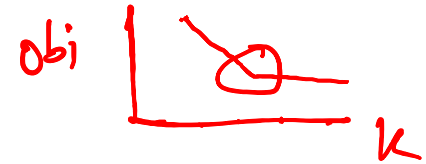
Mixture models & EM algorithm

Aarti Singh

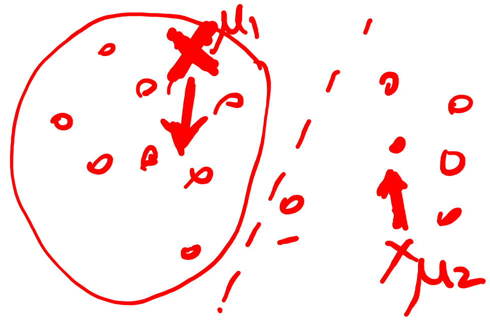
Machine Learning 10-315
Apr 11, 2022

Some slides courtesy of Eric Xing, Carlos Guestrin

Partitioning Algorithms



- K-means
  - **hard assignment**: each object belongs to only one cluster



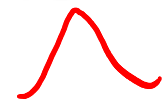
K

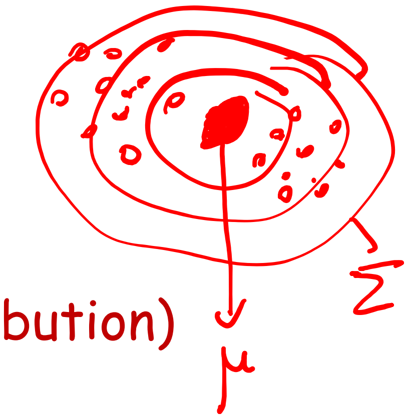
$$obj = \sum_j \underbrace{(\mu_{c(j)} - x_j)^2}_{d(\mu_{c(j)}, x_j)}$$

- Mixture modeling
  - **soft assignment**: probability that an object belongs to a cluster

Generative approach

Mixture models





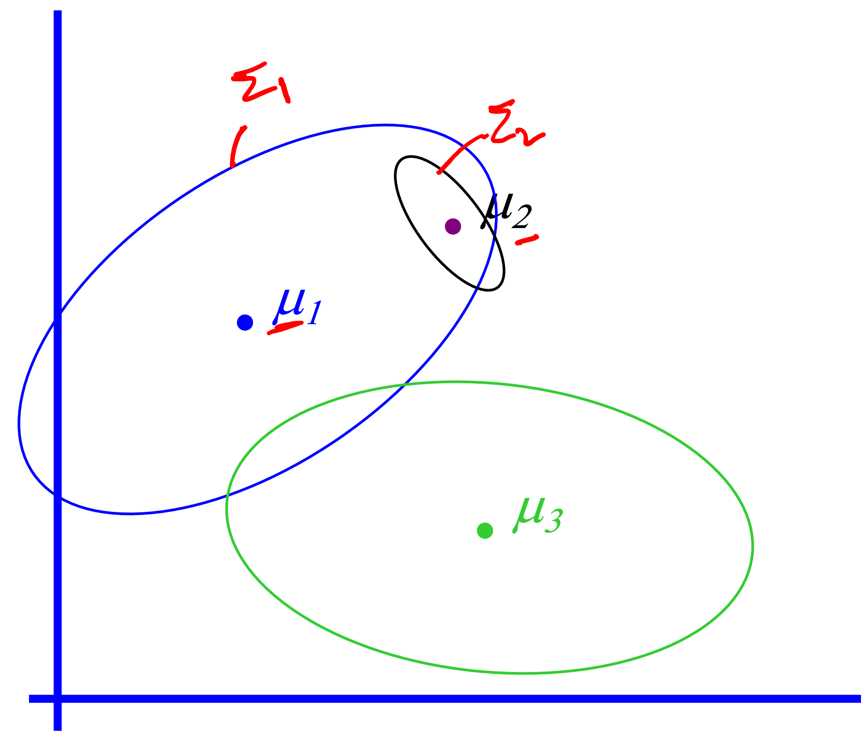
GMM – Gaussian Mixture Model (Multi-modal distribution)

$$p(x|y=i) \sim \mathrm{N}(\mu_i, \Sigma_i)$$

$$p(x) = \sum_i p(x|y=i)\, \boxed{P(y=i)}$$

Mixture component — $p(x|y=i)$

Mixture proportion — $P(y=i)$

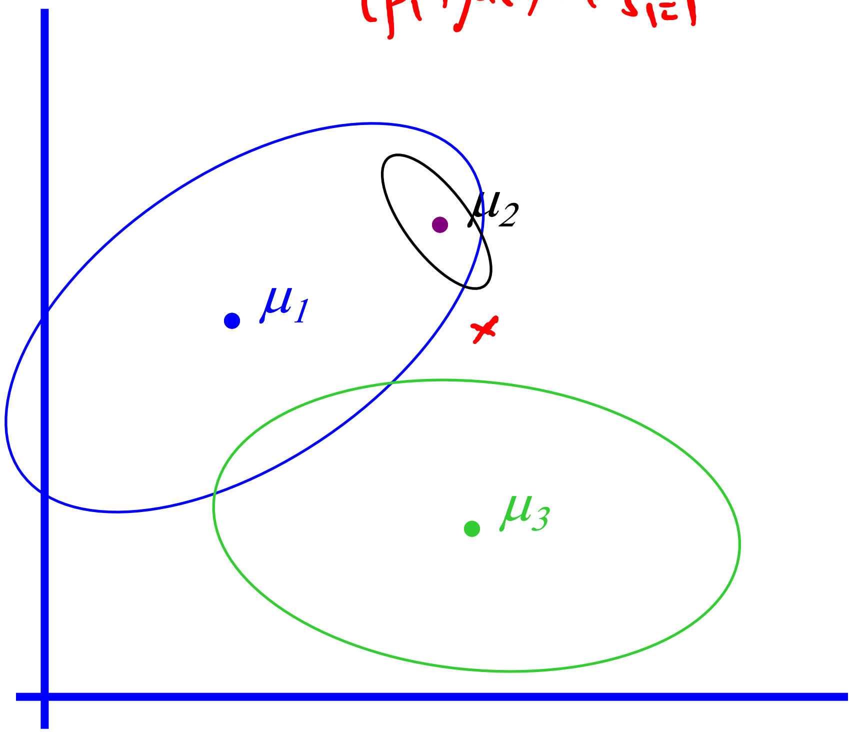


Mixture models

GMM – Gaussian Mixture Model (Multi-modal distribution)

$\{p_i, \mu_i, \Sigma_i\}_{i=1}^{K}$

- There are k components
- Component *i* has a probability of getting picked $p_i = P(y=i)$
- Each component generates data from a Gaussian with mean μ_i and covariance matrix Σ_i



Each data point is generated according to the following recipe:

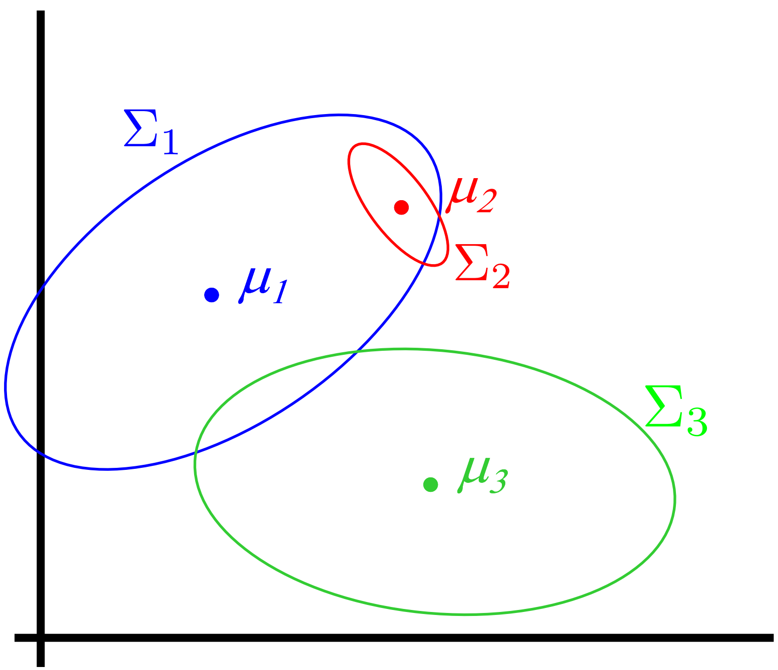
1) Pick a component at random: Choose component i with probability $p_i = P(y=i)$
2) Datapoint x $\sim \mathrm{N}(\mu_i, \Sigma_i)$

Mixture models (Gaussian)

$$x_1, \ldots, x_m \overset{\text{drawn from}}{\sim} p(x) = \sum_{i=1}^{k} \overbrace{p(x|Y=i)P(Y=i)}^{p(x,Y=i)}$$

Mixture component ($p(x|Y=i)$)

Mixture proportion, p_i ($P(Y=i)$)



Gaussian mixture model

$$p(x|Y=i) \sim \mathcal{N}(\mu_i, \Sigma_i)$$

Parameters: $\{p_i, \mu_i, \Sigma_i\}_{i=1}^{K}$

$P(Y=i \mid X)$ soft assignment

- How to estimate parameters? Max Likelihood
  But don't know labels Y (recall Gaussian Bayes classifier)

Expectation-Maximization (EM)

A general algorithm to deal with hidden data, but we will study it in the context of unsupervised learning (hidden labels)

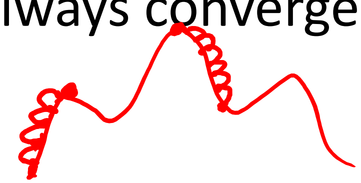
- No need to choose step size as in Gradient methods. ←
- EM is an Iterative algorithm with two linked steps: $\{p_i, \mu_i, \Sigma_i\}_{i=1}^{k}$

  E-step: fill-in hidden data (Y) using inference

  M-step: apply standard MLE/MAP method to estimate parameters

  $\{p_i, \mu_i, \Sigma_i\}^k_{i=1}$

- Not guaranteed to converge to global optimum. BUT…
- This procedure monotonically improves the marginal likelihood of observed data (or leaves it unchanged). Thus it always converges to a local optimum of the likelihood.



EM for spherical, same variance GMMs same mixture proportions

$\{p_i\}_{i=1}^{K} = \frac{1}{K}$

$\Sigma_i = \sigma^2 I$

$\{p_i, \mu_i, \Sigma_i\}_{i=1}^{K}$

Initialize: $\mu_1, \mu_2, \ldots, \mu_K$ randomly

E-step

Compute "expected" classes of all datapoints for each class

$$P(y = i \mid x_j, \mu_1 \ldots \mu_k) \propto \exp\left(-\frac{1}{2\sigma^2}\left\|x_j - \mu_i\right\|^2\right) P(y = i)$$

$$\propto \underbrace{p(x_j \mid y=i)}_{\sim N(\mu_i, \sigma^2 I)} P(y=i)$$

In K-means "E-step" we do hard assignment

EM does soft assignment

M-step

Compute Max. like μ given our data's class membership distributions (weights)

EM update

$$\hat{\mu}_i = \frac{\sum_{j \in C(i)} p(y=i \mid x_j)\, x_j}{\sum_{j \in C(i)} p(y=i \mid x_j)}$$

$$\hat{\mu}_i = \frac{\sum_{j \in C(i)} x_j}{\sum_{j \in C(i)} 1}$$

K-means

Iterate.

EM for spherical, same variance GMMs same mixture proportions

Initialize: $\mu_1, \mu_2, \ldots, \mu_K$ randomly

E-step

Compute "expected" classes of all datapoints for each class

$$P(y=i|x_j,\mu_1\ldots\mu_k) \propto \exp\left(-\frac{1}{2\sigma^2}\|x_j-\mu_i\|^2\right)P(y=i)$$

In K-means "E-step" we do hard assignment

EM does soft assignment

M-step

Compute Max. like **μ** given our data's class membership distributions (weights)

$$\mu_i = \frac{\sum_{j=1}^{m} P(y=i|x_j)\,x_j}{\sum_{j=1}^{m} P(y=i|x_j)}$$

Exactly same as MLE with weighted data

Iterate.

EM for general GMMs

Iterate. On iteration t let our estimates be

$\lambda_t = \{ \mu_1^{(t)}, \mu_2^{(t)} \ldots \mu_k^{(t)}, \Sigma_1^{(t)}, \Sigma_2^{(t)} \ldots \Sigma_k^{(t)}, p_1^{(t)}, p_2^{(t)} \ldots p_k^{(t)} \}$

$p_i^{(t)}$ is shorthand for estimate of $P(y=i)$ on t'th iteration

E-step

Compute "expected" classes of all datapoints for each class

$$\mathrm{P}\left(y=i \middle| x_j, \lambda_t\right) \propto p_i^{(t)} \mathrm{p}\left(x_j \middle| \mu_i^{(t)}, \Sigma_i^{(t)}\right)$$

Just evaluate a Gaussian at x_j

$$\propto \underbrace{p(x_j \mid y=i, \lambda_t)}_{\sim N(\mu_i, \Sigma_i)} \; \underline{p(y=i)}$$

M-step

Compute MLEs given our data's class membership distributions (weights)

EM for general GMMs

Iterate. On iteration t let our estimates be

$\lambda_t = \{ \mu_1^{(t)}, \mu_2^{(t)} \ldots \mu_k^{(t)}, \Sigma_1^{(t)}, \Sigma_2^{(t)} \ldots \Sigma_k^{(t)}, p_1^{(t)}, p_2^{(t)} \ldots p_k^{(t)} \}$

$p_i^{(t)}$ is shorthand for estimate of $P(y=i)$ on t'th iteration

E-step

Compute "expected" classes of all datapoints for each class

$$\mathrm{P}\left(y=i \middle| x_j, \lambda_t\right) \propto p_i^{(t)} \mathrm{p}\left(x_j \middle| \mu_i^{(t)}, \Sigma_i^{(t)}\right)$$

Just evaluate a Gaussian at x_j

M-step

Compute MLEs given our data's class membership distributions (weights)

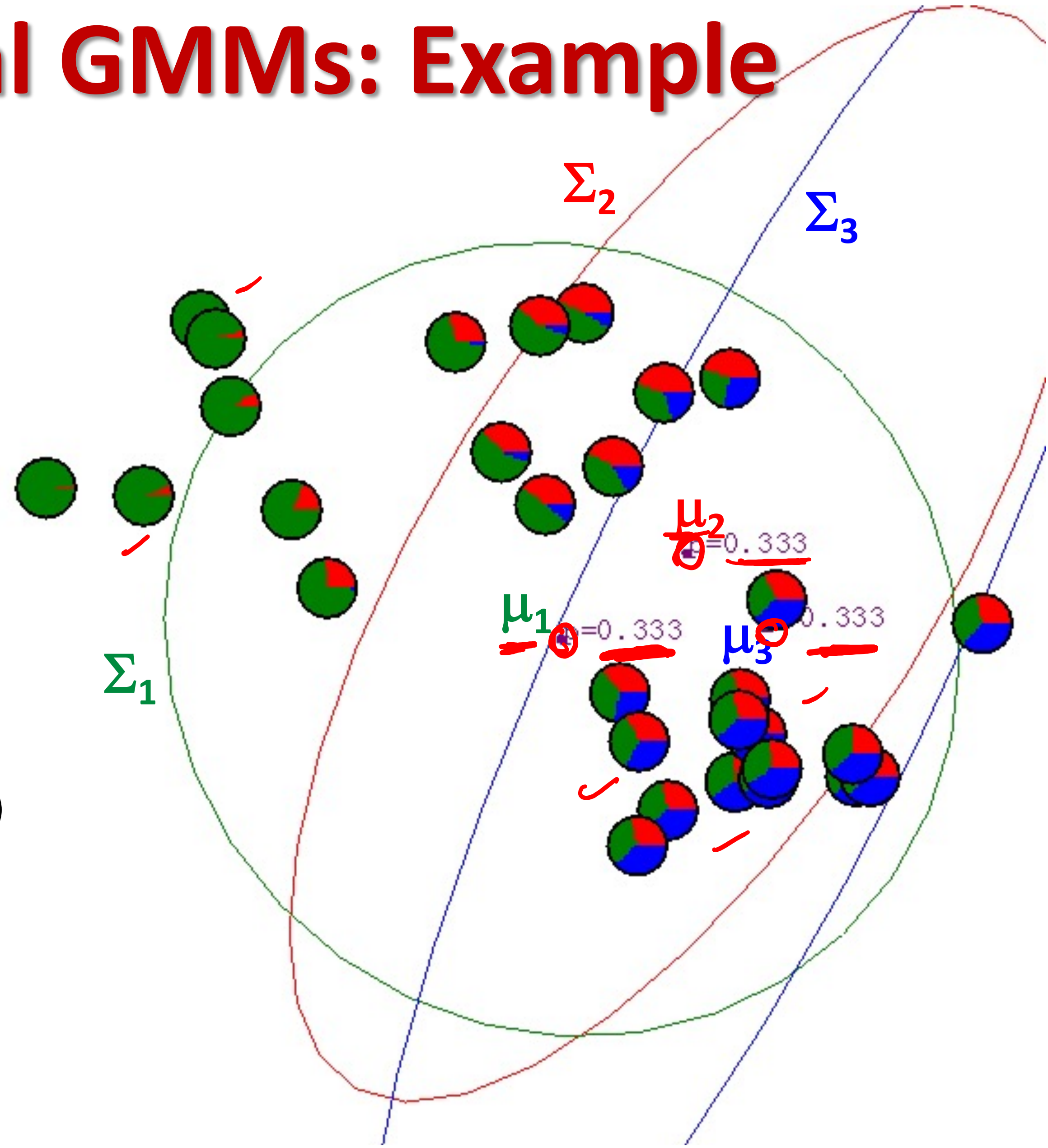
$$\mu_i^{(t+1)} = \frac{\sum_j \mathrm{P}\left(y=i \middle| x_j, \lambda_t\right) x_j}{\sum_j \mathrm{P}\left(y=i \middle| x_j, \lambda_t\right)} \qquad \Sigma_i^{(t+1)} = \frac{\sum_j \mathrm{P}\left(y=i \middle| x_j, \lambda_t\right)\left(x_j - \mu_i^{(t+1)}\right)\left(x_j - \mu_i^{(t+1)}\right)^{\mathrm{T}}}{\sum_j \mathrm{P}\left(y=i \middle| x_j, \lambda_t\right)}$$

$$p_i^{(t+1)} = \frac{\sum_j \mathrm{P}\left(y=i \middle| x_j, \lambda_t\right)}{m}$$

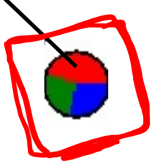
m = #data points

Iterate.

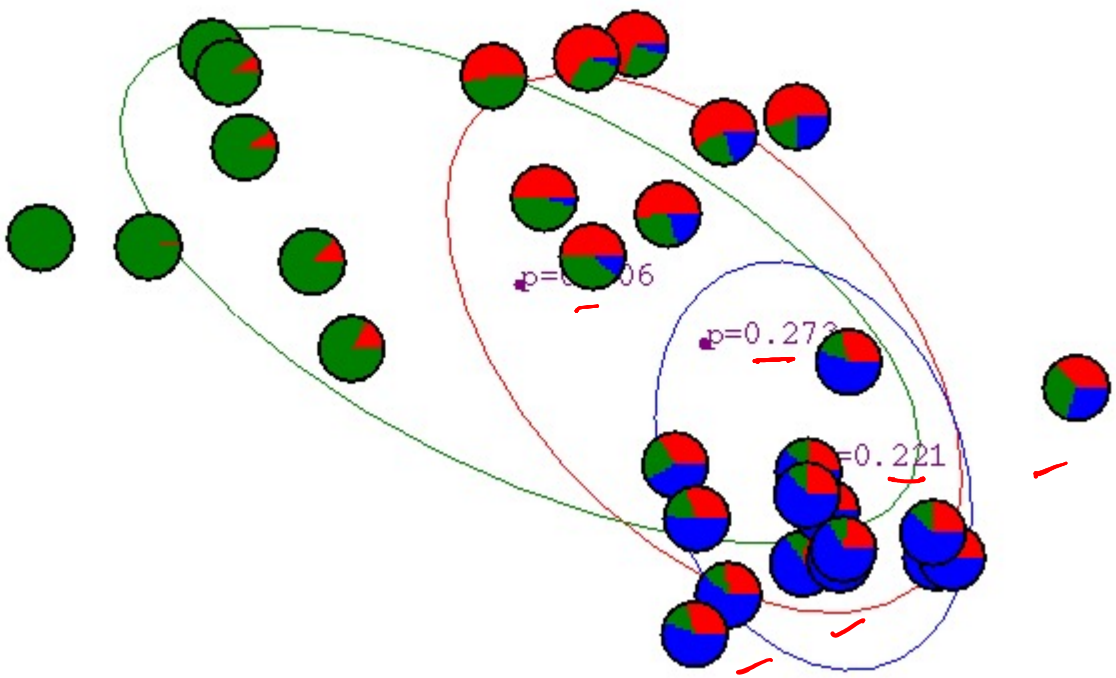
EM for general GMMs: Example



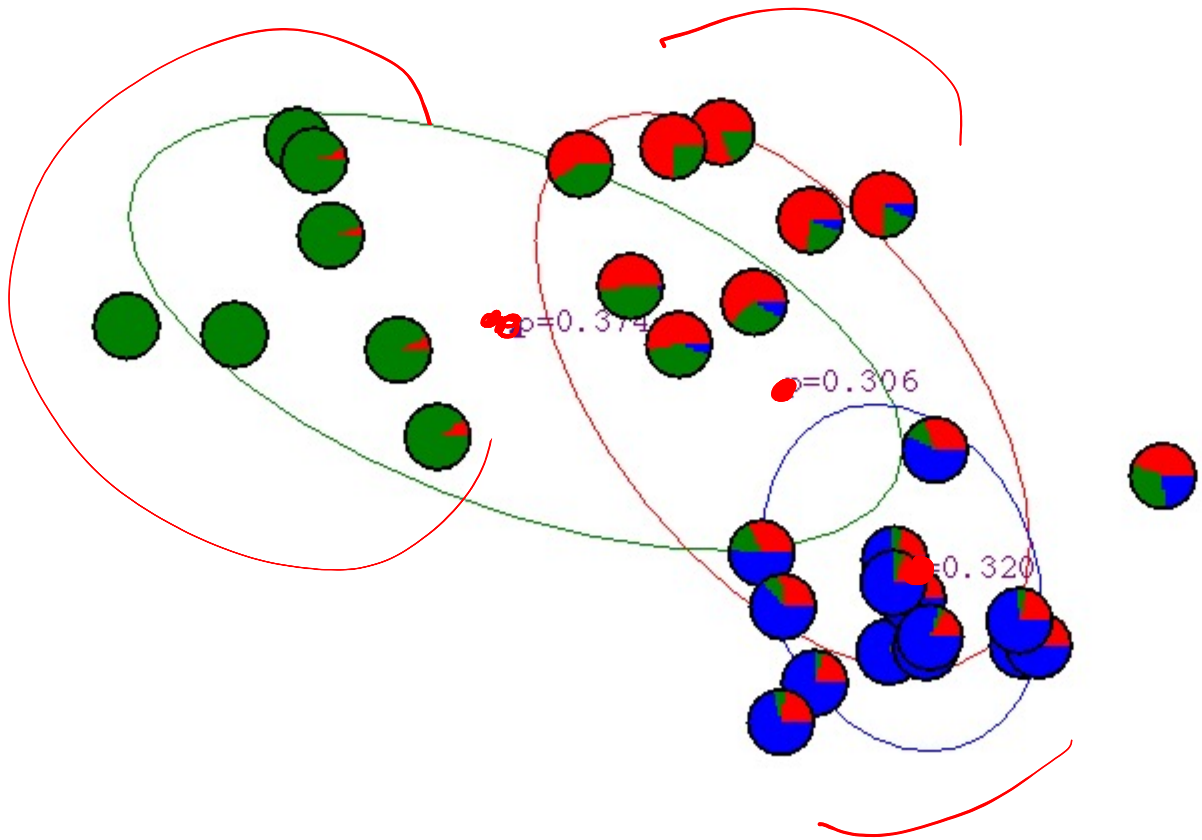
$P(y = \bullet \mid \underbrace{x_j, \mu_1, \mu_2, \mu_3, \Sigma_1, \Sigma_2, \Sigma_3, p_1, p_2, p_3}_{})$



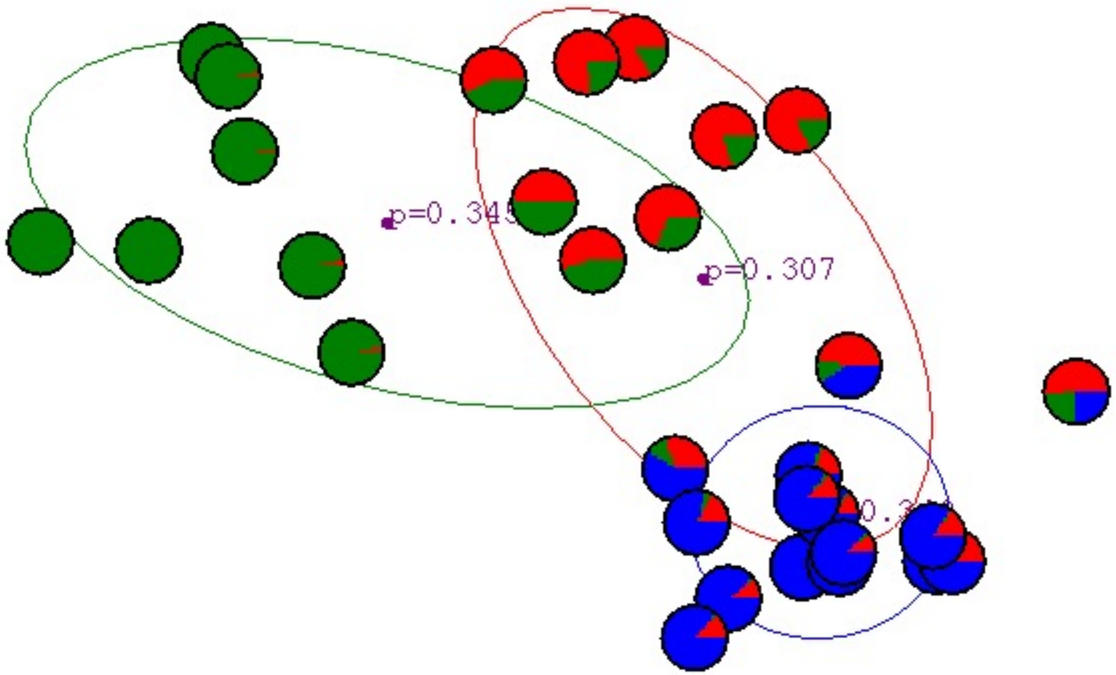
After 1st iteration



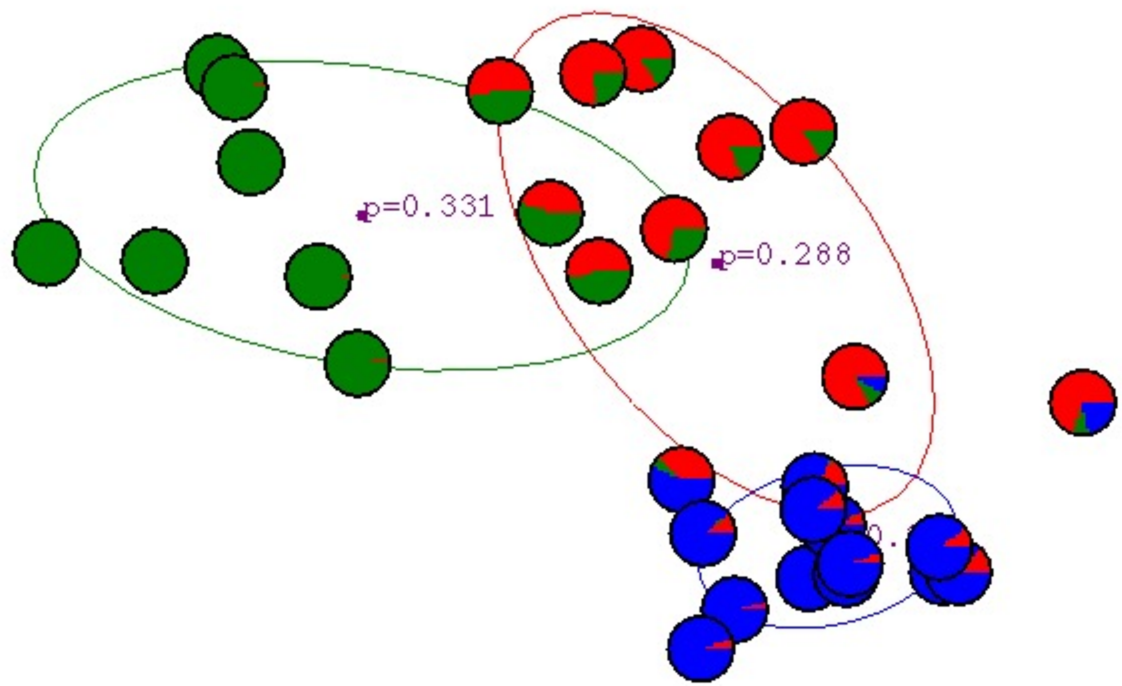
After 2nd iteration



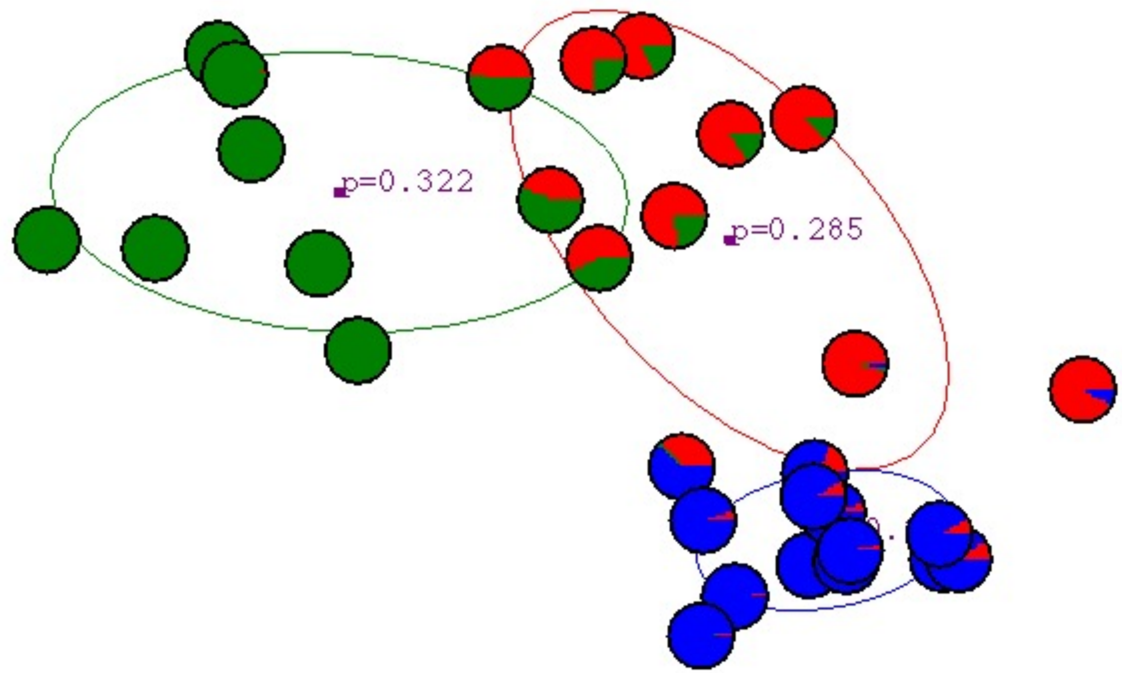
After 3rd iteration



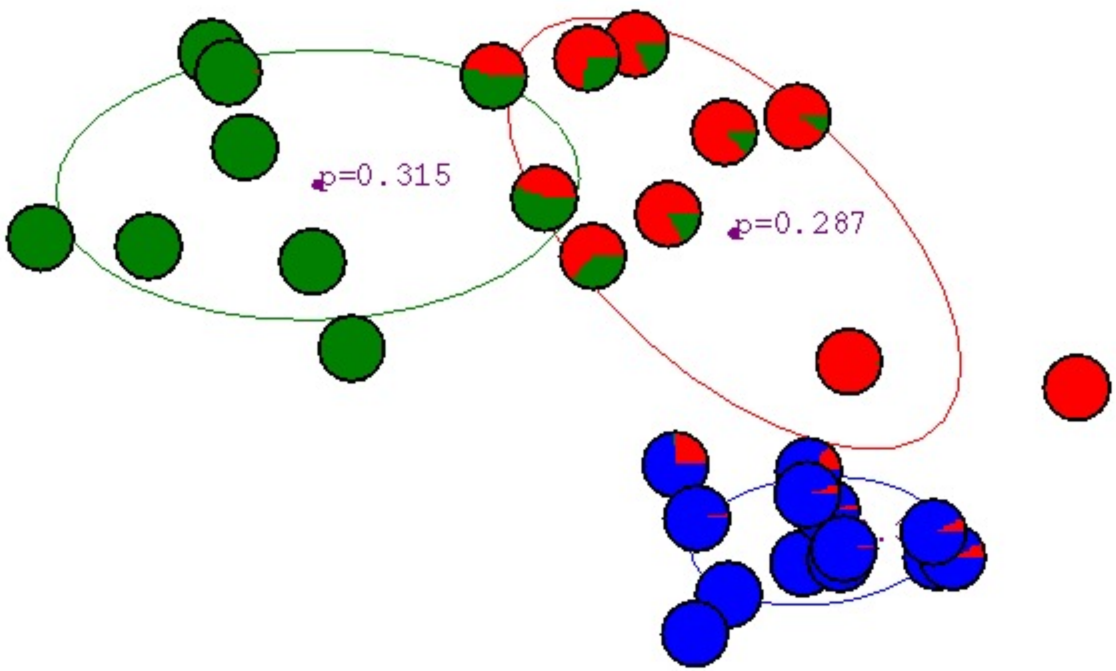
After 4th iteration



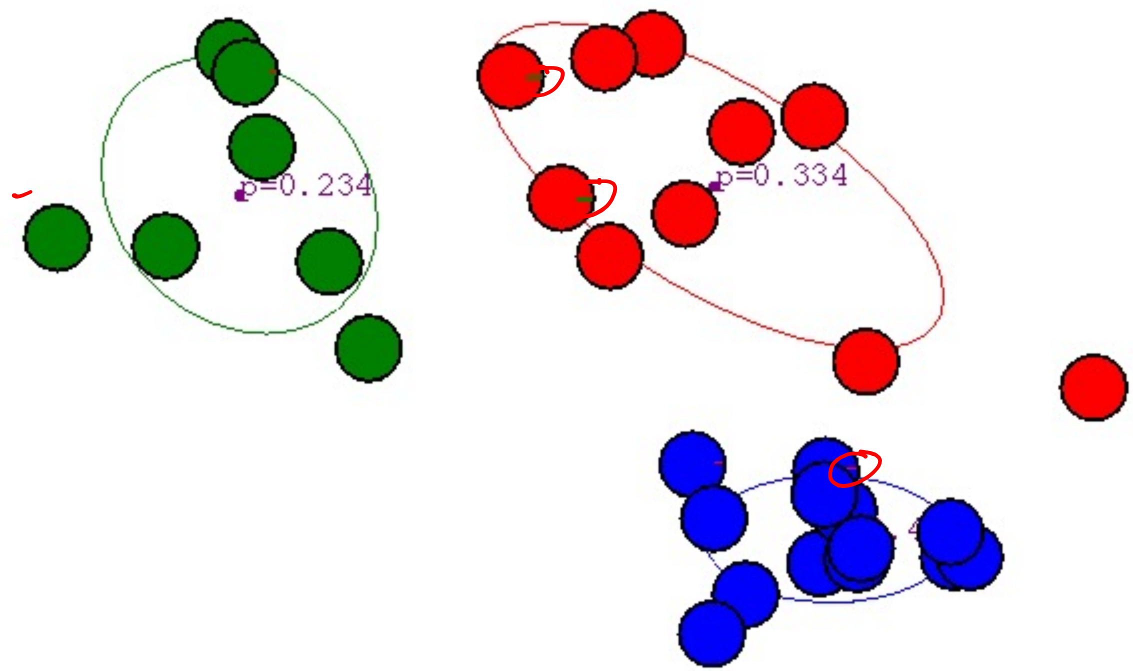
After 5th iteration



After 6th iteration



After 20th iteration



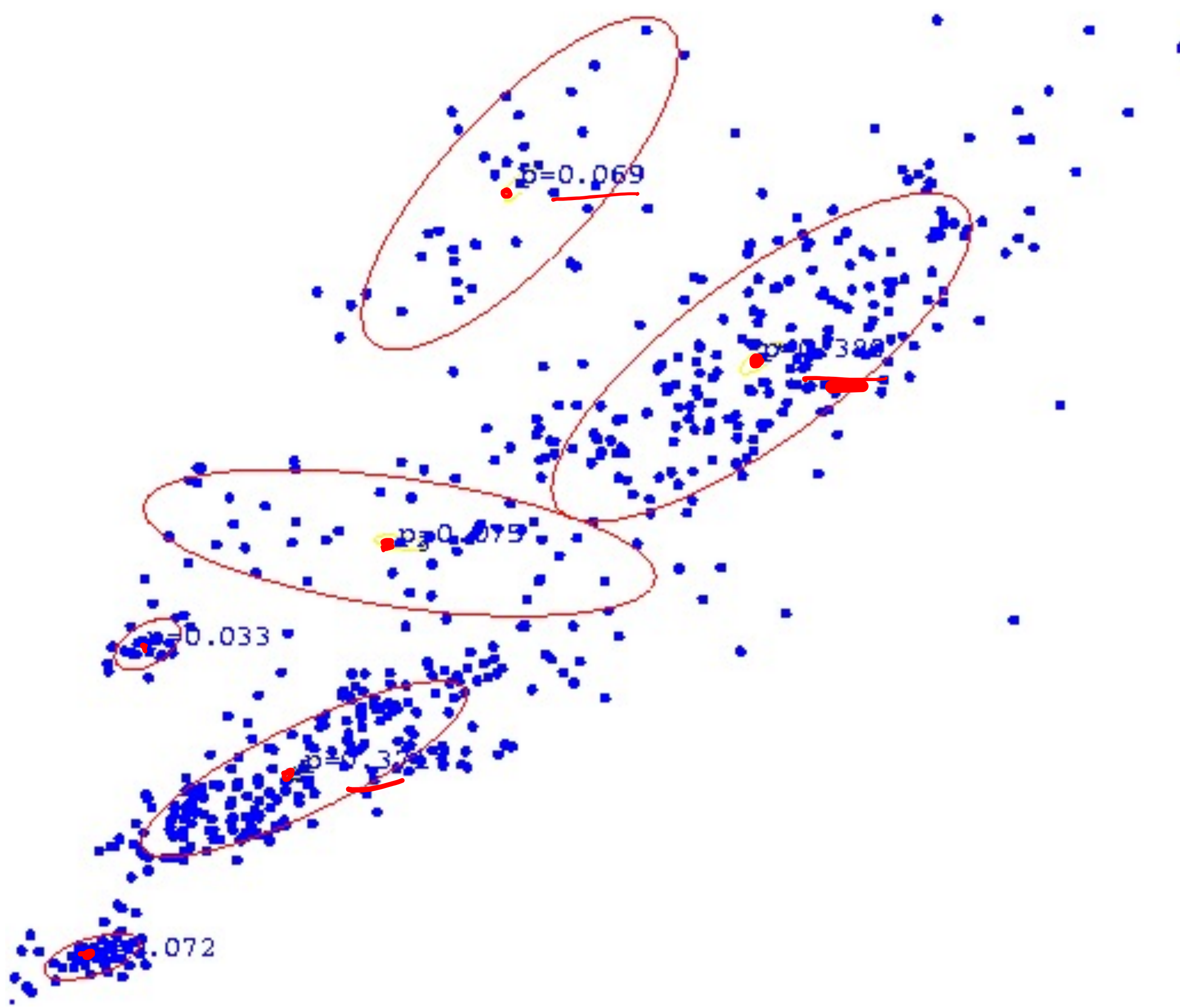
GMM clustering of assay data

$K=7$

$\{\mu_i\}_{i=1}^{K}$

$\{\Sigma_i\}_{i=1}^{K}$

$\{p_i\}_{i=1}^{K}$



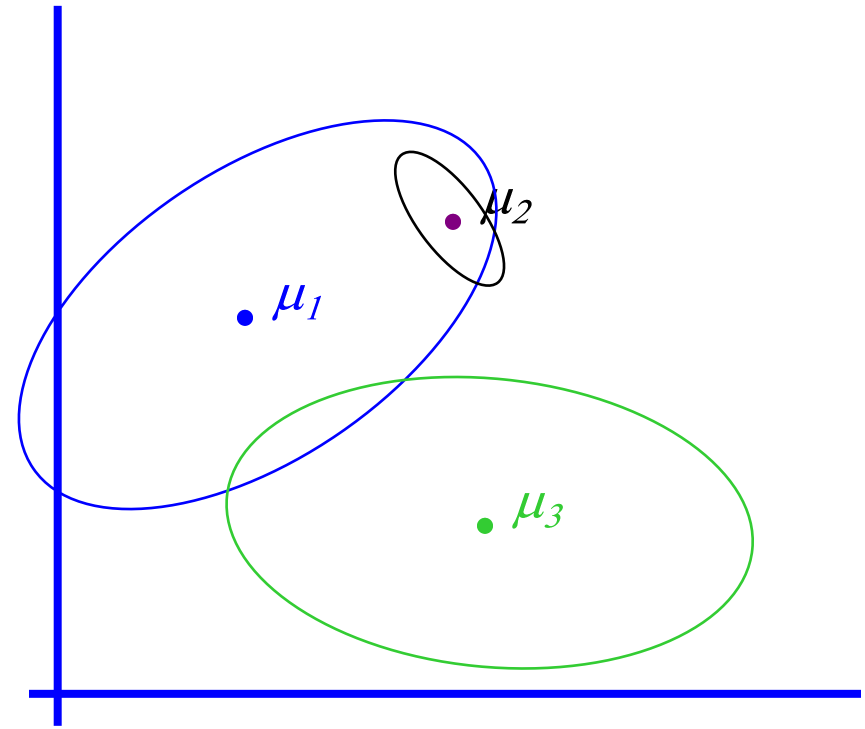
General GMM

GMM – Gaussian Mixture Model (Multi-modal distribution)

$$\boxed{p(x)} = \sum_i p(x|y=i)\, \overset{\pi_i}{P(y=i)}$$

Mixture component **Mixture proportion**

$$p(x|y=i) \sim \mathrm{N}(\mu_i, \Sigma_i)$$

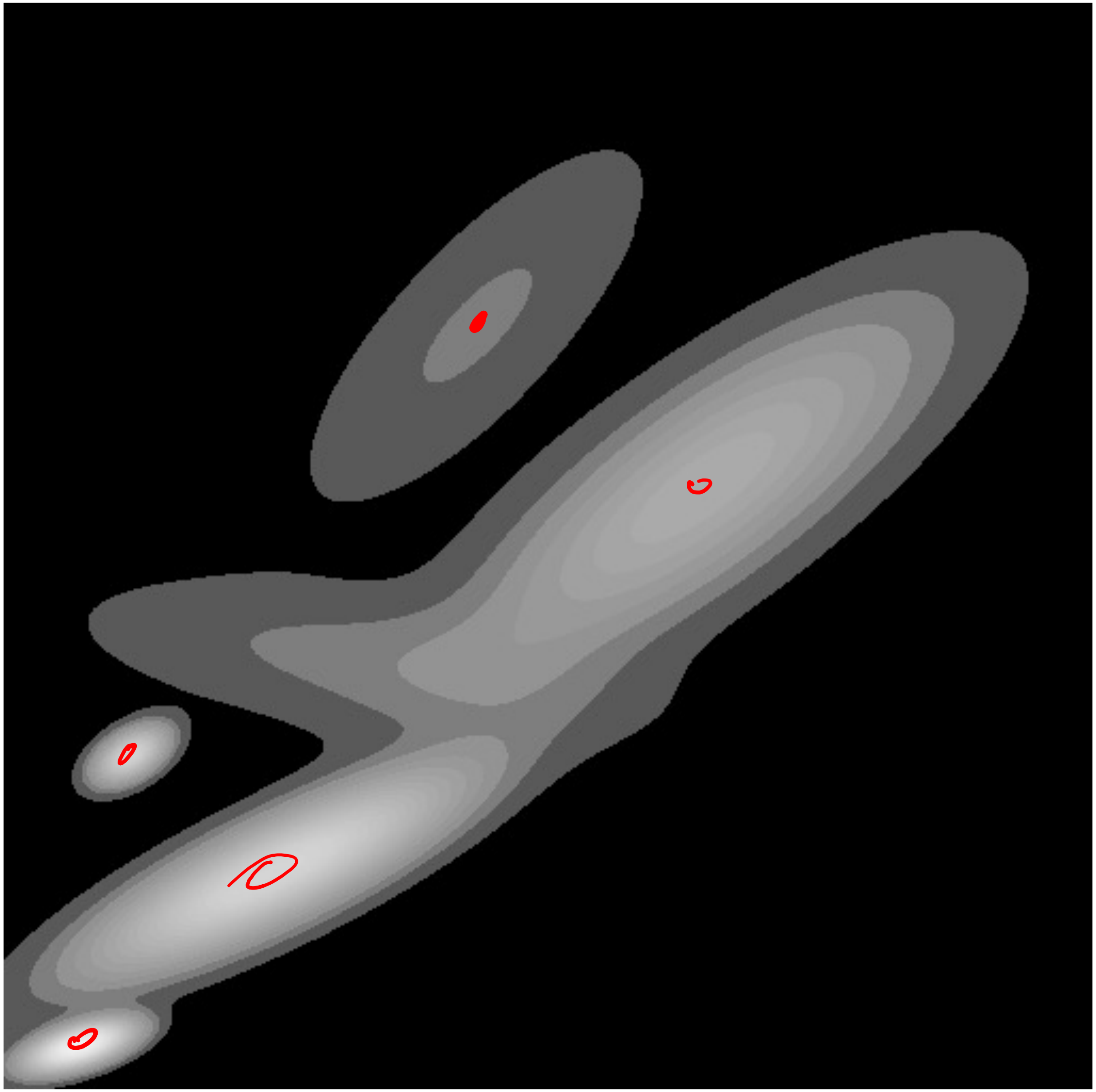


Resulting Density Estimator

$\{\mu_i, \Sigma_i, p_i\}_{i=1}^{K}$

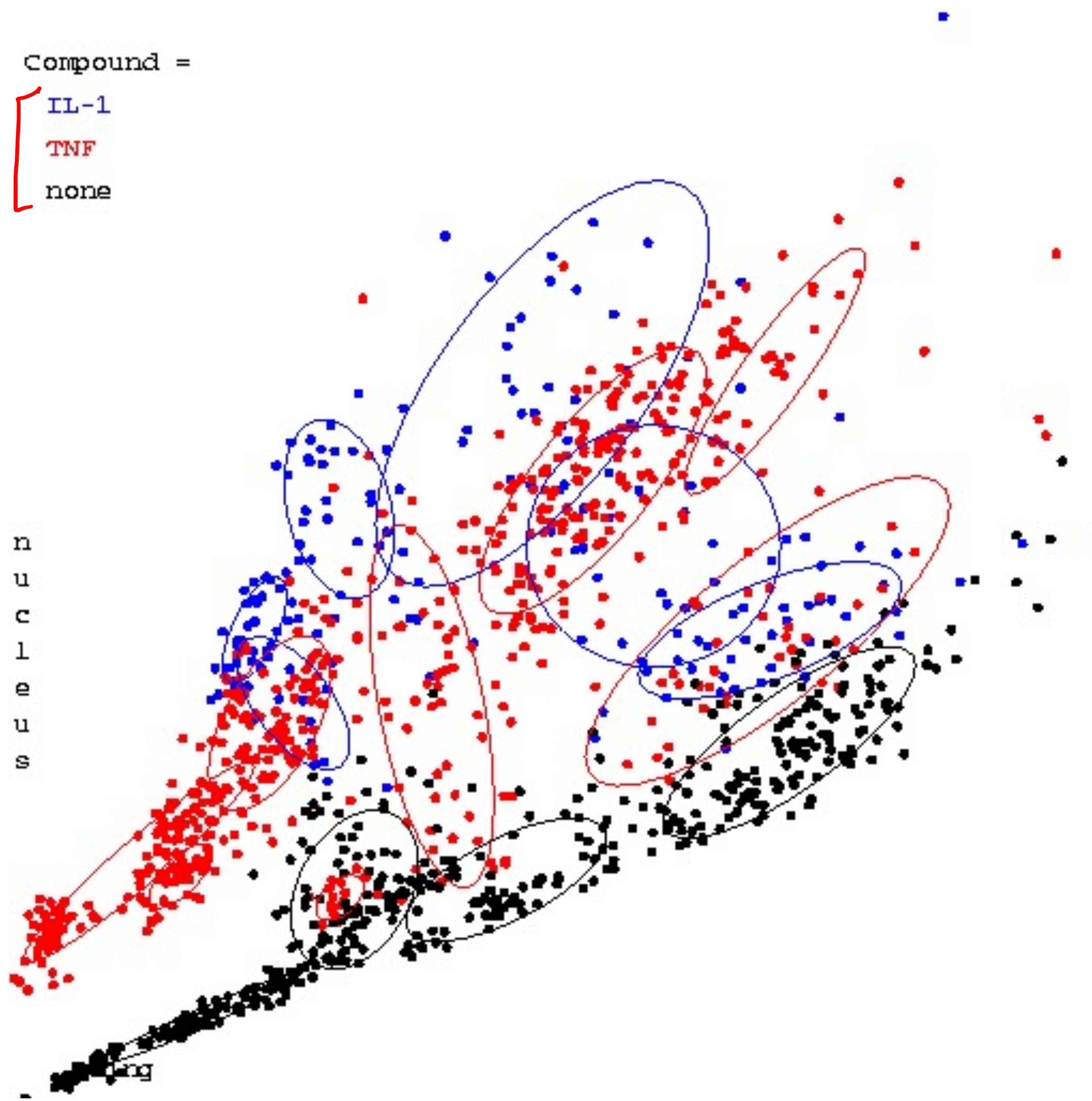
$p(x)$

Level sets



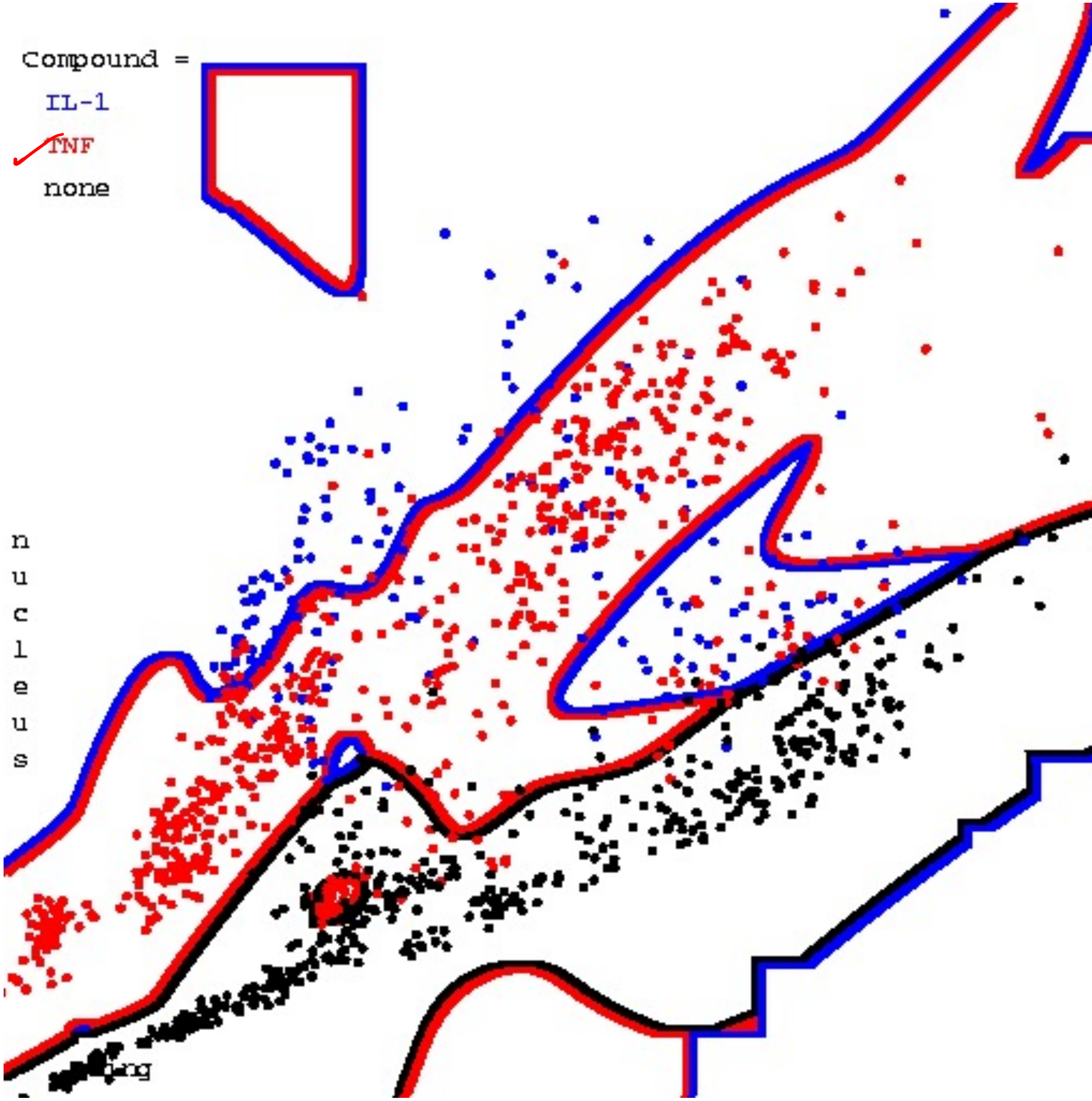
Three classes of assay

(each learned with it's own mixture model)



Resulting Bayes Classifier

$p(x \mid y = \text{TNF})$



Summary: EM Algorithm

- A way of maximizing likelihood function for hidden variable models. Finds MLE of parameters when the original (hard) problem can be broken up into two (easy) pieces:
  1. Estimate some "missing" or "unobserved" data from observed data and current parameters.
  2. Using this "complete" data, find the maximum likelihood parameter estimates.

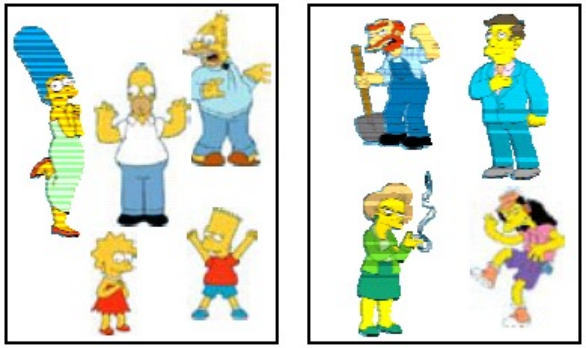
- Alternate between filling in the latent variables using the best guess (posterior) and updating the parameters based on this guess:
  1. E-step: soft cluster assignment for each data point
  2. M-step: update parameters of each mixture component

- EM can get stuck in local minima.

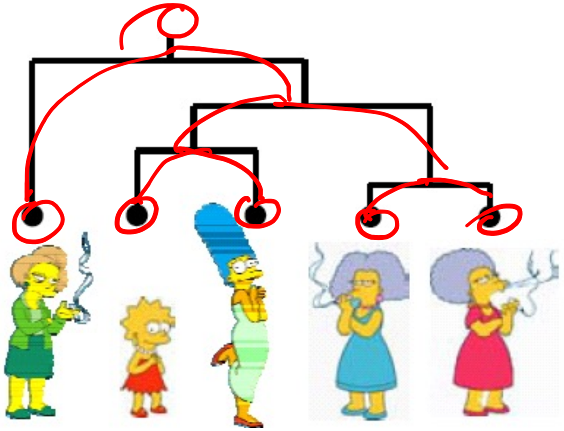
- BUT very popular in practice.

Clustering Algorithms

- Partition algorithms
  - K means clustering
  - Mixture-Model based clustering



- Hierarchical algorithms
  - Single-linkage
  - Average-linkage
  - Complete-linkage
  - Centroid-based



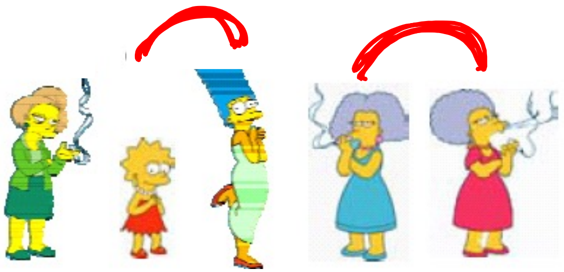
Hierarchical Clustering

- Bottom-Up Agglomerative Clustering

  Starts with each object in a separate cluster, and repeat:
  - Joins the most similar pair of clusters,
  - Update the similarity of the new cluster to others

  until there is only one cluster.

  Greedy - less accurate but simple to implement



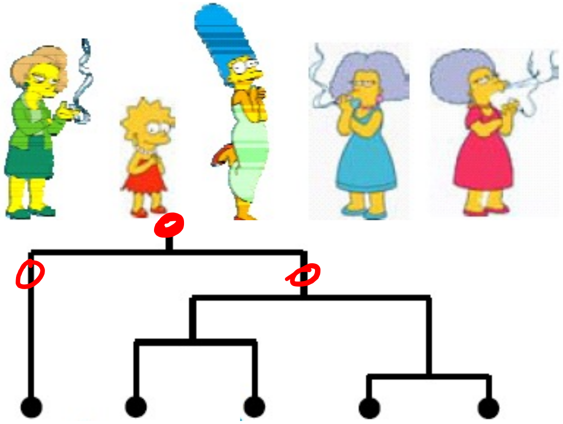
$d(x, x')$

- Top-Down divisive

  Starts with all the data in a single cluster, and repeat:
  - Split each cluster into two using a partition algorithm

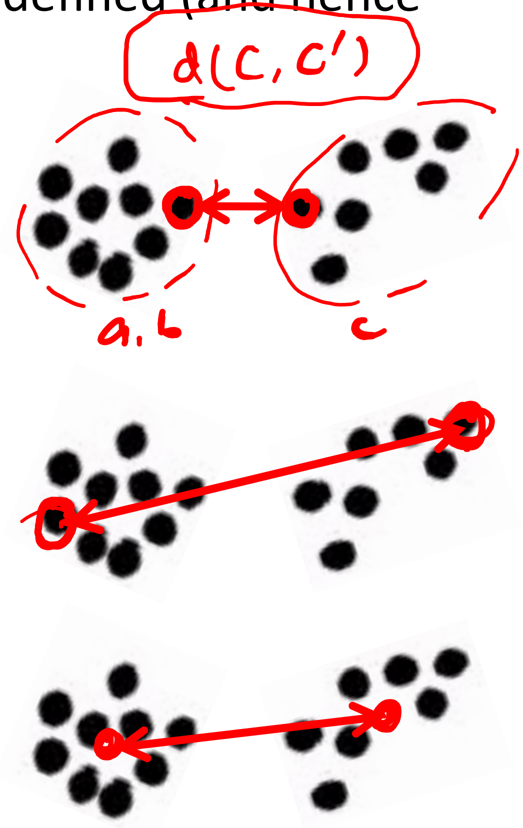
  Until each object is a separate cluster.

  More accurate but complex to implement



Bottom-up Agglomerative clustering

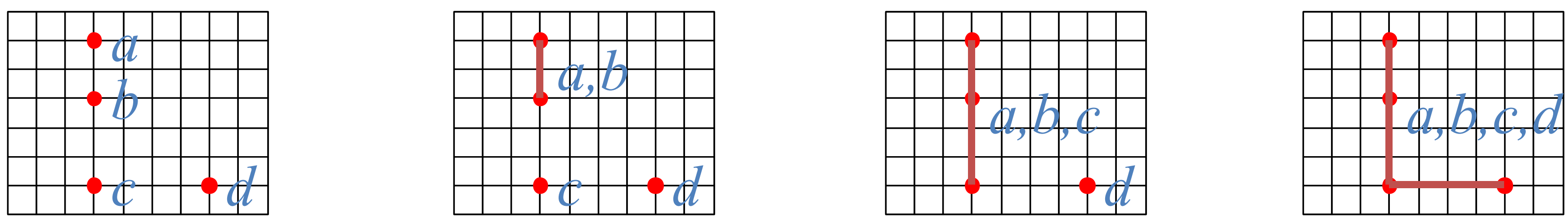
Different algorithms differ in how the similarities are defined (and hence updated) between two clusters



- Single-Linkage
  - Nearest Neighbor: similarity between their closest members.
- Complete-Linkage
  - Furthest Neighbor: similarity between their furthest members.
- Centroid
  - Similarity between the centers of gravity
- Average-Linkage
  - Average similarity of all cross-cluster pairs.

Single-Linkage Method

Euclidean Distance



(1)

(2)

(3)

	b	***c***	***d***
a	2	5	6
b		3	5
c			4

	b	***c***	***d***
a	2	5	6
b		3	5
c			4

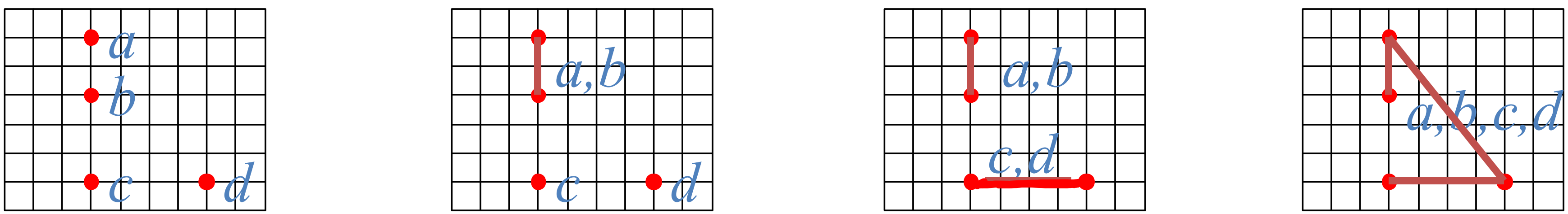
	c	***d***
a,b	3	5
c		4

	d
a,b,c	4

Distance Matrix

Complete-Linkage Method

Euclidean Distance



(1) (2) (3)

	b	c	d
a	2	5	6
b		3	5
c			4

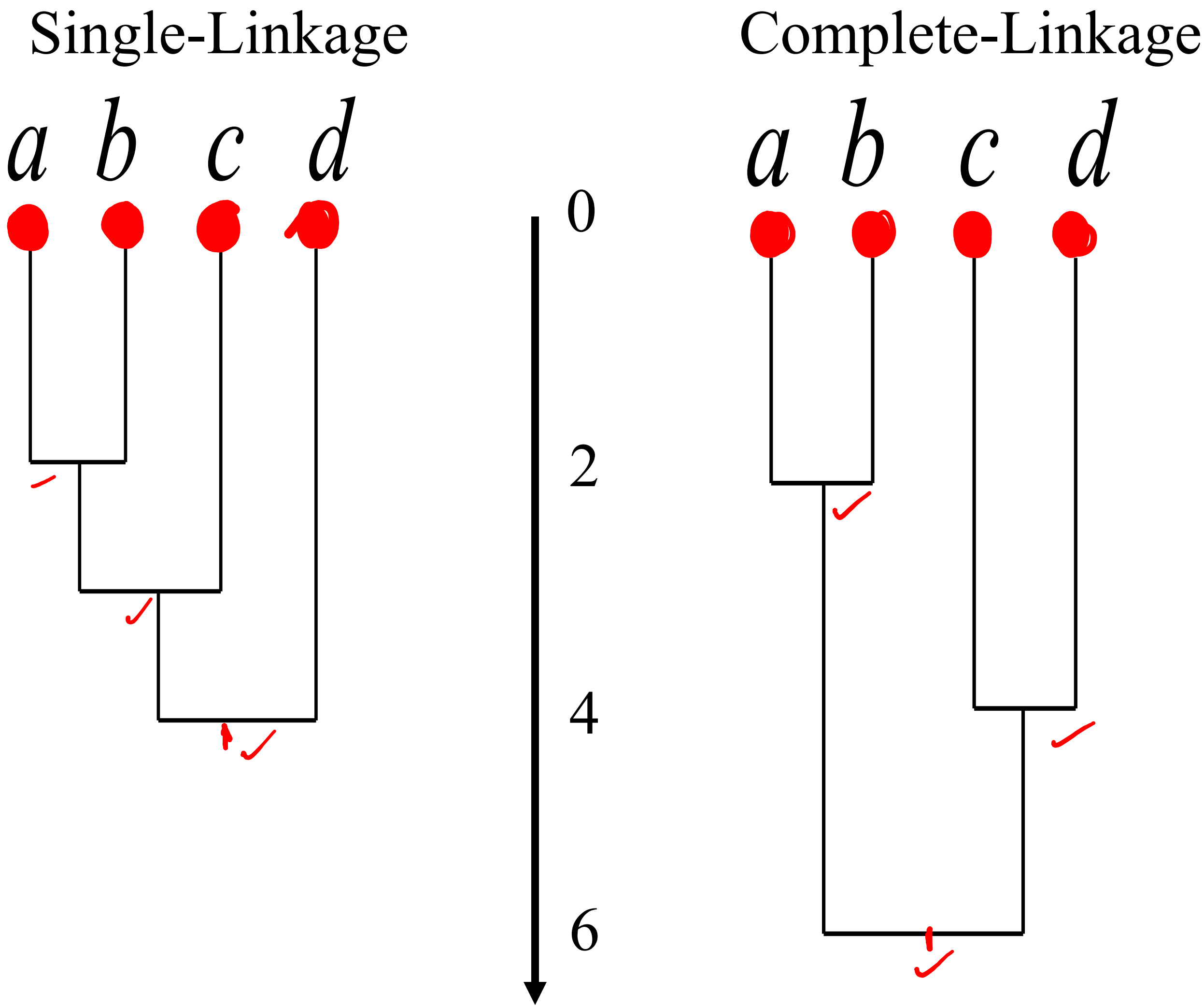
	b	c	d
a	2	5	6
b		3	5
c			4

	c	d
a,b	5	6
c		4

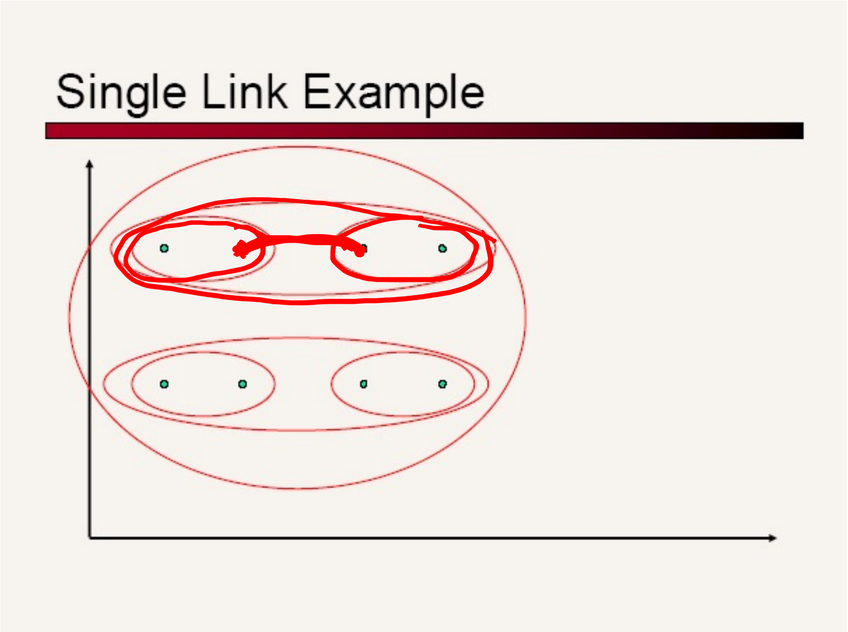
	c,d
a,b	6

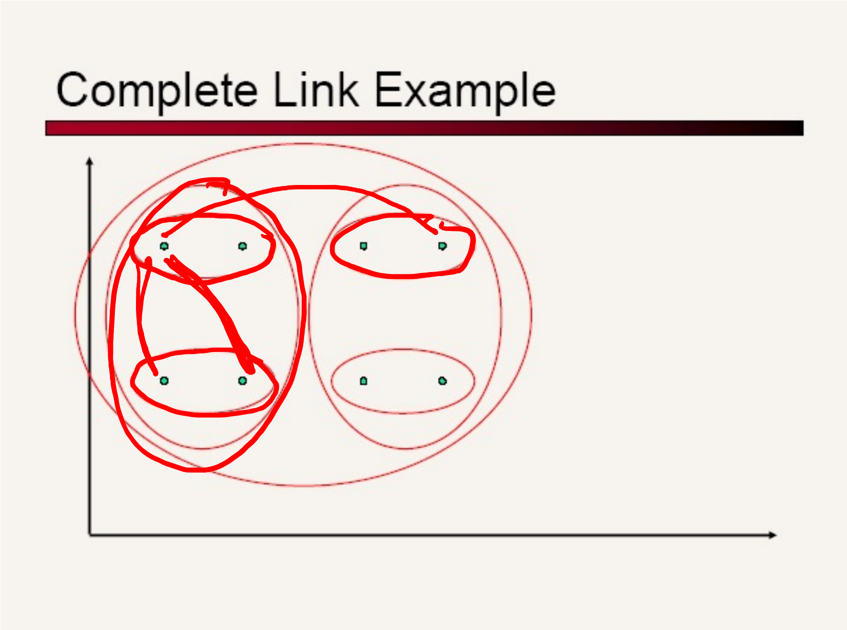
Distance Matrix

Dendrograms



Another Example

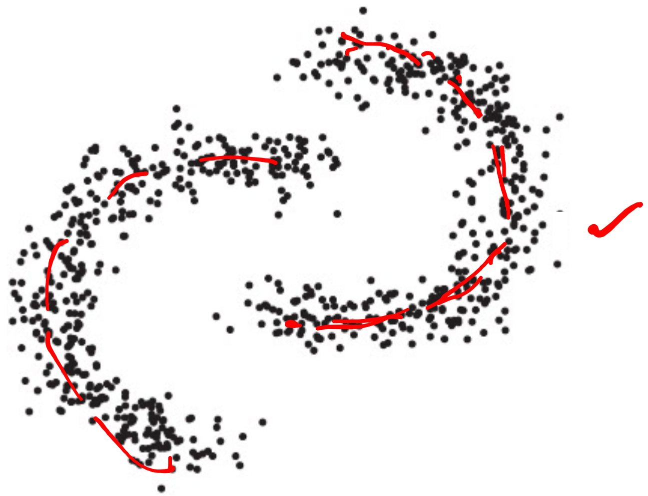




Single vs. Complete Linkage

Shape of clusters

Single-linkage	allows anisotropic and non-convex shapes
Complete-linkage	assumes isotopic, convex shapes



What you need to know…

- Partition based clustering algorithms
  - K-means
    - Coordinate descent
    - Seeding
    - Choosing K
  - Mixture models
    EM algorithm

- Hierarchical clustering algorithms
  - Single-linkage
  - Complete-linkage
  - Centroid-linkage
  - Average-linkage