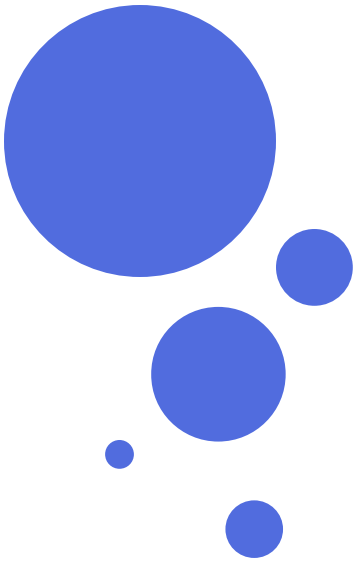
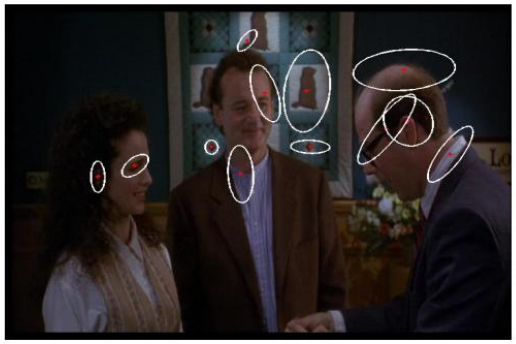
Rensselaer

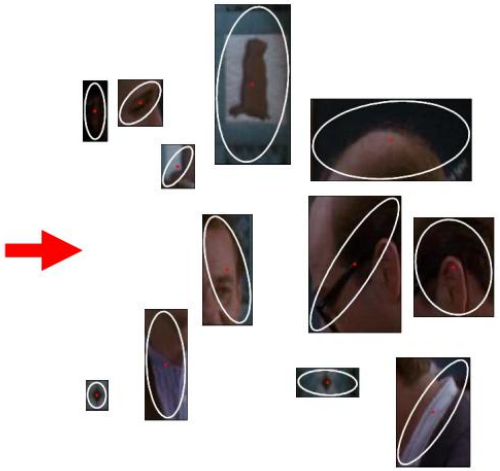


Lecture 22: Gaussian Mixture Model and Expectation Maximization Algorithm

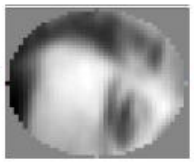
Dr. Chengjiang Long
Computer Vision Researcher at Kitware Inc.
Adjunct Professor at RPI.
Email: **longc3@rpi.edu**

Recap Previous Lecture





Collection of visual words

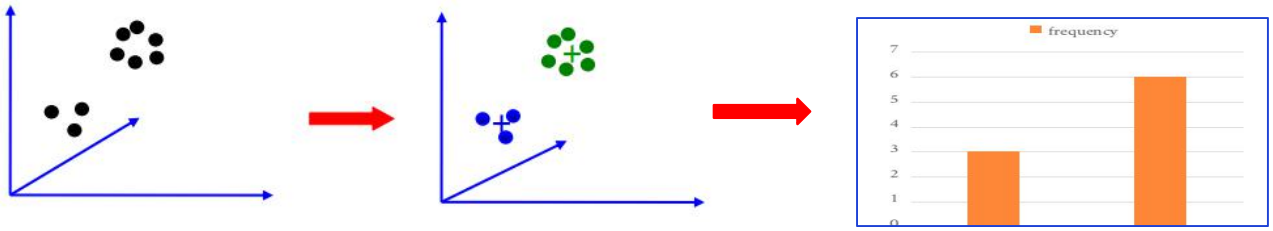


Normalize patch

Compute SIFT descriptor

[Lowe'99]

Vector quantize descriptors from a set of training images using k-means



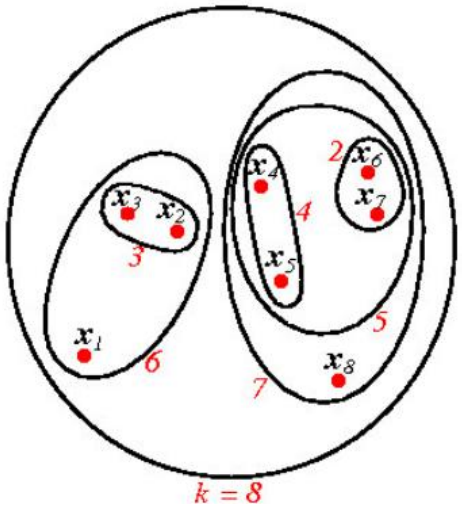


Image representation: a normalized histogram of visual words.

Outline

- Parametric Unsupervised Learning
- Mixture Density Model
- Gaussian Mixture Model
- Expectation Maximization (EM) Algorithm
- Applications

Outline

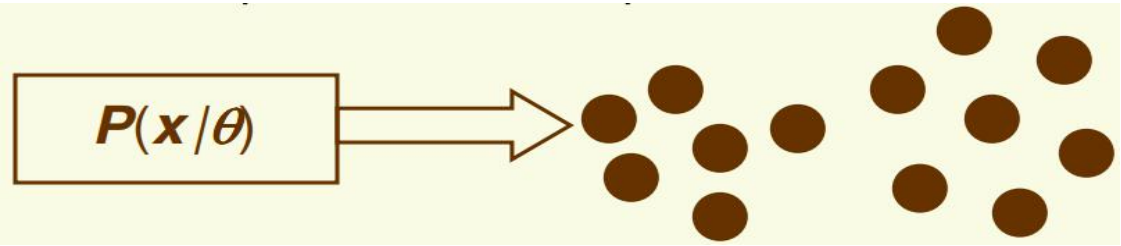
- **Parametric Unsupervised Learning**
- Mixture Density Model
- Gaussian Mixture Model
- Expectation Maximization (EM) Algorithm
- Applications

Unsupervised Learning

- In unsupervised learning, where we are only given samples $x_1, \ldots, x_n$ without class labels
- Last lecture: nonparametric approach (clustering)
- Today, parametric approach
  - assume parametric distribution of data
  - estimate parameters of this distribution
  - much "harder" than the supervised learning case

Parametric Unsupervised Learning

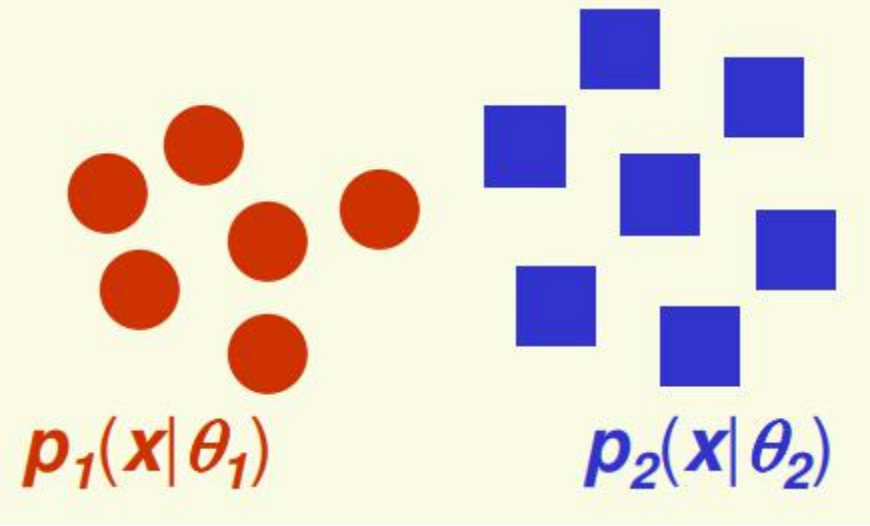
- Assume the data was generated by a model with known shape but unknown parameters



- Advantages of having a model
  - Gives a meaningful way to cluster data
    - adjust the parameters of the model to maximize the probability that the model produced the observed data
  - Can sensibly measure if a clustering is good
    - compute the likelihood of data induced by clustering
  - Can compare 2 clustering algorithms
    - which one gives the higher likelihood of the observed data?

Parametric Supervised Learning

- Let us recall supervised parametric learning
  - have m classes
  - have samples $x_1,\ldots, x_n$ each of class $1, 2,\ldots, m$
  - suppose D_i holds samples from class i
  - probability distribution for class i is $p_i(x|\theta_i)$

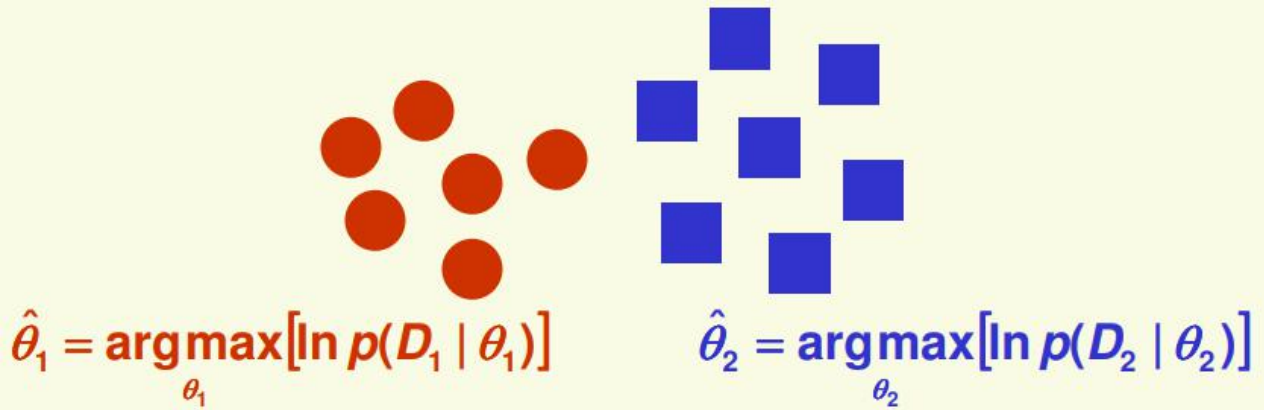


Parametric Supervised Learning

- Use the ML method to estimate parameters θ_i
  - find θ_i which maximizes the likelihood function $F(\theta_i)$
  - or, equivalently, find θ_i which maximizes the log likelihood $l(\theta_i)$

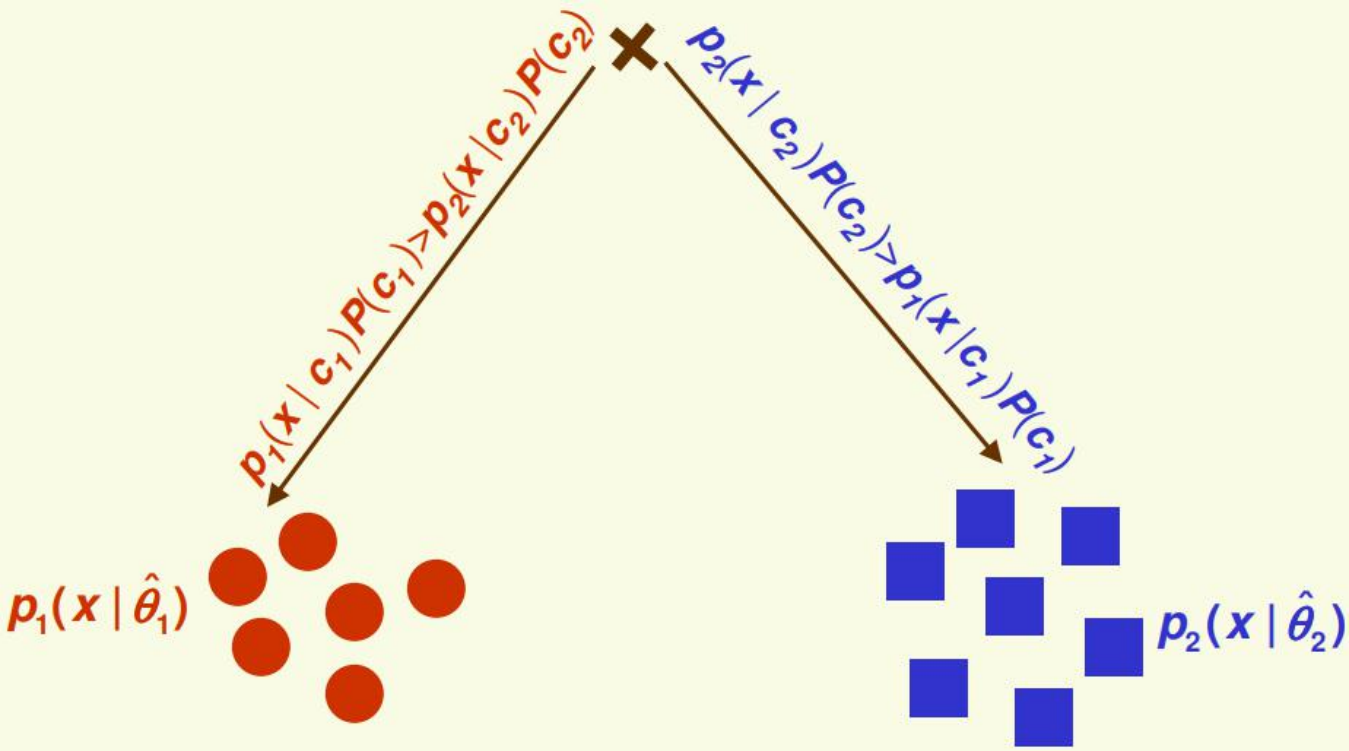
$$p(D_i \mid \theta_i) = \prod_{x \in D_i} p(x \mid \theta_i) = F(\theta_i)$$

$$l(\theta_i) = \ln p(D_i \mid \theta_i) = \sum_{x \in D_i} \ln p(x \mid \theta_i)$$



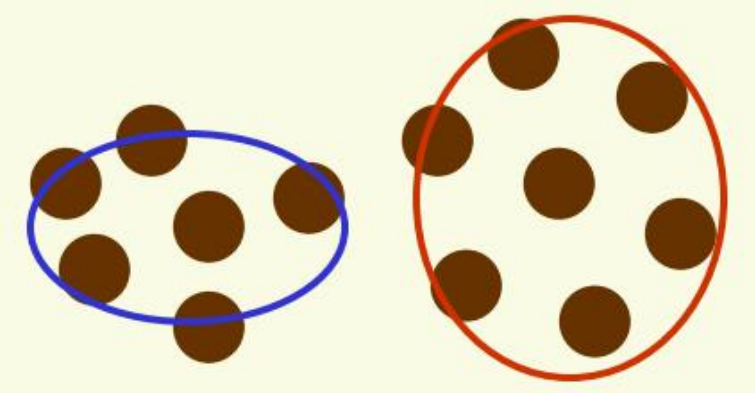
Parametric Supervised Learning

- now the distributions are fully specified
- can classify unknown sample using MAP classifier



Parametric Unsupervised Learning

- In unsupervised learning, no one tells us the true classes for samples. We still know
  - have m classes
  - have samples $x_1, \ldots, x_n$ each of unknown class
  - probability distribution for class i is $p_i(x|\theta_i)$
- Can we determine the classes and parameters simultaneously?



Outline

- Parametric Unsupervised Learning
- **Mixture Density Model**
- Gaussian Mixture Model
- Expectation Maximization (EM) Algorithm
- Applications

Mixture Density Model

- Model data with density model.

component densities

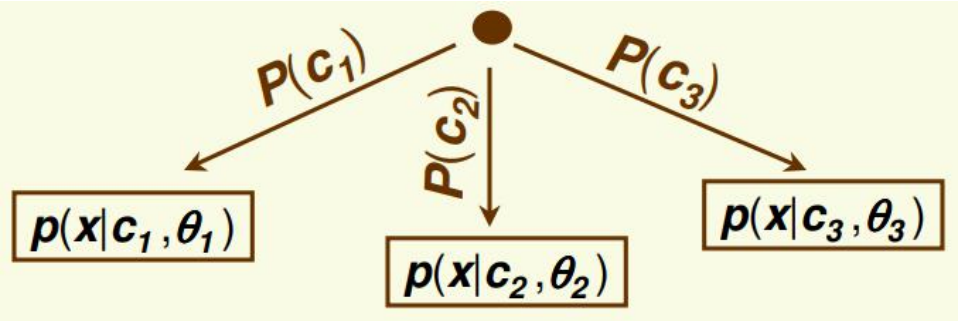
$$p(x \mid \theta) = \sum_{j=1}^{m} p(x \mid c_j, \theta_j) P(c_j)$$

mixing parameters

where $\theta = \{\theta_1, \ldots, \theta_m\}$

$$P(c_1) + P(c_2) + \ldots + P(c_m) = 1$$

- To generate a sample from distribution p(x|θ)
  - first select j with probability $p(c_j)$
  - then generate x according to probability law $p(x|c_j, \theta_j)$



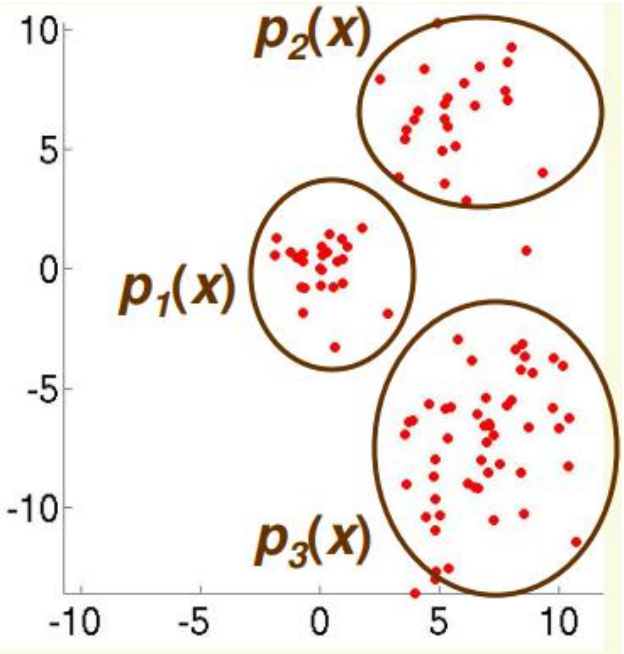
Example: Gaussian Mixture Density

- Mixture of 3 Gaussians

$$p_1(x) \cong N\left(\begin{bmatrix}0\\0\end{bmatrix}, \begin{bmatrix}1 & 0\\0 & 1\end{bmatrix}\right)$$

$$p_2(x) \cong N\left([6,6], \begin{bmatrix}4 & 0\\0 & 4\end{bmatrix}\right)$$

$$p_3(x) \cong N\left([7,-7], \begin{bmatrix}6 & 0\\0 & 6\end{bmatrix}\right)$$



$$p(x) = 0.2p_1(x) + 0.3p_2(x) + 0.5p_3(x)$$

Mixture Density

$$p(x \mid \theta) = \sum_{j=1}^{m} p(x \mid c_j, \theta_j) P(c_j)$$

- $P(c_1), \ldots, P(c_m)$ can be known or unknown
- Suppose we know how to estimate $\theta_1, \ldots, \theta_m$ and $P(c_1), \ldots, P(c_m)$
- Can "break apart" mixture p(x|θ) for classification
- To classify sample x, use MAP estimation, that is choose class i which maximizes

$$P(c_i \mid x, \theta_i) \propto \underbrace{p(x \mid c_i, \theta_i)}_{\text{probability of component } i \text{ to generate } x} \underbrace{P(c_i)}_{\text{probability of component } i}$$

ML Estimation for Mixture Density

$$p(x \mid \theta, \rho) = \sum_{j=1}^{m} p(x \mid c_j, \theta_j) P(c_j) = \sum_{j=1}^{m} p(x \mid c_j, \theta_j) \rho_i$$

- Can use Maximum Likelihood estimation for a mixture density; need to estimate

$$\theta_1, \ldots, \theta_m$$

$$\rho_1 = P(c_1), \ldots, \rho_m = P(c_m), \text{ and } \rho = \{\rho_1, \ldots, \rho_m\}$$

- As in the supervised case, form the logarithm likelihood function

$$l(\theta, \rho) = \ln p(D \mid \theta, \rho) = \sum_{k=1}^{n} \ln p(x_k \mid \theta, \rho) = \sum_{k=1}^{n} \ln \left[\sum_{j=1}^{m} p(x \mid c_j, \theta_j) \rho_i \right]$$

ML Estimation for Mixture Density

$$l(\theta,\rho)=\sum_{k=1}^{n}\ln\left[\sum_{j=1}^{m}p(x\mid c_j,\theta_j)\rho_i\right]$$

- need to maximize $l(\theta,\rho)$ with respect to θ and ρ
- As you may have guessed, $l(\theta,\rho)$ is not the easiest function to maximize
  - If we take partial derivatives with respect to θ, ρ and set them to 0, typically we have a "coupled" nonlinear system of equation
  - usually closed form solution cannot be found
- We could use the gradient ascent method in general, it is not the greatest method to use, should only be used as last resort
- There is a better algorithm, called Expectation Maximization (EM).

- Before EM, let's look at the mixture density again

$$p(x \mid \theta, \rho) = \sum_{j=1}^{m} p(x \mid c_j, \theta_j)\rho_j$$

- Suppose we know how to estimate θ and ρ
  - Estimating the class of x is easy with MAP, maximize

$$p(x \mid c_i, \theta_i)P(c_i) = p(x \mid c_i, \theta_i)\rho_i$$

- Suppose we know the class of samples $x_1, \ldots, x_n$
  - This is just the supervised learning case, so estimating θ and ρ is easy

$$\hat{\theta}_i = \arg\max_{\theta_i}[\ln p(D_i \mid \theta_i)] \qquad \hat{\rho}_i = \frac{|D_i|}{n}$$

- This is an example of chicken-and-egg problem
  - EM algorithm approaches this problem by adding "hidden" variables

Outline

- Parametric Unsupervised Learning
- Mixture Density Model
- **Gaussian Mixture Model**
- Expectation Maximization (EM) Algorithm
- Applications

What Model Should We Use?

- Depends on X!
- Here, maybe Gaussian Naïve Bayes?
  - – Multinomial over clusters Y
  - – (Independent) Gaussian for each Xi given Y

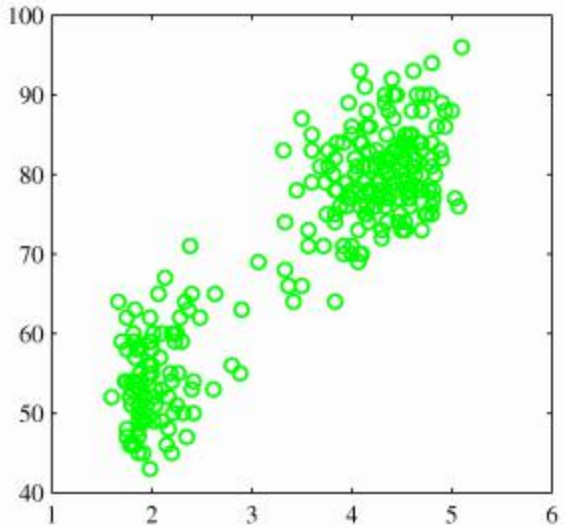
$$p(Y_i = y_k) = \theta_k$$

$$P(X_i = x \mid Y = y_k) = \frac{1}{\sigma_{ik}\sqrt{2\pi}} \; e^{\frac{-(x-\mu_{ik})^2}{2\sigma_{ik}^2}}$$

Y	X_1	X_2
??	0.1	2.1
??	0.5	-1.1
??	0.0	3.0
??	-0.1	-2.0
??	0.2	1.5
…	…	…

Could we make fewer assumptions?

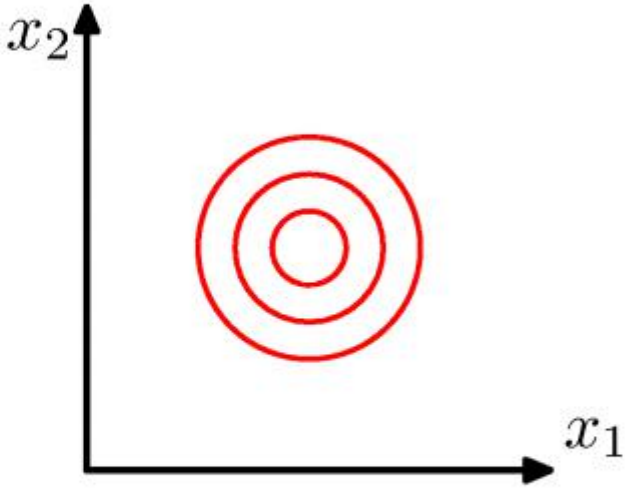
- What if the X_i co-vary?
- What if there are multiple peaks?
- Gaussian Mixture Models!
  - – P(Y) still multinomial
  - – P(X|Y) is a multivariate Gaussian distribution:



$$P(X = \mathbf{x}_j \mid Y = i) = \frac{1}{(2\pi)^{m/2} \|\Sigma_i\|^{1/2}} \exp\left[-\frac{1}{2}(\mathbf{x}_j - \mu_i)^T \Sigma_i^{-1} (\mathbf{x}_j - \mu_i)\right]$$

Multivariate Gaussians

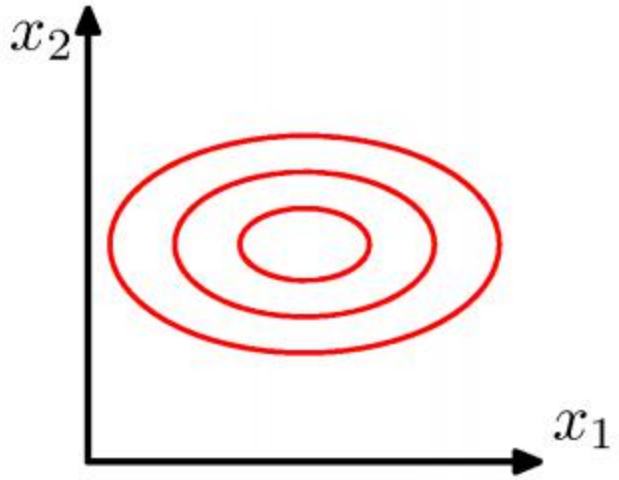
$$P(X=\mathbf{x}_j) = \frac{1}{(2\pi)^{m/2} \|\Sigma\|^{1/2}} \exp\left[-\frac{1}{2}(\mathbf{x}_j - \mu)^T \Sigma^{-1} (\mathbf{x}_j - \mu)\right]$$



$\Sigma \propto$ identity matrix

Multivariate Gaussians

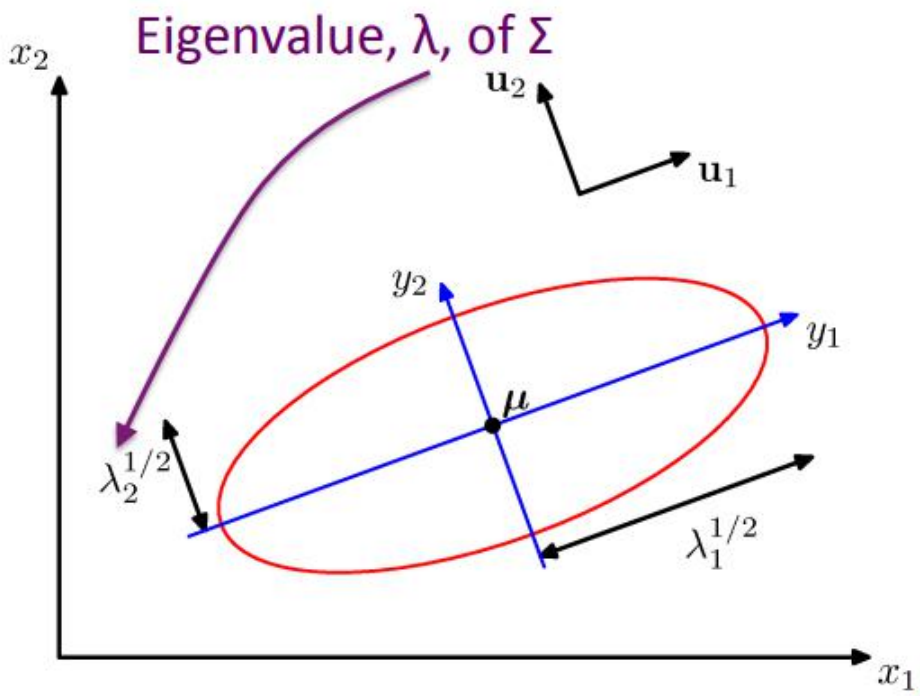
$$P(X=\mathbf{x}_j) = \frac{1}{(2\pi)^{m/2}\,\|\Sigma\|^{1/2}} \exp\left[-\frac{1}{2}(\mathbf{x}_j-\mu)^T \Sigma^{-1}(\mathbf{x}_j-\mu)\right]$$



Σ = diagonal matrix

X_i are independent *ala* Gaussian NB

Multivariate Gaussians

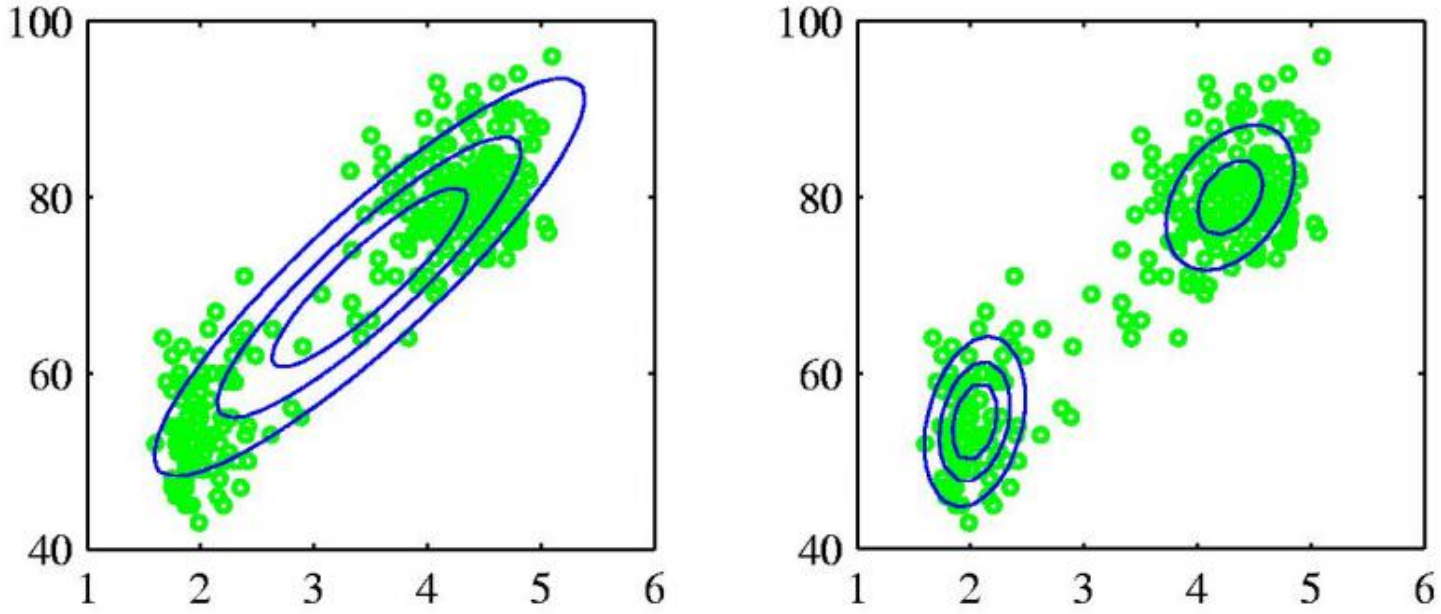


Covariance matrix, Σ, = degree to which x_i vary together

$$P(X=\mathbf{x}_j) = \frac{1}{(2\pi)^{m/2}\,\|\Sigma\|^{1/2}} \exp\left[-\frac{1}{2}(\mathbf{x}_j-\mu)^T \Sigma^{-1}(\mathbf{x}_j-\mu)\right]$$

Mixtures of Gaussians (1)

Old Faithful Data Set



Mixtures of Gaussians (2)

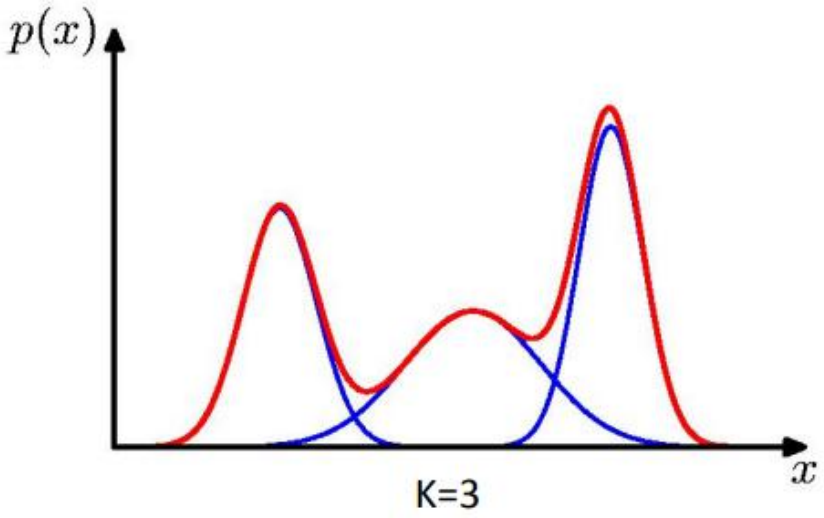
- Combine simple models into a complex model:

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

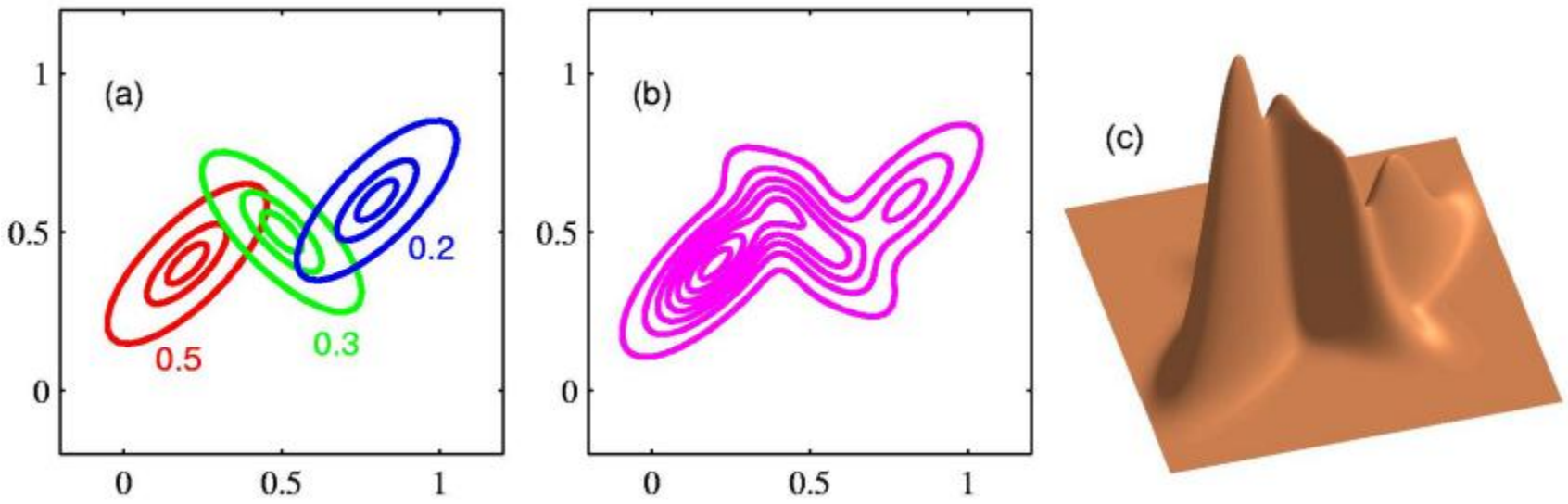
Component

Mixing coefficient

$$\forall k : \pi_k \geqslant 0 \qquad \sum_{k=1}^{K} \pi_k = 1$$

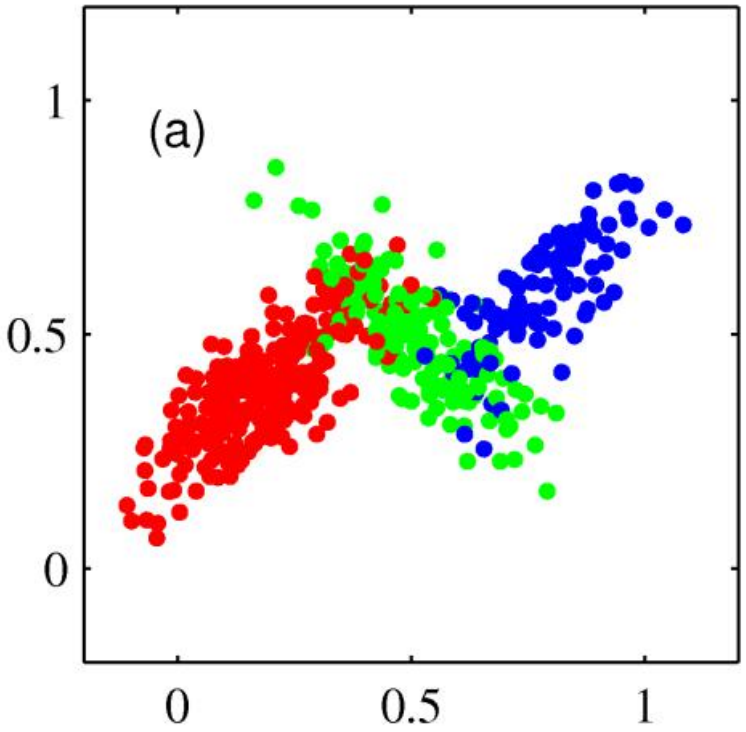


Mixtures of Gaussians (3)



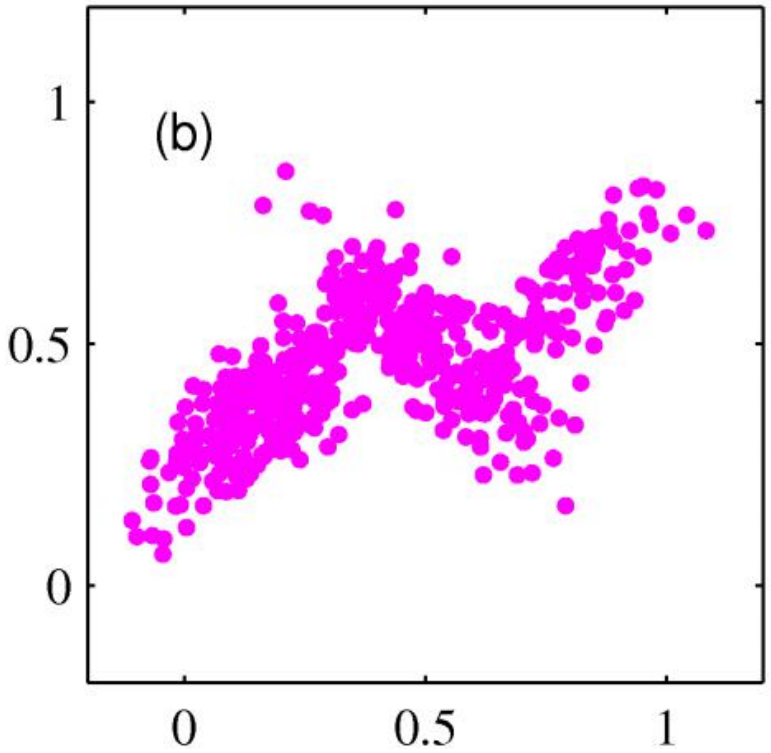
Eliminating Hard Assignments to Clusters

- Model data as mixture of multivariate Gaussians



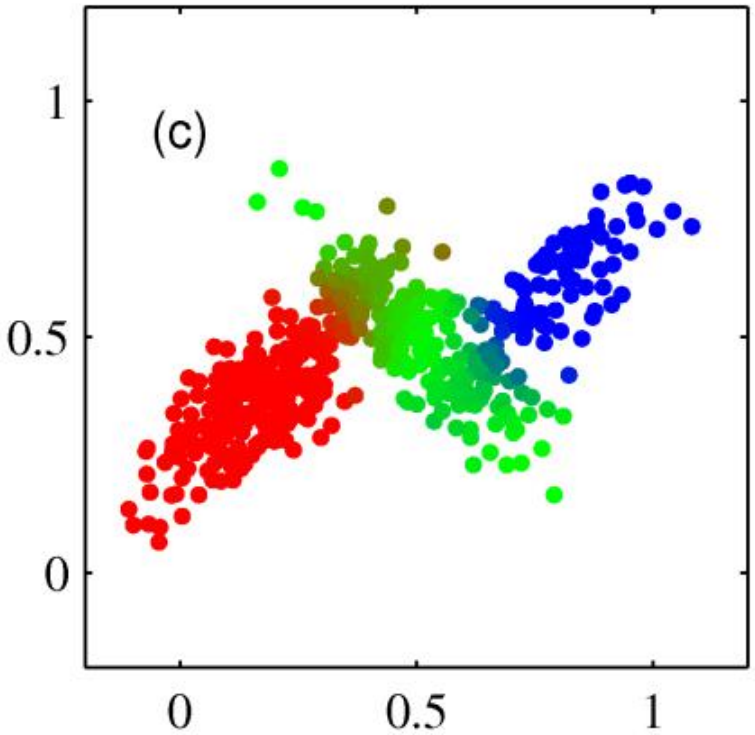
Eliminating Hard Assignments to Clusters

- Model data as mixture of multivariate Gaussians



Eliminating Hard Assignments to Clusters

- Model data as mixture of multivariate Gaussians



Shown is the posterior probability that a point was generated from i-th Gaussian: Pr(Y = i | x)

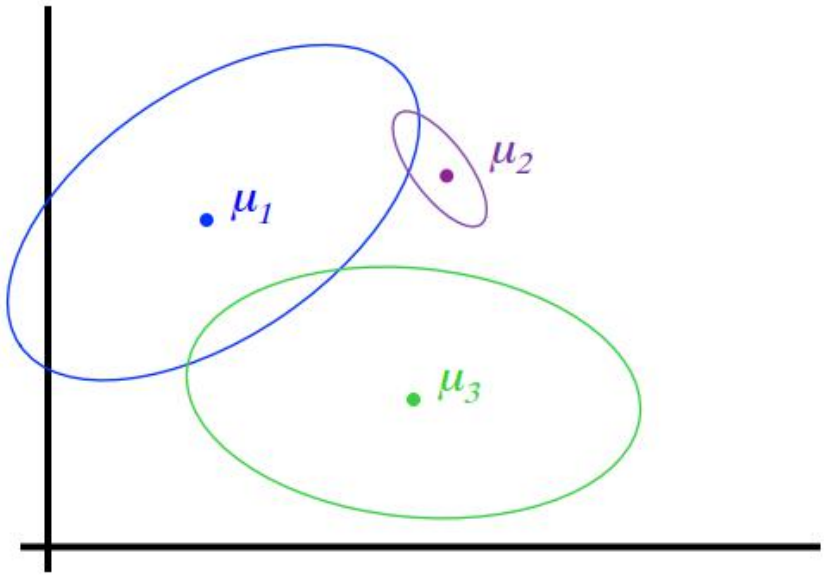
The General GMM Assumption

- P(Y): There are k components
- P(X|Y): Each component generates data from a **multivariate Gaussian** with mean μ_i and covariance matrix Σ_i

Each data point is sampled from a ***generative process***:

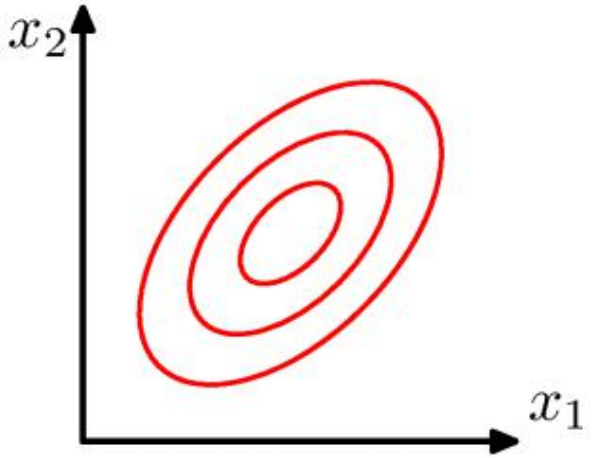
1. Choose component i with probability *P(y=i)*
2. Generate datapoint ~ N(m_i, Σ_i)

Gaussian mixture model (GMM)



Multivariate Gaussians

$$P(X=\mathbf{x}_j) = \frac{1}{(2\pi)^{m/2} \|\Sigma\|^{1/2}} \exp\left[-\frac{1}{2}(\mathbf{x}_j - \mu)^T \Sigma^{-1} (\mathbf{x}_j - \mu)\right]$$



Σ = arbitrary (semidefinite) matrix:
- specifies rotation (change of basis)
- eigenvalues specify relative elongation

ML Estimation in Supervised Setting

- Univariate Gaussian

$$\mu_{MLE} = \frac{1}{N}\sum_{i=1}^{N} x_i \qquad \sigma^2_{MLE} = \frac{1}{N}\sum_{i=1}^{N}(x_i - \hat{\mu})^2$$

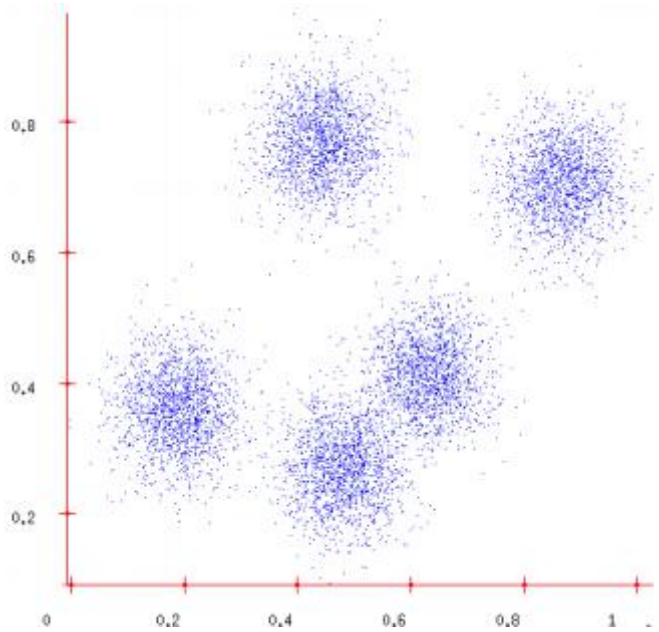
- Mixture of Multivariate Gaussians
  - ML estimate for each of the Multivariate Gaussians is given by:

$$\mu_{ML}^k = \frac{1}{n}\sum_{j=1}^{n} x_n \qquad \Sigma_{ML}^k = \frac{1}{n}\sum_{j=1}^{n}\left(\mathbf{x}_j - \mu_{ML}^k\right)\left(\mathbf{x}_j - \mu_{ML}^k\right)^T$$

Just sums over *x* generated from the *k*'th Gaussian

That was easy! But what if unobserved data?

- MLE:
  - $\text{argmax}_\theta \prod_j P(y_j, x_j)$
  - θ: all model parameters
    - eg, class probs, means, and variances



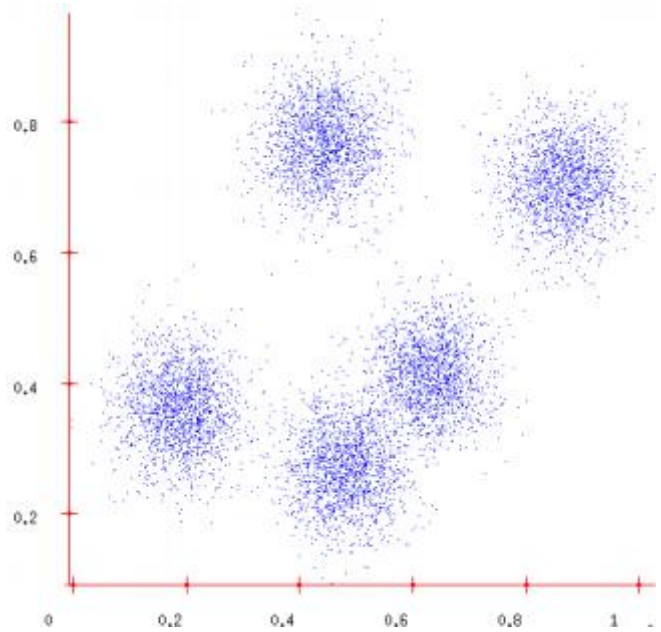
- But we don't know !!!
- Maximize marginal likelihood
  - $\text{argmax}_\theta \prod_j P(x_j) = \text{argmax} \prod_j \sum_{k=1}^{K} P(Y_j=k, x_j)$

How Do We Optimize? Closed Form?

- Maximize marginal likelihood:

$$- \text{argmax}_\theta \prod_j P(x_j) = \text{argmax} \prod_j \sum_{k=1}^{K} P(Y_j=k, x_j)$$

- Almost always a hard problem!
  - – Usually no closed-form solution
  - –Even when lgP(X,Y) is convex, lgP(X) generally isn't…
  - – For all but the simplest P(X), we will have to do gradient ascent, in a big messy space with lots of local optimum…



Learning General Mixtures of Gaussian

$$P(y=k \mid \mathbf{x}_j) \propto \frac{1}{(2\pi)^{m/2} \|\Sigma_k\|^{1/2}} \exp\left[-\frac{1}{2}\left(\mathbf{x}_j-\mu_k\right)^T \Sigma_k^{-1}\left(\mathbf{x}_j-\mu_k\right)\right] P(y=k)$$

- Marginal likelihood:

$$\prod_{j=1}^{m} P(\mathbf{x}_j) = \prod_{j=1}^{m} \sum_{k=1}^{K} P(\mathbf{x}_j, y=k)$$

$$= \prod_{j=1}^{m} \sum_{k=1}^{K} \frac{1}{(2\pi)^{m/2} \|\Sigma_k\|^{1/2}} \exp\left[-\frac{1}{2}\left(\mathbf{x}_j-\mu_k\right)^T \Sigma_k^{-1}\left(\mathbf{x}_j-\mu_k\right)\right] P(y=k)$$

- Need to differentiate and solve for μ_k, Σ_k, and P(Y=k) for k=1..K
- There will be no closed form solution, gradient is complex, lots of local optimum
- ***Wouldn't it be nice if there was a better way!?!***

Outline

- Parametric Unsupervised Learning
- Mixture Density Model
- Gaussian Mixture Model
- **Expectation Maximization (EM) Algorithm**
- Applications

Expectation Maximization Algorithm

- EM is an algorithm for ML parameter estimation when the data has missing values. It is used when
  - 1. data is incomplete (has missing values)
    - some features are missing for some samples due to data corruption, partial survey responces, etc.
    - This scenario is very useful
  - 2. Suppose data X is complete, but $p(X|\theta)$ is hard to optimize. Suppose further that introducing certain hidden variables U whose values are missing, and suppose it is easier to optimize the "complete" likelihood function $p(X,U|\theta)$. Then EM is useful.
    - This scenario is useful for the mixture density estimation, and is subject of our lecture today
- Notice that after we introduce artificial (hidden) variables U with missing values, case 2 is completely equivalent to case 1

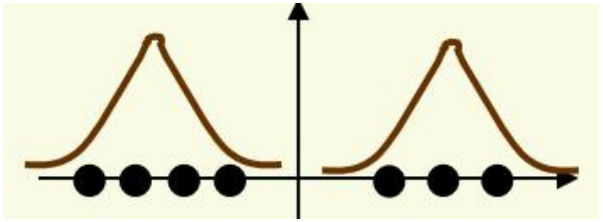
EM: Hidden Variables for Mixture Density

$$p(x \mid \theta) = \sum_{j=1}^{m} p(x \mid c_j, \theta_j) \rho_j$$

- For simplicity, assume component densities are

$$p(x \mid c_j, \theta_j) = \frac{1}{\sigma\sqrt{2\pi}} \exp\left(-\frac{(x-\mu_j)^2}{2\sigma^2} \right)$$

  - assume for now that the variance is known
  - need to estimate $\theta = \{\mu 1, \ldots, \mu m\}$



- If we knew which sample came from which component (that is the class label), the ML parameter estimation is easy
- Thus to get an easier problem, introduce hidden variables which indicate which component each sample belongs to

EM: Hidden Variables for Mixture Density

- For 1≤ i ≤ n, 1≤ k ≤m, define hidden variables

$$z_i^{(k)} = \begin{cases} 1 & \text{if sample } \boldsymbol{i} \text{ was generated by component } \boldsymbol{k} \\ 0 & \text{otherwise} \end{cases}$$

$$\boldsymbol{x}_i \rightarrow \left\{ \boldsymbol{x}_i, z_i^{(1)}, \ldots, z_i^{(m)} \right\}$$

- $z_i^{(k)}$ are indicator random variables, they indicate which Gaussian component generated sample x_i
- Let $z_i = \{z_i^{(1)}, \ldots, z_i^{(m)}\}$ indicator r.v. corresponding to sample x_i
- Conditioned on z_i , distribution of x_i is Gaussian

$$p(\boldsymbol{x}_i \mid z_i, \theta) \sim N(\mu_k, \sigma^2)$$

where $\boldsymbol{k}$ is s.t. $z_i^{(k)} = 1$

EM: Joint Likelihood

- Let $z_i = \{z_i^{(1)}, \ldots, z_i^{(m)}\}$, and $Z = \{z_1, \ldots, z_n\}$
- The complete likelihood is

$$p(X, Z \mid \theta) = p(x_1, \ldots, x_n, z_1, \ldots, z_n \mid \theta) = \prod_{i=1}^{n} p(x_i, z_i \mid \theta)$$

$$= \prod_{i=1}^{n} \underbrace{p(x_i \mid z_i, \theta)}_{\text{gaussian}} \underbrace{p(z_i)}_{\text{part of } \rho_c}$$

- If we actually observed Z, the log likelihood $\ln[p(X,Z \mid \theta)]$ would be trivial to maximize with respect to θ and ρ_i
- The problem, of course, is that the values of Z are missing, since we made it up (that is Z is hidden)

EM Derivation

- Instead of maximizing ln[p(X,Z| θ)] the idea behind EM is to maximize some function of ln[p(X,Z| θ)], usually its expected value

$$E_Z[\ln p(X,Z \mid \theta)]$$

  - If θ makes ln[p(X,Z|θ)] large, then θ tends to make E[ln p(X,Z|θ)] large
  - the expectation is with respect to the missing data Z
  - that is with respect to density p(Z |X,θ)

- however θ is our ultimate goal, we don't know θ !

EM Algorithm

- EM solution is to iterate

1. start with initial parameters $\theta^{(0)}$

iterate the following 2 step until convergence

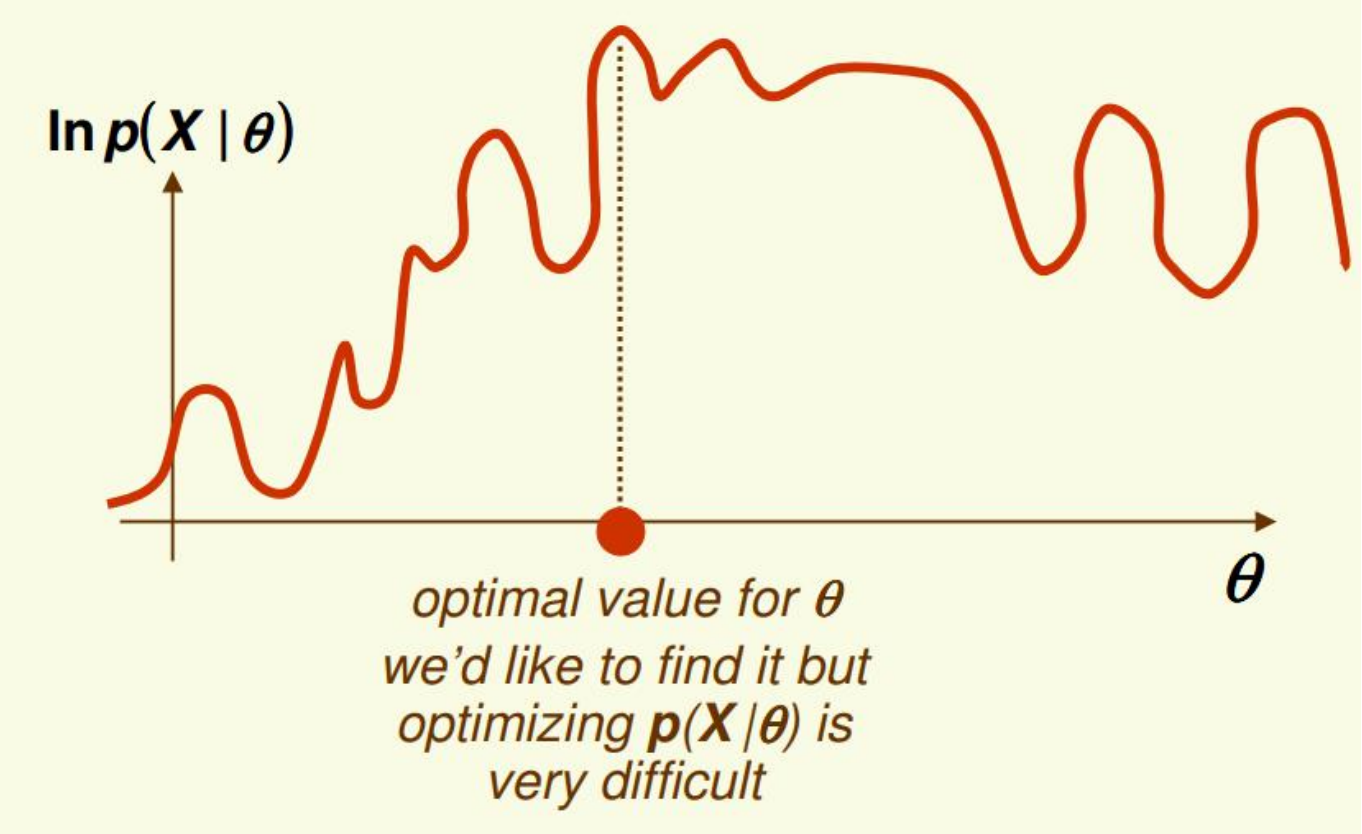
E. compute the expectation of log likelihood with respect to current estimate $\theta^{(t)}$ and $\boldsymbol{X}$. Let's call it $\boldsymbol{Q}(\theta|\theta^{(t)})$

$$\boldsymbol{Q}\left(\theta \mid \theta^{(t)}\right) = \boldsymbol{E}_{\boldsymbol{Z}}\left[\ln \boldsymbol{p}(\boldsymbol{X},\boldsymbol{Z} \mid \theta) \mid \boldsymbol{X},\theta^{(t)}\right]$$

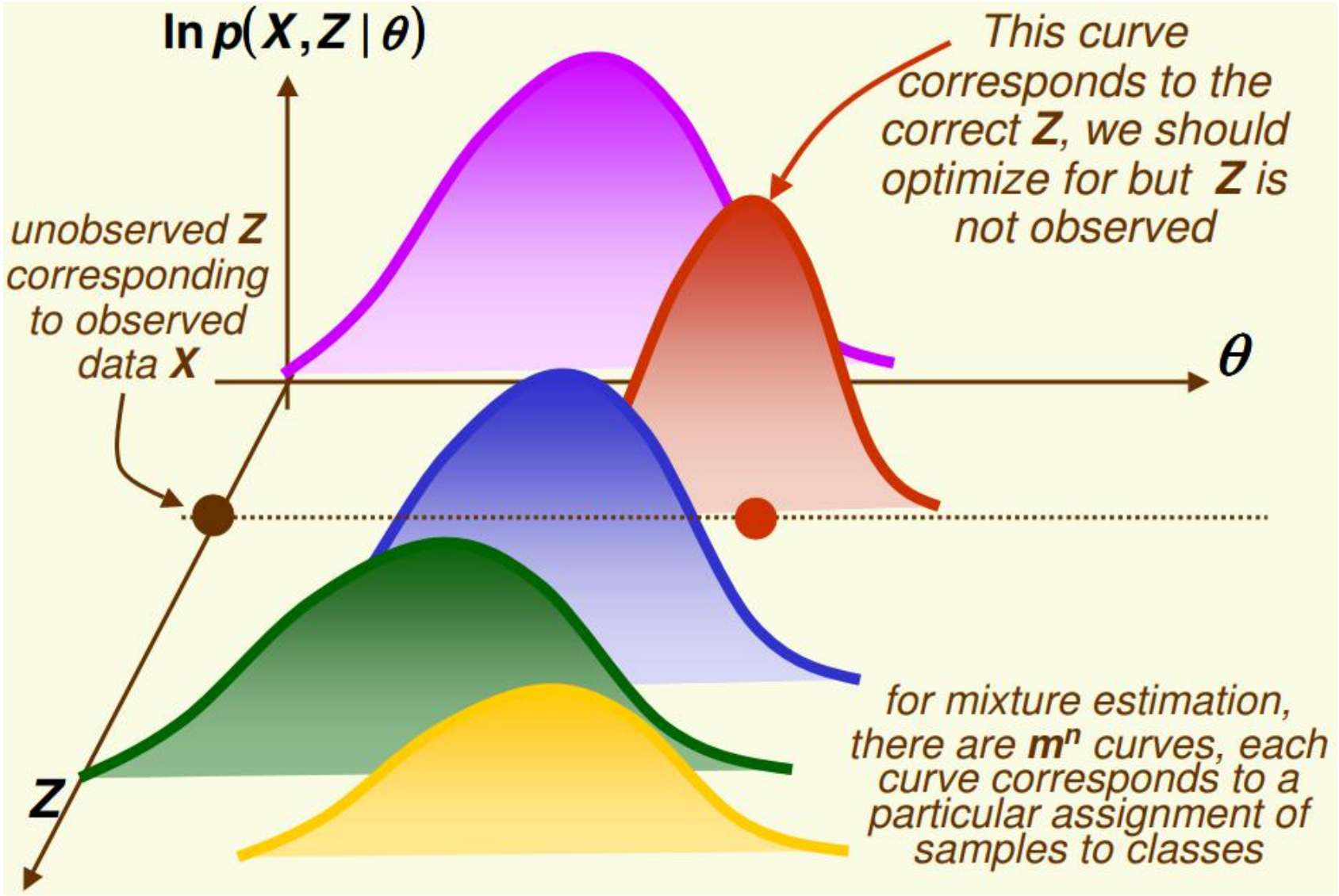
M. maximize $\boldsymbol{Q}(\theta|\theta^{(t)})$

$$\theta^{(t+1)} = \arg\max_{\theta} \boldsymbol{Q}\left(\theta \mid \theta^{(t)}\right)$$

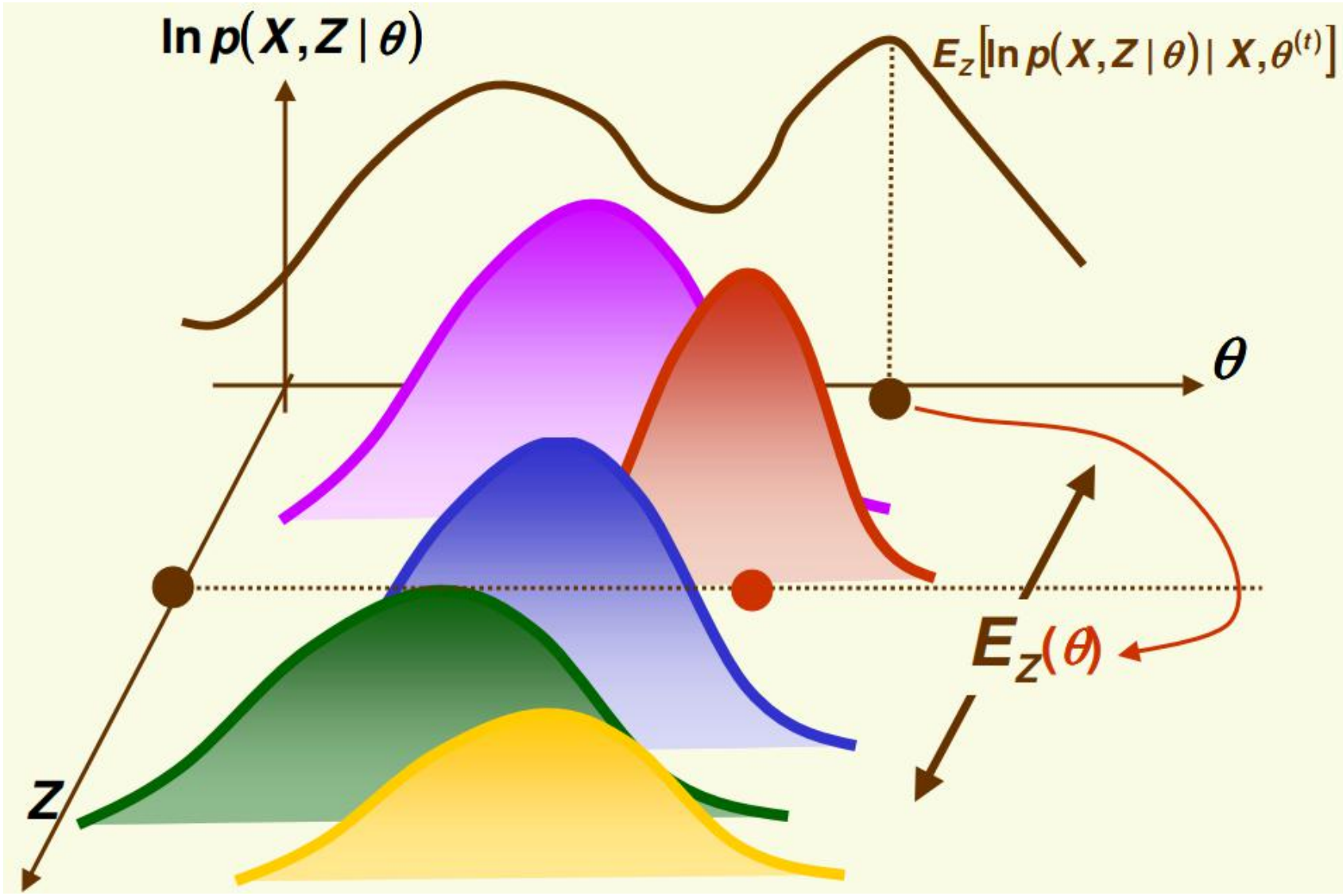
EM Algorithm: Picture



EM Algorithm: Picture



EM Algorithm: Picture



EM Algorithm

- It can be proven that EM algorithm converges to the local maximum of the log-likelihood

$$\ln p(\boldsymbol{X} \mid \boldsymbol{\theta})$$

- Why is it better than gradient ascent?
  - Convergence of EM is usually significantly faster, in the beginning, very large steps are made (that is likelihood function increases rapidly), as opposed to gradient ascent which usually takes tiny steps
  - gradient descent is not guaranteed to converge
  - recall all the difficulties of choosing the appropriate
  - learning rate

EM for Mixture of Gaussians: E step

- Let's come back to our example $p(x \mid \theta) = \sum_{j=1}^{m} p(x \mid c_j, \theta_j)\rho_j$

$$p(x \mid c_j, \theta_j) = \frac{1}{\sigma\sqrt{2\pi}} \exp\left(-\frac{(x-\mu_j)^2}{2\sigma^2}\right)$$

need to estimate $\theta = \{\mu_1, \ldots, \mu_m\}$ and $\rho_1, \ldots, \rho_m$

- For $1 \le i \le n$, $1 \le k \le m$, we define

$$z_i^{(k)} = \begin{cases} 1 & \text{if sample } \boldsymbol{i} \text{ was generated by component } \boldsymbol{k} \\ 0 & \text{otherwise} \end{cases}$$

- As before, $z_i = \{z_i^{(1)}, \ldots, z_i^{(m)}\}$ and $Z = \{z_1, \ldots, z_n\}$
- We need log-likelihood of observed X and hidden Z

$$\ln p(X, Z \mid \theta) = \ln \prod_{i=1}^{n} p(x_i, z_i \mid \theta) = \sum_{i=1}^{n} \ln p(x_i \mid z_i, \theta) P(z_i)$$

EM for Mixture of Gaussians: E step

- We need log-likelihood of observed X and hidden Z

$$\ln p(X,Z\mid\theta)=\ln\prod_{i=1}^{n}p(x_i,z_i\mid\theta)=\sum_{i=1}^{n}\ln p(x_i\mid z_i,\theta)P(z_i)$$

- First let's rewrite $p(x_i\mid z_i,\theta)P(z_i)$

$$p(x_i\mid z_i,\theta)P(z_i)=\begin{cases}p(x_i\mid z_i^{(1)}=1,\theta)P(z_i^{(1)}=1) & \text{if } z_i^{(1)}=1\\ \vdots & \vdots\\ p(x_i\mid z_i^{(m)}=1,\theta)P(z_i^{(m)}=1) & \text{if } z_i^{(m)}=1\end{cases}$$

$$=\prod_{k=1}^{m}\left[p(x_i\mid z_i^{(k)}=1,\theta)P(z_i^{(k)}=1)\right]^{z_i^{(k)}}$$

$$=\prod_{k=1}^{m}\left[\frac{1}{\sigma\sqrt{2\pi}}\exp\left(-\frac{(x_i-\mu_k)^2}{2\sigma^2}\right)P(z_i^{(k)}=1)\right]^{z_i^{(k)}}$$

EM for Mixture of Gaussians: E step

- log-likelihood of observed X and hidden Z is

$$\ln p(X,Z \mid \theta) = \sum_{i=1}^{n} \ln p(x_i \mid z_i,\theta) P(z_i)$$

$$= \sum_{i=1}^{n} \ln \prod_{k=1}^{m} \left[\frac{1}{\sigma\sqrt{2\pi}} \exp\left(-\frac{(x_i-\mu_k)^2}{2\sigma^2} \right) P\left(z_i^{(k)}=1\right) \right]^{z_i^{(k)}}$$

$$= \sum_{i=1}^{n} \sum_{k=1}^{m} \ln \left[\frac{1}{\sigma\sqrt{2\pi}} \exp\left(-\frac{(x_i-\mu_k)^2}{2\sigma^2} \right) P\left(z_i^{(k)}=1\right) \right]^{z_i^{(k)}}$$

$$= \sum_{i=1}^{n} \sum_{k=1}^{m} z_i^{(k)} \left[\ln \frac{1}{\sigma\sqrt{2\pi}} - \frac{(x_i-\mu_k)^2}{2\sigma^2} + \ln \underbrace{P\left(z_i^{(k)}=1\right)}_{P(\text{sample } x_i \text{ from class } k) = P(c_k) = \rho_k} \right]$$

$$= \sum_{i=1}^{n} \sum_{k=1}^{m} z_i^{(k)} \left[\ln \frac{1}{\sigma\sqrt{2\pi}} - \frac{(x_i-\mu_k)^2}{2\sigma^2} + \ln \rho_k \right]$$

EM for Mixture of Gaussians: E step

- log-likelihood of observed X and hidden Z is

$$\ln p(X,Z\mid\theta)=\sum_{i=1}^{n}\sum_{k=1}^{m} z_i^{(k)}\left[\ln\frac{1}{\sigma\sqrt{2\pi}}-\frac{(x_i-\mu_k)^2}{2\sigma^2}+\ln\rho_k\right]$$

- For the E step, we must compute

$$Q(\theta\mid\theta^{(t)})=Q(\theta\mid\mu_1^{(t)},\ldots,\mu_m^{(t)},\rho_1^{(t)},\ldots,\rho_m^{(t)})=E_Z\left[\ln p(X,Z\mid\theta)\mid X,\theta^{(t)}\right]$$

$$=E_Z\left(\sum_{i=1}^{n}\sum_{k=1}^{m} z_i^{(k)}\left[\ln\frac{1}{\sigma\sqrt{2\pi}}-\frac{(x_i-\mu_k)^2}{2\sigma^2}+\ln\rho_k^{(t)}\right]\right)$$

$$E_X\left[\sum_i a_i x_i+b\right]=\sum_i a_i E_X[x_i]+b$$

$$=\sum_{i=1}^{n}\sum_{k=1}^{m} E_Z\left[z_i^{(k)}\right]\left(\ln\frac{1}{\sigma\sqrt{2\pi}}-\frac{(x_i-\mu_k)^2}{2\sigma^2}+\ln\rho_k\right)$$

EM for Mixture of Gaussians: E step

$$Q(\theta \mid \theta^{(t)}) = \sum_{i=1}^{n}\sum_{k=1}^{m} E_Z[z_i^{(k)}]\left(\ln\frac{1}{\sigma\sqrt{2\pi}} - \frac{(x_i-\mu_k)^2}{2\sigma^2} + \ln\rho_k\right)$$

- need to compute $E_Z[z_i^{(k)}]$ in the above expression

$$E_Z[z_i^{(k)}] = 0 * P(z_i^{(k)} = 0 \mid \theta^{(t)}, x_i) + 1 * P(z_i^{(k)} = 1 \mid \theta^{(t)}, x_i)$$

$$= P(z_i^{(k)} = 1 \mid \theta^{(t)}, x_i) = \frac{p(x_i \mid \theta^{(t)}, z_i^{(k)} = 1) P(z_i^{(k)} = 1 \mid \theta^{(t)})}{p(x_i \mid \theta^{(t)})}$$

$$= \frac{\rho_k^{(t)} \exp\left(-\frac{1}{2}(x_i - \mu_k^{(t)})^2\right)}{\sum_{j=1}^{m} P(x_i \mid \theta^{(t)}, z_i^{(j)} = 1) P(z_i^{(j)} = 1 \mid \theta^{(t)})} = \frac{\rho_k^{(t)} \exp\left(-\frac{1}{2\sigma^2}(x_i - \mu_k^{(t)})^2\right)}{\sum_{j=1}^{m} \rho_j^{(t)} \exp\left(-\frac{1}{2\sigma^2}(x_i - \mu_j^{(t)})^2\right)}$$

- We are finally done with the E step
  - for implementation, just need to compute $E_Z[z_i^{(k)}]$s don't need to compute Q

EM for Mixture of Gaussians: M step

$$Q(\theta \mid \theta^{(t)}) = \sum_{i=1}^{n}\sum_{k=1}^{m} E_Z[z_i^{(k)}]\left(\ln\frac{1}{\sigma\sqrt{2\pi}} - \frac{(x_i-\mu_k)^2}{2\sigma^2} + \ln\rho_k\right)$$

- Need to maximize Q with respect to all parameters
- First differentiate with respect to μ_k

$$\frac{\partial}{\partial\mu_k}Q(\theta \mid \theta^{(t)}) = \sum_{i=1}^{n} E_Z[z_i^{(k)}]\frac{(x_i-\mu_k)}{\sigma^2} = 0$$

$$\Rightarrow \textbf{new}\ \mu_k = \mu_k^{(t+1)} = \boxed{\frac{1}{n}\sum_{i=1}^{n} E_Z[z_i^{(k)}]x_i}$$

*the mean for class **k** is weighted average of all samples, and this weight is proportional to the current estimate of probability that the sample belongs to class **k***

EM for Mixture of Gaussians: M step

$$Q(\theta \mid \theta^{(t)}) = \sum_{i=1}^{n}\sum_{k=1}^{m} E_Z\left[z_i^{(k)}\right]\left(\ln\frac{1}{\sigma\sqrt{2\pi}} - \frac{(x_i-\mu_k)^2}{2\sigma^2} + \ln\rho_k\right)$$

- For ρ_k we have to use Lagrange multipliers to preserve constraint $\sum_{j=1}^{m}\rho_j = 1$
- Thus we need to differentiate

$$F(\lambda,\rho) = Q(\theta \mid \theta^{(t)}) - \lambda\left(\sum_{j=1}^{m}\rho_j - 1\right)$$

$$\frac{\partial}{\partial\rho_k}F(\lambda,\rho) = \sum_{i=1}^{n}\frac{1}{\rho_k}E_Z\left[z_i^{(k)}\right] - \lambda = 0 \Rightarrow \sum_{i=1}^{n}E_Z\left[z_i^{(k)}\right] - \lambda\rho_k = 0$$

- Summing up over all components $\sum_{k=1}^{m}\sum_{i=1}^{n}E_Z\left[z_i^{(k)}\right] = \sum_{k=1}^{m}\lambda\rho_k$
- Since $\sum_{k=1}^{m}\sum_{i=1}^{n}E_Z\left[z_i^{(k)}\right] = n$ and $\sum_{k=1}^{m}\rho_k = 1$, we get $\lambda = n$

$$\rho_k^{(t+1)} = \frac{1}{n}\sum_{i=1}^{n}E_Z\left[z_i^{(k)}\right]$$

EM Algorithm

- The algorithm on this slide applies ONLY to univariate gaussian case with known variances

1. Randomly initialize $\mu_1, \ldots, \mu_m, \rho_1, \ldots, \rho_m$ (with constraint $\Sigma \rho_i = 1$)

iterate until no change in $\mu_1, \ldots, \mu_m, \rho_1, \ldots, \rho_m$

E. for all i, k, compute

$$E_z\left[z_i^{(k)}\right] = \frac{\rho_k \exp\left(-\frac{1}{2\sigma^2}(x_i - \mu_k)^2\right)}{\sum_{j=1}^{m} \rho_j \exp\left(-\frac{1}{2\sigma^2}(x_i - \mu_j)^2\right)}$$

M. for all k, do parameter update

$$\mu_k = \frac{1}{n}\sum_{i=1}^{n} E_z\left[z_i^{(k)}\right] x_i \qquad \rho_k = \frac{1}{n}\sum_{i=1}^{n} E_z\left[z_i^{(k)}\right]$$

EM Algorithm

- For the more general case of multivariate
- Gaussians with unknown means and variances

E step: $$E_z\left[z_i^{(k)}\right] = \frac{\rho_k\, p(x \mid \mu_k, \Sigma_k)}{\sum_{j=1}^{m} \rho_j p(x \mid \mu_j, \Sigma_j)}$$

where $$p(x \mid \mu_k, \Sigma_k) = \frac{1}{(2\pi)^{d/2}\left|\Sigma_k^{-1}\right|^{1/2}} exp\left[-\frac{1}{2}(x-\mu_k)^t \Sigma_k^{-1}(x-\mu_k)\right]$$

M step:

$$\rho_k = \frac{1}{n}\sum_{i=1}^{n} E_z\left[z_i^{(k)}\right]$$

$$\mu_k = \frac{\sum_{i=1}^{n} E_z\left[z_i^{(k)}\right] x_i}{\sum_{i=1}^{n} E_z\left[z_i^{(k)}\right]}$$

$$\Sigma_k = \frac{\sum_{i=1}^{n} E_z\left[z_i^{(k)}\right](x_i-\mu_k)(x_i-\mu_k)^T}{\sum_{i=1}^{n} E_z\left[z_i^{(k)}\right]}$$

EM Algorithm and K-means

- k-means can be derived from EM algorithm
- Setting mixing parameters equal for all classes,

$$E_Z\left[z_i^{(k)}\right]=\frac{\rho_k \exp\left(-\frac{1}{2\sigma^2}(x_i-\mu_k)^2\right)}{\sum_{j=1}^{m}\rho_j \exp\left(-\frac{1}{2\sigma^2}(x_i-\mu_j)^2\right)}=\frac{\exp\left(-\frac{1}{2\sigma^2}(x_i-\mu_k)^2\right)}{\sum_{j=1}^{m}\exp\left(-\frac{1}{2\sigma^2}(x_i-\mu_j)^2\right)}$$

- If we let $\sigma \to \infty$, then

$$E_Z\left[z_i^{(k)}\right]=\begin{cases}1 & \text{if } \forall j,\ \| x_i-\mu_k \| > \| x_i-\mu_j \| \\ 0 & \text{otherwise}\end{cases}$$

  - so at the E step, for each current mean, we find all points closest to it and form new clusters
  - at the M step, we compute the new means inside current clusters

$$\mu_k=\frac{1}{n}\sum_{i=1}^{n}E_Z\left[z_i^{(k)}\right]x_i$$

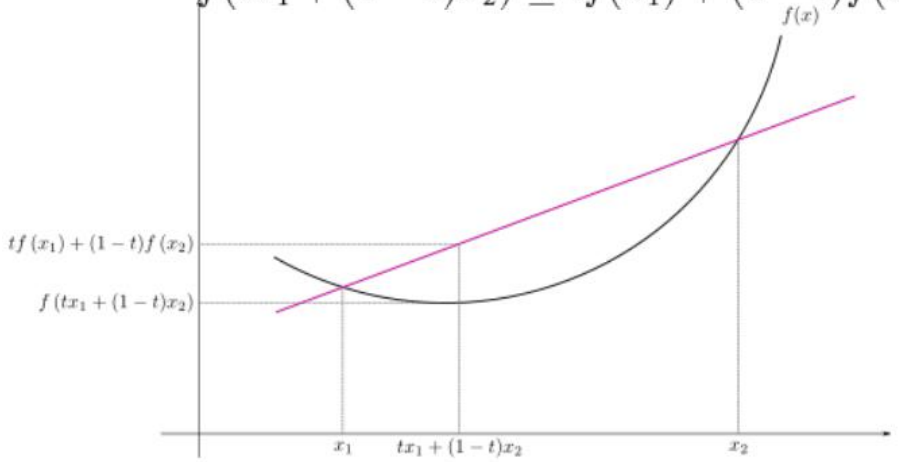
Properties of EM

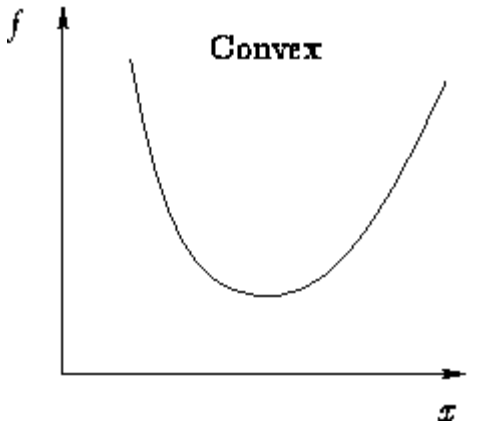
- We will prove that
  - – EM converges to a local minima
  - – Each iteration improves the logZ likelihood
- How? (Same as k-means)
  - – E-step can never decrease likelihood
  - – M-step can never decrease likelihood

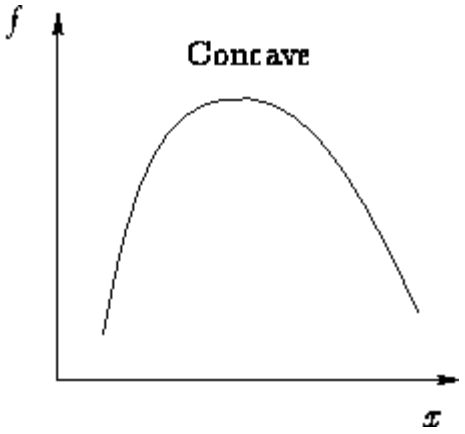
Jensen's Inequality

- **Theorem**: $\log \sum_z P(z) f(z) \geq \sum_z P(z) \log f(z)$
  - **e.g., Binary case for convex function f:**

$$f(tx_1 + (1-t)x_2) \leq tf(x_1) + (1-t)f(x_2).$$







Applying Jensen's Inequality

- Use: $\log \sum_z P(z)\, f(z) \geq \sum_z P(z) \log f(z)$

$$\ell(\theta^{(t)} : \mathcal{D}) = \sum_{j=1}^{m} \log \sum_{\mathbf{z}} Q^{(t+1)}(\mathbf{z} \mid \mathbf{x}_j) \frac{P(\mathbf{z}, \mathbf{x}_j \mid \theta^{(t)})}{Q^{(t+1)}(\mathbf{z} \mid \mathbf{x}_j)}$$

$$\geq \sum_{j=1}^{m} \sum_{z} Q^{(t+1)}(z \mid x_j) \log \left(\frac{p(z, x_j \mid \theta^{(t)})}{Q^{(t+1)}(z \mid x_j)} \right)$$

$$= \sum_{j=1}^{m} \sum_{z} Q^{(t+1)}(z \mid x_j) \log \left(p(z, x_j \mid \theta^{(t)}) \right) - \sum_{j=1}^{m} \sum_{z} Q^{(t+1)}(z \mid x_j) \log \left(Q^{(t+1)}(z \mid x_j) \right)$$

$$\ell(\theta^{(t)} : \mathcal{D}) \geq \sum_{j=1}^{m} \sum_{\mathbf{z}} Q^{(t+1)}(\mathbf{z} \mid \mathbf{x}_j) \log P(\mathbf{z}, \mathbf{x}_j \mid \theta^{(t)}) + \sum_{j=1}^{m} H(Q^{(t+1),j})$$

The M-step

- Lower bound:

$$\ell(\theta^{(t)} : \mathcal{D}) \geq \sum_{j=1}^{m} \sum_{\mathbf{z}} Q^{(t+1)}(\mathbf{z} \mid \mathbf{x}_j) \log P(\mathbf{z}, \mathbf{x}_j \mid \theta^{(t)}) + \sum_{j=1}^{m} H(Q^{(t+1),j})$$

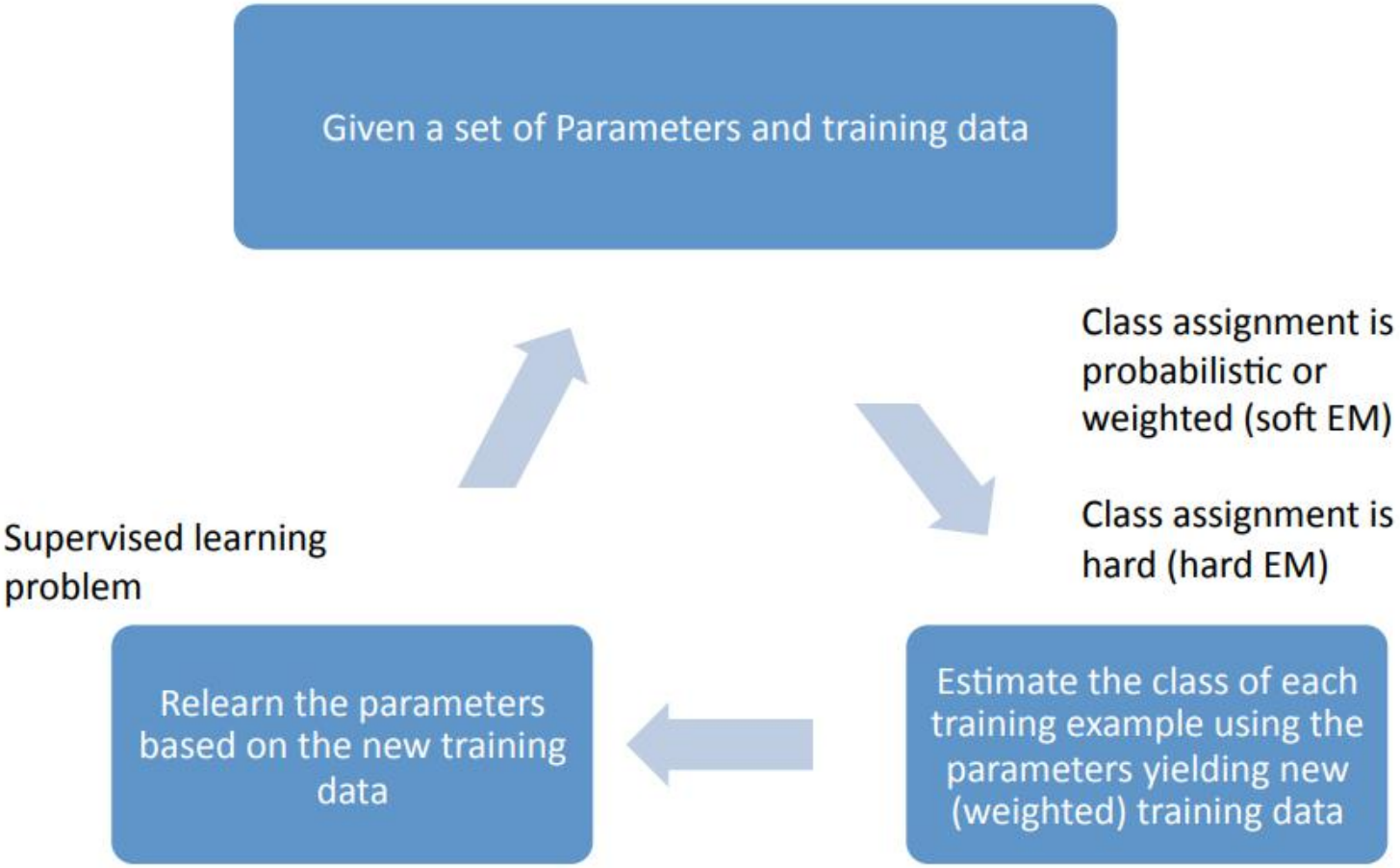
This term is a constant with respect to θ

- Maximization step

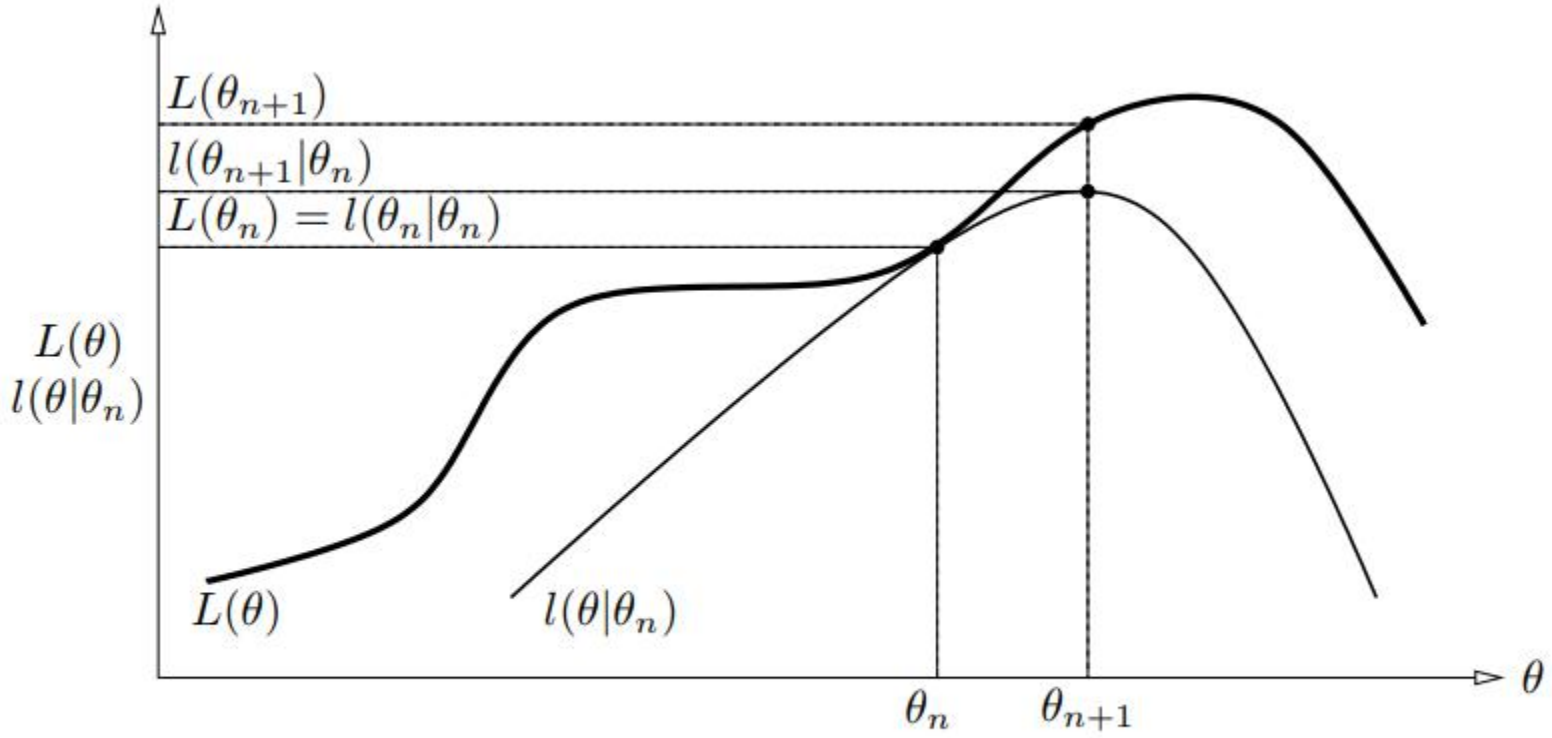
$$\theta^{(t+1)} \leftarrow \arg\max_{\theta} \sum_{j=1}^{m} \sum_{\mathbf{z}} Q^{(t+1)}(\mathbf{z} \mid \mathbf{x}_j) \log P(\mathbf{z}, \mathbf{x}_j \mid \theta)$$

- We are optimizing a lower bound!

EM Algorithm: Pictorial View



EM Pictorially

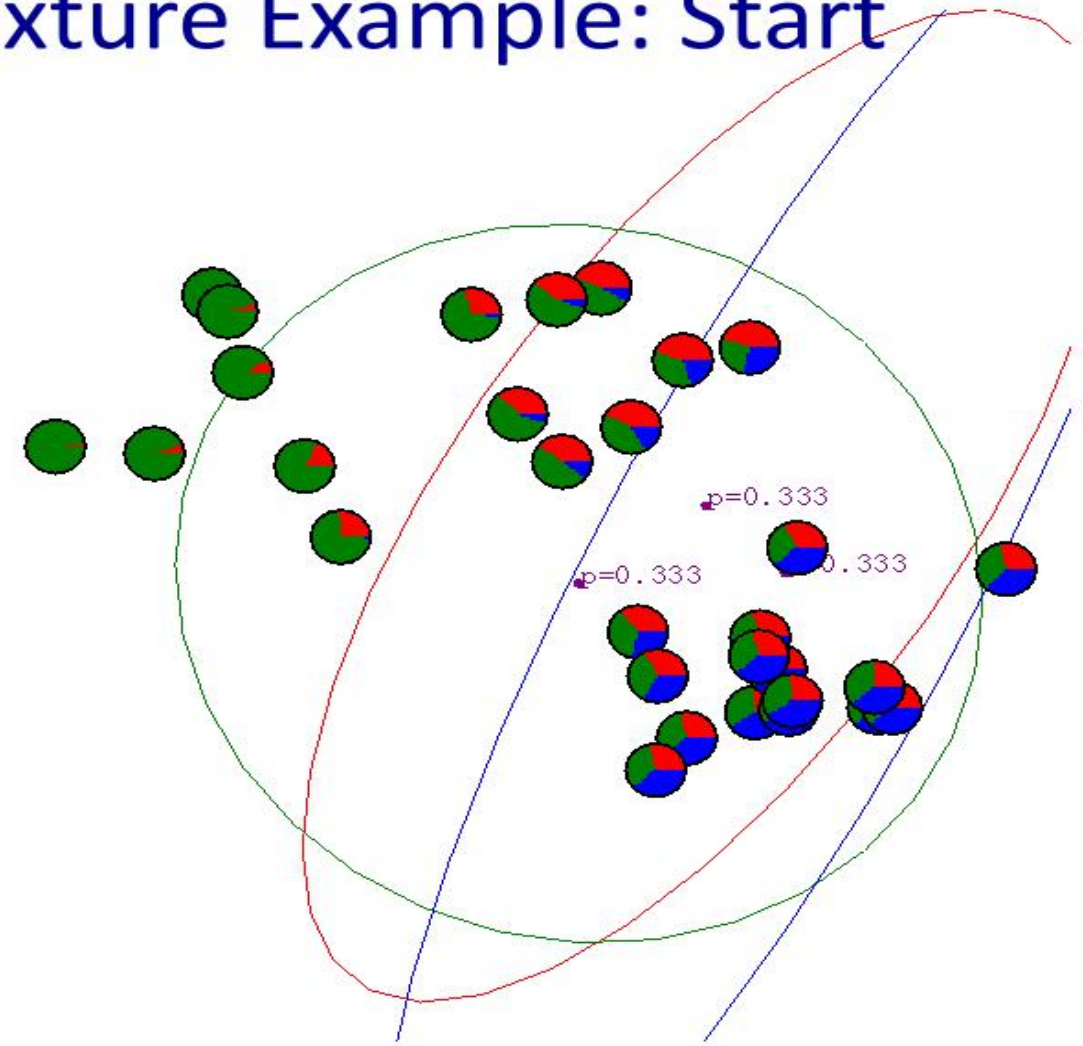


(Figure from tutorial by Sean Borman)

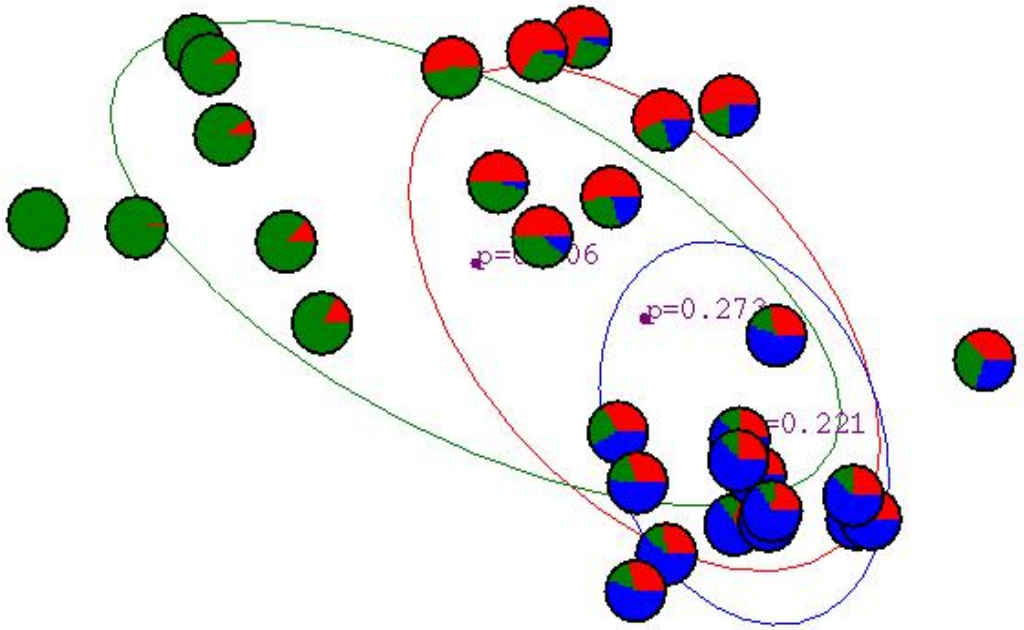
Outline

- Parametric Unsupervised Learning
- Mixture Density Model
- Gaussian Mixture Model
- Expectation Maximization (EM) Algorithm
- **Applications**

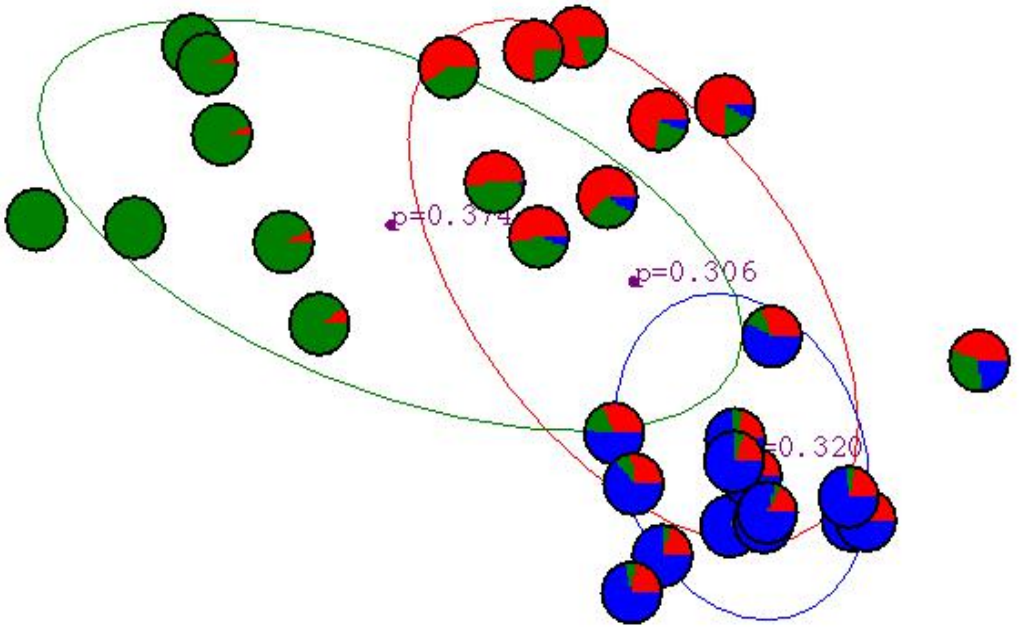
Gaussian Mixture Example: Start



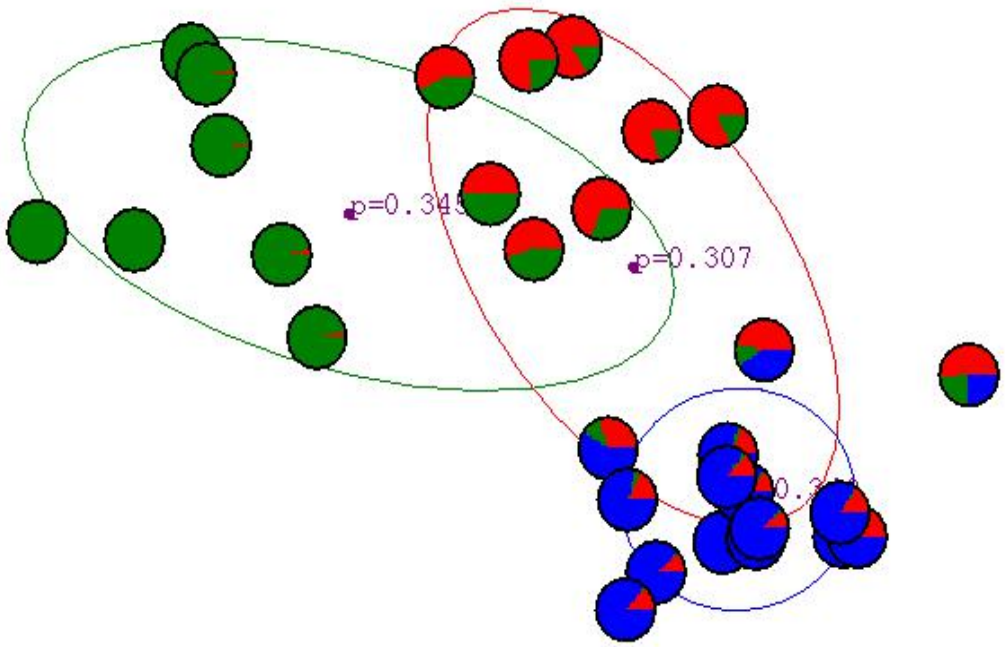
After first iteration



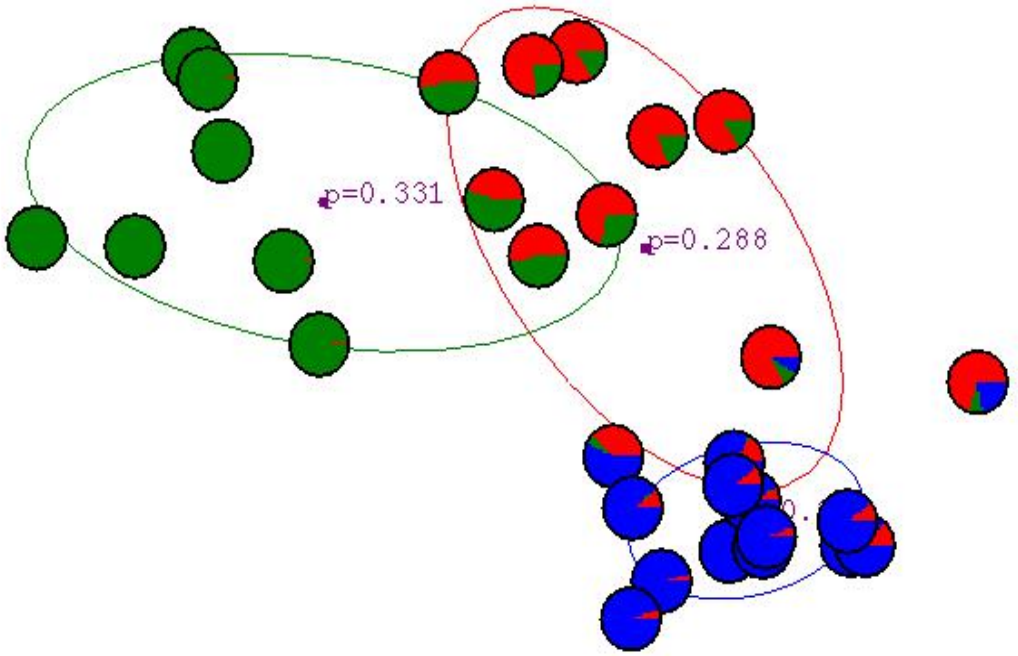
After 2nd iteration



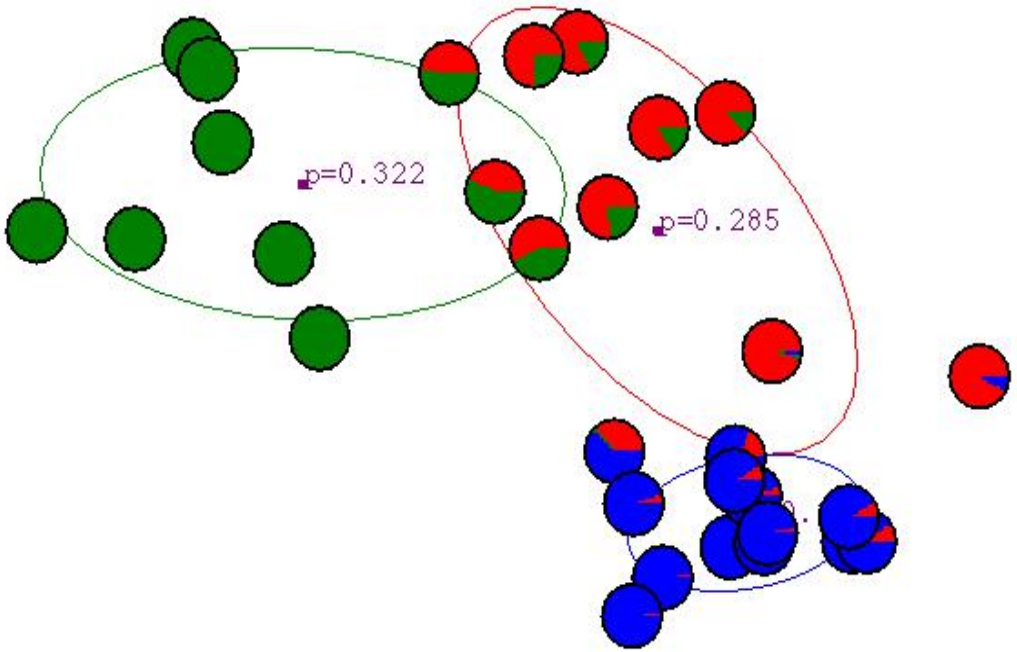
After 3rd iteration



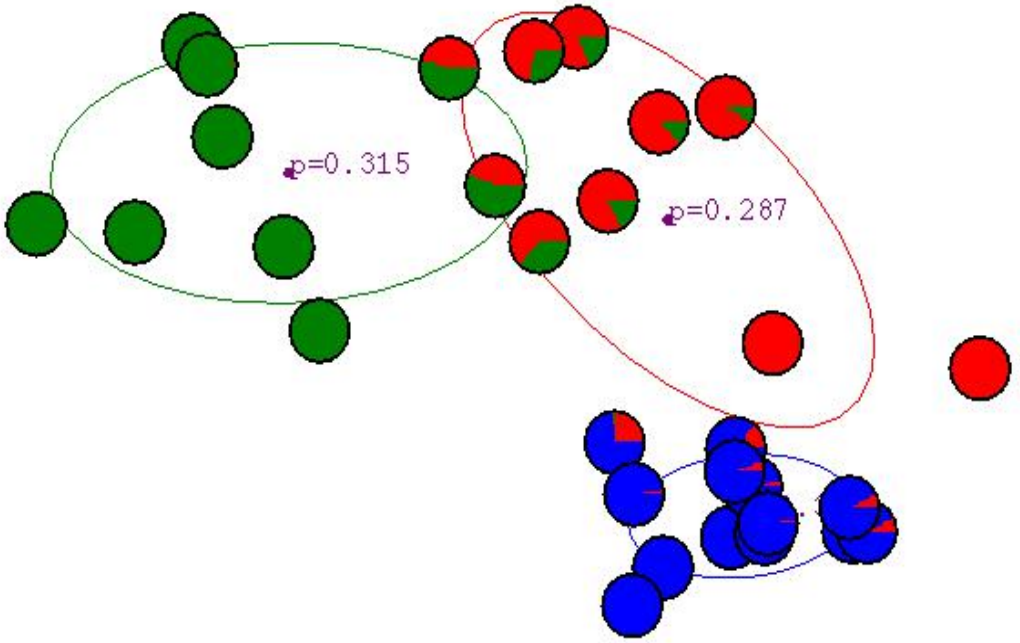
After 4th iteration



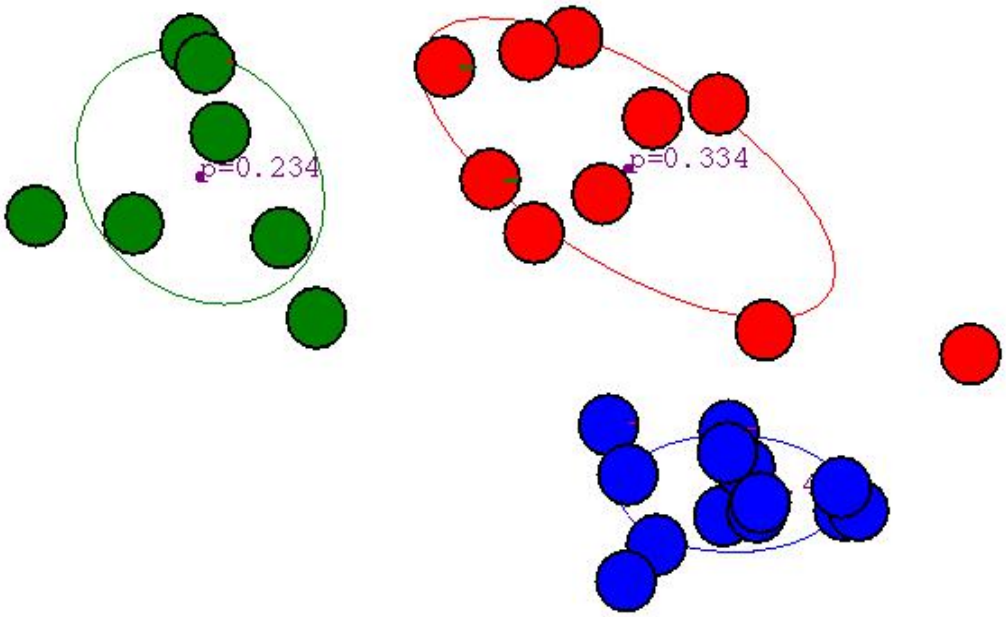
After 5th iteration



After 6th iteration



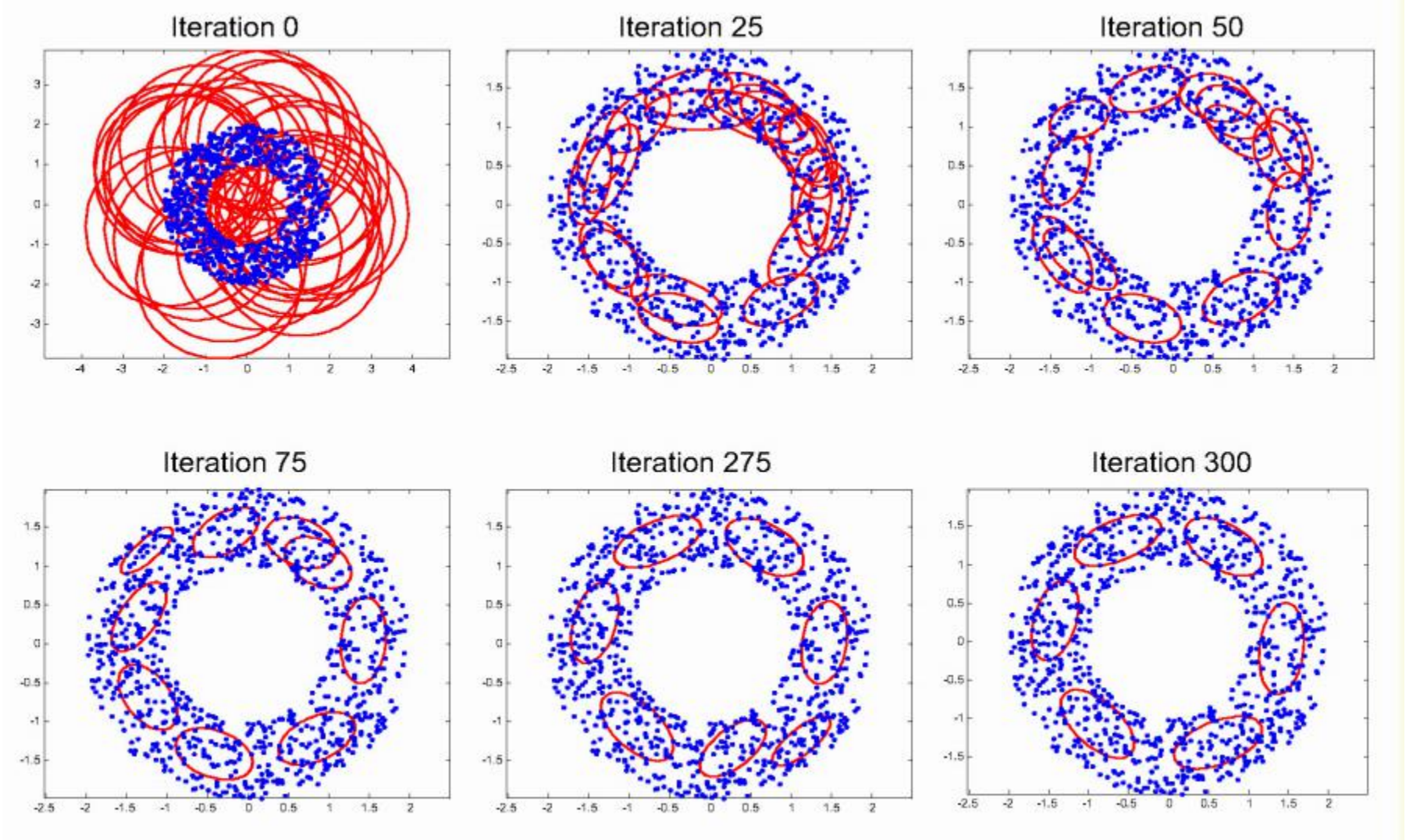
After 20th iteration



EM Example

- Example from R. Gutierrez-Osuna
- Training set of 900 examples forming an annulus
- Mixture model with m = 30 Gaussian components of unknown mean and variance is used
- Training:
  - Initialization:
    - means to 30 random examples
    - covaraince matrices initialized to be diagonal, with large variances on the diagonal (compared to the training data variance)
  - During EM training, components with small mixing coefficients were trimmed
    - This is a trick to get in a more compact model, with fewer than 30 Gaussian components

EM Example



from R. Gutierrez-Osuna

EM Texture Segmentation Example

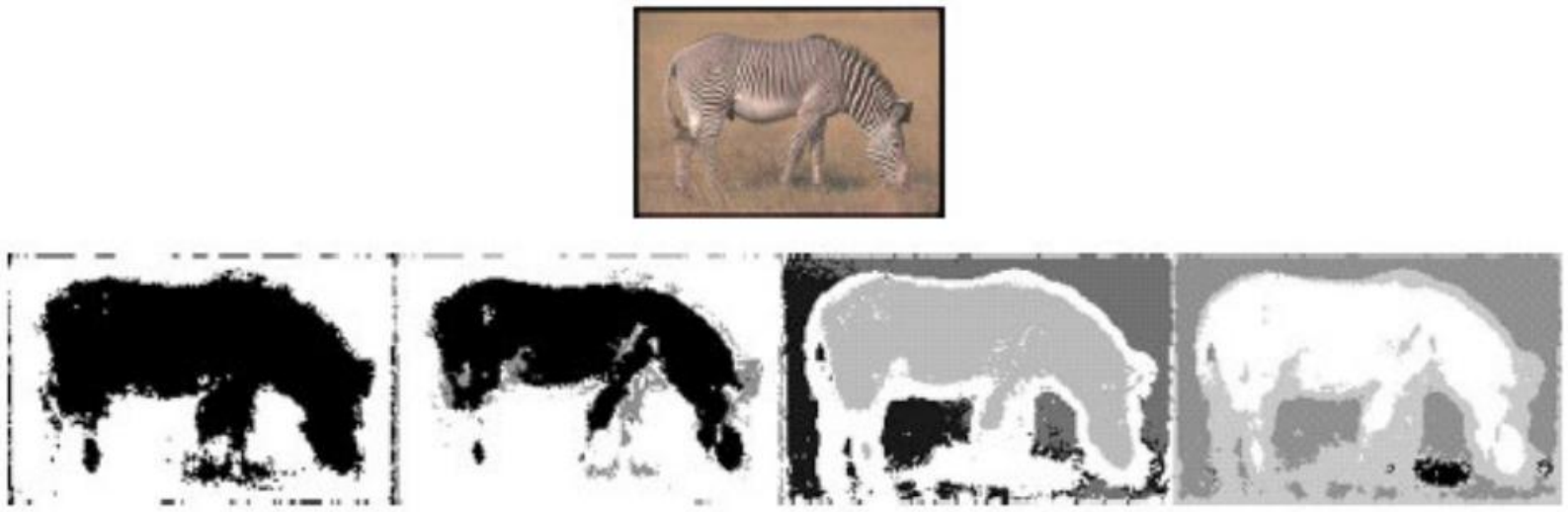


Figure from “Color and Texture Based Image Segmentation Using EM and Its Application to Content Based Image Retrieval”,S.J. Belongie et al., ICCV 1998

EM Motion Segmentation Example

Three frames from the MPEG “flower garden” sequence

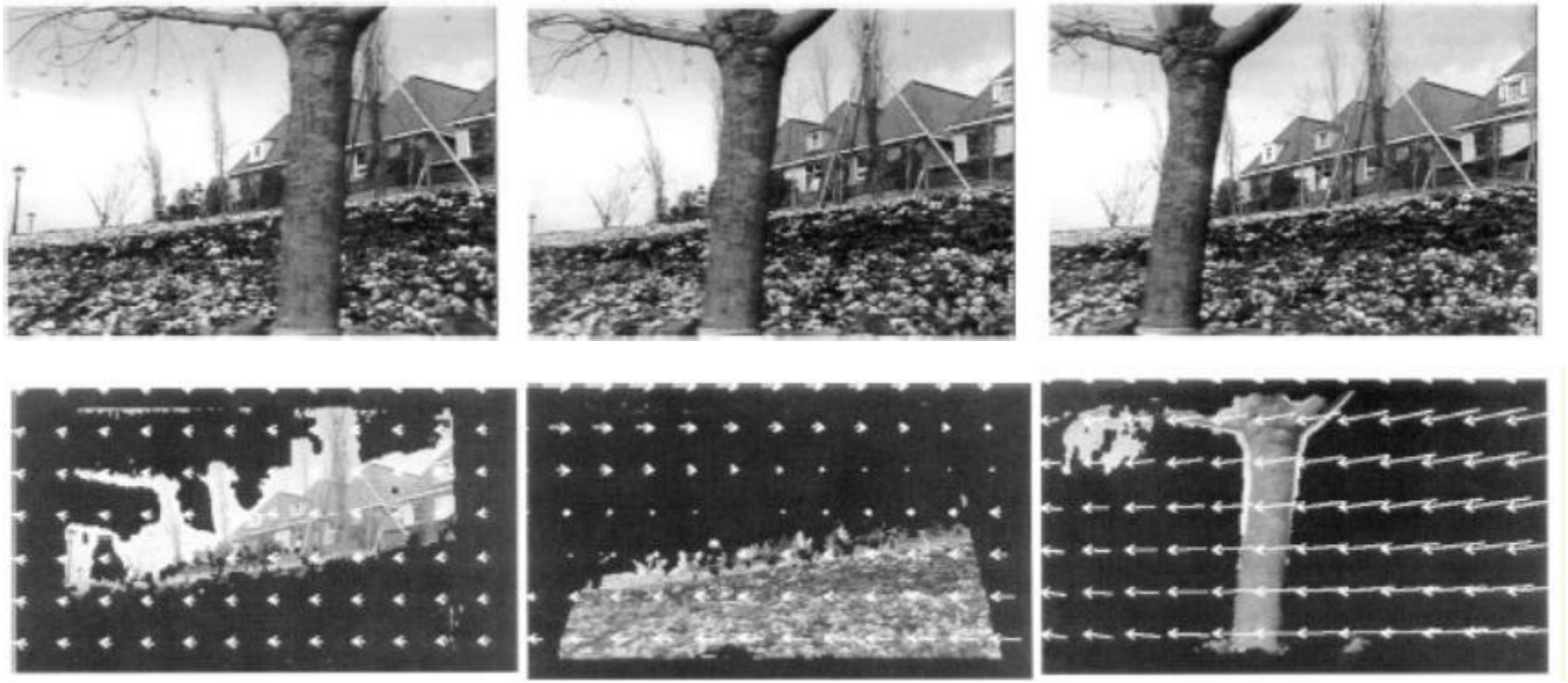


Figure from “Representing Images with layers,”, by J. Wang and E.H. Adelson, IEEE Transactions on Image Processing, 1994.

Summary

- Advantages
  - If the assumed data distribution is correct, the algorithm works well
- Disadvantages
  - If assumed data distribution is wrong, results can be quite bad.
  - In particular, bad results if use incorrect number of classes (i.e. the number of mixture components)

What You Should Know

- Mixture of Gaussians
- EM for mixture of Gaussians:
  - – Coordinate ascent, just like k-means
  - – How to “learn” maximum likelihood parameters (locally max. like.) in the case of unlabeled data
  - – Relation to k-means
- Hard / soft clustering
- Probabilistic model
- Remember, EM can get stuck in local minima,
  - – And empirically it DOES

Q & A