14.02.2020

Statistical Methods in AI (CSE/ECE 471)

Lecture-11: Unsupervised Learning (GMM, Hierarchical Clustering)

Ravi Kiran (ravi.kiran@iiit.ac.in)

Vineet Gandhi (v.gandhi@iiit.ac.in)

Center for Visual Information Technology (CVIT)

IIIT Hyderabad

Unsupervised Learning → Clustering

Group similar things e.g. images

[Goldberger et al.]

$$\{x^{(1)}, \ldots, x^{(m)}\} \qquad x^{(i)} \in \mathbb{R}^n$$

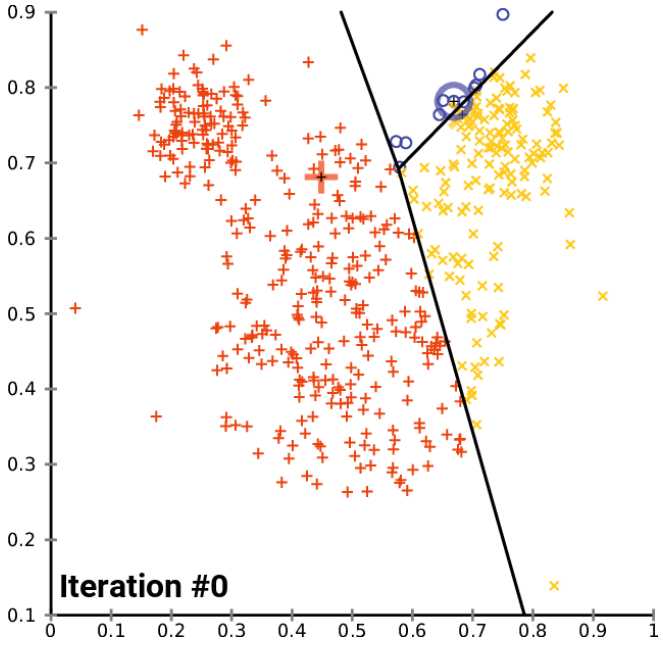
The k-means clustering algorithm is as follows:

1. Initialize **cluster centroids** $\mu_1, \mu_2, \ldots, \mu_k \in \mathbb{R}^n$ randomly.
2. Repeat until convergence: {

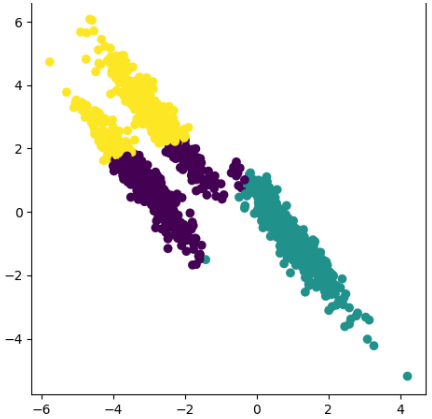
   For every i, set
   $$c^{(i)} := \arg\min_j ||x^{(i)} - \mu_j||^2.$$

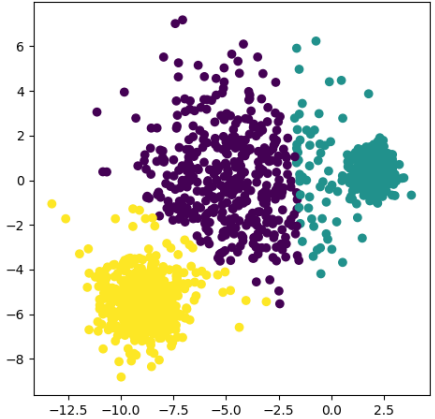
   For each j, set
   $$\mu_j := \frac{\sum_{i=1}^m 1\{c^{(i)} = j\} x^{(i)}}{\sum_{i=1}^m 1\{c^{(i)} = j\}}.$$

   }



- Can we have a distance-from-center based on ‘shape’ of the cluster ?
- Can we go beyond ‘hard’ assignments of points to clusters ?





Probabilistic Generative Model

Observed data is the ‘realization’ of a probabilistic model

Estimating Parameters of a Probabilistic Model [Maximum Likelihood Approach]

- Consider the estimation of heads probability of a coin tossed *n* times
- Heads probability p
- Data = HHTTHTHHTTT
- $L(p) = \Pr(D|p) = pp(1-p)(1-p)p(1-p)pp(1-p)(1-p)(1-p) = p^5(1-p)^6$

$$Data = \{X_1, X_2, \ldots X_n\}$$

- The likelihood function is the simultaneous density of the observation, as a function of the model parameters.

$$L(\Theta) = \Pr(Data|\Theta)$$

- If the observations are independent, we can decompose the term into

$$\Pr(Data\,|\,\Theta) = \prod_{i=1}^{n} \Pr(X_i\,|\,\Theta)$$

$$\boxed{\Theta^* = \arg\max_{\Theta} Pr(Data|\Theta)}$$

Maximum Likelihood

Take the derivative of L with respect to p:

$$\frac{dL}{dp} = 5p^4(1-p)^6 - 6p^5(1-p)^5$$

Equate it to zero and solve:

$$\hat{p} = 5/11$$

$\ln L = 5 \ln p + 6 \ln(1-p)$

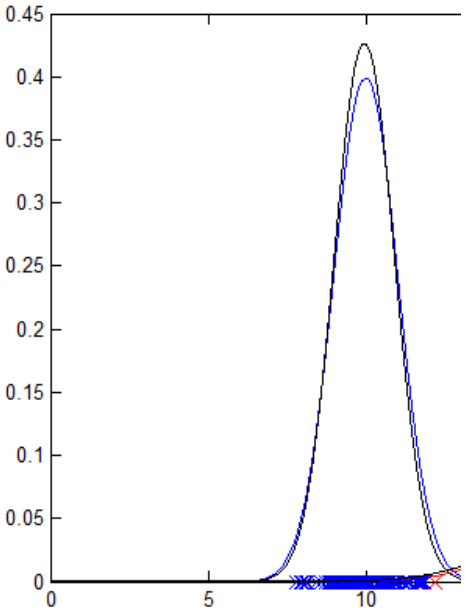
with derivative

$$\frac{d(\ln L)}{dp} = \frac{5}{p} - \frac{6}{(1-p)} = 0$$

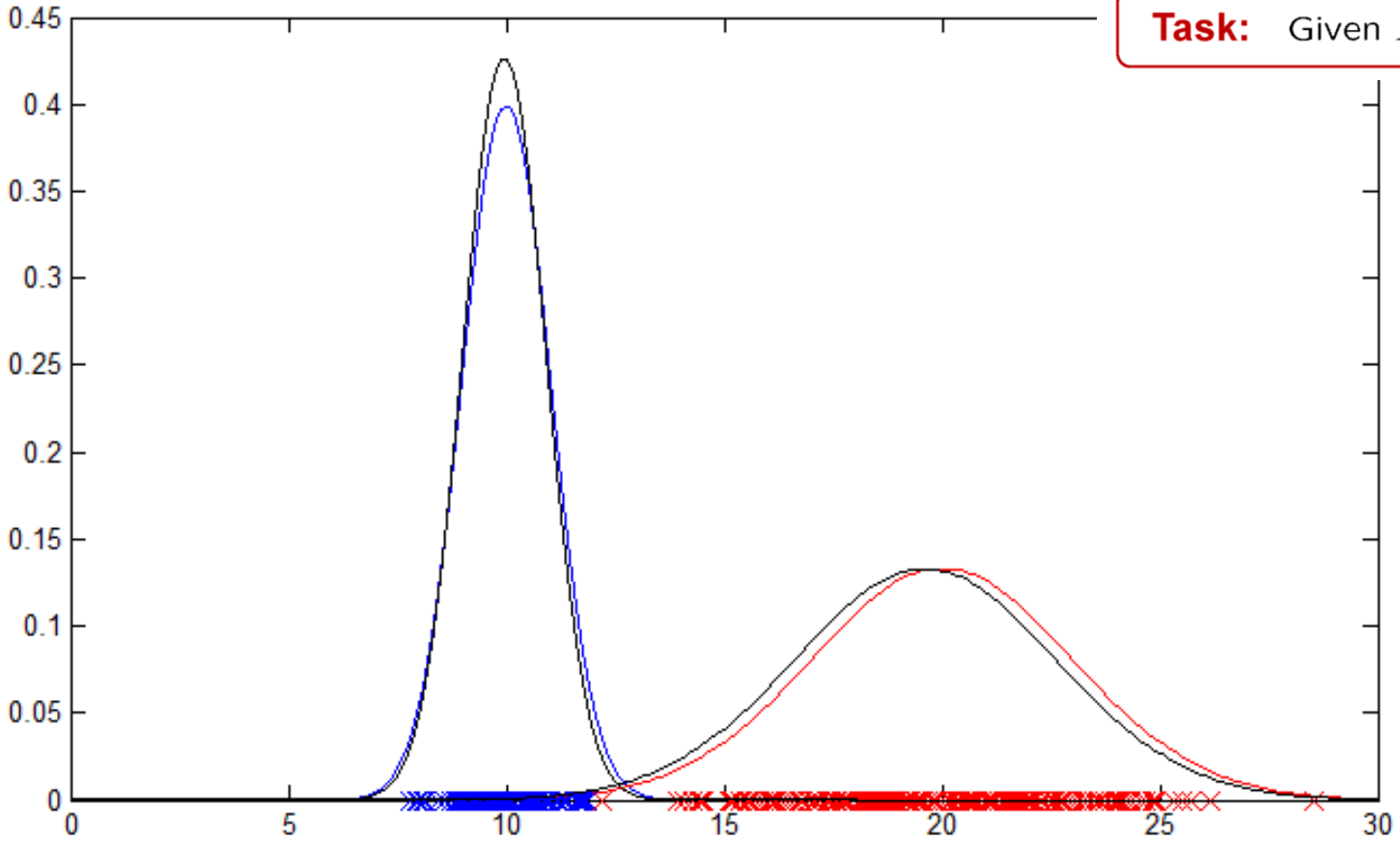
$$\hat{p} = 5/11$$

- Note: Knowing the parameters allows us to compute probability (density) of data
- Previously (k-means): Obtain cluster centers from cluster memberships
- Alternative: Obtain from probabilistic modelling of ‘cluster data density’

$$y = \frac{1}{\sigma\sqrt{2\pi}} e^{-\frac{(x-\mu)^2}{2\sigma^2}}$$



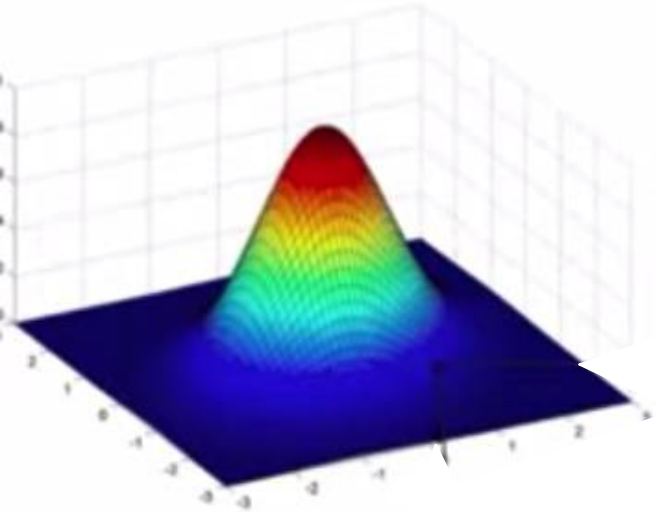
Data Probability Density is often Multi-modal



Task: Given $X \in \mathcal{X}$, learn $f(X)$.

Multivariate Gaussian

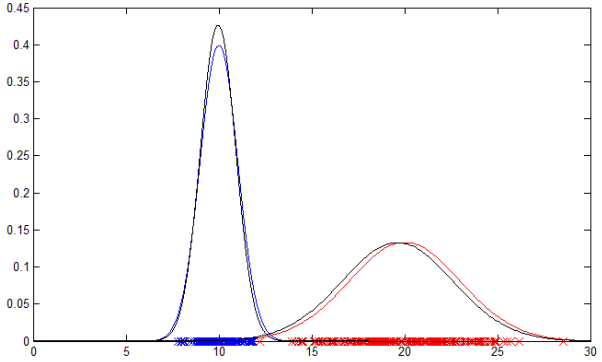
$$\mathcal{N}(\underline{x}\ ;\ \underline{\mu}, \Sigma) = \frac{1}{(2\pi)^{d/2}} |\Sigma|^{-1/2} \exp\left\{ -\frac{1}{2} (\underline{x} - \underline{\mu}) \Sigma^{-1} (\underline{x} - \underline{\mu})^T \right\}$$



μ = **length-d row vector**
Σ = **d x d matrix**

$|\Sigma|$ = **matrix determinant**

Mixture of Gaussians



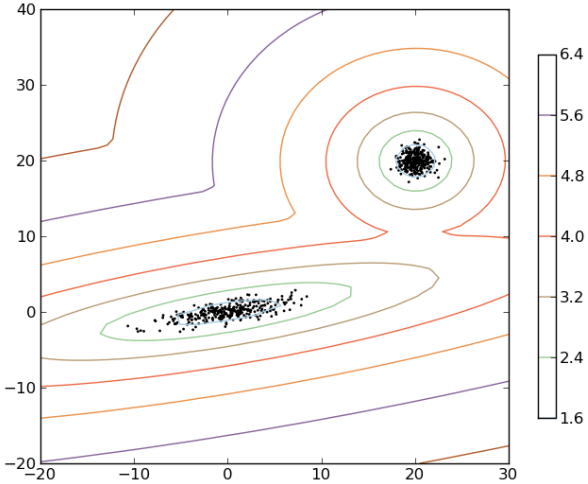
- Convex Combination of Distributions

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

- Normalization and positivity require

$$\sum_{k=1}^{K} \pi_k = 1 \qquad 0 \leqslant \pi_k \leqslant 1$$

$$p(\mathbf{x}) = \sum_{k=1}^{K} p(k)p(\mathbf{x}|k)$$

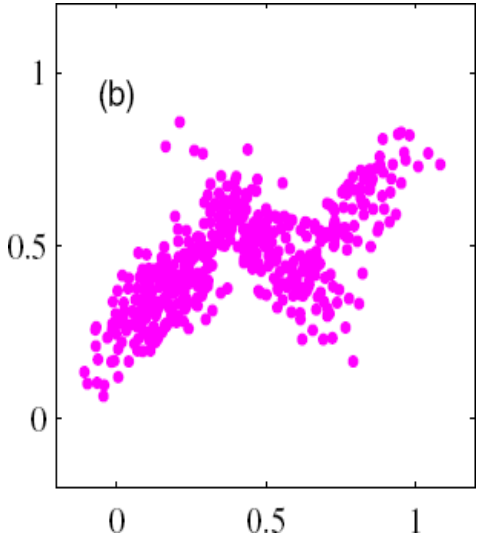


Bishop, 2003

MLE of Mixture Parameters

- However, MLE of mixture parameters is HARD!
- Joint distribution:

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



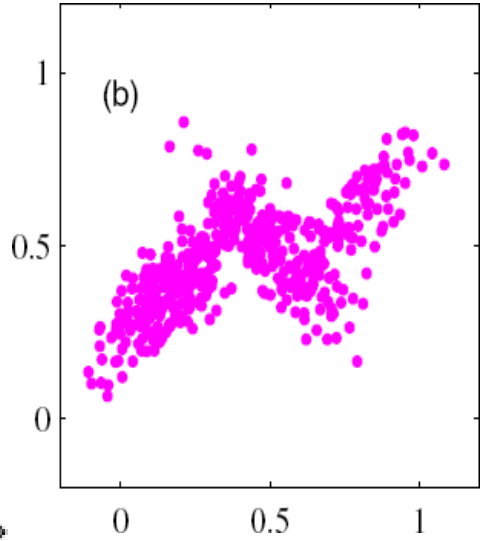
MLE of Mixture Parameters

- However, MLE of mixture parameters is HARD!
- Joint distribution:

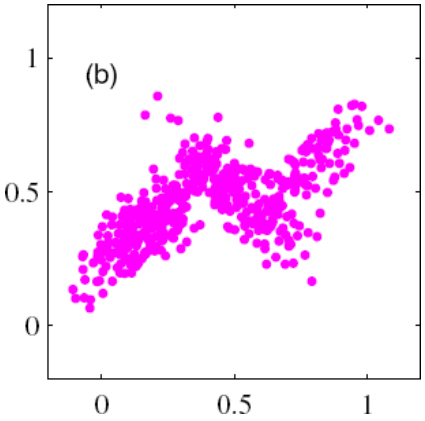
$$p(\mathbf{X}|\pi,\boldsymbol{\mu},\boldsymbol{\Sigma}) = \prod_{n=1}^{N} \sum_{k=1}^{K} \pi_k \mathcal{N}(\mathbf{x}_n \,|\boldsymbol{\mu}_k, \boldsymbol{\Sigma}_k)$$

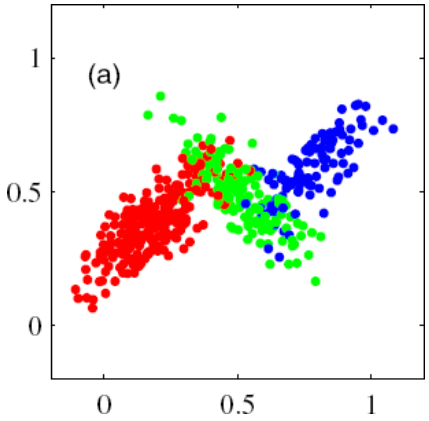
- Log likelihood

$$\ln p(\mathbf{X}|\pi,\boldsymbol{\mu},\boldsymbol{\Sigma}) = \sum_{n=1}^{N} \ln \underbrace{\left\{ \sum_{k=1}^{K} \pi_k \mathcal{N}(\mathbf{x}_n|\boldsymbol{\mu}_k, \boldsymbol{\Sigma}_k) \right\}}_{\text{Uh-oh, log of a sum}}$$



EM Algorithm



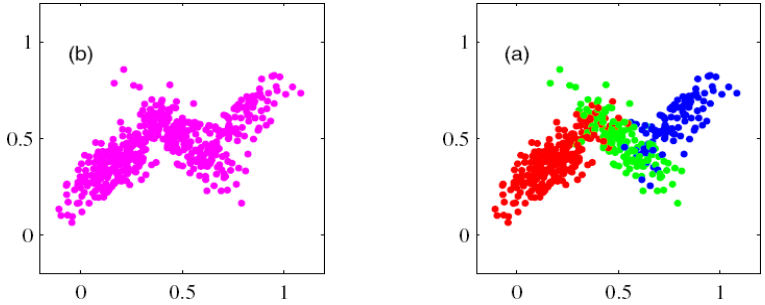


Suppose some oracle told us which point comes from which Gaussian.

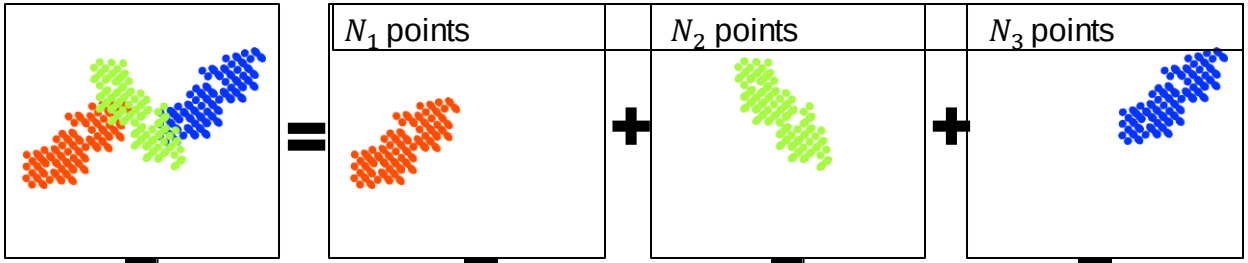
Robert Collins
CSE586

EM Algorithm

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



We can easily estimate each Gaussian, along with the mixture weights!



estimate $\pi_k = N_k/\mathrm{N}$

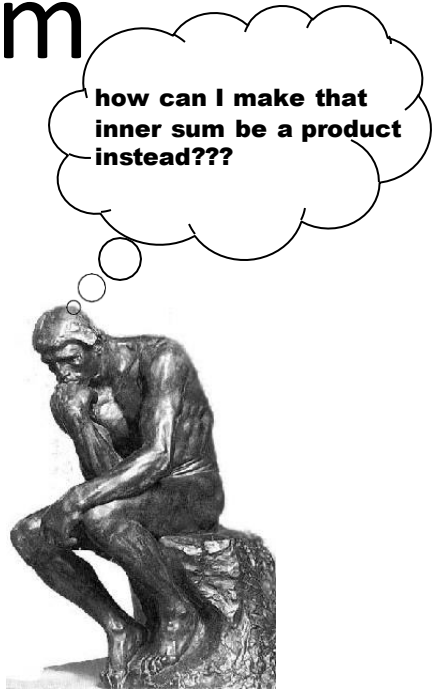
Estimate μ_1, Σ_1

Estimate μ_2, Σ_2

Estimate μ_3, Σ_3

Robert Collins
CSE586

EM Algorithm



Remember that this was a problem...

$$\ln p(\mathbf{X}|\boldsymbol{\pi}, \boldsymbol{\mu}, \boldsymbol{\Sigma}) = \sum_{n=1}^{N} \ln \left\{ \sum_{k=1}^{K} \pi_k \mathcal{N}(\mathbf{x}_n | \boldsymbol{\mu}_k, \boldsymbol{\Sigma}_k) \right\}$$

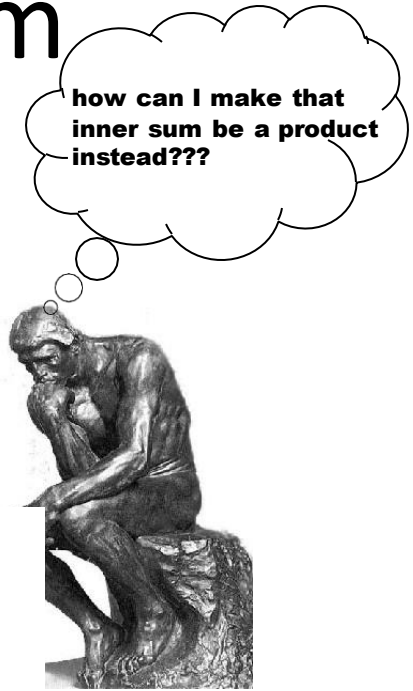
Suppose some oracle told us which point comes from which Gaussian.

How? By providing a "latent" variable $z_{n,k}$

- $z_{n,k}$= 1 if point n comes from the kth Gaussian
- = 0 otherwise

Robert Collins
CSE586

EM Algorithm



Remember that this was a problem...

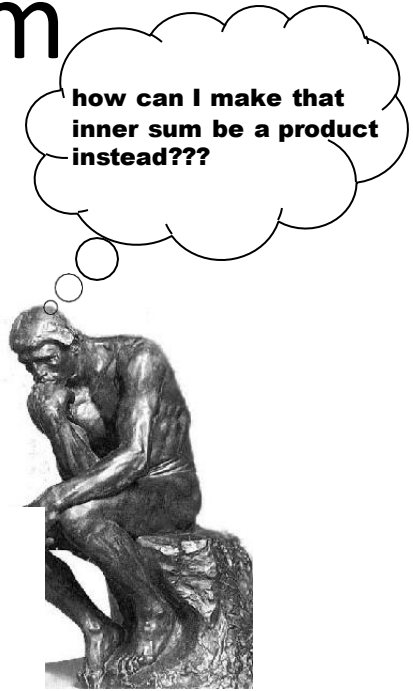
$$\ln p(\mathbf{X}|\boldsymbol{\pi},\boldsymbol{\mu},\boldsymbol{\Sigma}) = \sum_{n=1}^{N} \ln\left\{\sum_{k=1}^{K} \pi_k \mathcal{N}(\mathbf{x}_n|\boldsymbol{\mu}_k,\boldsymbol{\Sigma}_k)\right\}$$

If some oracle told us which point comes from which Gaussian, we could work with the complete log likelihood

$$p(\mathbf{X},\mathbf{Z}|\boldsymbol{\mu},\boldsymbol{\Sigma},\boldsymbol{\pi}) = \prod_{n=1}^{N}\prod_{k=1}^{K} \pi_k^{z_{nk}} \mathcal{N}(\mathbf{x}_n|\boldsymbol{\mu}_k,\boldsymbol{\Sigma}_k)^{z_{nk}}$$

Robert Collins
CSE586

EM Algorithm



Remember that this was a problem...

$$\ln p(\mathbf{X}|\boldsymbol{\pi},\boldsymbol{\mu},\boldsymbol{\Sigma}) = \sum_{n=1}^{N} \ln\left\{\sum_{k=1}^{K} \pi_k \mathcal{N}(\mathbf{x}_n|\boldsymbol{\mu}_k,\boldsymbol{\Sigma}_k)\right\}$$

If some oracle told us which point comes from which Gaussian, we could work with the complete log likelihood

$$p(\mathbf{X},\mathbf{Z}|\boldsymbol{\mu},\boldsymbol{\Sigma},\boldsymbol{\pi}) = \prod_{n=1}^{N}\prod_{k=1}^{K} \pi_k^{z_{nk}} \mathcal{N}(\mathbf{x}_n|\boldsymbol{\mu}_k,\boldsymbol{\Sigma}_k)^{z_{nk}}$$

and the log of that looks much better!

$$\ln p(\mathbf{X},\mathbf{Z}|\boldsymbol{\mu},\boldsymbol{\Sigma},\boldsymbol{\pi}) = \sum_{n=1}^{N}\sum_{k=1}^{K} z_{nk}\left\{\ln \pi_k + \ln \mathcal{N}(\mathbf{x}_n|\boldsymbol{\mu}_k,\boldsymbol{\Sigma}_k)\right\}.$$

Latent Variable View

$$\ln p(\mathbf{X}, \mathbf{Z}|\boldsymbol{\mu}, \boldsymbol{\Sigma}, \boldsymbol{\pi}) = \sum_{n=1}^{N} \sum_{k=1}^{K} z_{nk} \{\ln \pi_k + \ln \mathcal{N}(\mathbf{x}_n|\boldsymbol{\mu}_k, \boldsymbol{\Sigma}_k)\}.$$

note: for a given n, there are k of these latent variables, and only ONE of them is 1 (all the rest are 0)

Latent Variable View

$$\ln p(\mathbf{X}, \mathbf{Z}|\boldsymbol{\mu}, \boldsymbol{\Sigma}, \boldsymbol{\pi}) = \sum_{n=1}^{N} \sum_{k=1}^{K} z_{nk} \{\ln \pi_k + \ln \mathcal{N}(\mathbf{x}_n|\boldsymbol{\mu}_k, \boldsymbol{\Sigma}_k)\}.$$

note: for a given n, there are k of these latent variables, and only ONE of them is 1 (all the rest are 0)

This is thus equivalent to

$$\sum_{\substack{\text{all n for which} \\ z_{n,1}=1}} \ln \pi_1 + \ln \mathcal{N}(x_n|\mu_1, \Sigma_1)$$

$$+ \sum_{\substack{\text{all n for which} \\ z_{n,2}=1}} \ln \pi_2 + \ln \mathcal{N}(x_n|\mu_2, \Sigma_2) \quad + \quad \cdots \quad +$$

$$+ \sum_{\substack{\text{all n for which} \\ z_{n,K}=1}} \ln \pi_K + \ln \mathcal{N}(x_n|\mu_K, \Sigma_K)$$

Latent Variable View

$$\sum_{\substack{\text{all n for which} \\ z_{n,1}=1}} \ln \pi_1 + \sum_{\substack{\text{all n for which} \\ z_{n,1}=1}} \ln \mathcal{N}(x_n|\mu_1, \Sigma_1)$$

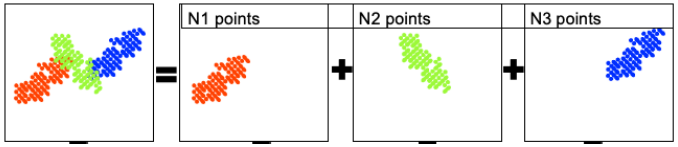
$$+ \sum_{\substack{\text{all n for which} \\ z_{n,2}=1}} \ln \pi_2 + \sum_{\substack{\text{all n for which} \\ z_{n,2}=1}} \ln \mathcal{N}(x_n|\mu_2, \Sigma_2)$$

$$+ \quad \cdots \quad +$$

$$+ \sum_{\substack{\text{all n for which} \\ z_{n,K}=1}} \ln \pi_K + \sum_{\substack{\text{all n for which} \\ z_{n,K}=1}} \ln \mathcal{N}(x_n|\mu_K, \Sigma_K)$$

Robert Collins
CSE586

Latent Variable View



$$\sum_{\substack{\text{all n for which}\\ z_{n,1}=1}} \ln \pi_1 + \boxed{\sum_{\substack{\text{all n for which}\\ z_{n,1}=1}} \ln \mathcal{N}(x_n|\mu_1,\Sigma_1)}$$

can be estimated separately

$$+ \sum_{\substack{\text{all n for which}\\ z_{n,2}=1}} \ln \pi_2 + \boxed{\sum_{\substack{\text{all n for which}\\ z_{n,2}=1}} \ln \mathcal{N}(x_n|\mu_2,\Sigma_2)}$$

can be estimated separately

$$+ \quad \cdots \quad +$$

$$+ \sum_{\substack{\text{all n for which}\\ z_{n,K}=1}} \ln \pi_K + \boxed{\sum_{\substack{\text{all n for which}\\ z_{n,K}=1}} \ln \mathcal{N}(x_n|\mu_K,\Sigma_K)}$$

can be estimated separately

Latent Variable View

$$\sum_{\substack{\text{all n for which}\\ z_{n,1}=1}} \ln \pi_1 + \sum_{\substack{\text{all n for which}\\ z_{n,1}=1}} \ln \mathcal{N}(x_n|\mu_1,\Sigma_1)$$

can be estimated separately

$$+ \sum_{\substack{\text{all n for which}\\ z_{n,2}=1}} \ln \pi_2 + \sum_{\substack{\text{all n for which}\\ z_{n,2}=1}} \ln \mathcal{N}(x_n|\mu_2,\Sigma_2)$$

can be estimated separately

$$+ \cdots +$$

$$+ \sum_{\substack{\text{all n for which}\\ z_{n,K}=1}} \ln \pi_K + \sum_{\substack{\text{all n for which}\\ z_{n,K}=1}} \ln \mathcal{N}(x_n|\mu_K,\Sigma_K)$$

can be estimated separately

these are coupled because the mixing weights all sum to 1, but it is no big deal to solve

Insight

- Since we don't know the latent variables, we instead take the expected value of the log likelihood with respect to their posterior distribution P(z|x,theta).
- In the GMM case, this is equivalent to "softening" the binary latent variables to continuous ones (the expected values of the latent variables)

$$\ln p(\mathbf{x}, \mathbf{z}|\boldsymbol{\theta}) = \sum_{n=1}^{N} \sum_{k=1}^{K} \underbrace{z_{nk}}_{\text{unknown discrete value 0 or 1}} \{\ln \pi_k + \ln \mathcal{N}(\mathbf{x}_n|\boldsymbol{\mu}_k, \boldsymbol{\Sigma}_k)\}$$

$$\mathrm{E}_{\mathbf{z}}[\ln p(\mathbf{x}, \mathbf{z}|\boldsymbol{\theta})] = \sum_{n=1}^{N} \sum_{k=1}^{K} \underbrace{\gamma_k(\mathbf{x}_n)}_{\text{known continuous value between 0 and 1}} \{\ln \pi_k + \ln \mathcal{N}(\mathbf{x}_n|\boldsymbol{\mu}_k, \boldsymbol{\Sigma}_k)\}$$

Where $\gamma_j(\mathbf{x}_n)$ is $P(z_{nk} = 1)$

Robert Collins
CSE586

EM Algorithm for GMM

E

$$\gamma_j(\mathbf{x}_n) = \frac{\pi_j \mathcal{N}(\mathbf{x}_n|\boldsymbol{\mu}_j, \boldsymbol{\Sigma}_j)}{\sum_k \pi_k \mathcal{N}(\mathbf{x}_n|\boldsymbol{\mu}_k, \boldsymbol{\Sigma}_k)}$$

ownership weights

M

$$\boldsymbol{\mu}_j = \frac{\sum_{n=1}^{N} \gamma_j(\mathbf{x}_n)\mathbf{x}_n}{\sum_{n=1}^{N} \gamma_j(\mathbf{x}_n)}$$

means

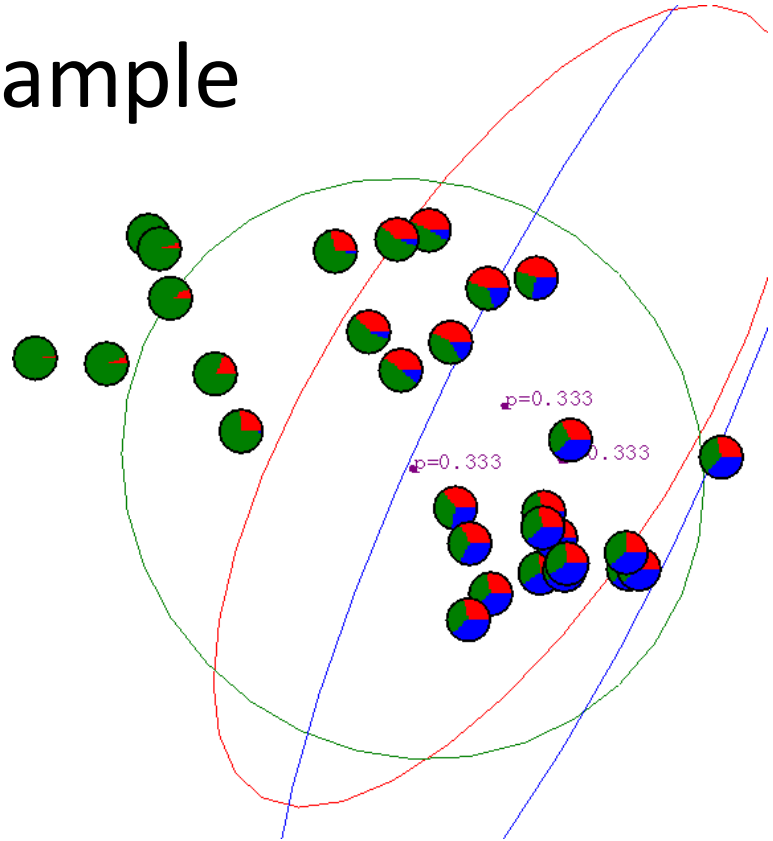
$$\boldsymbol{\Sigma}_j = \frac{\sum_{n=1}^{N} \gamma_j(\mathbf{x}_n)(\mathbf{x}_n - \boldsymbol{\mu}_j)(\mathbf{x}_n - \boldsymbol{\mu}_j)^\top}{\sum_{n=1}^{N} \gamma_j(\mathbf{x}_n)}$$

covariances

$$\pi_j = \frac{1}{N} \sum_{n=1}^{N} \gamma_j(\mathbf{x}_n)$$

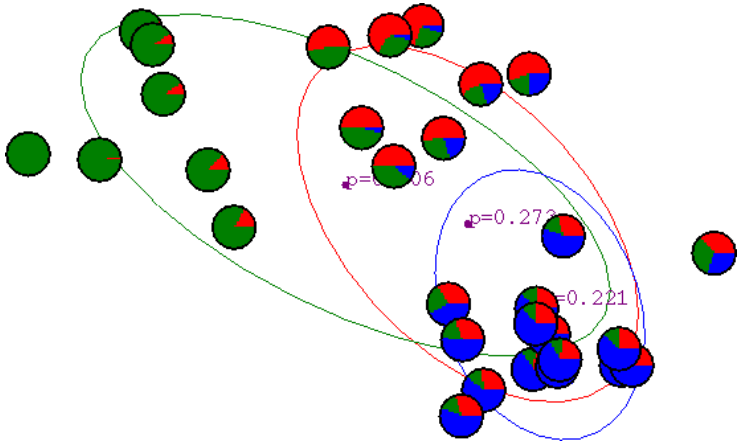
mixing probabilities

Gaussian Mixture Example

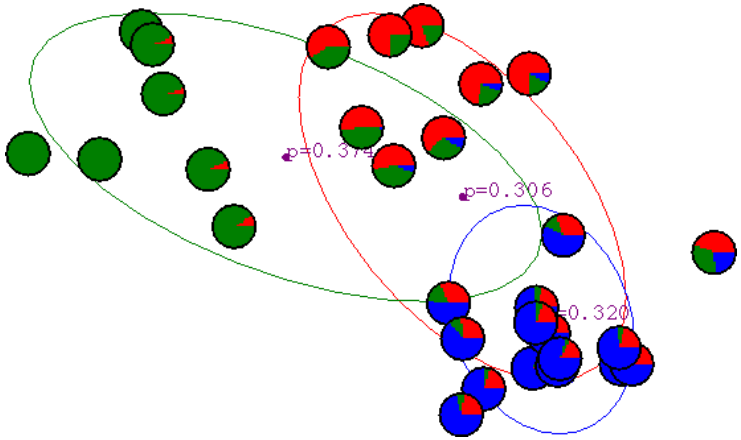


Robert Collins
CSE586

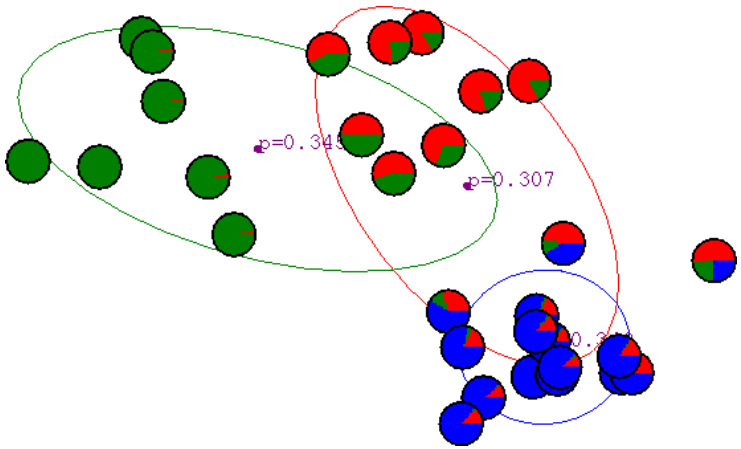
After first iteration



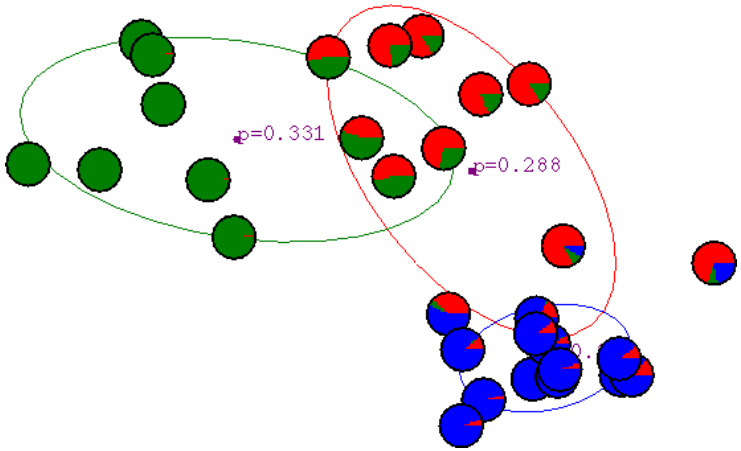
After 2nd iteration



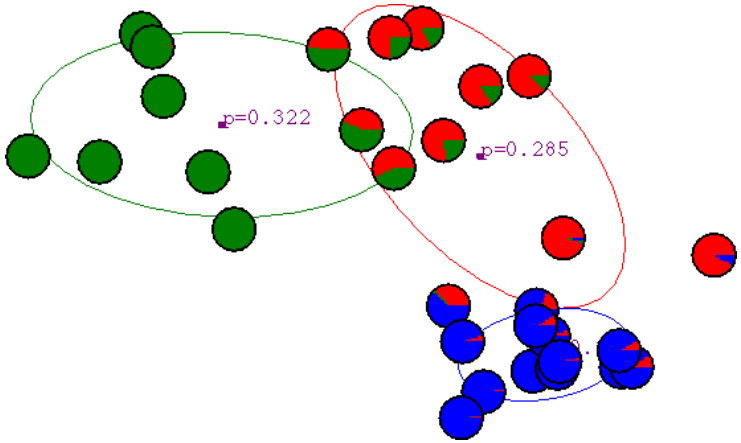
After 3rd iteration



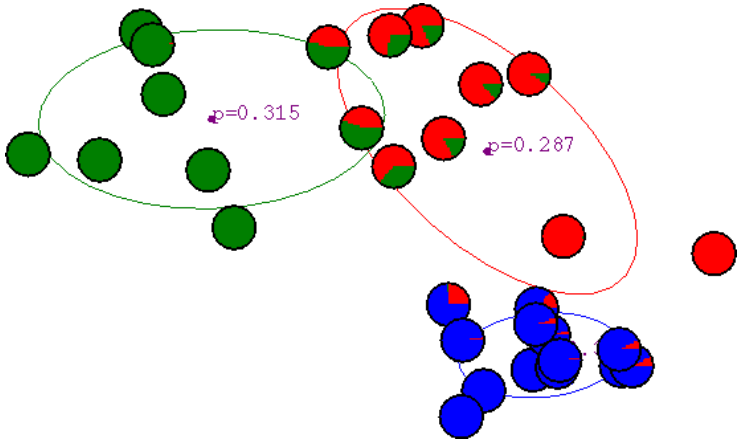
After 4th iteration



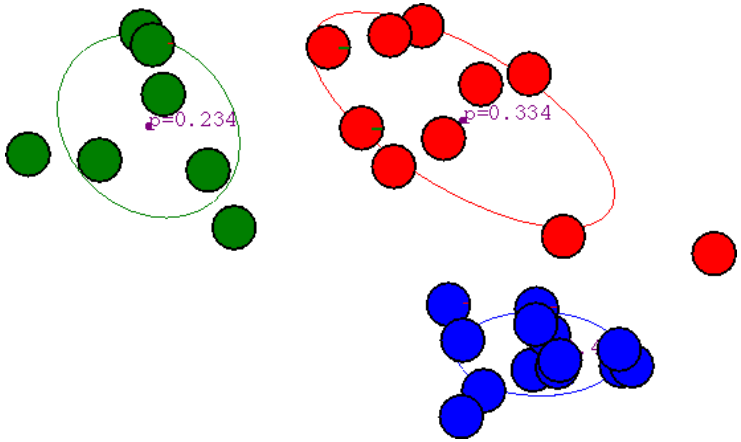
After 5th iteration



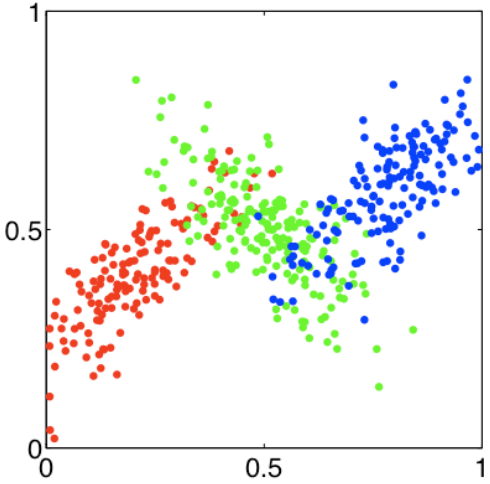
After 6th iteration



After 20th iteration

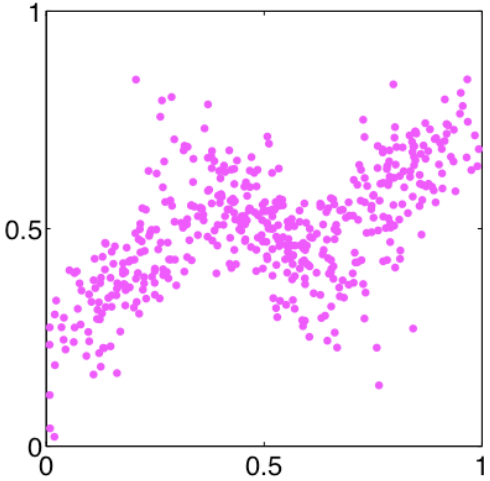


Recall: Labeled vs Unlabeled Data



labeled

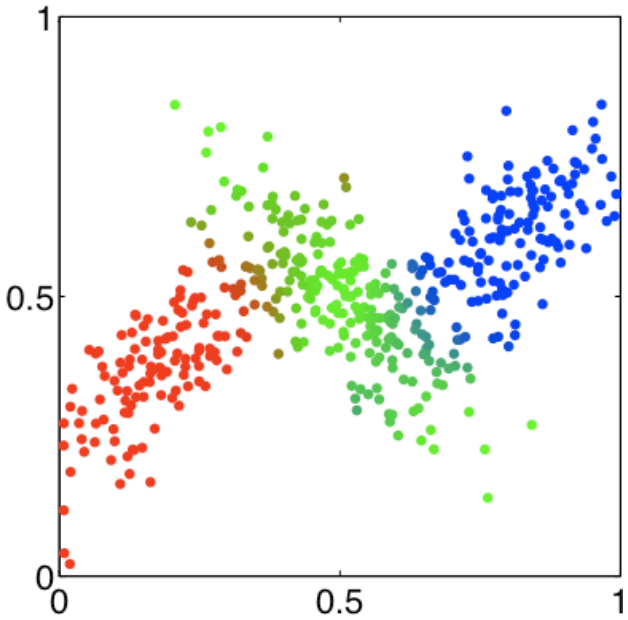
Easy to estimate params
(do each color separately)



unlabeled

Hard to estimate params
(we need to assign colors)

EM produces a “Soft” labeling



each point makes a weighted contribution
to the estimation of ALL components

Insight

E $$\gamma_j(\mathbf{x}_n) = \frac{\pi_j \mathcal{N}(\mathbf{x}_n|\boldsymbol{\mu}_j, \boldsymbol{\Sigma}_j)}{\sum_k \pi_k \mathcal{N}(\mathbf{x}_n|\boldsymbol{\mu}_k, \boldsymbol{\Sigma}_k)}$$ ownership weights

M $$\boldsymbol{\mu}_j = \frac{\sum_{n=1}^{N} \gamma_j(\mathbf{x}_n)\mathbf{x}_n}{\sum_{n=1}^{N} \gamma_j(\mathbf{x}_n)}$$ means

$$\boldsymbol{\Sigma}_j = \frac{\sum_{n=1}^{N} \gamma_j(\mathbf{x}_n)(\mathbf{x}_n - \boldsymbol{\mu}_j)(\mathbf{x}_n - \boldsymbol{\mu}_j)^\top}{\sum_{n=1}^{N} \gamma_j(\mathbf{x}_n)}$$ covariances

$$\pi_j = \frac{1}{N}\sum_{n=1}^{N} \gamma_j(\mathbf{x}_n)$$ mixing probabilities

Replacing the binary latent variables with their continuous expected values:

- all points contribute to the estimation of all components
- each point has unit mass to contribute, but splits it across the K components
- the amount of weight a point contributes to a component is proportional to the relative likelihood that the point was generated by that component

Review of EM for GMMs

E

$$\gamma(z_{nj}) = \frac{\pi_j \mathcal{N}(\mathbf{x}_n | \boldsymbol{\mu}_j, \boldsymbol{\Sigma}_j)}{\sum_k \pi_k \mathcal{N}(\mathbf{x}_n | \boldsymbol{\mu}_k, \boldsymbol{\Sigma}_k)}$$

ownership weights
(soft labels)

M

$$\boldsymbol{\mu}_j = \frac{\sum_{n=1}^{N} \gamma(z_{nj})\, \mathbf{x}_n}{\sum_{n=1}^{N} \gamma(z_{nj})}$$

means

$$\boldsymbol{\Sigma}_j = \frac{\sum_{n=1}^{N} \gamma(z_{nj})\, (\mathbf{x}_n - \boldsymbol{\mu}_j)(\mathbf{x}_n - \boldsymbol{\mu}_j)^\top}{\sum_{n=1}^{N} \gamma(z_{nj})}$$

covariances

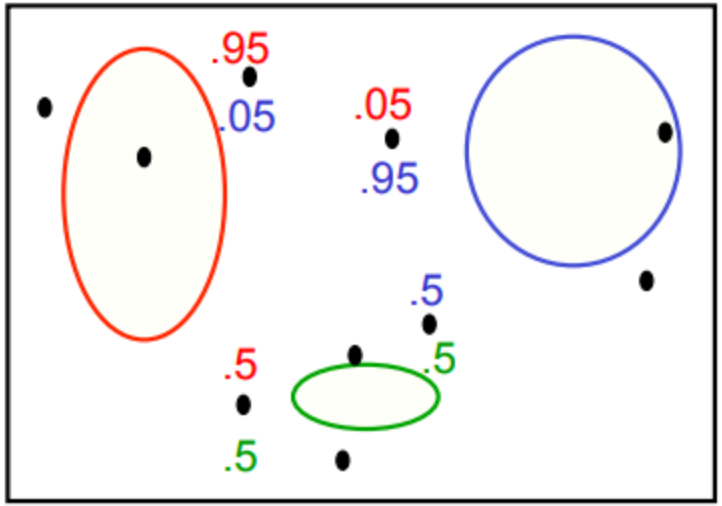
$$\pi_j = \frac{1}{N} \sum_{n=1}^{N} \gamma(z_{nj})$$

mixing weights

Alternate E and M steps to convergence.

- Optimization uses the Expectation Maximization algorithm, which alternates between two steps:
  1. E-step: Compute the posterior probability over z given our current model - i.e. how much do we think each Gaussian generates each datapoint.

$$\ell(\boldsymbol{\pi}, \mu, \Sigma) = \ln p(\mathbf{X}|\pi, \mu, \Sigma) = \sum_{n=1}^{N} \ln p(\mathbf{x}^{(n)}|\pi, \mu, \Sigma)$$
$$= \sum_{n=1}^{N} \ln \sum_{z^{(n)}=1}^{K} p(\mathbf{x}^{(n)}|\, z^{(n)}; \mu, \Sigma) p(z^{(n)}|\, \boldsymbol{\pi})$$

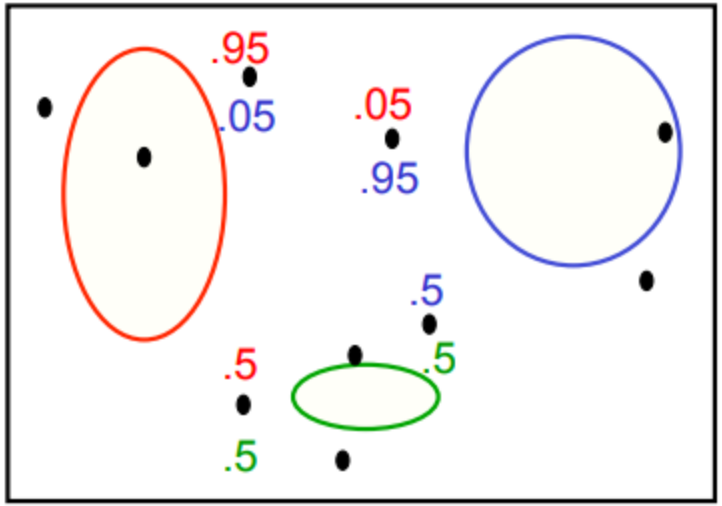


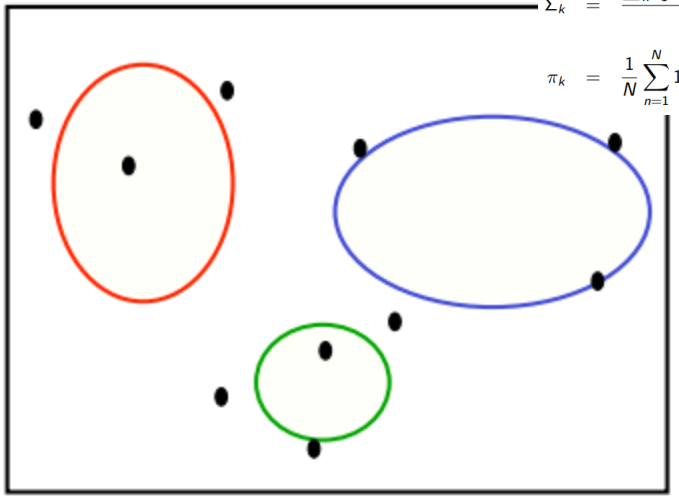
- Optimization uses the Expectation Maximization algorithm, which alternates between two steps:
  1. E-step: Compute the posterior probability over z given our current model - i.e. how much do we think each Gaussian generates each datapoint.
  2. M-step: Assuming that the data really was generated this way, change the parameters of each Gaussian to maximize the probability that it would generate the data it is currently responsible for.

$$\mu_k = \frac{\sum_{n=1}^{N} 1_{[z^{(n)}=k]} \mathbf{x}^{(n)}}{\sum_{n=1}^{N} 1_{[z^{(n)}=k]}}$$

$$\Sigma_k = \frac{\sum_{n=1}^{N} 1_{[z^{(n)}=k]} (\mathbf{x}^{(n)} - \mu_k)(\mathbf{x}^{(n)} - \mu_k)^T}{\sum_{n=1}^{N} 1_{[z^{(n)}=k]}}$$

$$\pi_k = \frac{1}{N} \sum_{n=1}^{N} 1_{[z^{(n)}=k]}$$





GMM - Algorithm

- Initialize the means μ_k, covariances Σ_k and mixing coefficients π_k
- Iterate until convergence:
  - E-step: Evaluate the responsibilities given current parameters

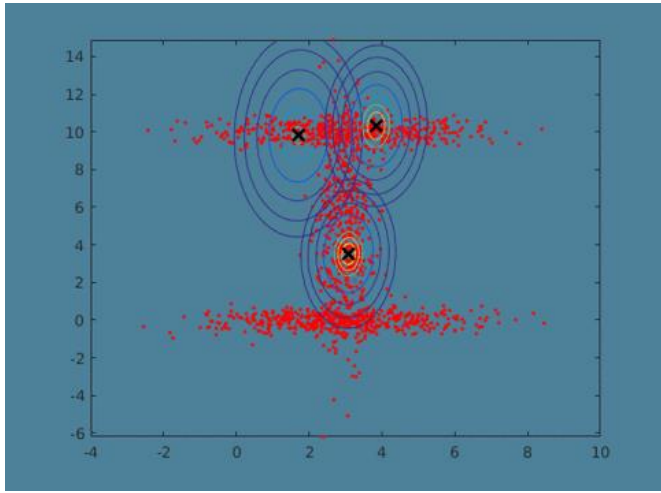
$$\gamma_k^{(n)} = p(z^{(n)}|\mathbf{x}) = \frac{\pi_k \mathcal{N}(\mathbf{x}^{(n)}|\mu_k, \Sigma_k)}{\sum_{j=1}^{K} \pi_j \mathcal{N}(\mathbf{x}^{(n)}|\mu_j, \Sigma_j)}$$

  - M-step: Re-estimate the parameters given current responsibilities

$$\begin{aligned} \mu_k &= \frac{1}{N_k} \sum_{n=1}^{N} \gamma_k^{(n)} \mathbf{x}^{(n)} \\ \Sigma_k &= \frac{1}{N_k} \sum_{n=1}^{N} \gamma_k^{(n)} (\mathbf{x}^{(n)} - \mu_k)(\mathbf{x}^{(n)} - \mu_k)^T \\ \pi_k &= \frac{N_k}{N} \quad \text{with} \quad N_k = \sum_{n=1}^{N} \gamma_k^{(n)} \end{aligned}$$

  - Evaluate log likelihood and check for convergence

$$\ln p(\mathbf{X}|\pi, \mu, \Sigma) = \sum_{n=1}^{N} \ln \left(\sum_{k=1}^{K} \pi_k \mathcal{N}(\mathbf{x}^{(n)}|\mu_k, \Sigma_k) \right)$$



- The K-Means Algorithm:
  1. Assignment step: Assign each data point to the closest cluster
  2. Refitting step: Move each cluster center to the center of gravity of the data assigned to it
- The EM Algorithm:
  1. E-step: Compute the posterior probability over z given our current model
  2. M-step: Maximize the probability that it would generate the data it is currently responsible for.

From EM to K–Means

Alternative explanation of K-means!

- Fix all mixing weights to 1/K
  [drop out of the estimation]
- Fix all covariances to $\sigma^2 I$
  [drop out of the estimation so we only have to estimate the means; each Gaussian likelihood becomes inversely proportional to distance from a mean]
- Take limit as σ^2 goes to 0
  [this forces weights to become binary]

From EM to K–Means

E

$$\gamma(z_{nj}) = \frac{\pi_j \mathcal{N}(\mathbf{x}_n|\boldsymbol{\mu}_j, \boldsymbol{\Sigma}_j)}{\sum_k \pi_k \mathcal{N}(\mathbf{x}_n|\boldsymbol{\mu}_k, \boldsymbol{\Sigma}_k)}$$

ownership weights (soft labels)

$$= \frac{\exp(-\frac{1}{2\sigma^2}\|x_n - \mu_j\|^2)}{\sum_{k=1}^{K} \exp(-\frac{1}{2\sigma^2}\|x_n - \mu_k\|^2)}$$

after fixing mixing weights and covariances as described on last slide

now divide top and bottom by $\exp(-\frac{1}{2\sigma^2} d_{min}^2)$ where $d_{min}^2 = \min_k \|x_n - \mu_k\|^2$

and take limit as σ^2 goes to 0

$$\gamma(z_{nj}) = z_{nj} = \begin{cases} 1 & \text{if } \mu_j \text{ is closest mean to } x_n \\ 0 & \text{otherwise} \end{cases}$$

hard labels, as in the K–means algorithm

K–Means Algorithm

- Given N data points $x_1, x_2, \ldots, x_N$
- Find K cluster centers $\mu_1, \mu_2, \ldots, \mu_K$ to minimize $\sum_{n=1}^{N} \sum_{k=1}^{K} z_{nk} \|x_n - \mu_j\|^2$
  (z_{nk} is 1 if point n belongs to cluster k; 0 otherwise)
- Algorithm:
  - initialize K cluster centers $\mu_1, \mu_2, \ldots, \mu_K$
  - repeat
    - set z_{nk} labels to assign each point to closest cluster center **E**
    - revise each cluster center μ_j to be center of mass of points in that cluster $\mu_j = \frac{\sum_{n=1}^{N} z_{nj}\, x_n}{\sum_{n=1}^{N} z_{nj}}$ **M**
  - until convergence (e.g. z_{nk} labels don't change)

How to choose k ?

- Simple: Pick a ‘k’ which generates maximum likelihood for a ‘hold out’ set

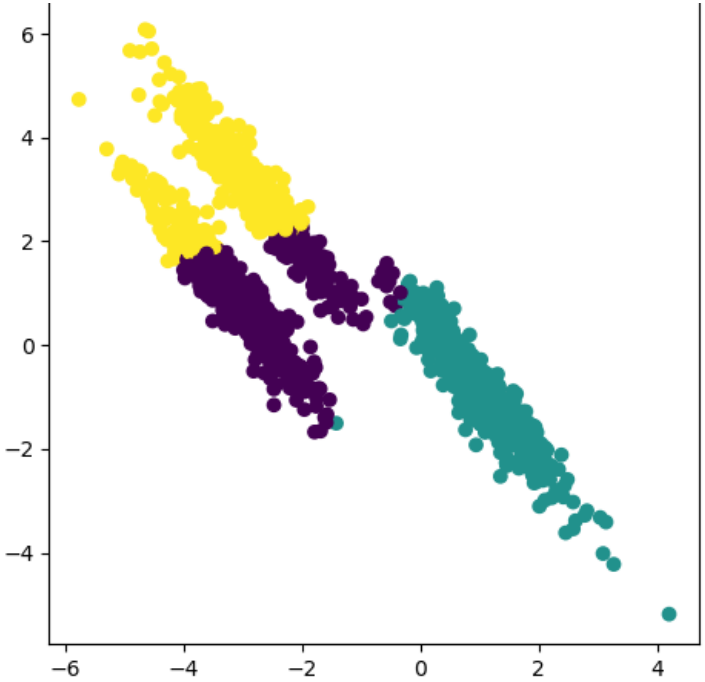
- Log-likelihood:

$$\ell(\boldsymbol{\pi}, \mu, \Sigma) = \ln p(\mathbf{X}|\pi, \mu, \Sigma) = \sum_{n=1}^{N} \ln p(\mathbf{x}^{(n)}|\pi, \mu, \Sigma)$$

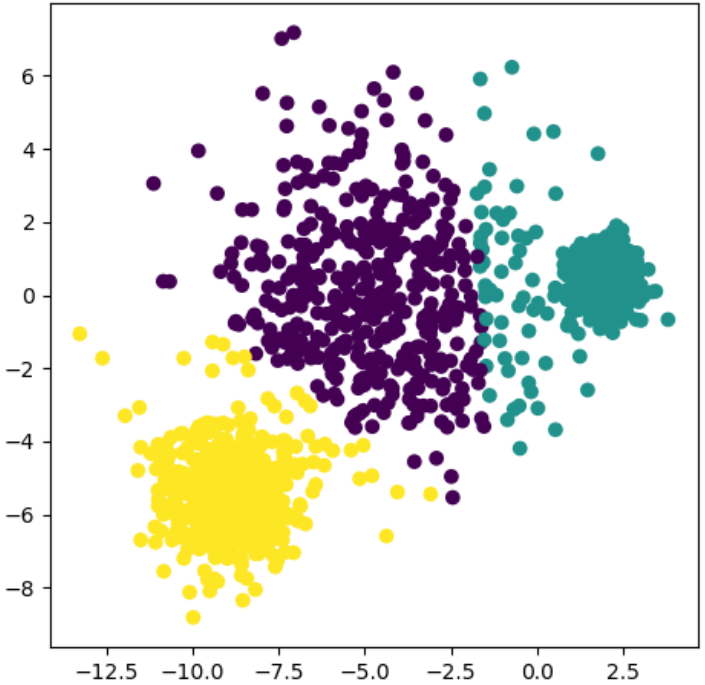
$$= \sum_{n=1}^{N} \ln \sum_{z^{(n)}=1}^{K} p(\mathbf{x}^{(n)}|\, z^{(n)}; \mu, \Sigma) p(z^{(n)}|\, \boldsymbol{\pi})$$

Better criteria exist → Cross-validation, Information-Theoretic (AIC, BIC)

Issues with k-means



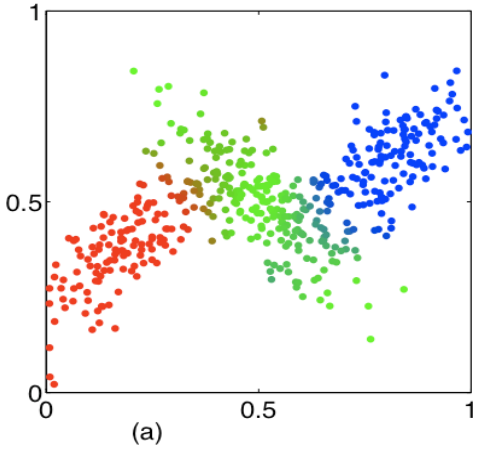
- Euclidean distance encourages spherical clusters



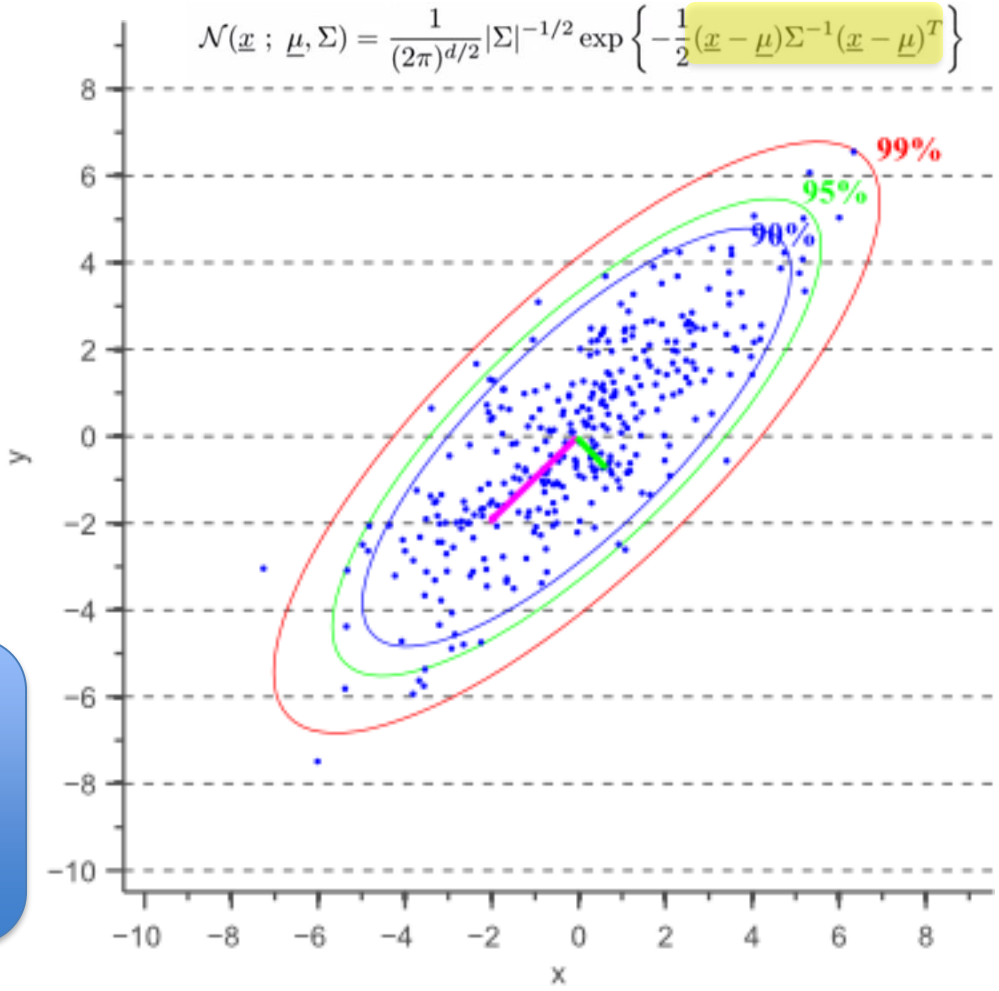
- Hard assignments → hard to characterize 'border cases'

- Can we have a distance-from-center based on ‘shape’ of the cluster ?
- Can we go beyond ‘hard’ assignments of points to clusters ?

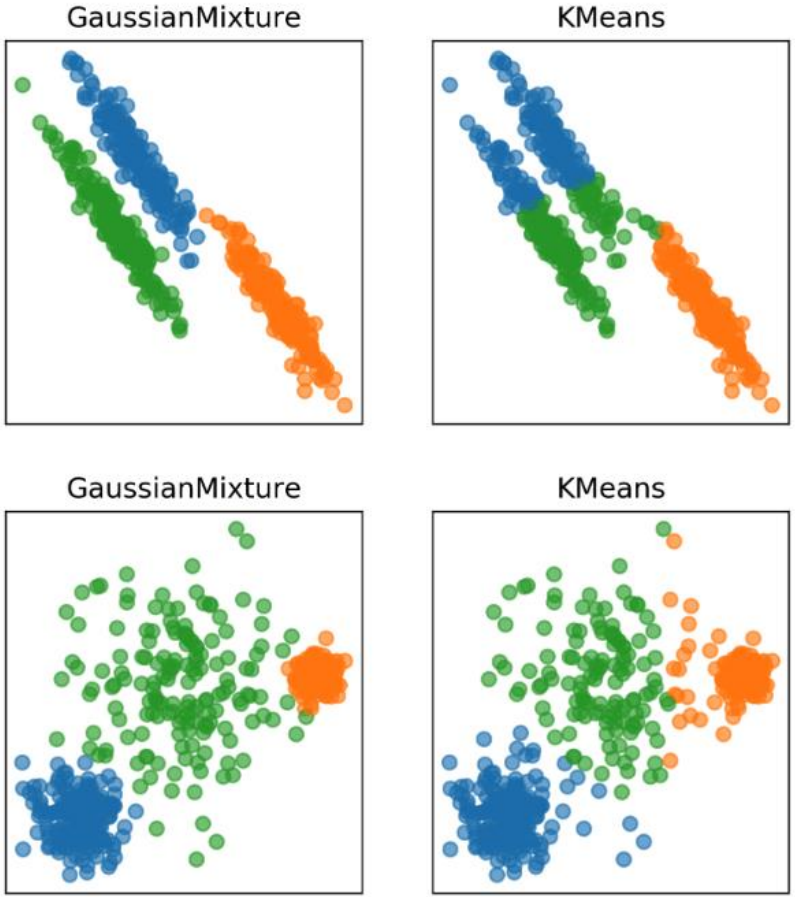
$$\gamma(z_{nj}) = \frac{\pi_j \mathcal{N}(\mathbf{x}_n|\boldsymbol{\mu}_j, \boldsymbol{\Sigma}_j)}{\sum_k \pi_k \mathcal{N}(\mathbf{x}_n|\boldsymbol{\mu}_k, \boldsymbol{\Sigma}_k)}$$



Models the uncertainty of cluster assignment



GMM vs KMeans

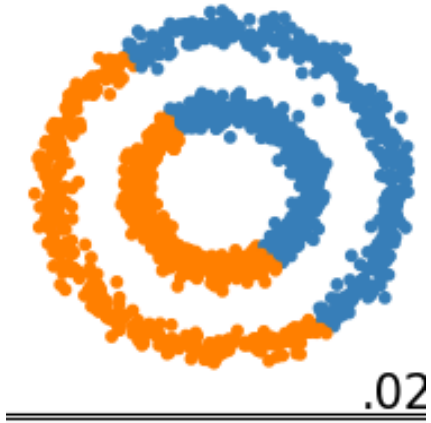


https://amueller.github.io/COMS4995-s18/slides/aml-16-032118-clustering-and-mixture-models/#49

Additional features of GMM

- GMM lets us cluster with ‘missing feature’ data
- GMM lets us generate new data with statistical properties of given data x

GMM on non-convex clusters

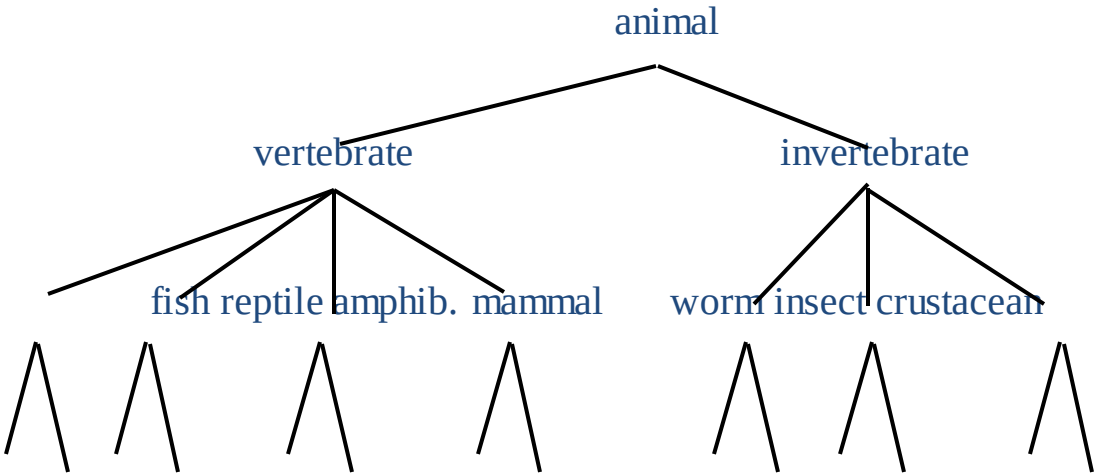


Hierarchical Clustering

Adapted from slides by Prabhakar Raghavan, Christopher Manning, Ray Mooney , Soumen Chakrabarti and A. Mueller

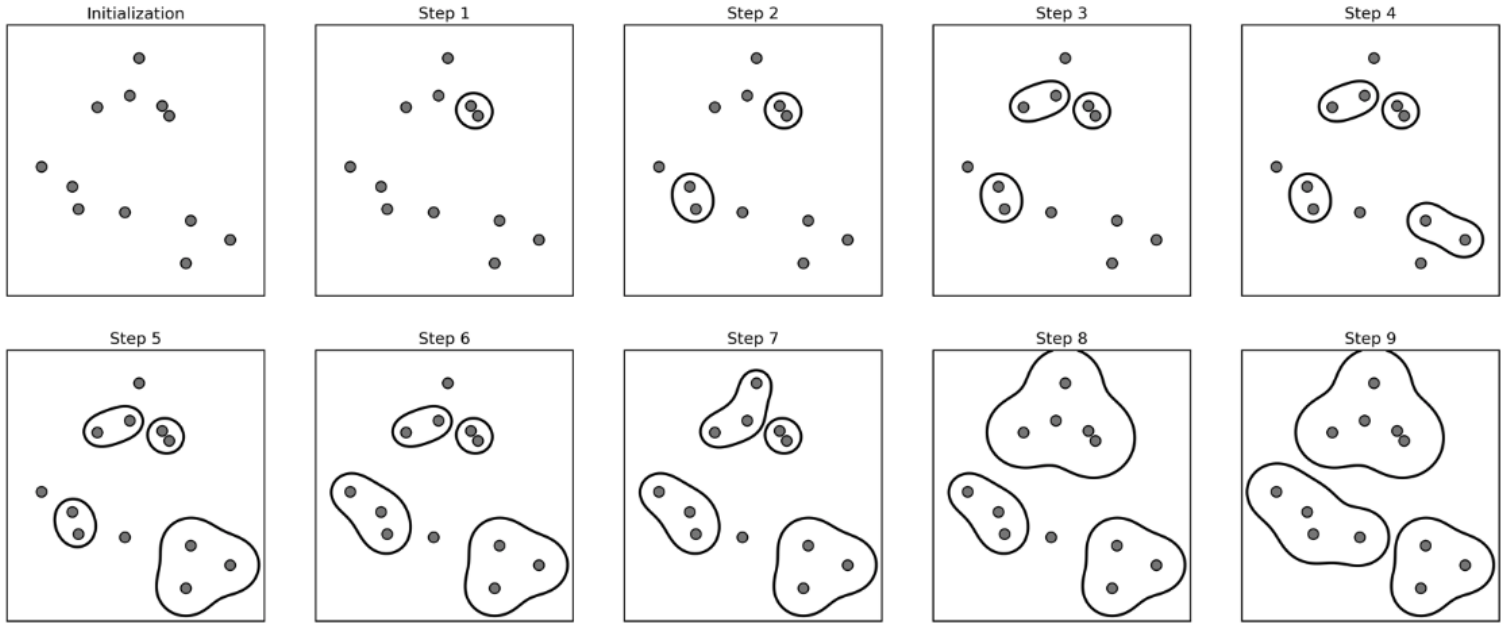
Hierarchical Clustering

- Data often has multiple, hierarchical layers of structure
- GMM, k-means clustering : Optimize for one/few layers
- How to get the entire structure ?
- Idea: Build a tree-based hierarchical taxonomy (*dendrogram*) from data.



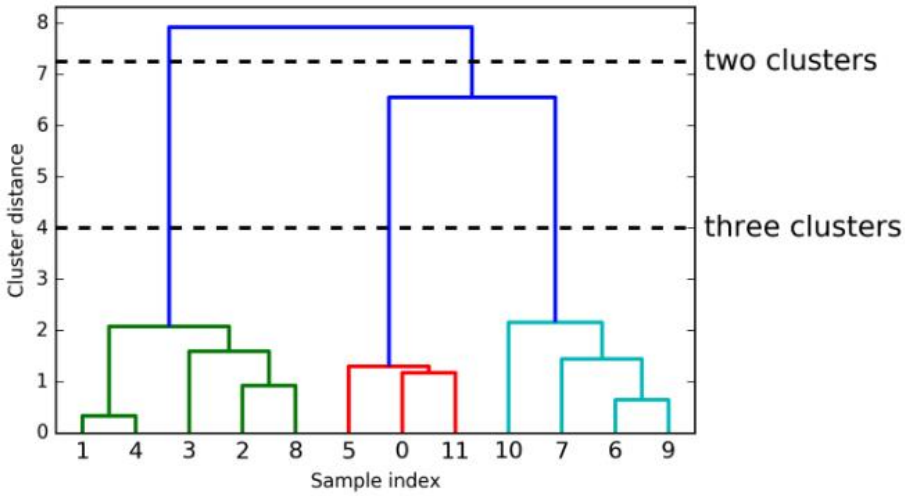
Agglomerative Clustering

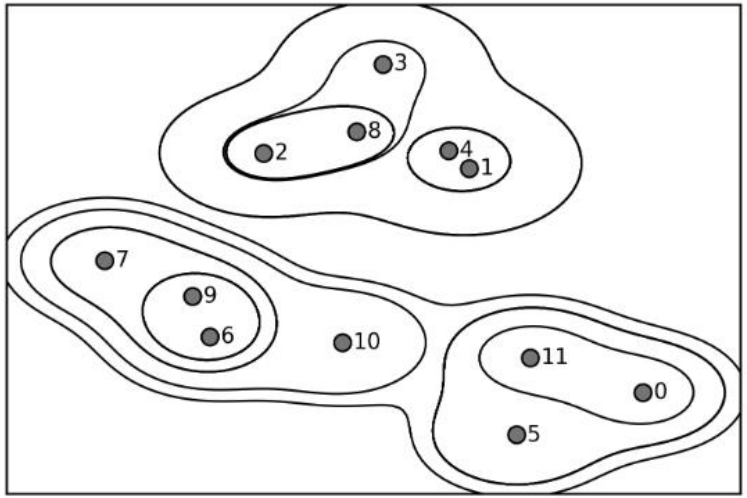
- Start with all points in their own cluster.
- Greedily merge the two most similar clusters.



https://amueller.github.io/COMS4995-s18/slides/aml-16-032118-clustering-and-mixture-models/#26

Dendograms





https://amueller.github.io/COMS4995-s18/slides/aml-16-032118-clustering-and-mixture-models/#27

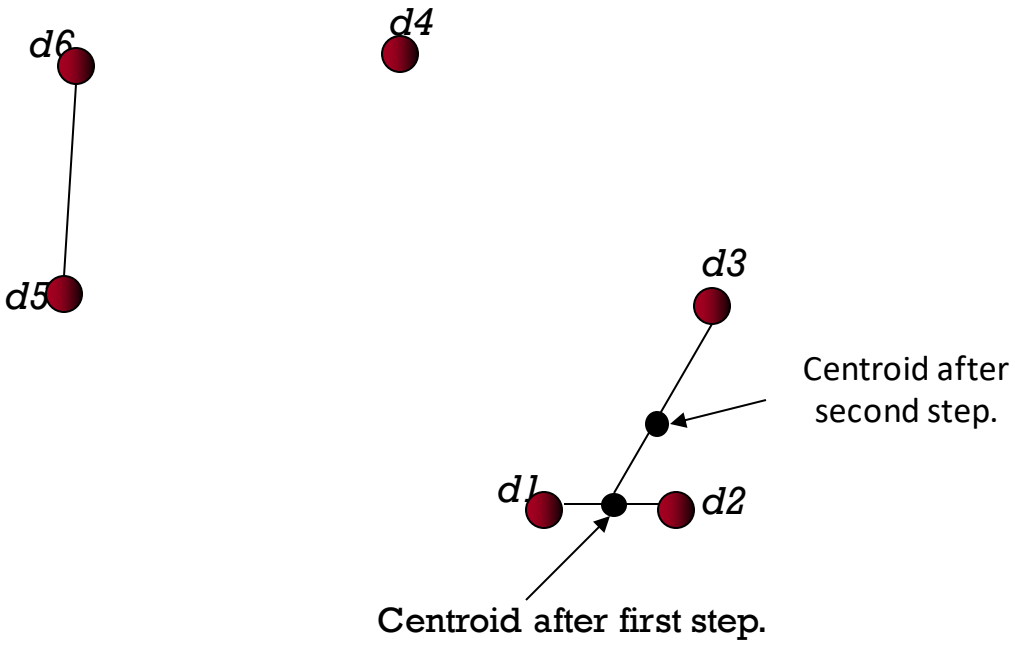
Hierarchical Agglomerative Clustering (HAC) Algorithm

Start with all instances in their own cluster.
Until there is only one cluster:
    Among the current clusters, determine the two clusters, c_i and c_j, that are most similar.
    Replace c_i and c_j with a single cluster $c_i \cup c_j$

How to determine similarity/distance between “clusters” ?

- Have a “cluster representative”
- Representative should be some sort of “typical” or central point in the cluster, e.g., Centroid or center of gravity

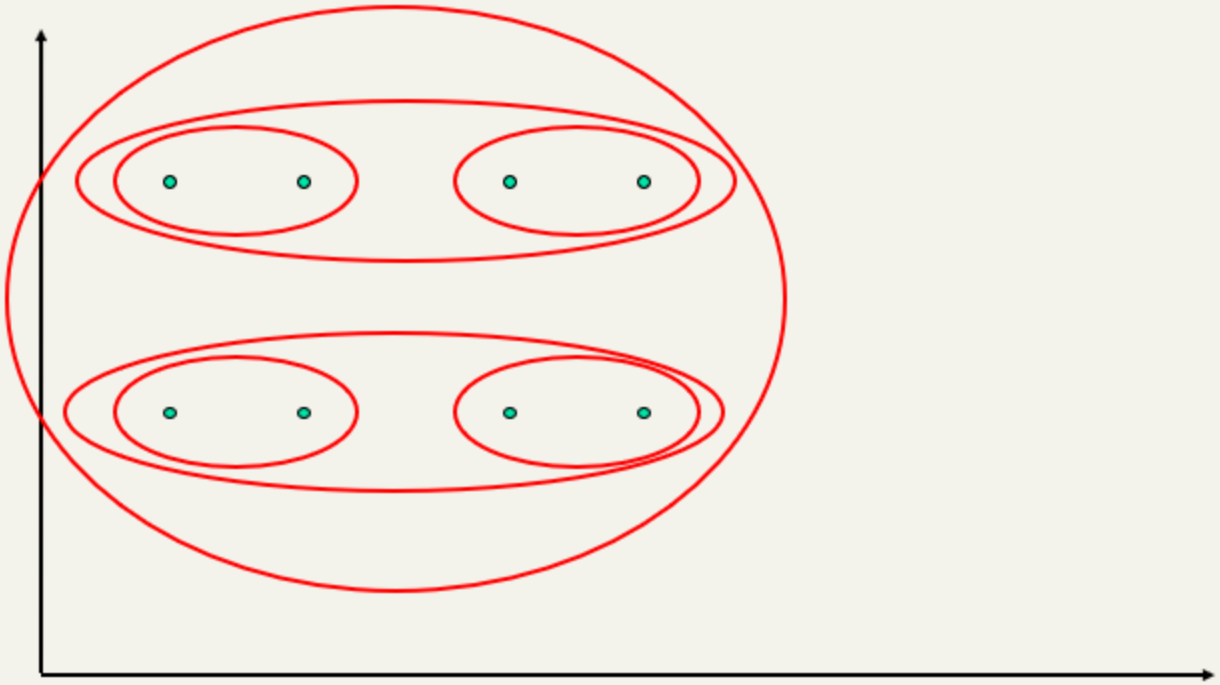
Example: *n=6, k=3,* closest pair of centroids



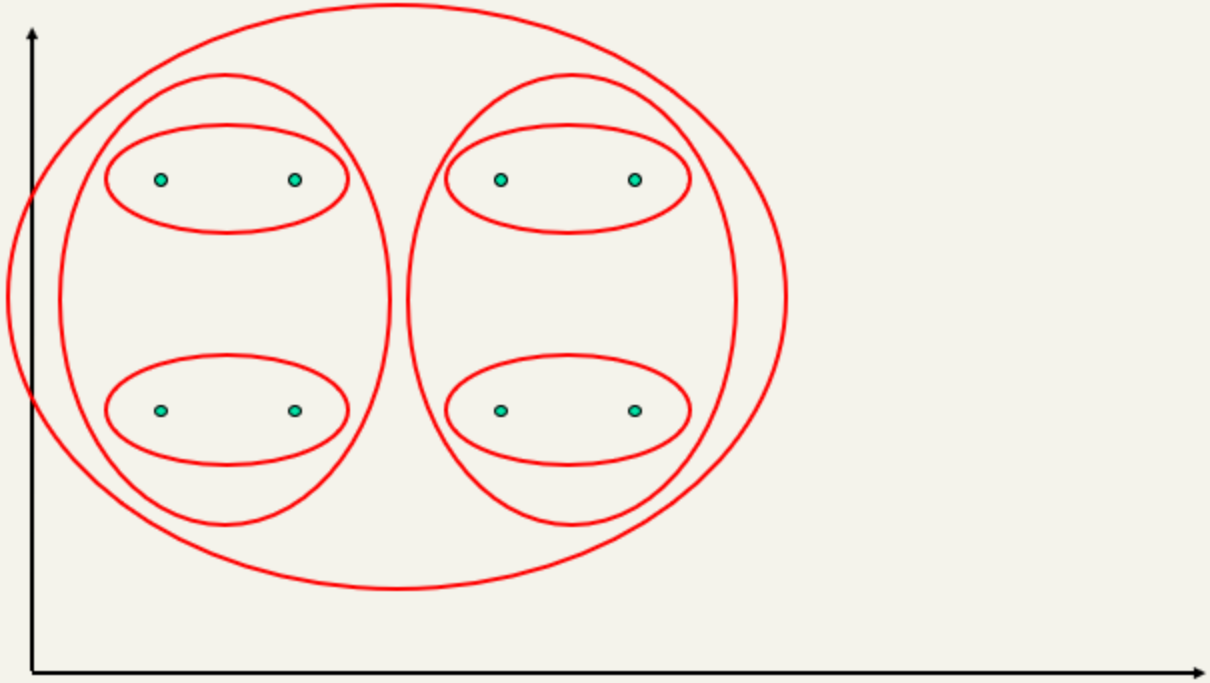
Closest pair of clusters

- **Single-link**
  - Smallest ‘minimum distance’
- **Complete-link**
  - Smallest ‘maximum distance’
- **Average-link**
  - Smallest ‘average distance’
- **Ward**
  - Smallest increase in within-cluster variance
  - Default in sk-learn
  - Tends to create equal-sized clusters

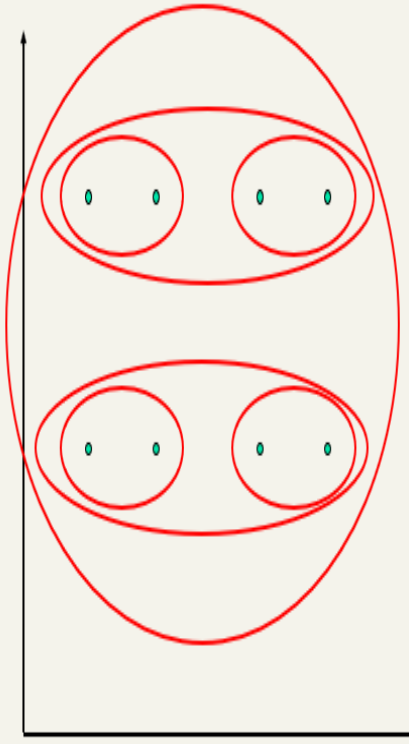
Single Link Example



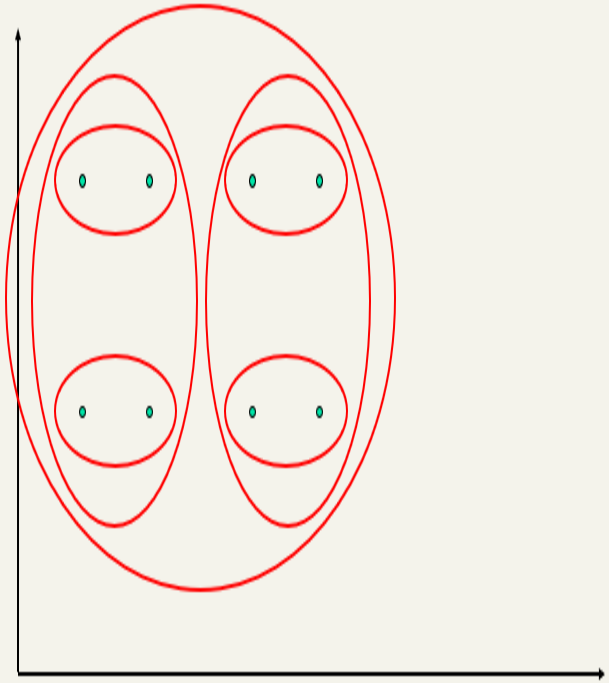
Complete Link Example



Single Link Example



Complete Link Example



Hierarchical Clustering algorithms

- **Agglomerative (bottom-up):**
  - Start with each document being a single cluster.
  - Eventually all documents belong to the same cluster.
- **Divisive (top-down):**
  - Start with all documents belong to the same cluster.
  - Eventually each node forms a cluster on its own.
- Does not require the number of clusters ***k*** in advance

What is a Good Clustering?

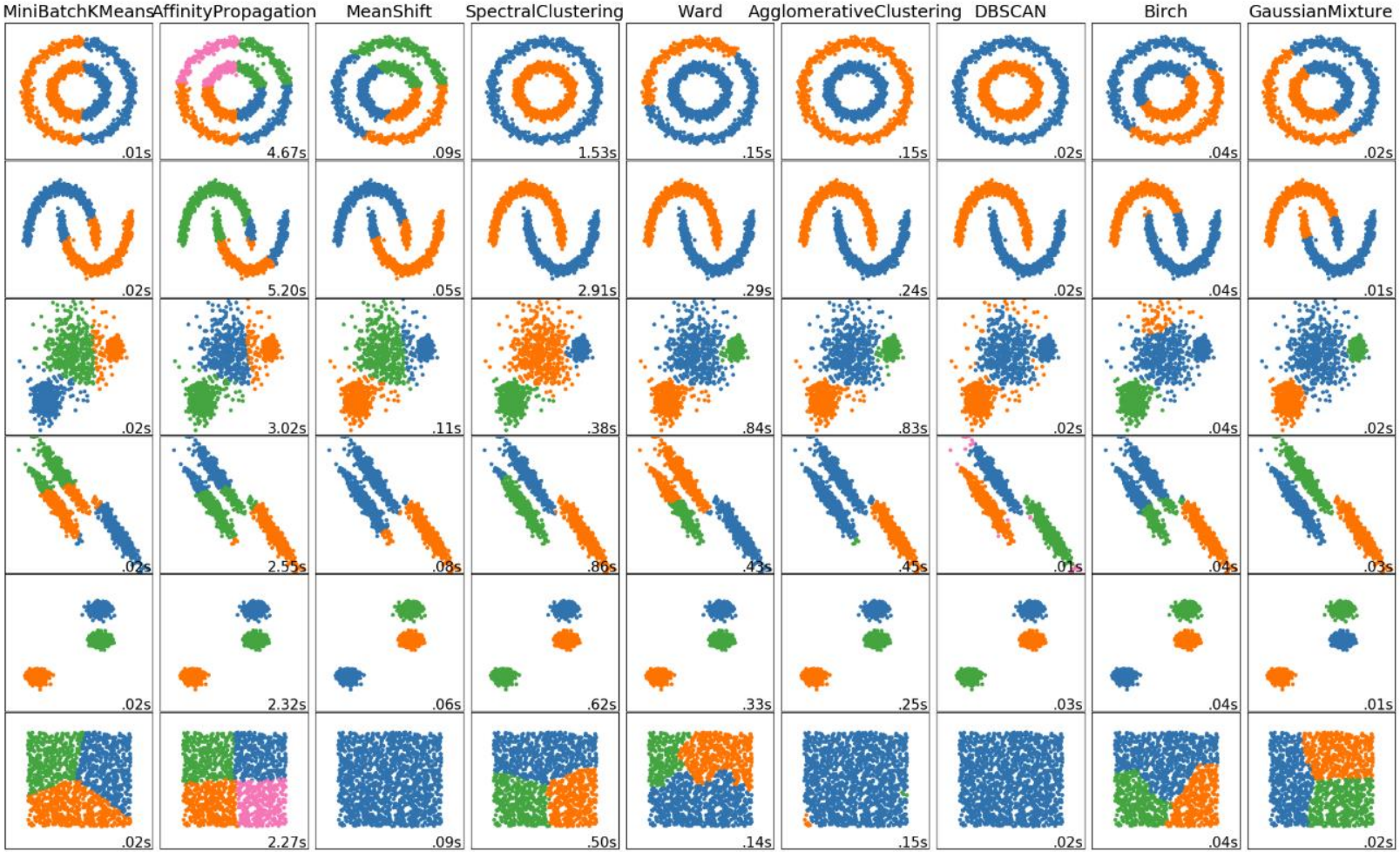
- *Internal criterion*: A good clustering will produce high quality clusters in which:
  - the intra-class (that is, intra-cluster) similarity is high
  - the inter-class similarity is low
  - The measured quality of a clustering depends on both the data representation and the similarity measure used

“The Curse of Dimensionality”

- Why clustering is difficult
  - While clustering looks intuitive in 2 dimensions, many applications involve 10,000 or more dimensions…
  - High-dimensional spaces look different
    - The probability of random points being close drops quickly as the dimensionality grows.
    - Furthermore, random pair of vectors are all almost perpendicular.

Summary

- K-means : Simple. Only convex cluster shapes, determined by cluster centers.
- Gaussian Mixture Models: Probabilistic. Soft clustering. Can be hard to fit.
- Hierarchical: Take input topology into account, produces hierarchy of clusters.



http://scikit-learn.org/dev/auto_examples/cluster/plot_cluster_comparison.html

References

- PRML (Bishop) – Chapter 9: 9.1,9.2,9.3.2
- Pattern Classification (Duda, Hart, Stork)
  - 10.4.3,10.6.1,10.7.1,10.7.2, 10.8, 10.10
  - 10.9 (Hierarchical Clustering)