Data mining & Machine Learning

CS 373
Purdue University

Dan Goldwasser
dgoldwas@purdue.edu

Today's Lecture

Expectation Maximization

How can we deal with cluster membership ambiguity?

Assume the data is generated by different processes.

Can you estimate the parameters of these processes?

Are we already doing it with K-Means?

But before that a quick review of

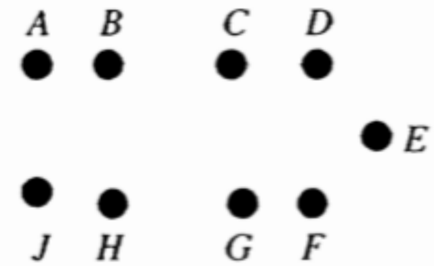
Hierarchical clustering

Agglomerative

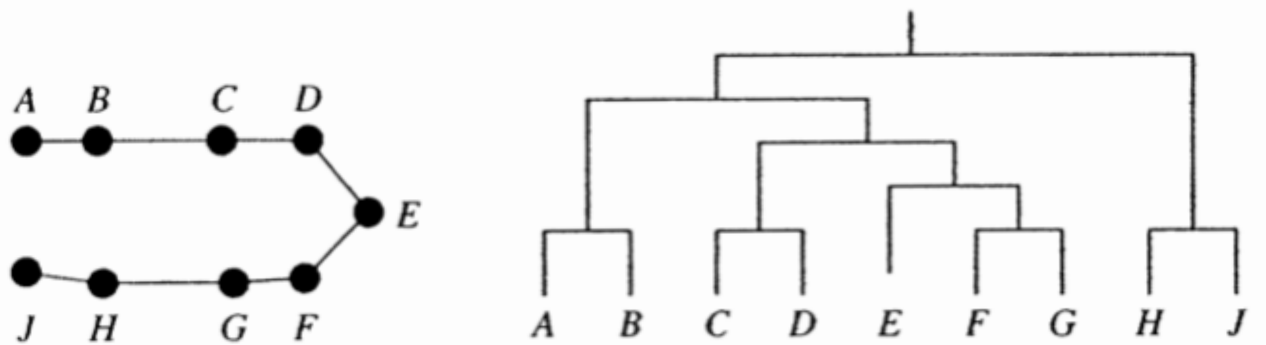
- For i = 1 to n:
  - Let $C_i = \{x(i)\}$
- While $|C|>1$:
  - Let C_i and C_j be the pair of clusters with min $D(C_i,C_j)$
  - $C_i=C_i \cup C_j$
  - Remove C_j

Recall: Distance measures between clusters

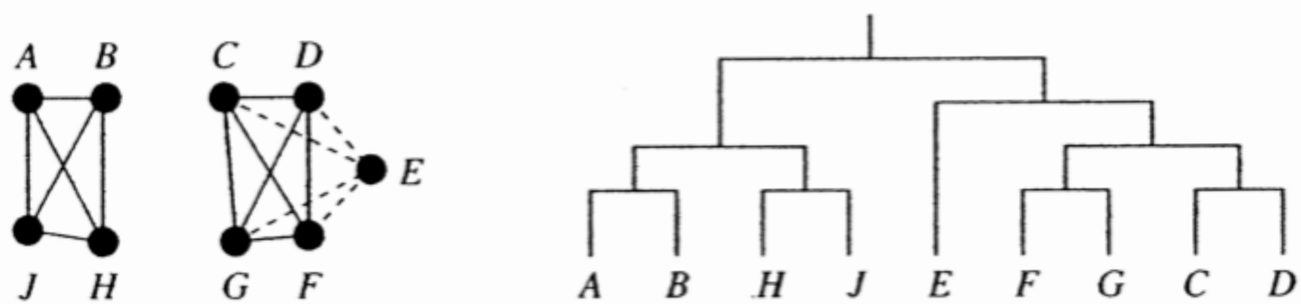
- ***Single-link***/nearest neighbor:
  - $D(C_i,C_j) = \mathbf{min}\{ d(x,y) \mid x \in C_i, y \in C_j \}$
    ⇒ can produce long thin clusters
- ***Complete-link***/furthest neighbor:
  - $D(C_i,C_j) = \mathbf{max}\{ d(x,y) \mid x \in C_i, y \in C_j \}$
    ⇒ is sensitive to outliers
- ***Average link***:
  - $D(C_i,C_j) = \mathbf{avg}\{ d(x,y) \mid x \in C_i, y \in C_j \}$
    ⇒ compromise between the two



(a) Data set



(b) Clustering using single linkage

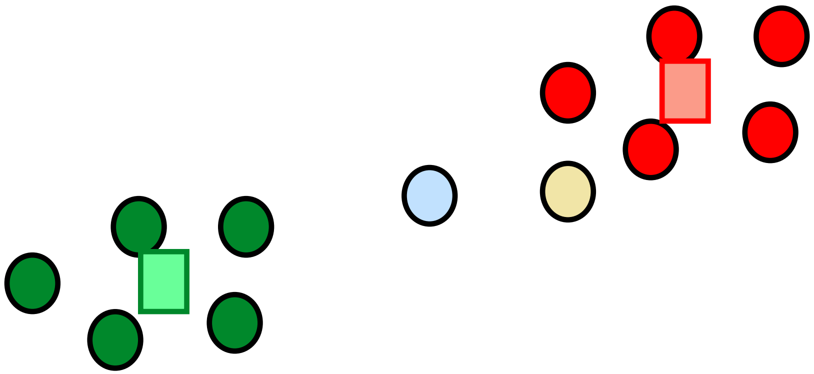


(c) Clustering using complete linkage

Agglomerative clustering

- Knowledge representation?
    - Dendograms: hierarchy of groupings from size 1 to n
- Score function?
    - Distance measure between two clusters (e.g., single link), considers pairwise distances between two sets of nodes
- Search?
    - Greedy, heuristic search successively chooses pair of clusters to merge that minimize distance

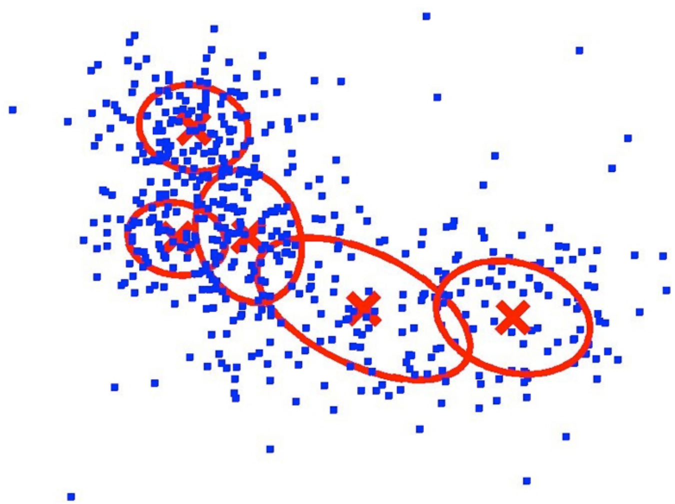
Some thought on K-Means



- What should be the cluster assignment for blue and yellow circles?
- What can we say about the certainty of this assignment?
- What would it matter?

How can we put these observations into an algorithm?
Stay tuned.. Coming up!

Probabilistic mixture model



probability of observing x

$$f(x) = \sum_{k=1}^{K} w_k f_k(x;\theta)$$

likelihood of point belonging to cluster k

likelihood of x being generated from cluster k

Mixture models

- *How to learn the model from data?*
- We don't know the mixing coefficients ($w_{1...k}$) or the component parameters (θ)
- **Solution**:
  - Interpret mixing coefficients as **prior** probabilities of cluster membership
  - Use **Expectation-Maximization** algorithm to estimate model *(Dempster, Laird, Rubin, 1977)*

$$
\begin{aligned}
f(x) &= \sum_{k=1}^{K} w_k f_k(x;\theta) \\
p(x) &= \sum_{k=1}^{K} p(k) p(x|k)
\end{aligned}
$$

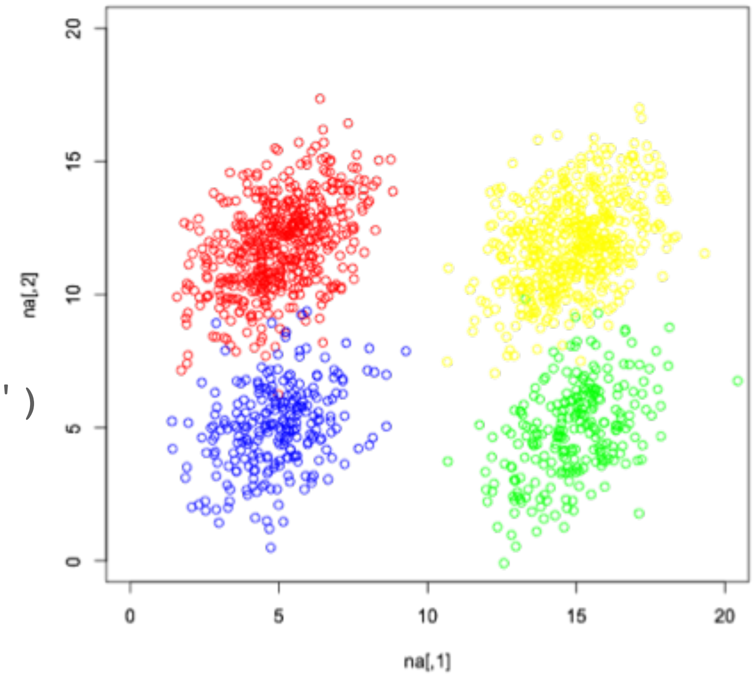
Generative Story: Generative process for GMM

- Assume that the data are generated from a mixture of k multi-dimensional Gaussians
  - Each component is has parameters: $N_k(\mu_k, \Sigma_k)$
- For each data point:
  - Pick component Gaussian randomly with probability p(k)
  - Draw point from that Gaussian by sampling from: $N_k(\mu_k, \Sigma_k)$

$$p(x) = \sum_{k=1}^{K} p(k)p(x|k) = \sum_{k=1}^{K} p(k)p\Big(x|x \sim N(\mu_k, \Sigma_k)\Big)$$

Example generative process

```
sigma <- matrix(c(2,1,1,3),2,2)
na=mvrnorm(n=500, c(5,12), sigma)
nb=mvrnorm(n=250, c(5,5), sigma)
nc=mvrnorm(n=250, c(15,5), sigma)
nd=mvrnorm(n=500, c(15,12), sigma)
d=rbind(na,nb,nc,nd)
plot(na,xlim=c(0,20),ylim=c(0,20),col='red')
points(nb,col='blue')
points(nc,col='green')
points(nd,col='yellow')
```



Parameters

$$p(k) = [0.333, 0.167, 0.167, 0.333]$$

$$\mu_1 = [5, 15], \mu_2 = [5, 5], \mu_3 = [15, 5], \mu_2 = [15, 12]$$

$$\Sigma = \begin{bmatrix} Var(X_1) & Cov(X_1, X_2) \\ Cov(X_1, X_2) & Var(X_2) \end{bmatrix} \qquad \Sigma_1 = \Sigma_2 = \Sigma_3 = \Sigma_4 = \begin{bmatrix} 2 & 1 \\ 1 & 3 \end{bmatrix}$$

Learning the model from data

- **We want to invert this process**
- Given the data, find the parameters of the generating process
  - Mixing coefficients $p(k)$
  - Component means and covariance matrix $N_k(\mu_k, \Sigma_k)$
- ***If we knew*** which component generated each point then the MLE solution would **involve fitting each component distribution to the appropriate cluster points**
- **Problem**: the cluster memberships are **hidden**

What is the equivalent the K-Means problem?

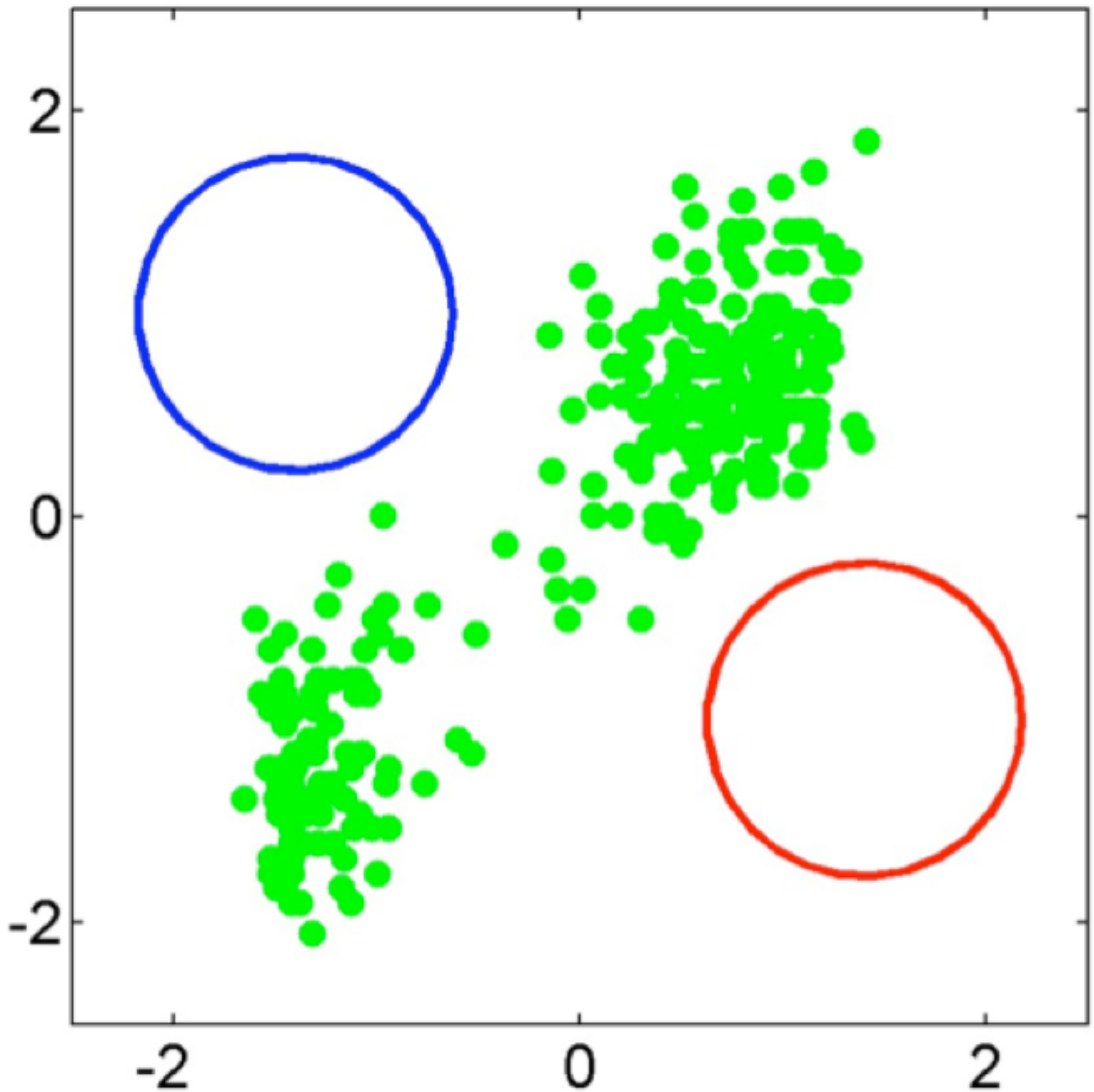
Expectation-maximization (EM) algorithm

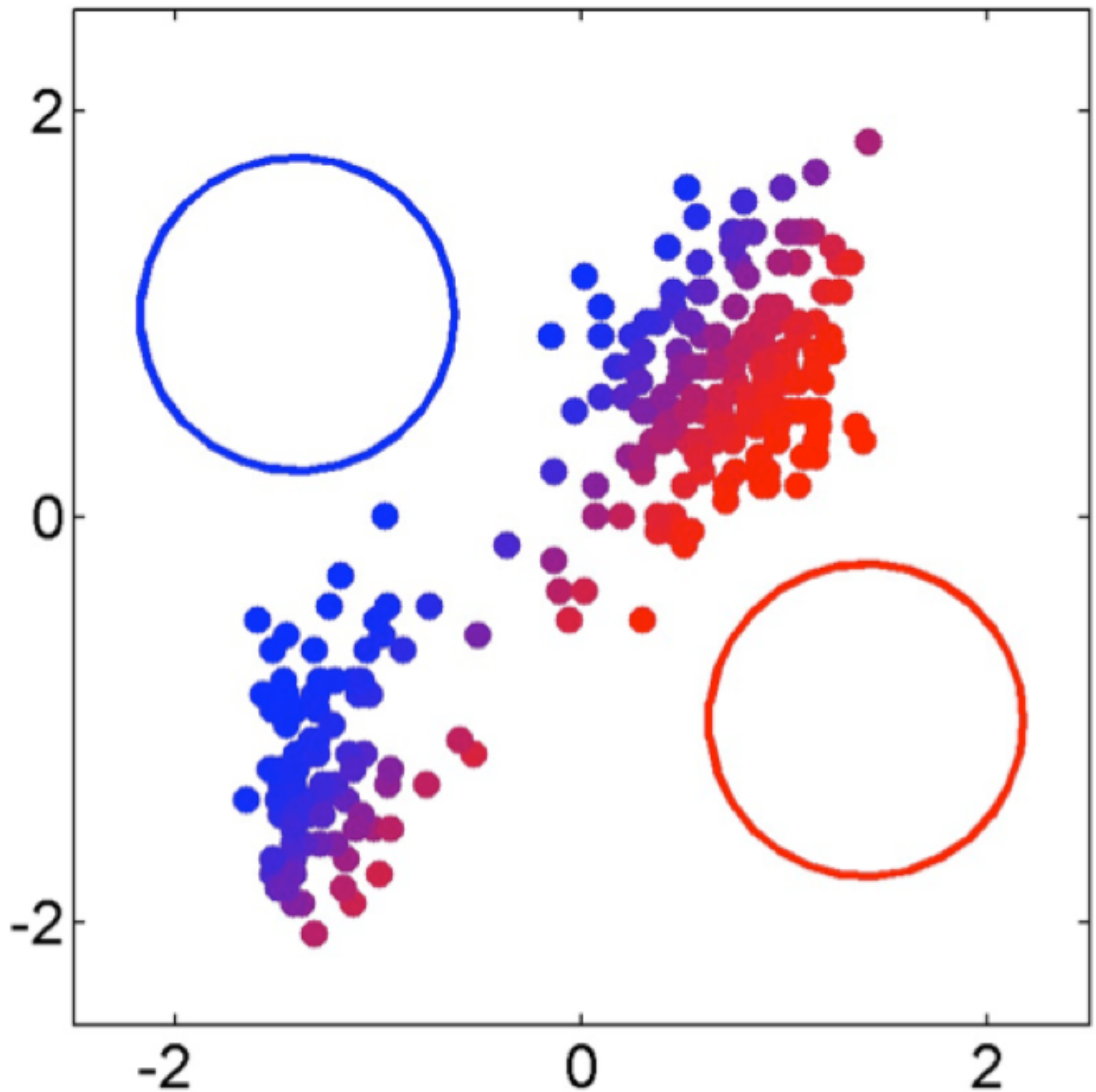
- Popular algorithm for parameter estimation in data with *hidden/unobserved* values
  - **Hidden variables=cluster membership**
- **Basic idea**
  - Initialize hidden variables and parameters
  - Predict values for hidden variables given current parameters

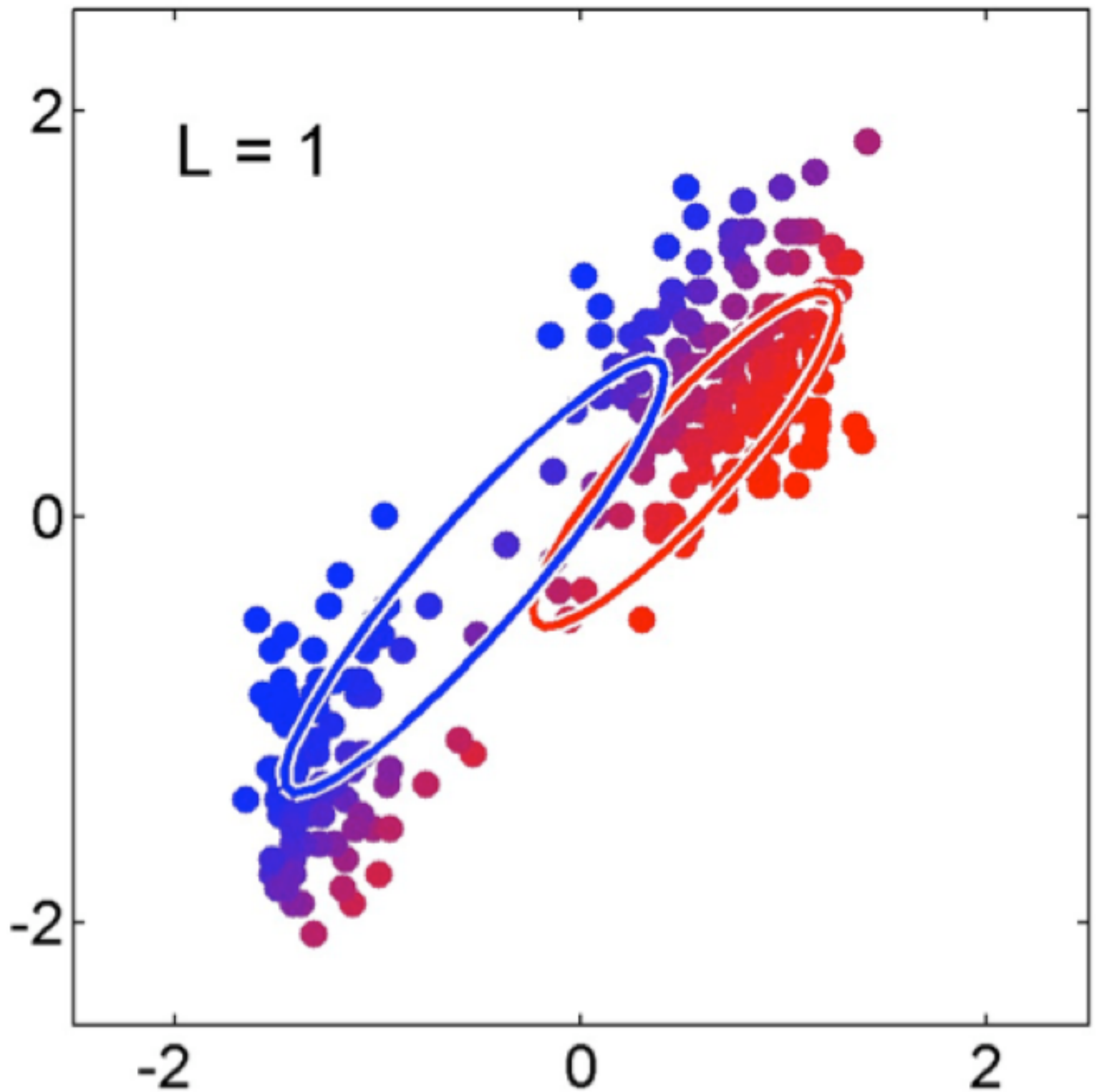
    E Step
  - Estimate parameters given current prediction for hidden variables

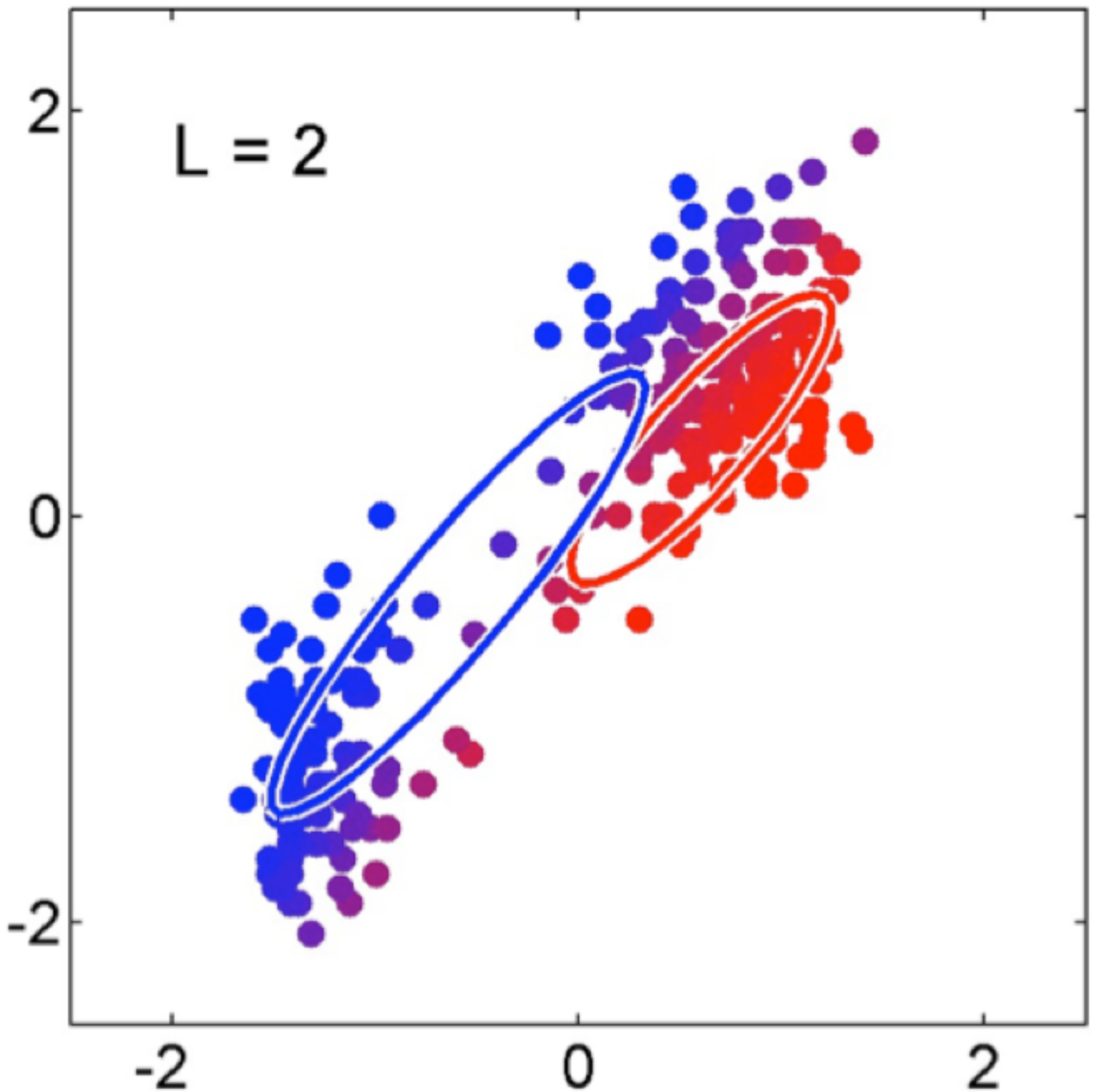
    M-Step
  - Repeat

GMM example

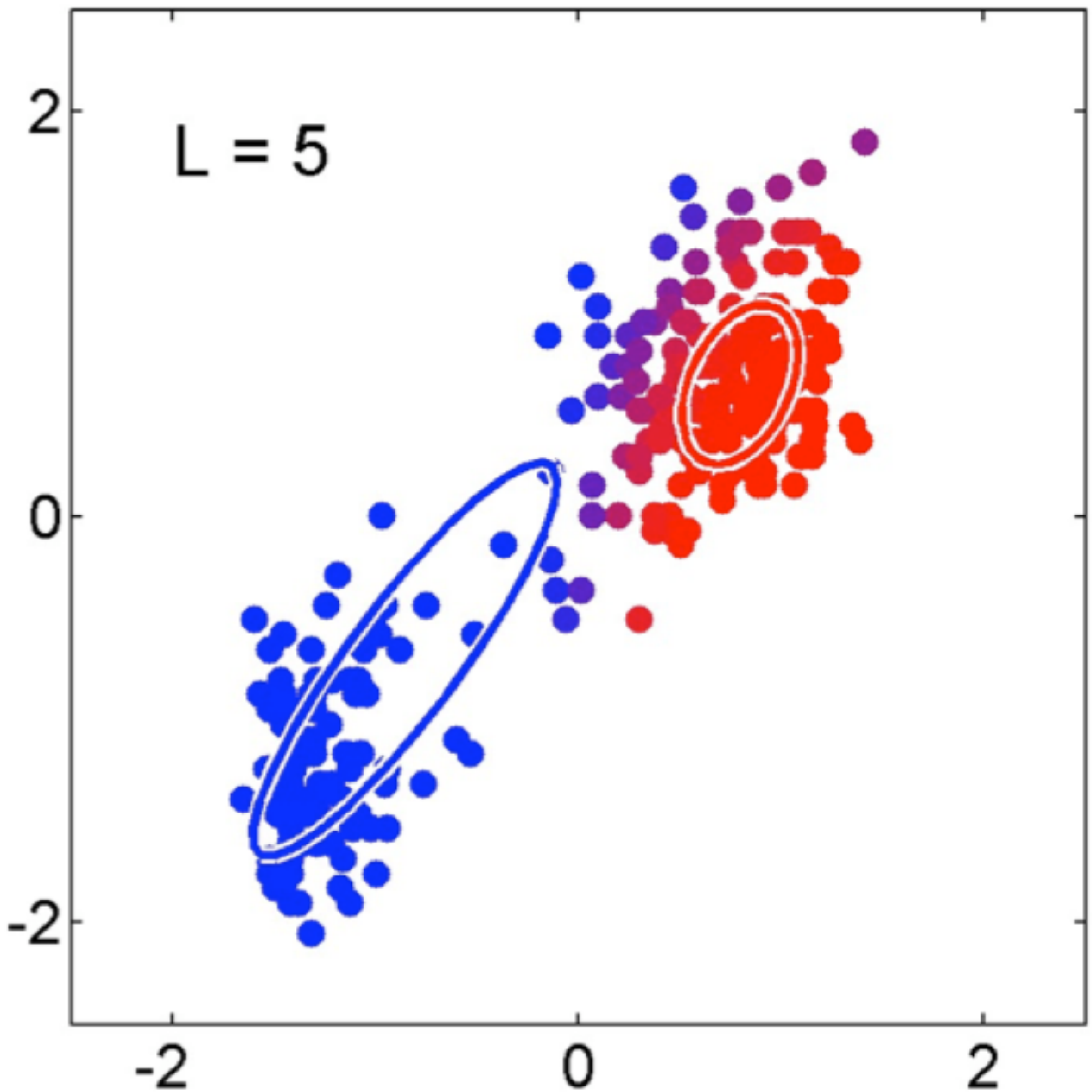

2
0
-2
-2
0
2


2
0
-2
-2
0
2

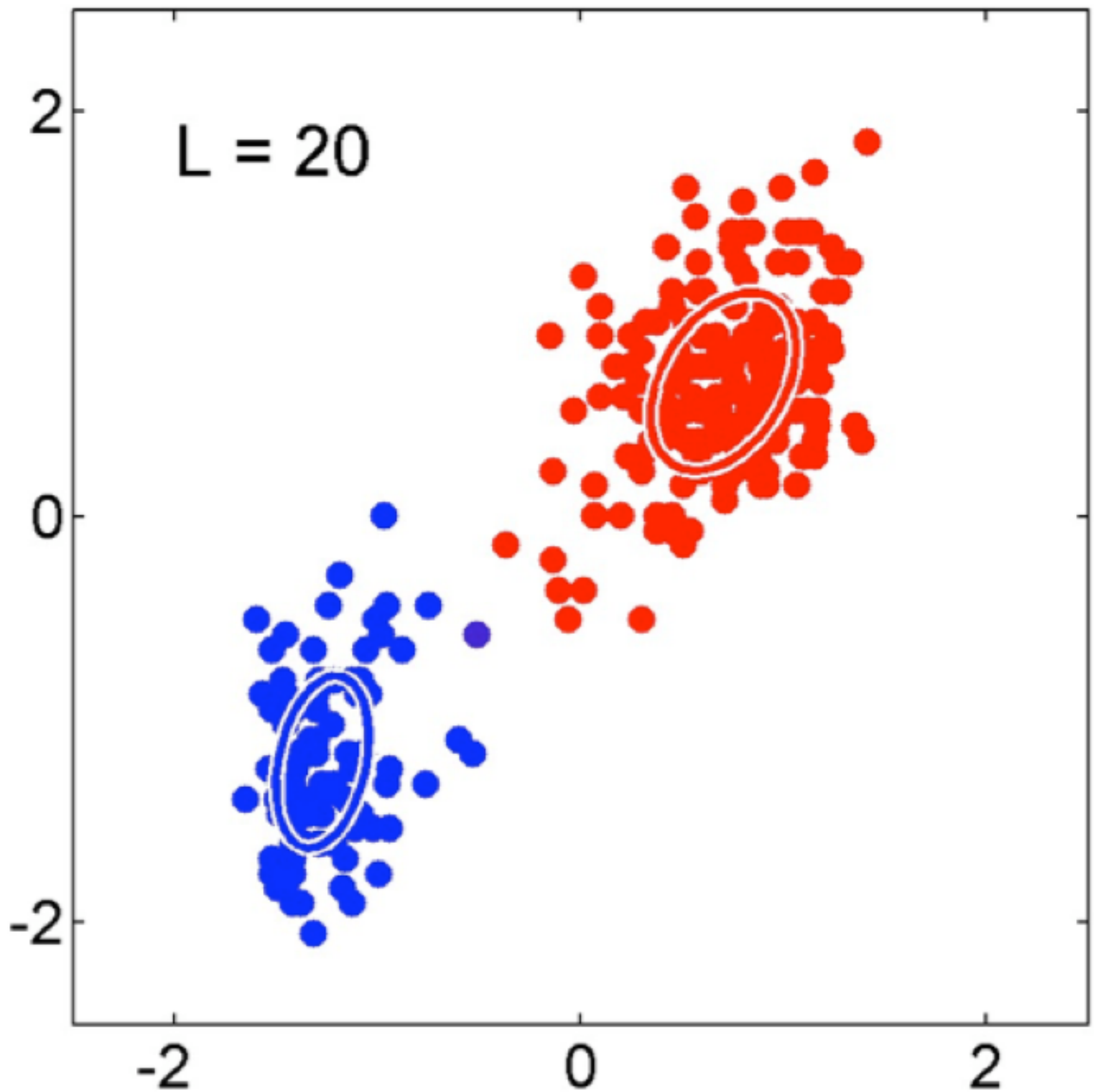

L = 1
2
0
-2
-2
0
2

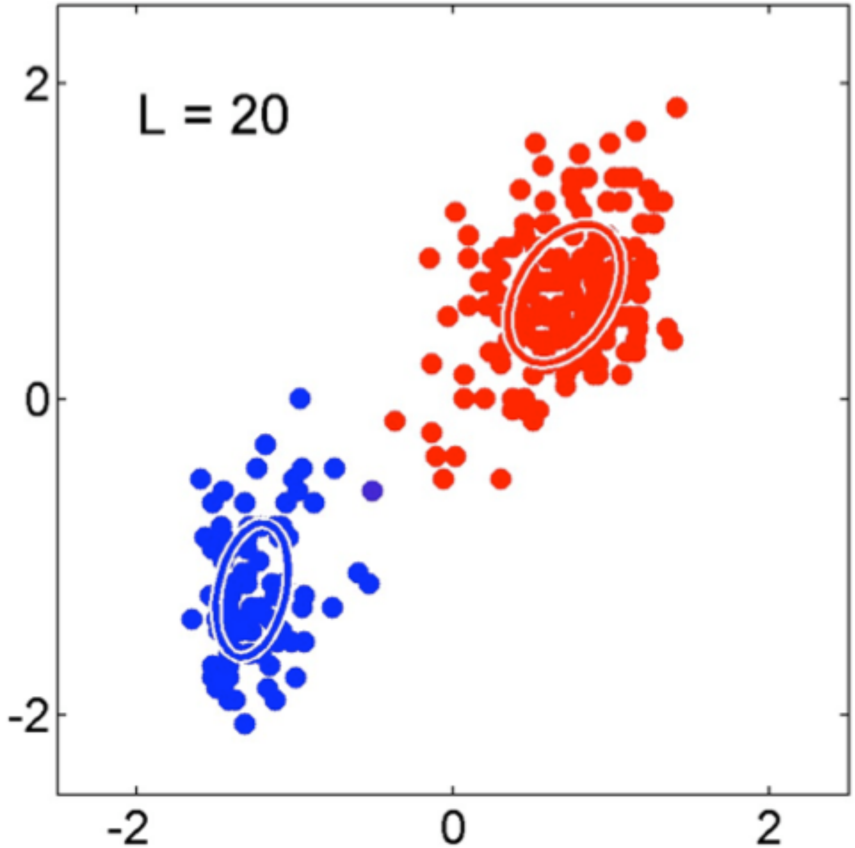


L = 2
2
0
-2
-2
0
2



L = 5
2
0
-2
-2
0
2


L = 20
2
0
-2
-2
0
2



Note the difference compared to K-Means.

K-Means:
Calculate the distance to centroid

EM:
Calculate the probability based on the underlying distribution (can account for different COV matrix) and the prior probability

How to learn GMMs?

Score function for GMM

- **Log likelihood** takes the following form (for model M={w,μ,Σ}):

$$\begin{aligned} log\, p(D|w,\mu,\Sigma) &= \sum_{i=1}^{N} log\, p(x_n|M) \\ &= \sum_{i=1}^{N} log\left[\sum_{k=1}^{K} p(x_n|k,M)P(k|M)\right] \\ &= \sum_{i=1}^{N} log\left[\sum_{k=1}^{K} w_k N(x_n|\mu_k,\Sigma_k)\right] \end{aligned}$$

- Note the sum over components is inside the log
- There is no closed form solution for the MLE

Hidden cluster membership variables

- Consider k cluster indicator variables for example x_n: $\mathbf{z_n} = [z_{n1}, ..., z_{nk}]$ which equals 1 for the cluster that x_n is a member of, and 0 otherwise
- ***If we knew*** *the values of the hidden cluster membership variables (z)* we could easily maximize the complete data log-likelihood, which has a closed form solution:

$$
\begin{aligned}
log\, p(D, \mathbf{z}|w, \mu, \Sigma) &= \sum_{i=1}^{N} log\Big[\sum_{k=1}^{K} z_{nk} \cdot w_k N(x_n|\mu_k, \Sigma_k)\Big] \\
&= \sum_{i=1}^{N} log\left[w_{k'} N(x_n|\mu_{k'}, \Sigma_{k'})\right] \quad \text{where } z_{nk'} \neq 0 \\
&= \sum_{i=1}^{N} log\, w_{k'} + log\, w_{k'} N(x_n|\mu_{k'}, \Sigma_{k'}) \quad \text{where } z_{nk'} \neq 0
\end{aligned}
$$

- Unfortunately we don’t know the values for the hidden variables!
- But, for given set of parameters we can compute the ***expected values*** of the hidden variables (cluster memberships)

Posterior probabilities of cluster membership

- We can think of the mixing coefficients as **prior** probabilities for cluster membership
- Then for a given example x_n, we can evaluate the corresponding **posterior** probabilities of **cluster membership** with Bayes theorem:

$$\gamma_k(x_n) \equiv p(z_{nk} = 1 | x_n) = \frac{p(x_n | z_{nk} = 1) p(z_{nk} = 1)}{p(x_n)}$$

$$= \frac{w_k N(x_n | \mu_k, \Sigma_k)}{\sum_{j=1}^{K} w_j N(x_n | \mu_j, \Sigma_j)}$$

cluster membership for x

What is the equivalent K-Means step?

Expected Log Likelihood

- We can now define the expected log likelihood based on the Posterior probabilities of cluster membership

$$log\ p(x, z|\theta) = \sum_{n=1}^{N} \sum_{k=1}^{K} \gamma_i(x_n) \left[log\ w_k + log\ N(x_n|\mu_k, \Sigma_k) \right]$$

- The M step: Find the parameters that maximize it

What is the equivalent K-Means step?

EM for GMM

- Suppose we make a guess for the parameters values
- Use these to evaluate cluster memberships $\Gamma(x_n) = [\gamma_1(x_n), ..., \gamma_K(x_n)]$

E-Step

- Now compute the log-likelihood using predicted cluster memberships

M-Step

$$log\ p(x, z|\theta) = \sum_{n=1}^{N} \sum_{k=1}^{K} \gamma_i(x_n) \left[log\ w_k + log\ N(x_n|\mu_k, \Sigma_k)\right]$$

- Use completed likelihood to determine MLE for parameters

EM for GMM

- Evaluate cluster memberships
  - *Based on your **current** model parameters*

$$\gamma_k(x_n) \equiv p(z_{nk} = 1|x_n) = \frac{p(x_n|z_{nk} = 1)p(z_{nk} = 1)}{p(x_n)}$$

$$= \frac{w_k N(x_n|\mu_k, \Sigma_k)}{\sum_{j=1}^{K} w_j N(x_n|\mu_j, \Sigma_j)}$$

E-Step

EM for GMM

- MLE of the new parameters of the model

M-Step

$$N_k = \sum_i \gamma_k(x_i)$$

Total weight assigned to cluster k

$$w_k = \frac{N_k}{N}$$

Normalize into “fractional points” assigned to cluster k

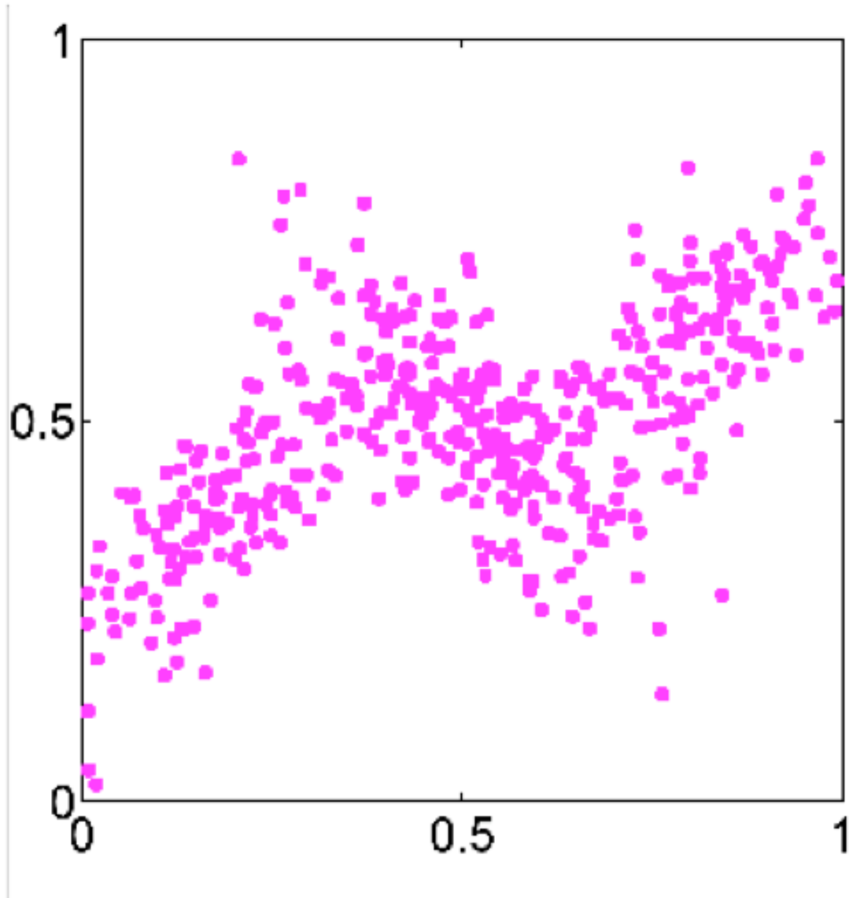
$$\mu_k = \frac{1}{N_k} \sum_i \gamma_k(x_i) x_i$$

Weighted mean for the assigned data points

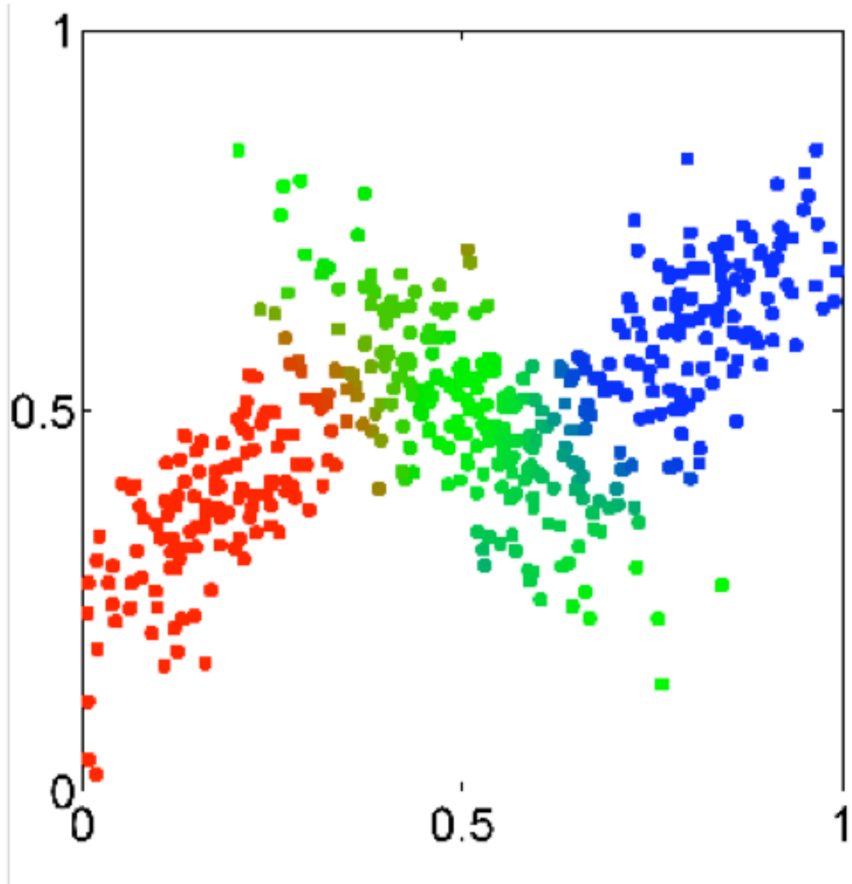
$$\Sigma_k = \frac{1}{N_k} \sum_i \gamma_k(x_i)(x_i - \mu_k)(x_i - \mu_k)^T$$

Weighted covariance for the assigned data, **based on the new weighted mean**

Unlabeled dataset



Posterior probabilities of cluster membership



Probabilistic clustering

- Model provides full distributional description for each component
  - May be able to interpret differences in the distributions
- Soft clustering (compared to k-mean hard clustering)
  - Given the model, each point has a k-component vector of membership probabilities
- **Key cost**: assumption of parametric model

Mixture models

- Knowledge representation?
  - **Parametric model**
    parameters = mixture coefficient and component parameters
- Score function?
  - **Likelihood**
- Search?
  - **Expectation maximization**
    iteratively find parameters that maximize likelihood and predicts cluster memberships
- Optimal? Exhaustive?

Score functions for selecting k

How to choose k?

- Choose k to maximize likelihood?
  - *As k increases the value of the maximum likelihood cannot decrease*
- Thus more complex models will always improve likelihood
- **How to compare models with different complexities?**

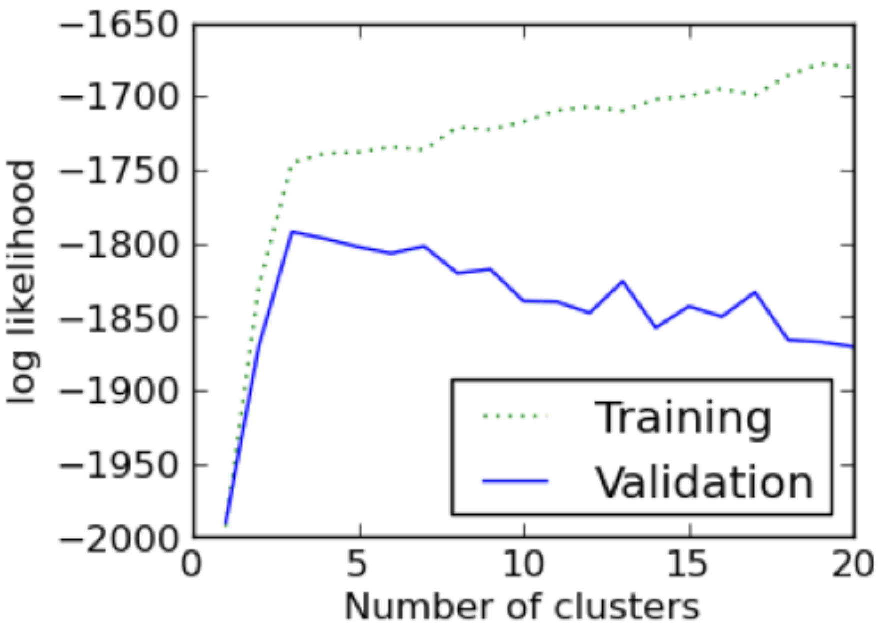
Model selection scoring functions

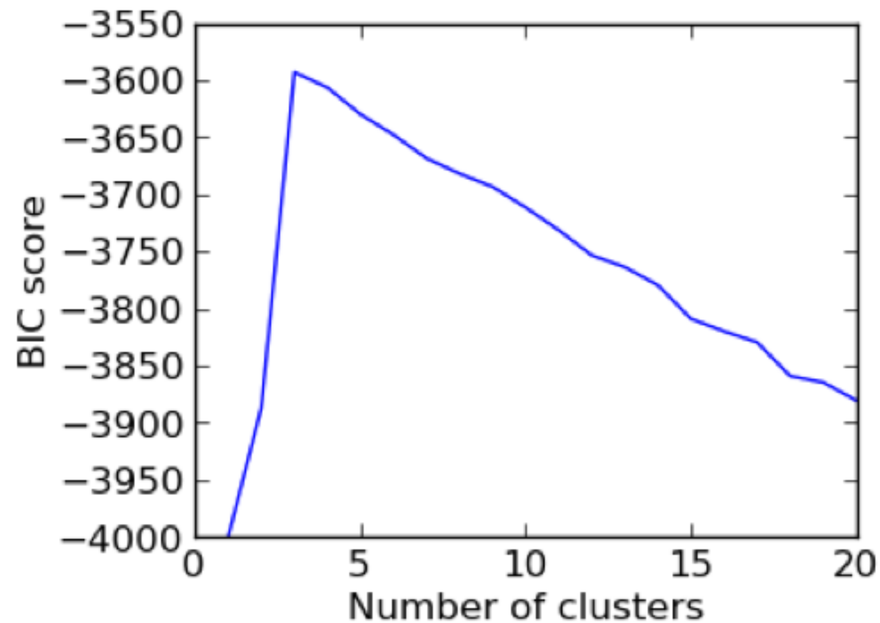
- **Goal 1**: *Describe* data as precisely as possible
    - General approach based on data compression and information theory uses score function:
- **Goal 2**: *Generalize* to new data
    - Goodness of fit is part of the evaluation, but since the data is not the entire population, we want to learn a model that will generalize to other new data instances
- Thus, want to strike a balance between between how well the model fits and the data and the simplicity of the model

Penalized score functions

- Penalized score functions include a term that reflects how well the model fits and the data and another (penalty) term to value the simplicity of the model
- Score(θ,M) = error(M) + penalty(M)
    - Penalty may depend on the number of parameters in the model (p) and the number of data points (n)
    - Error is generally based on likelihood of the data given the model (L)
- **AIC** (Akaike information criterion):
  $\text{Score}_{AIC} = -2 \log L + 2p$
- **BIC** (Bayesian information criterion):
  $\text{Score}_{BIC} = -2 \log L + p \log n$
- **Other functions**: *minimum description length, structural risk minimization*

Example: GMMs





Source: http://pypr.sourceforge.net/examples.html