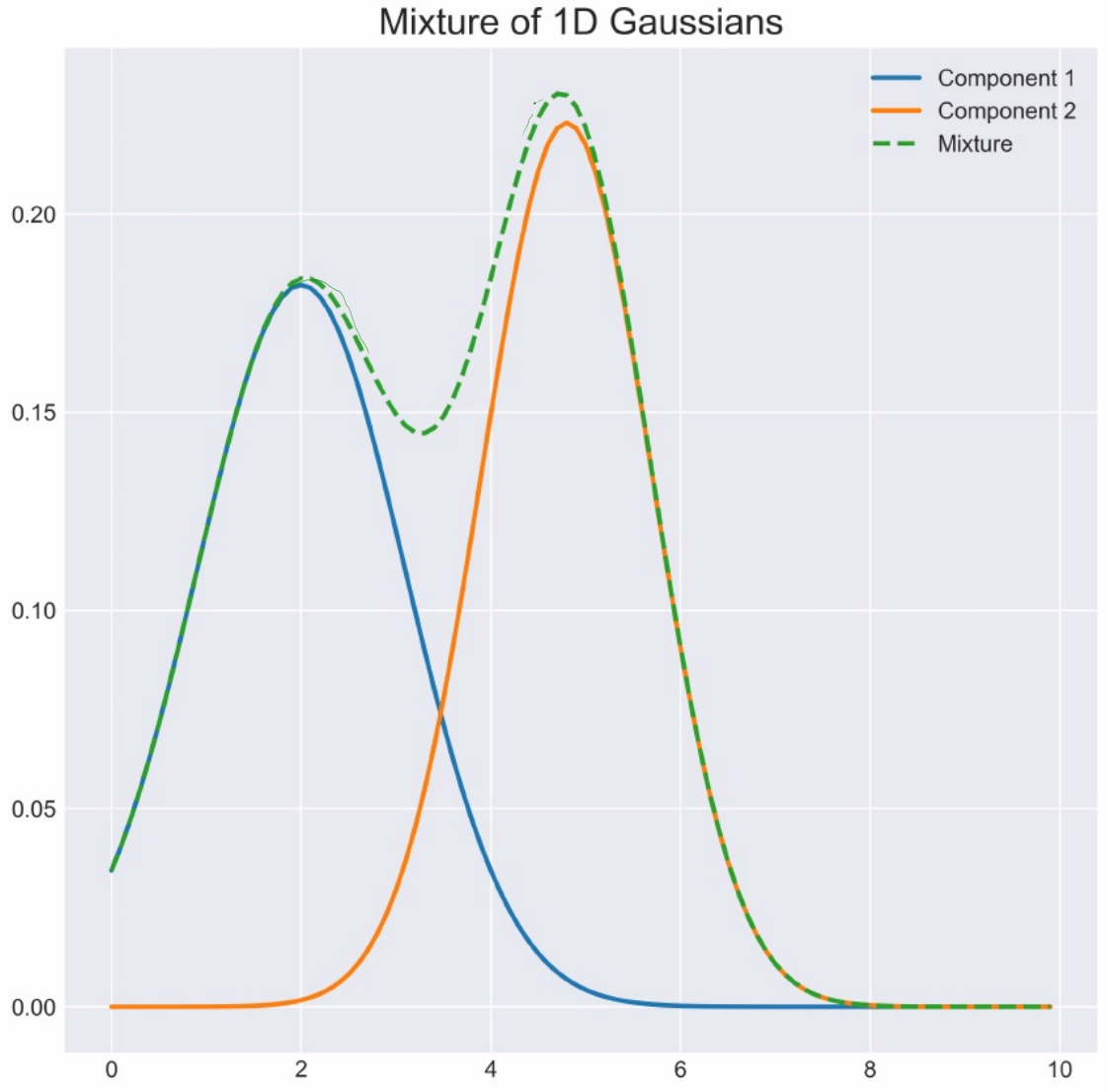
$$p(\boldsymbol{x}|z = k) = N(\boldsymbol{x}|\boldsymbol{\mu}_k, \boldsymbol{\Sigma}_k)$$

Then, the marginal distribution of $\boldsymbol{x}$ is

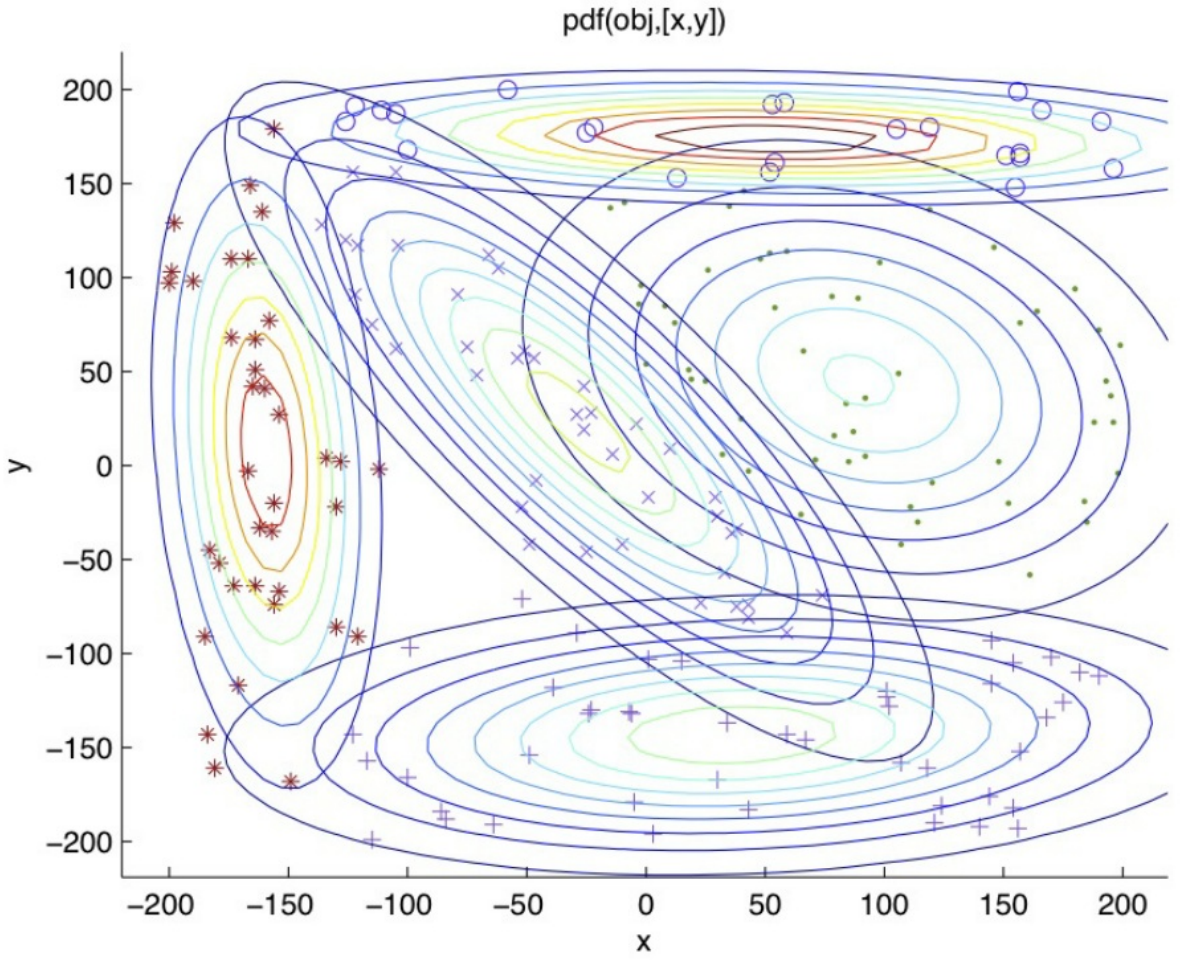
$$p(\boldsymbol{x}) = \sum_{k=1}^{K} \omega_k N(\boldsymbol{x}|\boldsymbol{\mu}_k, \boldsymbol{\Sigma}_k)$$

Namely, the Gaussian mixture model

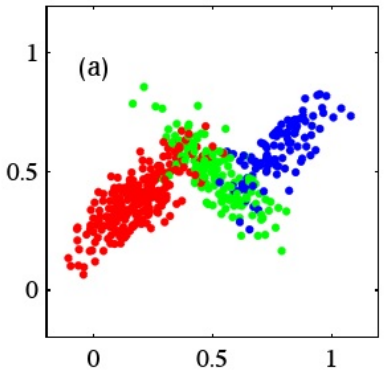
Gaussian mixtures in 1D



Gaussian mixture model for clustering



GMMs: example



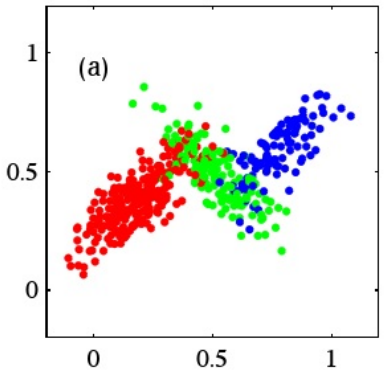
The conditional distribution between $\boldsymbol{x}$ and z (representing color) are

$$p(\boldsymbol{x}|z = red) = N(\boldsymbol{x}|\boldsymbol{\mu}_1, \boldsymbol{\Sigma}_1)$$

$$p(\boldsymbol{x}|z = blue) = N(\boldsymbol{x}|\boldsymbol{\mu}_2, \boldsymbol{\Sigma}_2)$$

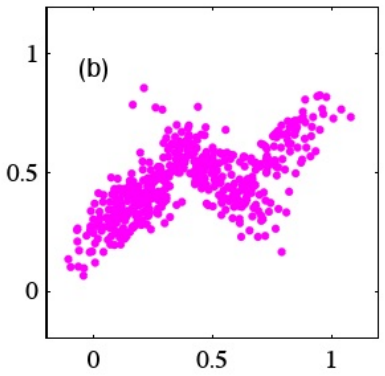
$$p(\boldsymbol{x}|z = green) = N(\boldsymbol{x}|\boldsymbol{\mu}_3, \boldsymbol{\Sigma}_3)$$

GMMs: example



The conditional distribution between $\boldsymbol{x}$ and z (representing color) are

$$p(\boldsymbol{x}|z = red) = N(\boldsymbol{x}|\boldsymbol{\mu}_1, \boldsymbol{\Sigma}_1)$$
$$p(\boldsymbol{x}|z = blue) = N(\boldsymbol{x}|\boldsymbol{\mu}_2, \boldsymbol{\Sigma}_2)$$
$$p(\boldsymbol{x}|z = green) = N(\boldsymbol{x}|\boldsymbol{\mu}_3, \boldsymbol{\Sigma}_3)$$



The marginal distribution is thus

$$p(\boldsymbol{x}) = p(red)N(\boldsymbol{x}|\boldsymbol{\mu}_1, \boldsymbol{\Sigma}_1) + p(blue)N(\boldsymbol{x}|\boldsymbol{\mu}_2, \boldsymbol{\Sigma}_2) + p(green)N(\boldsymbol{x}|\boldsymbol{\mu}_3, \boldsymbol{\Sigma}_3)$$

Parameter estimation for Gaussian mixture models

The parameters in GMMs are:

Parameter estimation for Gaussian mixture models

The parameters in GMMs are:

$$\boldsymbol{\theta} = \{\omega_k, \boldsymbol{\mu}_k, \boldsymbol{\Sigma}_k\}_{k=1}^{K}$$

Let's first consider the simple/unrealistic case where *we have labels z*

Define $\mathcal{D}' = \{\boldsymbol{x}_n, z_n\}_{n=1}^{N}$, $\mathcal{D} = \{\boldsymbol{x}_n\}_{n=1}^{N}$

- $\mathcal{D}'$ is the **complete** data
- $\mathcal{D}$ the **incomplete** data

How can we learn our parameters?

Parameter estimation for Gaussian mixture models

The parameters in GMMs are:

$$\boldsymbol{\theta} = \{\omega_k, \boldsymbol{\mu}_k, \boldsymbol{\Sigma}_k\}_{k=1}^{K}$$

Let's first consider the simple/unrealistic case where *we have labels z*

Define $\mathcal{D}' = \{\boldsymbol{x}_n, z_n\}_{n=1}^{N}$, $\mathcal{D} = \{\boldsymbol{x}_n\}_{n=1}^{N}$

- $\mathcal{D}'$ is the **complete** data
- $\mathcal{D}$ the **incomplete** data

$$p(x) = \sum_{z=1}^{K} p(x, z)$$

How can we learn our parameters?

Given $\mathcal{D}'$, the maximum likelihood estimation of the $\boldsymbol{\theta}$ is given by

$$\boldsymbol{\theta} = \arg\max \log \mathcal{D}' = \sum_{n} \log p(\boldsymbol{x}_n, z_n)$$

$$\log p(\mathcal{D}'; \theta) = \log \prod_{n=1}^{N} p(x_n, z_n; \theta)$$

Parameter estimation for GMMs: complete data

The complete likelihood is decomposable

$$\sum_n \log p(\boldsymbol{x}_n, z_n) = \sum_n \log p(z_n)p(\boldsymbol{x}_n|z_n) = \sum_k \sum_{n:z_n=k} \log p(z_n)p(\boldsymbol{x}_n|z_n)$$

where we have grouped data by cluster labels z_n.

Parameter estimation for GMMs: complete data

The complete likelihood is decomposable

$$\sum_n \log p(\boldsymbol{x}_n, z_n) = \sum_n \log p(z_n)p(\boldsymbol{x}_n|z_n) = \sum_k \sum_{n:z_n=k} \log p(z_n)p(\boldsymbol{x}_n|z_n)$$

where we have grouped data by cluster labels z_n.

Let $\gamma_{nk} \in \{0, 1\}$ be a binary variable that indicates whether $z_n = k$:

Parameter estimation for GMMs: complete data

The complete likelihood is decomposable

$$\sum_n \log p(\boldsymbol{x}_n, z_n) = \sum_n \log p(z_n)p(\boldsymbol{x}_n|z_n) = \sum_k \sum_{n:z_n=k} \log p(z_n)p(\boldsymbol{x}_n|z_n)$$

where we have grouped data by cluster labels z_n.

Let $\gamma_{nk} \in \{0, 1\}$ be a binary variable that indicates whether $z_n = k$:

$$\sum_n \log p(\boldsymbol{x}_n, z_n) = \sum_k \sum_n \gamma_{nk} \log p(z = k)p(\boldsymbol{x}_n|z = k)$$

Parameter estimation for GMMs: complete data

The complete likelihood is decomposable

$$\sum_n \log p(\boldsymbol{x}_n, z_n) = \sum_n \log p(z_n) p(\boldsymbol{x}_n | z_n) = \sum_k \sum_{n:z_n=k} \log p(z_n) p(\boldsymbol{x}_n | z_n)$$

where we have grouped data by cluster labels z_n.

Let $\gamma_{nk} \in \{0, 1\}$ be a binary variable that indicates whether $z_n = k$:

$$\begin{aligned}\sum_n \log p(\boldsymbol{x}_n, z_n) &= \sum_k \sum_n \gamma_{nk} \log p(z = k) p(\boldsymbol{x}_n | z = k) \\ &= \sum_k \sum_n \gamma_{nk} \left[\log \omega_k + \log N(\boldsymbol{x}_n | \boldsymbol{\mu}_k, \boldsymbol{\Sigma}_k)\right]\end{aligned}$$

Note: in the complete setting the γ_{nk} just add to the notation, but later we will 'relax' these variables and allow them to take on fractional values

Parameter estimation for GMMs: complete data

From our previous discussion, we have

$$\sum_n \log p(\boldsymbol{x}_n, z_n) = \sum_k \sum_n \gamma_{nk} \left[\log \omega_k + \log N(\boldsymbol{x}_n | \boldsymbol{\mu}_k, \boldsymbol{\Sigma}_k)\right]$$

Parameter estimation for GMMs: complete data

From our previous discussion, we have

$$\sum_n \log p(\boldsymbol{x}_n, z_n) = \sum_k \sum_n \gamma_{nk} \left[\log \omega_k + \log N(\boldsymbol{x}_n | \boldsymbol{\mu}_k, \boldsymbol{\Sigma}_k)\right]$$

Regrouping, we have

$$\sum_n \log p(\boldsymbol{x}_n, z_n) = \sum_k \sum_n \gamma_{nk} \log \omega_k + \sum_k \left\{ \sum_n \gamma_{nk} \log N(\boldsymbol{x}_n | \boldsymbol{\mu}_k, \boldsymbol{\Sigma}_k) \right\}$$

Parameter estimation for GMMs: complete data

From our previous discussion, we have

$$\sum_n \log p(\boldsymbol{x}_n, z_n) = \sum_k \sum_n \gamma_{nk} \left[\log \omega_k + \log N(\boldsymbol{x}_n | \boldsymbol{\mu}_k, \boldsymbol{\Sigma}_k)\right]$$

Regrouping, we have

$$\ell(\theta) = \sum_n \log p(\boldsymbol{x}_n, z_n) = \sum_k \sum_n \gamma_{nk} \log \omega_k + \sum_k \left\{ \sum_n \gamma_{nk} \log N(\boldsymbol{x}_n | \boldsymbol{\mu}_k, \boldsymbol{\Sigma}_k) \right\} + \lambda \left(\sum_{k=1}^{K} \omega_k - 1 \right)$$

$\frac{\partial \ell}{\partial \omega_k} =$

The term inside the braces depends on k-th component's parameters. It is now easy to show that (left as an exercise) the MLE is:

HW: Check!

How to compute ω_k?

$$\omega_k = \frac{\sum_n \gamma_{nk}}{\sum_k \sum_n \gamma_{nk}}, \quad \boldsymbol{\mu}_k = \frac{1}{\sum_n \gamma_{nk}} \sum_n \gamma_{nk} \boldsymbol{x}_n$$

$$\boldsymbol{\Sigma}_k = \frac{1}{\sum_n \gamma_{nk}} \sum_n \gamma_{nk} (\boldsymbol{x}_n - \boldsymbol{\mu}_k)(\boldsymbol{x}_n - \boldsymbol{\mu}_k)^\top$$

What's the intuition?

$$\frac{\partial \ell}{\partial \omega_k} = \sum_n \gamma_{nk} \frac{1}{\omega_k} + \lambda = 0 \Rightarrow \sum_k \sum_n \gamma_{nk} + \sum_k \lambda \omega_k = 0$$

$$\Rightarrow \omega_k = \frac{1}{N} \sum_n \gamma_{nk} \qquad \Rightarrow \lambda = -N$$

Intuition

Since γ_{nk} is binary, the previous solution is nothing but:

- ω_k: fraction of total data points whose cluster label z_n is k
  - note that $\sum_k \sum_n \gamma_{nk} = N$
- $\boldsymbol{\mu}_k$: mean of all data points whose z_n is k
- $\boldsymbol{\Sigma}_k$: covariance of all data points whose z_n is k

Intuition

Since γ_{nk} is binary, the previous solution is nothing but:

- ω_k: fraction of total data points whose cluster label z_n is k
  - note that $\sum_k \sum_n \gamma_{nk} = N$
- $\boldsymbol{\mu}_k$: mean of all data points whose z_n is k
- $\boldsymbol{\Sigma}_k$: covariance of all data points whose z_n is k

Recall that this depends on us knowing the true cluster labels z_n

This intuition will help us develop an algorithm for estimating $\boldsymbol{\theta}$ when we *do not* know z_n (incomplete data)

GMMs and Incomplete Data

Parameter estimation for GMMs: Incomplete data

GMM Parameters

$$\boldsymbol{\theta} = \{\omega_k, \boldsymbol{\mu}_k, \boldsymbol{\Sigma}_k\}_{k=1}^{K}$$

Incomplete Data

Our data contains observed and unobserved data, and hence is incomplete

- Observed: $\mathcal{D} = \{\boldsymbol{x}_n\}$
- Unobserved (hidden): $\{\boldsymbol{z}_n\}$

Parameter estimation for GMMs: Incomplete data

GMM Parameters

$$\boldsymbol{\theta} = \{\omega_k, \boldsymbol{\mu}_k, \boldsymbol{\Sigma}_k\}_{k=1}^{K}$$

Incomplete Data

Our data contains observed and unobserved data, and hence is incomplete

- Observed: $\mathcal{D} = \{\boldsymbol{x}_n\}$
- Unobserved (hidden): $\{\boldsymbol{z}_n\}$

Goal Obtain the maximum likelihood estimate of $\boldsymbol{\theta}$:

$$\boldsymbol{\theta} = \arg\max \ell(\boldsymbol{\theta}) = \arg\max \log \mathcal{D} = \arg\max \sum_n \log p(\boldsymbol{x}_n; \boldsymbol{\theta})$$

$$\arg\max \log \prod_{n=1}^{N} p(x_n; \theta)$$

Parameter estimation for GMMs: Incomplete data

GMM Parameters

$$\boldsymbol{\theta} = \{\omega_k, \boldsymbol{\mu}_k, \boldsymbol{\Sigma}_k\}_{k=1}^{K}$$

Incomplete Data

Our data contains observed and unobserved data, and hence is incomplete

- Observed: $\mathcal{D} = \{\boldsymbol{x}_n\}$
- Unobserved (hidden): $\{\boldsymbol{z}_n\}$

Goal Obtain the maximum likelihood estimate of $\boldsymbol{\theta}$:

$$\boldsymbol{\theta} = \arg\max \ell(\boldsymbol{\theta}) = \arg\max \log \mathcal{D} = \arg\max \sum_n \log p(\boldsymbol{x}_n|\boldsymbol{\theta})$$

$$= \arg\max \sum_n \log \sum_{\boldsymbol{z}_n} p(\boldsymbol{x}_n, \boldsymbol{z}_n|\boldsymbol{\theta})$$

The objective function $\ell(\boldsymbol{\theta})$ is called the *incomplete* log-likelihood.

Issue with Incomplete log-likelihood

No simple way to optimize the incomplete log-likelihood (exercise: try to take derivative with respect to parameters, set it to zero and solve)

Issue with Incomplete log-likelihood

No simple way to optimize the incomplete log-likelihood (exercise: try to take derivative with respect to parameters, set it to zero and solve)

EM algorithm provides a strategy for iteratively optimizing this function

Issue with Incomplete log-likelihood

No simple way to optimize the incomplete log-likelihood (exercise: try to take derivative with respect to parameters, set it to zero and solve)

EM algorithm provides a strategy for iteratively optimizing this function

Two steps as they apply to GMM:

- E-step: 'guess' values of the z_n using existing values of $\boldsymbol{\theta}$
- M-step: solve for new values of $\boldsymbol{\theta}$ given imputed values for z_n (i.e., maximize complete likelihood!)

E-step: Soft cluster assignments

We define γ_{nk} as $p(z_n = k|\boldsymbol{x}_n, \boldsymbol{\theta})$

- This is the posterior distribution of z_n given $\boldsymbol{x}_n$ and $\boldsymbol{\theta}$

E-step: Soft cluster assignments

We define γ_{nk} as $p(z_n = k|\boldsymbol{x}_n, \boldsymbol{\theta})$

- This is the posterior distribution of z_n given $\boldsymbol{x}_n$ and $\boldsymbol{\theta}$
- Recall that in complete data setting γ_{nk} was binary

E-step: Soft cluster assignments

We define γ_{nk} as $p(z_n = k|\boldsymbol{x}_n, \boldsymbol{\theta})$

- This is the posterior distribution of z_n given $\boldsymbol{x}_n$ and $\boldsymbol{\theta}$
- Recall that in complete data setting γ_{nk} was binary
- Now it's a "soft" assignment of $\boldsymbol{x}_n$ to k-th component, with $\boldsymbol{x}_n$ assigned to each component with some probability

E-step: Soft cluster assignments

We define γ_{nk} as $p(z_n = k|\boldsymbol{x}_n, \boldsymbol{\theta})$

- This is the posterior distribution of z_n given $\boldsymbol{x}_n$ and $\boldsymbol{\theta}$
- Recall that in complete data setting γ_{nk} was binary
- Now it's a "soft" assignment of $\boldsymbol{x}_n$ to k-th component, with $\boldsymbol{x}_n$ assigned to each component with some probability

E-step: Soft cluster assignments

We define γ_{nk} as $p(z_n = k|\boldsymbol{x}_n, \boldsymbol{\theta})$

- This is the posterior distribution of z_n given $\boldsymbol{x}_n$ and $\boldsymbol{\theta}$
- Recall that in complete data setting γ_{nk} was binary
- Now it's a "soft" assignment of $\boldsymbol{x}_n$ to k-th component, with $\boldsymbol{x}_n$ assigned to each component with some probability

Given an estimate of $\boldsymbol{\theta} = \{\omega_k, \boldsymbol{\mu}_k, \boldsymbol{\Sigma}_k\}_{k=1}^K$, we can compute γ_{nk} as follows:

$$\begin{aligned}\gamma_{nk} &= p(z_n = k|\boldsymbol{x}_n) \\ &= \frac{p(\boldsymbol{x}_n|z_n = k)p(z_n = k)}{p(\boldsymbol{x}_n)}\end{aligned}$$

P(A|B) = P(B|A) * P(A) / P(B)

E-step: Soft cluster assignments

We define γ_{nk} as $p(z_n = k|\boldsymbol{x}_n, \boldsymbol{\theta})$

- This is the posterior distribution of z_n given $\boldsymbol{x}_n$ and $\boldsymbol{\theta}$
- Recall that in complete data setting γ_{nk} was binary
- Now it's a "soft" assignment of $\boldsymbol{x}_n$ to k-th component, with $\boldsymbol{x}_n$ assigned to each component with some probability

Given an estimate of $\boldsymbol{\theta} = \{\omega_k, \boldsymbol{\mu}_k, \boldsymbol{\Sigma}_k\}_{k=1}^K$, we can compute γ_{nk} as follows:

$$\begin{aligned}
\gamma_{nk} &= p(z_n = k|\boldsymbol{x}_n) \\
&= \frac{p(\boldsymbol{x}_n|z_n = k)p(z_n = k)}{p(\boldsymbol{x}_n)} \\
&= \frac{p(\boldsymbol{x}_n|z_n = k)p(z_n = k)}{\sum_{k'=1}^K p(\boldsymbol{x}_n|z_n = k')p(z_n = k')}
\end{aligned}$$

M-step: Maximimize complete likelihood

Recall definition of complete likelihood from earlier:

$$\sum_n \log p(\boldsymbol{x}_n, z_n) = \sum_k \sum_n \gamma_{nk} \log \omega_k + \sum_k \left\{ \sum_n \gamma_{nk} \log N(\boldsymbol{x}_n | \boldsymbol{\mu}_k, \boldsymbol{\Sigma}_k) \right\}$$

Previously γ_{nk} was binary, but now we define $\gamma_{nk} = p(z_n = k | \boldsymbol{x}_n)$ (E-step)

M-step: Maximimize complete likelihood

Recall definition of complete likelihood from earlier:

$$\sum_n \log p(\boldsymbol{x}_n, z_n) = \sum_k \sum_n \gamma_{nk} \log \omega_k + \sum_k \left\{ \sum_n \gamma_{nk} \log N(\boldsymbol{x}_n | \boldsymbol{\mu}_k, \boldsymbol{\Sigma}_k) \right\}$$

Previously γ_{nk} was binary, but now we define $\gamma_{nk} = p(z_n = k | \boldsymbol{x}_n)$ (E-step)

We get the same simple expression for the MLE as before!

$$\omega_k = \frac{\sum_n \gamma_{nk}}{\sum_k \sum_n \gamma_{nk}}, \quad \boldsymbol{\mu}_k = \frac{1}{\sum_n \gamma_{nk}} \sum_n \gamma_{nk} \boldsymbol{x}_n$$

$$\boldsymbol{\Sigma}_k = \frac{1}{\sum_n \gamma_{nk}} \sum_n \gamma_{nk} (\boldsymbol{x}_n - \boldsymbol{\mu}_k)(\boldsymbol{x}_n - \boldsymbol{\mu}_k)^\top$$

M-step: Maximimize complete likelihood

Recall definition of complete likelihood from earlier:

$$\sum_n \log p(\boldsymbol{x}_n, z_n) = \sum_k \sum_n \gamma_{nk} \log \omega_k + \sum_k \left\{ \sum_n \gamma_{nk} \log N(\boldsymbol{x}_n | \boldsymbol{\mu}_k, \boldsymbol{\Sigma}_k) \right\}$$

Previously γ_{nk} was binary, but now we define $\gamma_{nk} = p(z_n = k | \boldsymbol{x}_n)$ (E-step)

We get the same simple expression for the MLE as before!

$$\omega_k = \frac{\sum_n \gamma_{nk}}{\sum_k \sum_n \gamma_{nk}}, \quad \boldsymbol{\mu}_k = \frac{1}{\sum_n \gamma_{nk}} \sum_n \gamma_{nk} \boldsymbol{x}_n$$

$$\boldsymbol{\Sigma}_k = \frac{1}{\sum_n \gamma_{nk}} \sum_n \gamma_{nk} (\boldsymbol{x}_n - \boldsymbol{\mu}_k)(\boldsymbol{x}_n - \boldsymbol{\mu}_k)^\top$$

Intuition: Each point now contributes some fractional component to each of the parameters, with weights determined by γ_{nk}

EM procedure for GMM

Alternate between estimating γ_{nk} and estimating θ

- Initialize $\boldsymbol{\theta}$ with some values (random or otherwise)
- Repeat
  - E-Step: Compute γ_{nk} using the current $\boldsymbol{\theta}$
  - M-Step: Update $\boldsymbol{\theta}$ using the γ_{nk} we just computed
- Until Convergence

EM procedure for GMM

Alternate between estimating γ_{nk} and estimating $\boldsymbol{\theta}$

- Initialize $\boldsymbol{\theta}$ with some values (random or otherwise)
- Repeat
  - E-Step: Compute γ_{nk} using the current $\boldsymbol{\theta}$
  - M-Step: Update $\boldsymbol{\theta}$ using the γ_{nk} we just computed
- Until Convergence

Questions to be answered next

- How does GMM relate to K-means?
- Is this procedure reasonable, i.e., are we optimizing a sensible criterion?
- Will this procedure converge?

GMMs and K-means

GMMs provide probabilistic interpretation for K-means

GMMs and K-means

GMMs provide probabilistic interpretation for K-means

HW:

GMMs reduce to K-means under the following assumptions (in which case EM for GMM parameter estimation simplifies to K-means):

- Assume all Gaussians have $\sigma^2 \boldsymbol{I}$ covariance matrices
- Further assume $\sigma \to 0$, so we only need to estimate $\boldsymbol{\mu}_k$, i.e., means

K-means is often called "hard" GMM or GMMs is called "soft" K-means

The posterior γ_{nk} provides a probabilistic assignment for $\boldsymbol{x}_n$ to cluster k

$$\gamma_{nk} = p(z_n = k \mid \underline{x}_n ; \theta)$$

$$= \frac{p(\underline{x}_n \mid z_n = k)\, p(z_n = k)}{p(\underline{x}_n)}$$

$$= \frac{\omega_k \, \mathcal{N}(\underline{x}_n ; \underline{\mu}_k, \Sigma_k)}{\sum_{j=1}^{K} \omega_j \, \mathcal{N}(\underline{x}_n ; \underline{\mu}_j, \Sigma_j)}$$

$$= \frac{\omega_k \frac{1}{(2\pi)^{d/2} |\Sigma_k|^{1/2}} \exp\left\{ -\frac{1}{2} (\underline{x}_n - \underline{\mu}_k)^T \Sigma_k^{-1} (\underline{x}_n - \underline{\mu}_k) \right\}}{\sum_{j=1}^{K} \omega_j \frac{1}{(2\pi)^{d/2} |\Sigma_j|^{1/2}} \exp\left\{ -\frac{1}{2} (\underline{x}_n - \underline{\mu}_j)^T \Sigma_j^{-1} (\underline{x}_n - \underline{\mu}_j) \right\}}$$

Substituting $\Sigma_k = \sigma^2 I$

$$\gamma_{nk} = \frac{\omega_k \exp\left\{ -\frac{1}{2\sigma^2} \|x_n - \mu_k\|^2 \right\}}{\sum_{j=1}^{K} \omega_j \exp\left\{ -\frac{1}{2\sigma^2} \|x_n - x_j\|^2 \right\}} = \frac{\omega_k}{\sum_{j=1}^{K} \omega_j \exp\left\{ -\frac{1}{2\sigma^2} \left(\|x_n - \mu_j\| - \|x_n - \mu_k\|^2 \right) \right\}}$$

if $\sigma \to 0$, $\gamma_{nk} \to$ $\begin{cases} 1 & \text{if } \underline{x}_n \text{ is closest to } \mu_k \\ 0 & \text{if } \underline{x}_n \text{ is not closest to } \mu_k \end{cases}$ | K-means labels

GMMs vs. k-means

Pros/Cons

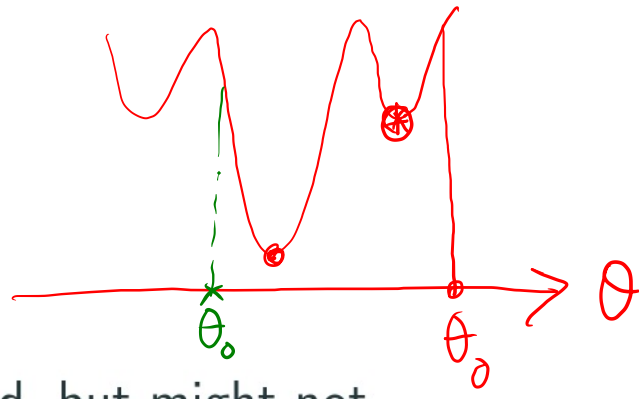
- k-means is a simpler, more straightforward method, but might not be as accurate because of deterministic clustering

GMMs vs. k-means

Pros/Cons

- k-means is a simpler, more straightforward method, but might not be as accurate because of deterministic clustering
- GMMs can be more accurate, as they model more information (soft clustering, variance), but can be more expensive to compute

GMMs vs. k-means



Pros/Cons

- k-means is a simpler, more straightforward method, but might not be as accurate because of deterministic clustering
- GMMs can be more accurate, as they model more information (soft clustering, variance), but can be more expensive to compute
- Both methods have a similar set of practical issues (having to select k, the distance, and the initialization)