Gaussian Mixture Models

Aarti Singh

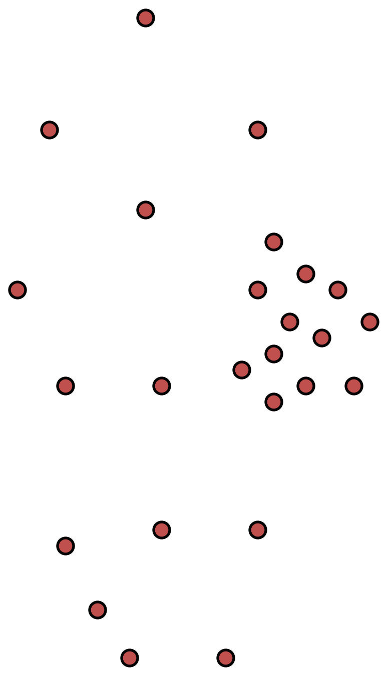
Machine Learning 10-315
Nov 13, 2019

Some slides courtesy of Eric Xing, Carlos Guestrin

MACHINE LEARNING DEPARTMENT

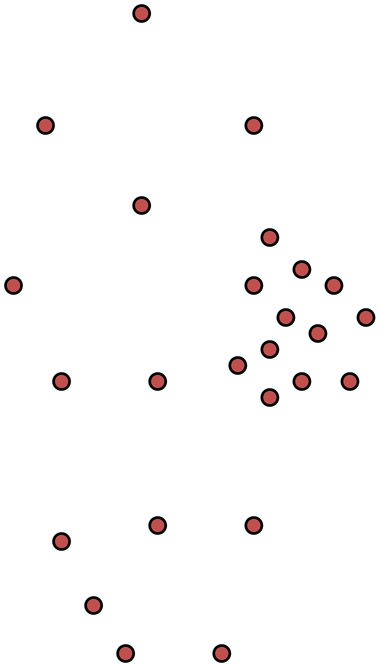
Carnegie Mellon.
School of Computer Science

(One) bad case for K-means



- Clusters may overlap
- Some clusters may be “wider” than others
- Clusters may not be linearly separable

(One) bad case for K-means



- Clusters may overlap
- Some clusters may be “wider” than others
- Clusters may not be linearly separable

Partitioning Algorithms

- K-means
  - **hard assignment**: each object belongs to only one cluster
- Mixture modeling
  - **soft assignment**: probability that an object belongs to a cluster

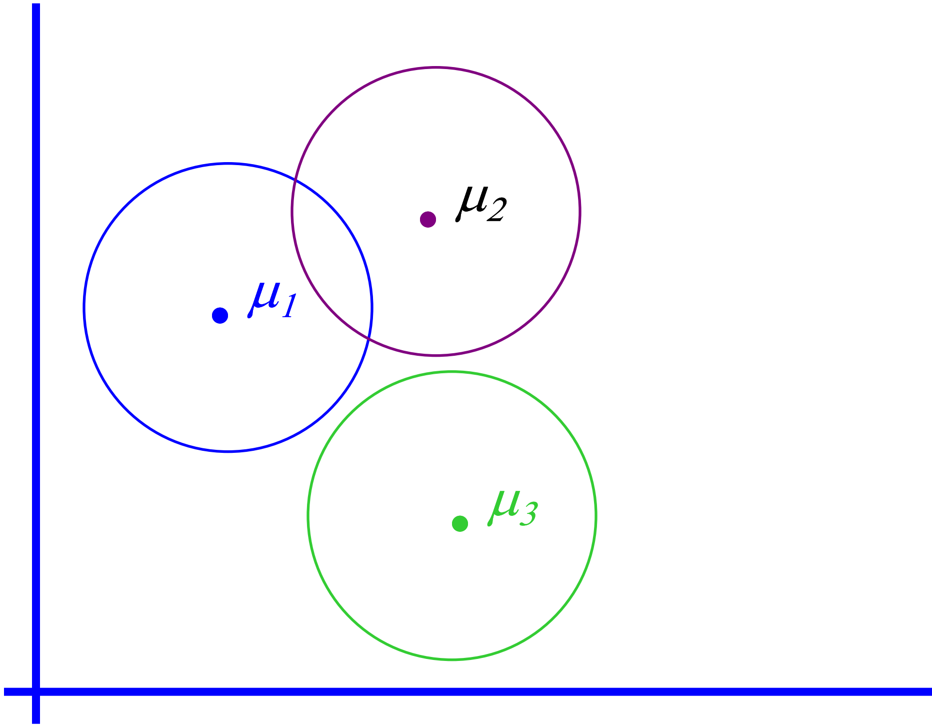
Generative approach

Gaussian Mixture Model

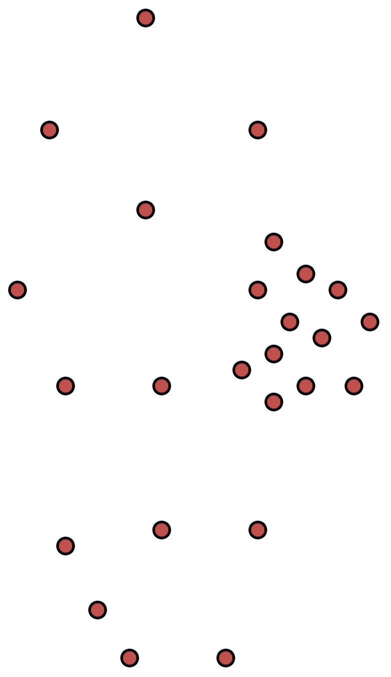
Mixture of K Gaussian distributions: (Multi-modal distribution)

$$p(x|y=i) \sim \mathrm{N}(\mu_i, \sigma^2 \boldsymbol{I})$$

$$p(x) = \sum_i \underset{\substack{\big\downarrow \\ \textbf{Mixture} \\ \textbf{component}}}{p(x|y=i)} \; \underset{\substack{\big\downarrow \\ \textbf{Mixture} \\ \textbf{proportion}}}{P(y=i)}$$



(One) bad case for K-means



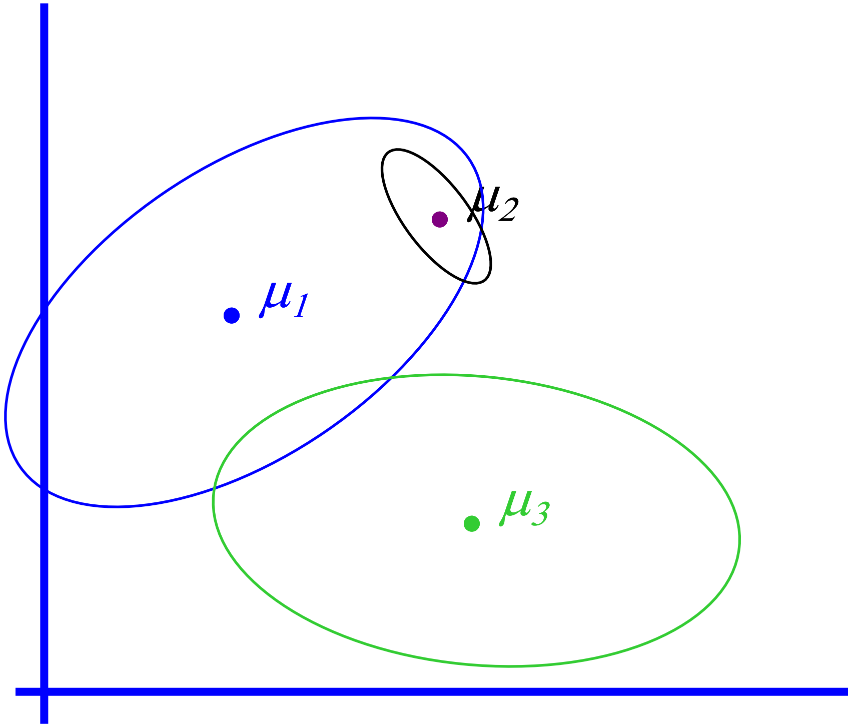
- Clusters may overlap
- Some clusters may be “wider” than others
- Clusters may not be linearly separable

General GMM

GMM – Gaussian Mixture Model (Multi-modal distribution)

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

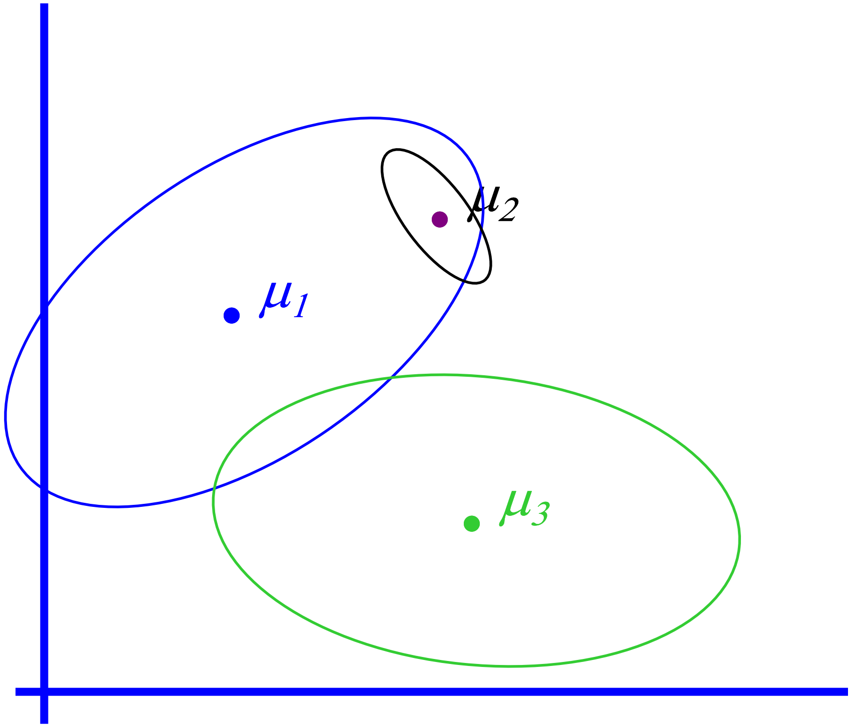
$$p(x) = \sum_i \underset{\substack{\big\downarrow \\ \textbf{Mixture} \\ \textbf{component}}}{p(x|y=i)}\ \underset{\substack{\big\downarrow \\ \textbf{Mixture} \\ \textbf{proportion}}}{P(y=i)}$$



General GMM

GMM – Gaussian Mixture Model (Multi-modal distribution)

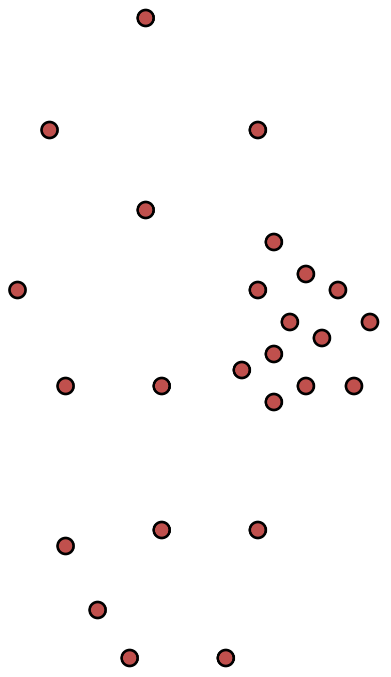
- There are k components
- Component *i* has an associated mean vector μ_i
- Each component generates data from a Gaussian with mean μ_i and covariance matrix Σ_i



Each data point is generated according to the following recipe:

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

(One) bad case for K-means



- Clusters may overlap
- Some clusters may be “wider” than others
- Clusters may not be linearly separable

General GMM

GMM – Gaussian Mixture Model (Multi-modal distribution)

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

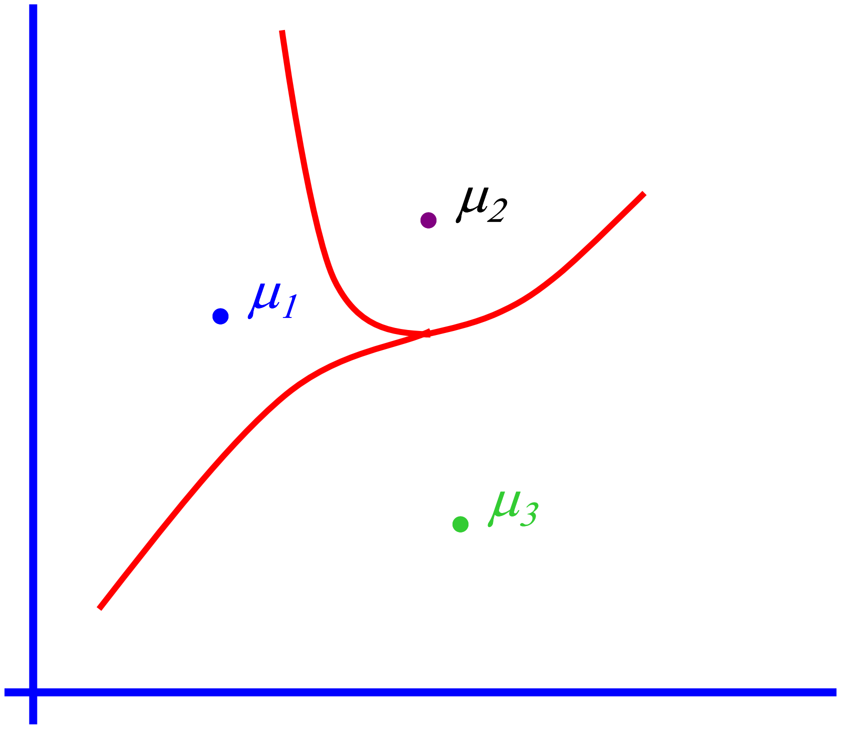
Gaussian Bayes Classifier:

$$\log \frac{P(y=i|x)}{P(y=j|x)}$$

$$= \log \frac{p(x|y=i)P(y=i)}{p(x|y=j)P(y=j)}$$

$$= x^T \boxed{W} x + \boxed{w}^T x$$

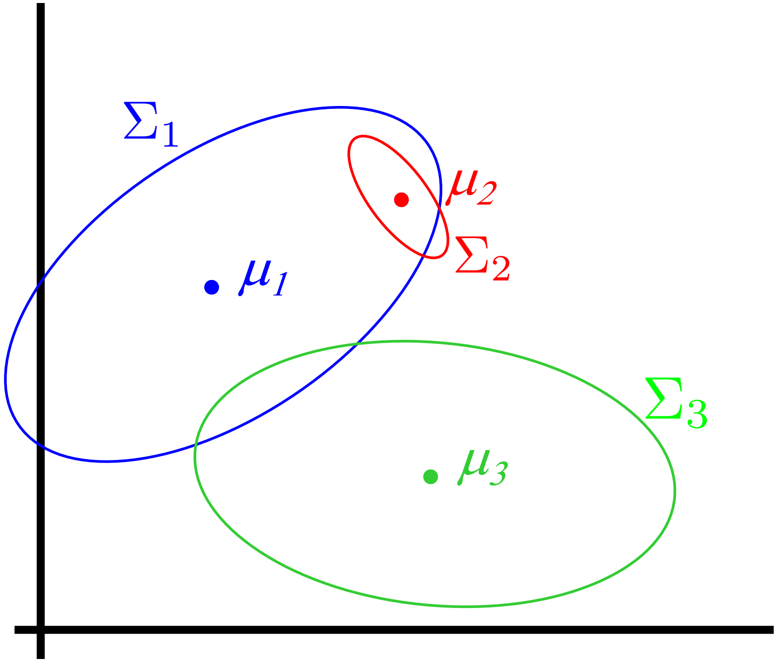
Depend on $\mu_1, \mu_2, .., \mu_K, \Sigma_1, \Sigma_2, .., \Sigma_K$, P(y=1),…, P(Y=k)



"Quadratic Decision boundary" – second-order terms don't cancel out

Learning General GMM

$$x_1, \ldots, x_m \sim p(x) = \sum_{i=1}^{k} \underset{\substack{\downarrow \\ \textbf{Mixture} \\ \textbf{component}}}{p(x|Y=i)} \; \underset{\substack{\downarrow \\ \textbf{Mixture} \\ \textbf{proportion, } p_i}}{P(Y=i)}$$



Gaussian mixture model

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

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

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

Learning General GMM

Maximize marginal likelihood:

$$\text{argmax} \prod_j P(x_j) = \text{argmax} \prod_j \sum_{i=1}^{K} P(y_j=i, x_j)$$

$$= \text{argmax} \prod_j \sum_{i=1}^{K} P(y_j=i) p(x_j | y_j=i)$$

$P(y_j=i) = P(y=i)$ Mixture component i is chosen with prob $P(y = i)$

$$= \arg\max \prod_{j=1}^{m} \sum_{i=1}^{k} P(y=i) \frac{1}{\sqrt{\det(\Sigma_i)}} \exp\left[-\frac{1}{2}(x_j - \mu_i)^T \Sigma_i (x_j - \mu_i) \right]$$

How do we find the μ_i, Σ_i s and P(y=i)s which give max. marginal likelihood?

* Set $\frac{\partial}{\partial \mu_i}$ log Prob (....) = 0 and solve for μ_i's. Non-linear not-analytically solvable

* Use gradient descent: Doable, but often slow

GMM vs. k-means

Maximize marginal likelihood:

$$\text{argmax} \prod_j P(x_j) = \text{argmax} \prod_j \sum_{i=1}^{K} P(y_j{=}i,x_j)$$

$$= \text{argmax} \prod_j \sum_{i=1}^{K} P(y_j{=}i)p(x_j|y_j{=}i)$$

- What happens if we assume **Hard assignment**?

$$P(y_j = i) = 1 \text{ if } i = C(j)$$

$$= 0 \text{ otherwise}$$

$$\text{argmax} \prod_j P(x_j) = \text{argmax} \prod_j p(x_j|y_j{=}C(j))$$

$$= \text{argmax} \prod_{j=1}^{n} \exp\left(\frac{-1}{2\sigma^2}\|x_j - \mu_{C(j)}\|^2\right)$$

$$= \text{argmin} \sum_{j=1}^{n} \|x_j - \mu_{C(j)}\|^2) = \arg\min_{\mu,C} F(\mu, C)$$

Same as k-means!

Expectation-Maximization (EM)

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

- No need to choose step size as in Gradient methods.
- EM is an Iterative algorithm with two linked steps:

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

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

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

- We will see that this procedure monotonically improves the likelihood (or leaves it unchanged). Thus it always converges to a local optimum of the likelihood.

EM for spherical, same variance GMMs

E-step

Compute "expected" classes of all datapoints for each class

$$\mathrm{P}\left(\mathrm{y}=\mathrm{i}\middle|\mathrm{x}_{\mathrm{j}},\mu_1\ldots\mu_{\mathrm{k}}\right)\propto \exp\left(-\frac{1}{2\sigma^2}\left\|\mathrm{x}_{\mathrm{j}}-\mu_{\mathrm{i}}\right\|^2\right)\mathrm{P}(\mathrm{y}=\mathrm{i})$$

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)

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

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

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

Just evaluate a Gaussian at x_j

M-step

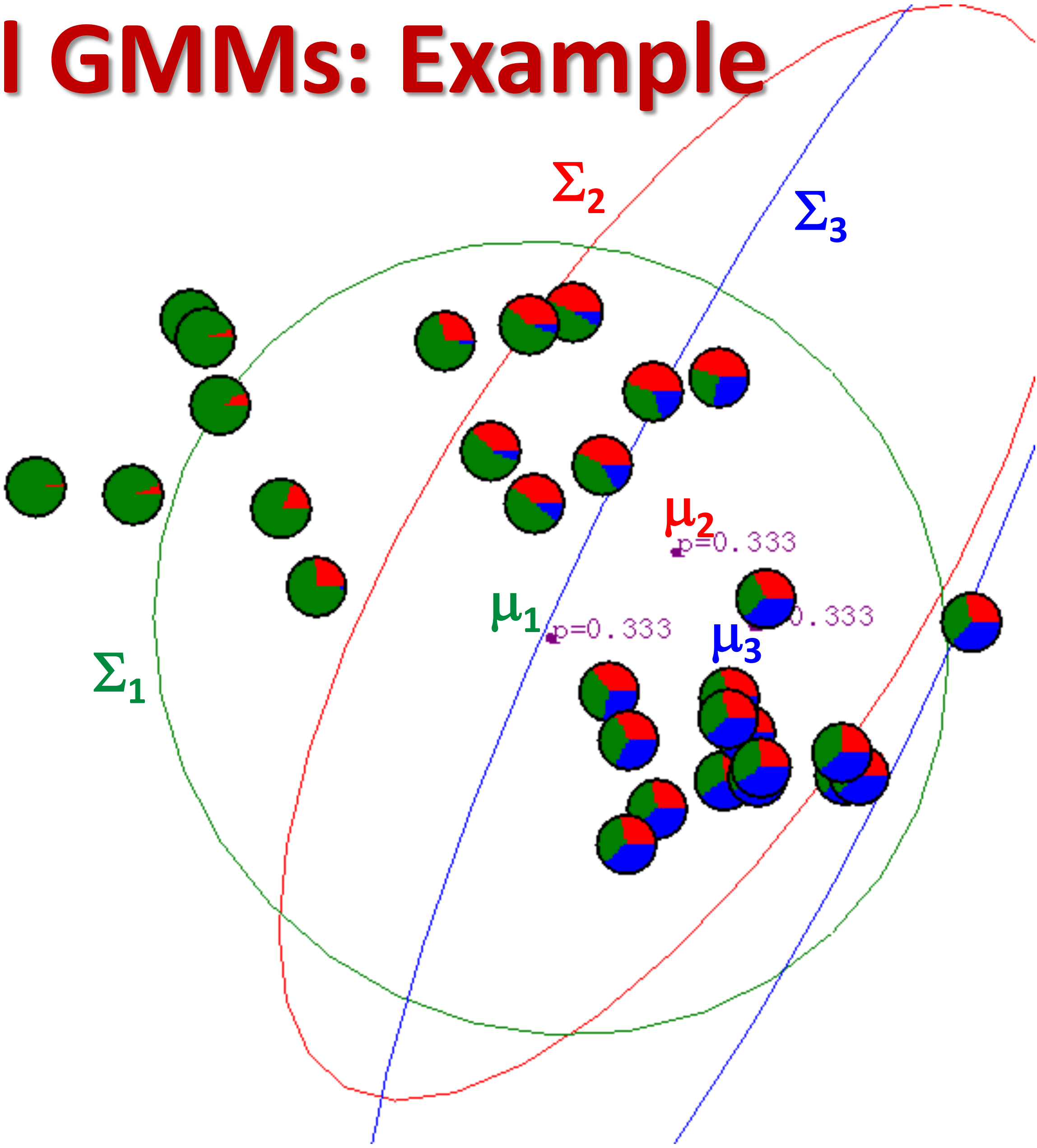
Compute MLEs given our data's class membership distributions (weights)

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

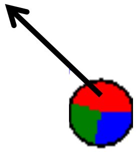
$$p_i^{(t+1)} = \frac{\sum_j P\left(y=i \middle| x_j, \lambda_t\right)}{m}$$

m = #data points

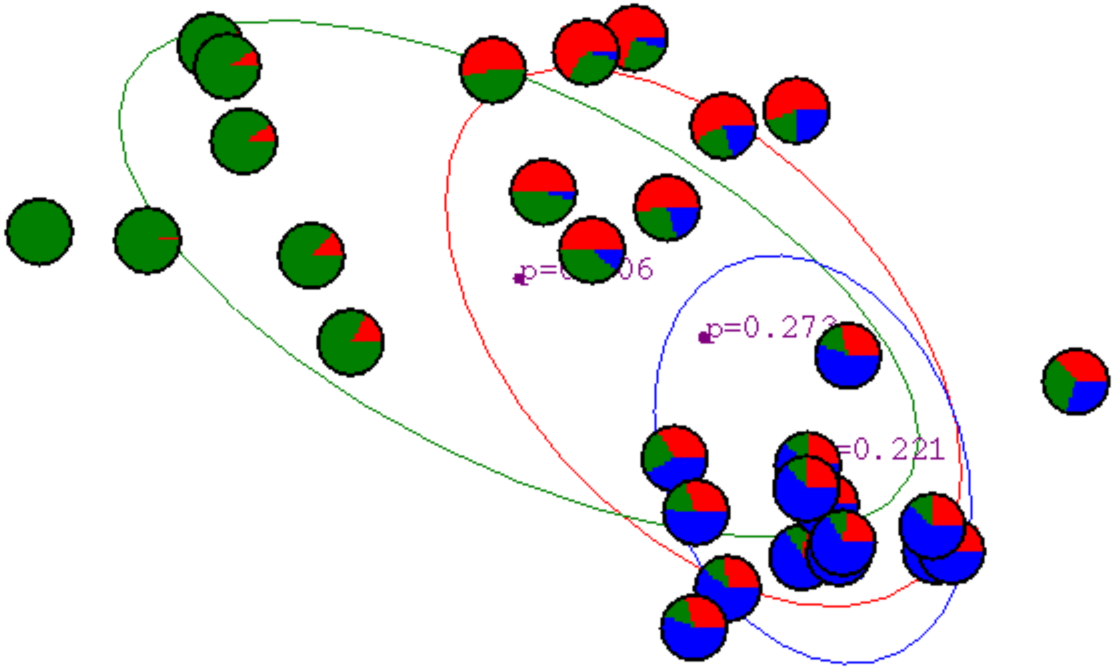
EM for general GMMs: Example



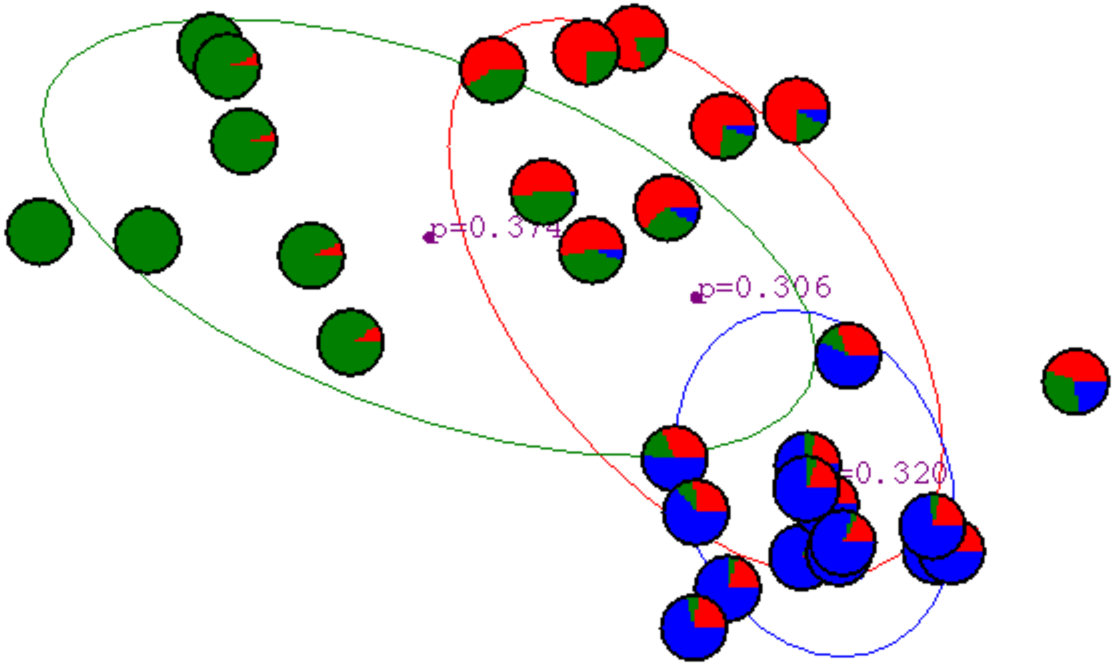
$P(y =$ ● $| x_j,\mu_1,\mu_2,\mu_3,\Sigma_1,\Sigma_2,\Sigma_3,p_1,p_2,p_3)$



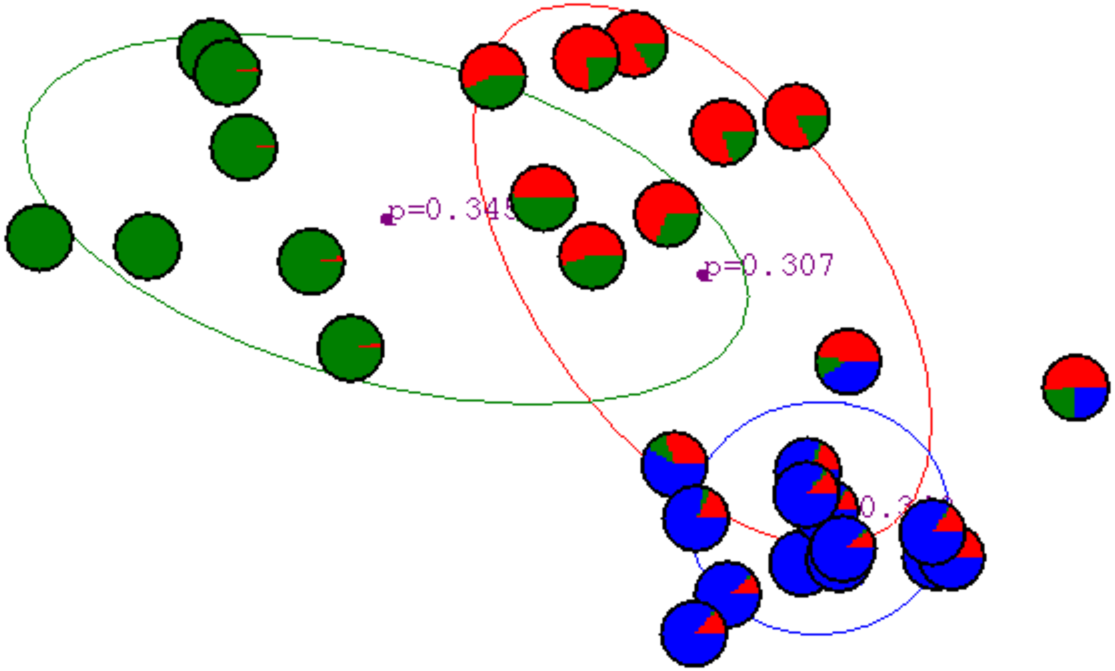
After 1st iteration



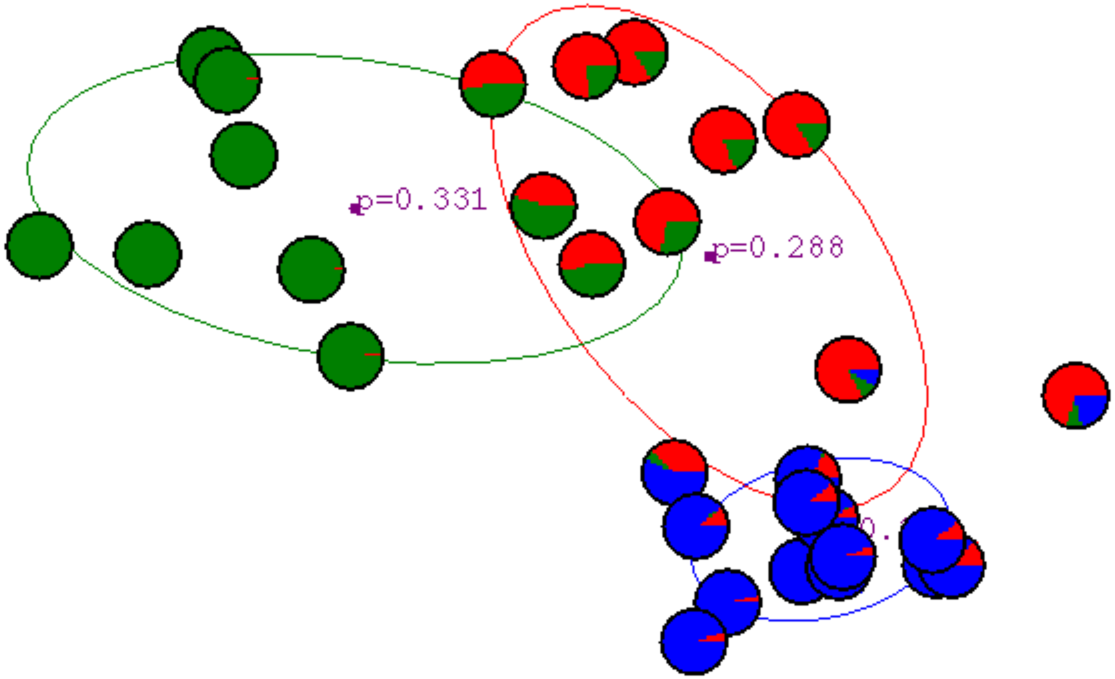
After 2nd iteration



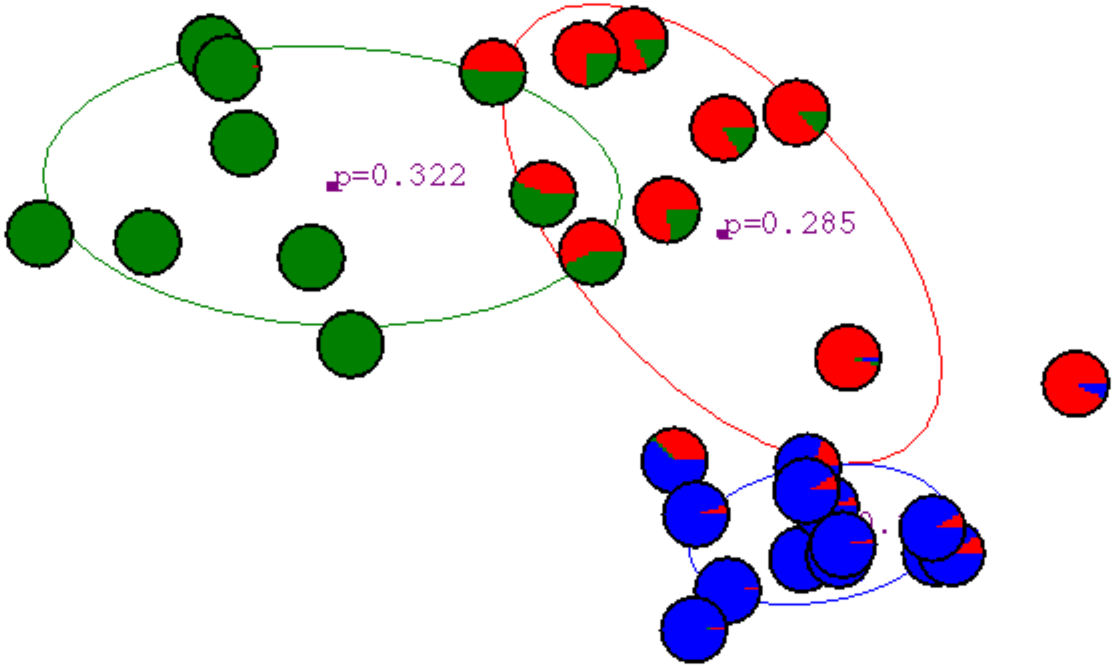
After 3rd iteration



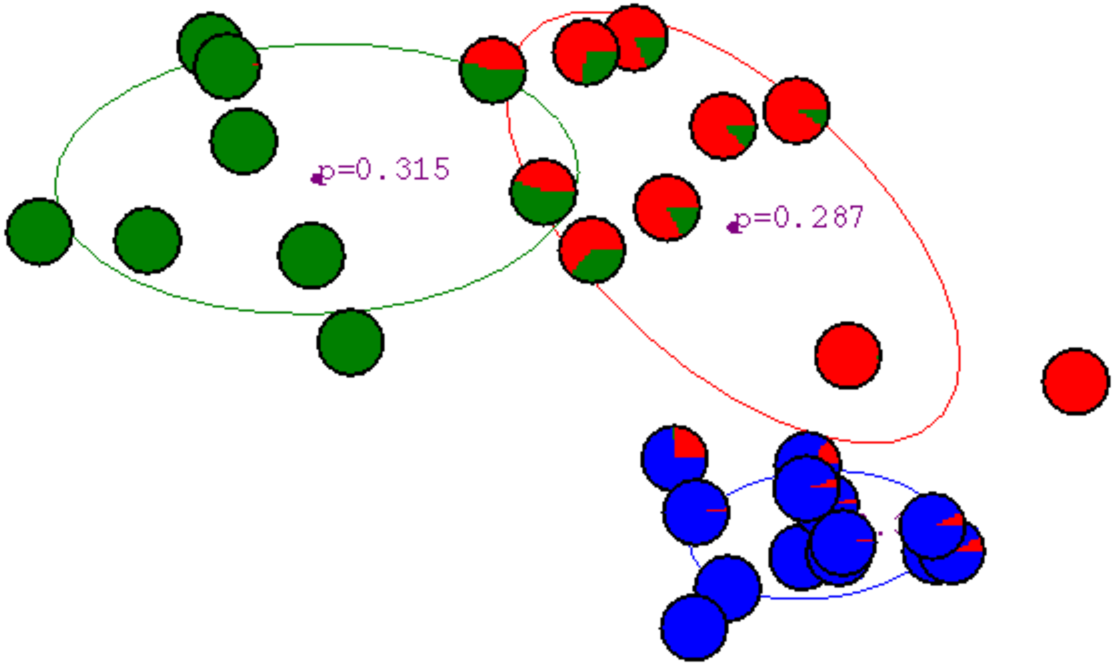
After 4th iteration



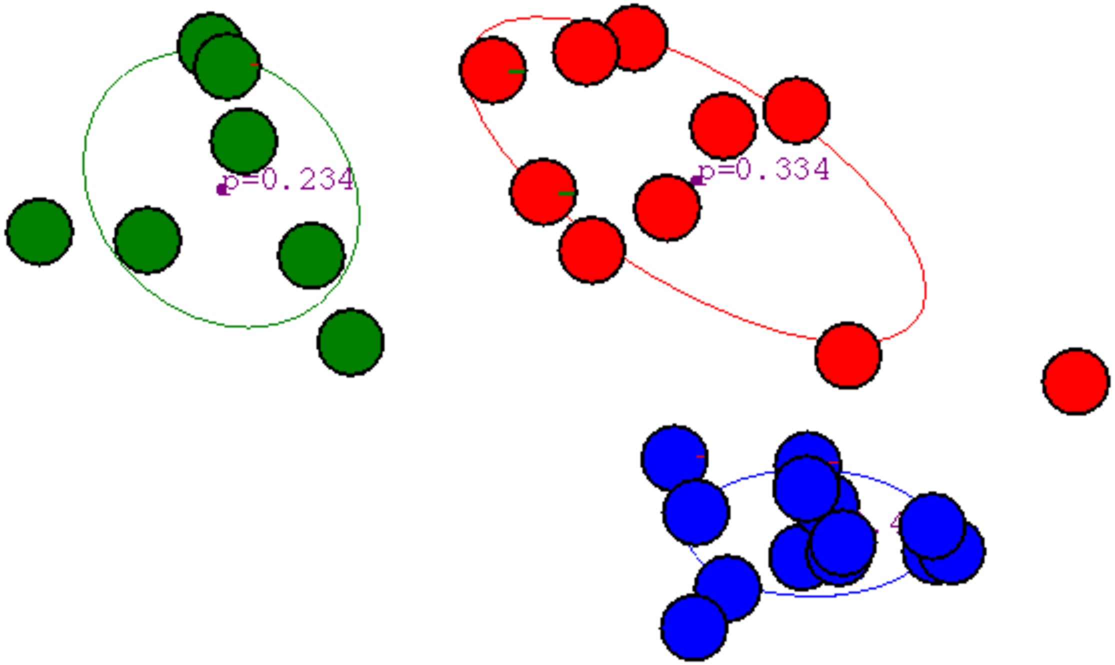
After 5th iteration



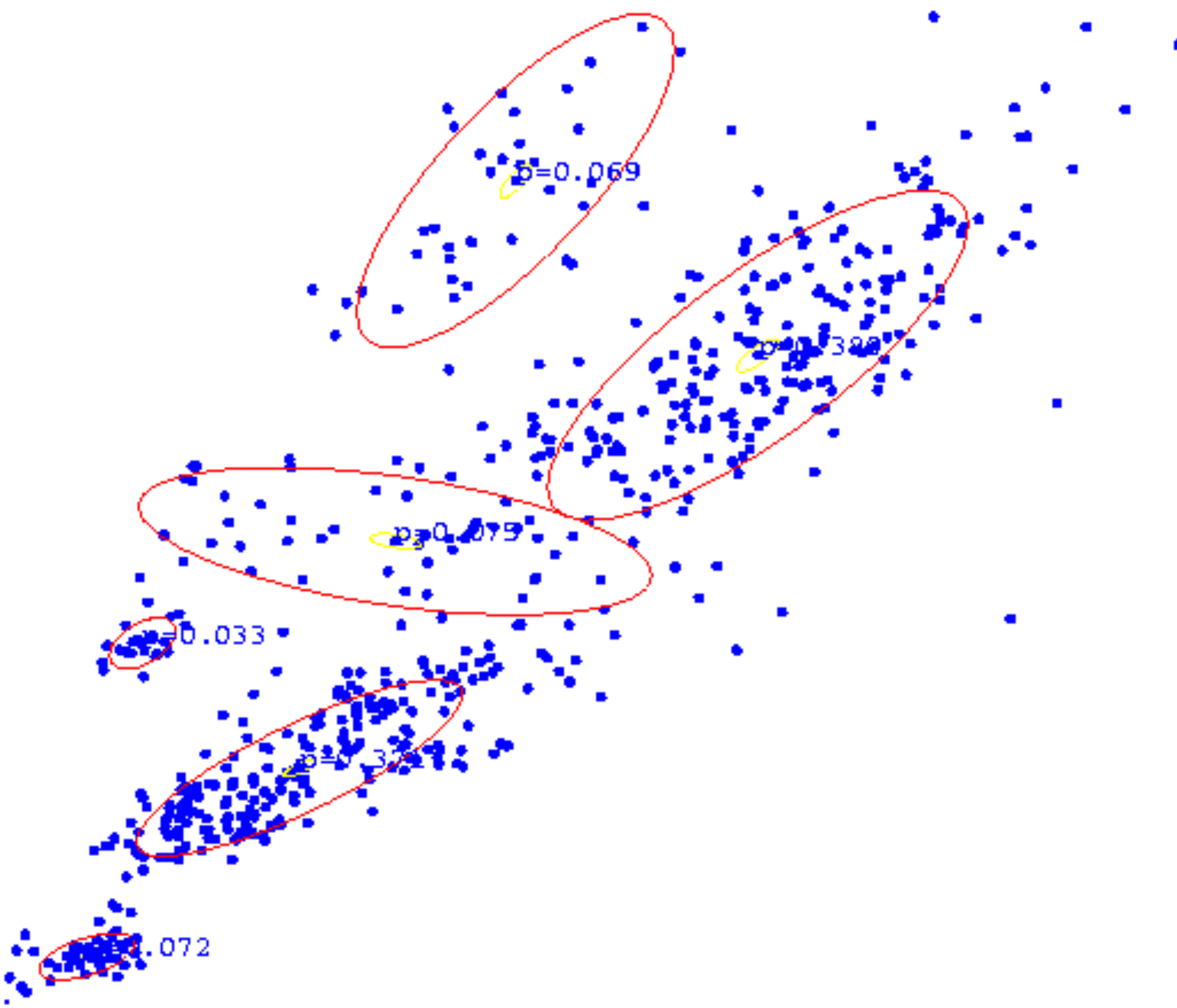
After 6th iteration



After 20th iteration



GMM clustering of assay data



General GMM

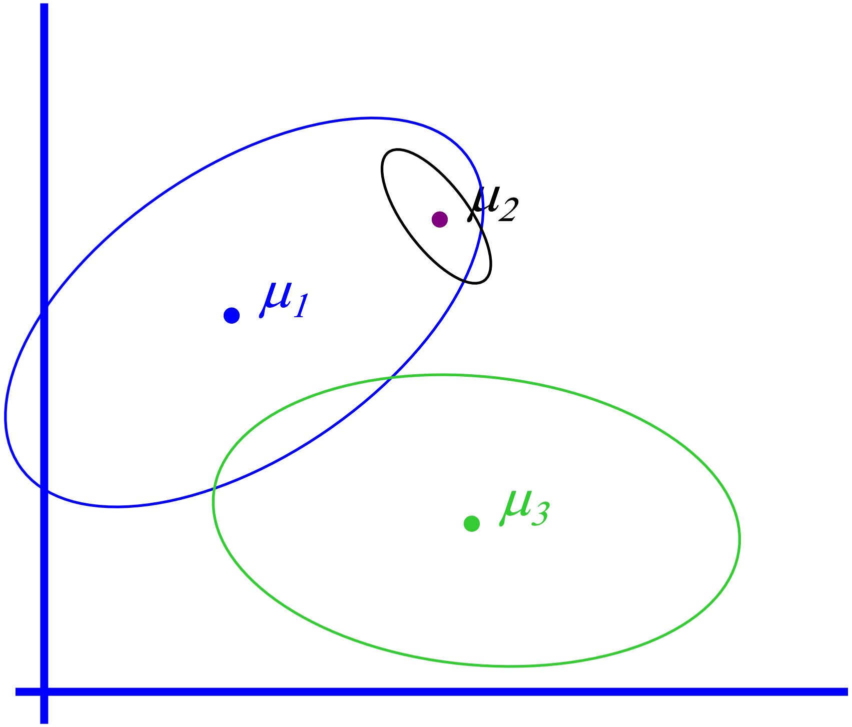
GMM – Gaussian Mixture Model (Multi-modal distribution)

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

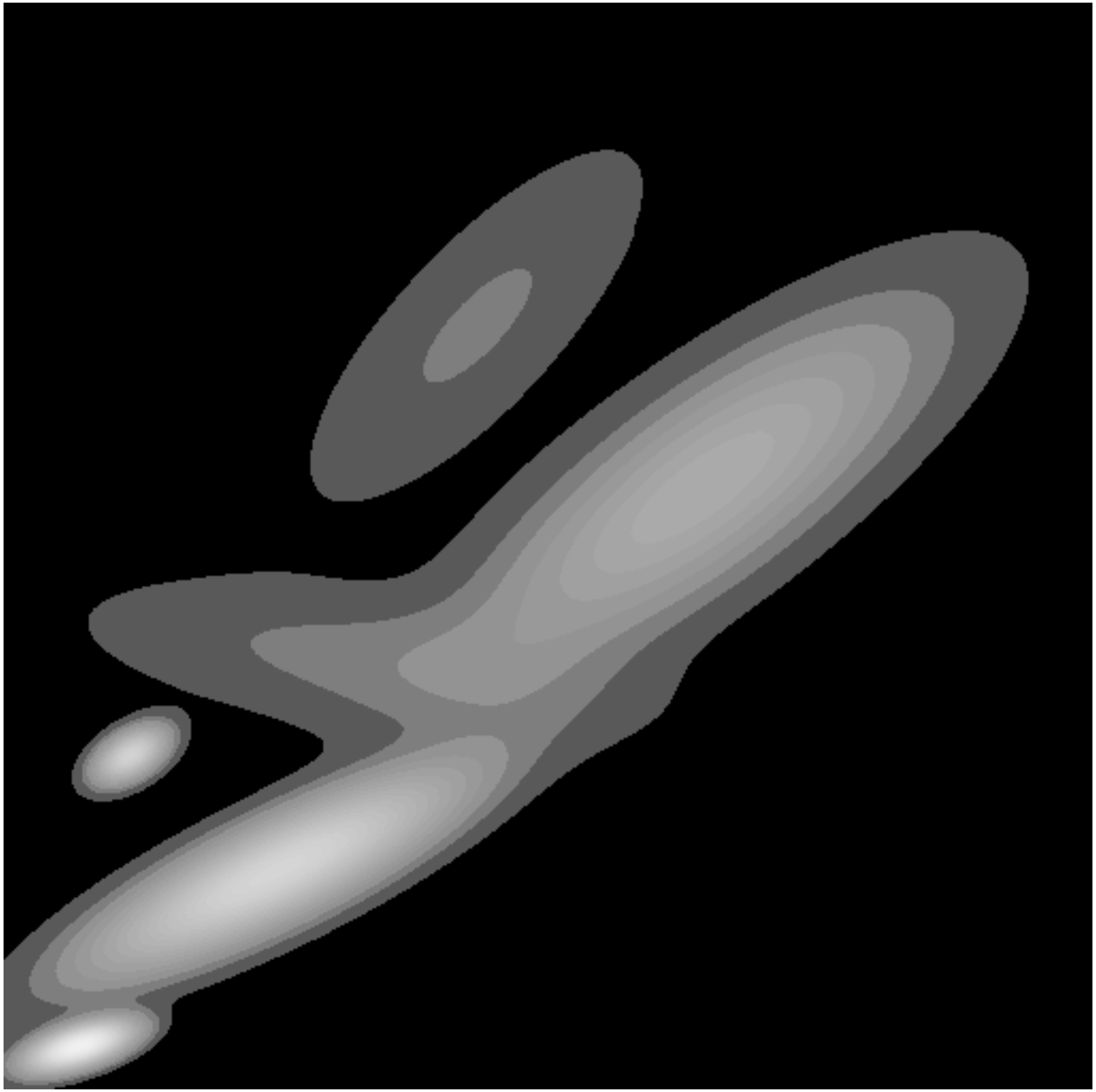
↓ **Mixture component**

↓ **Mixture proportion**

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

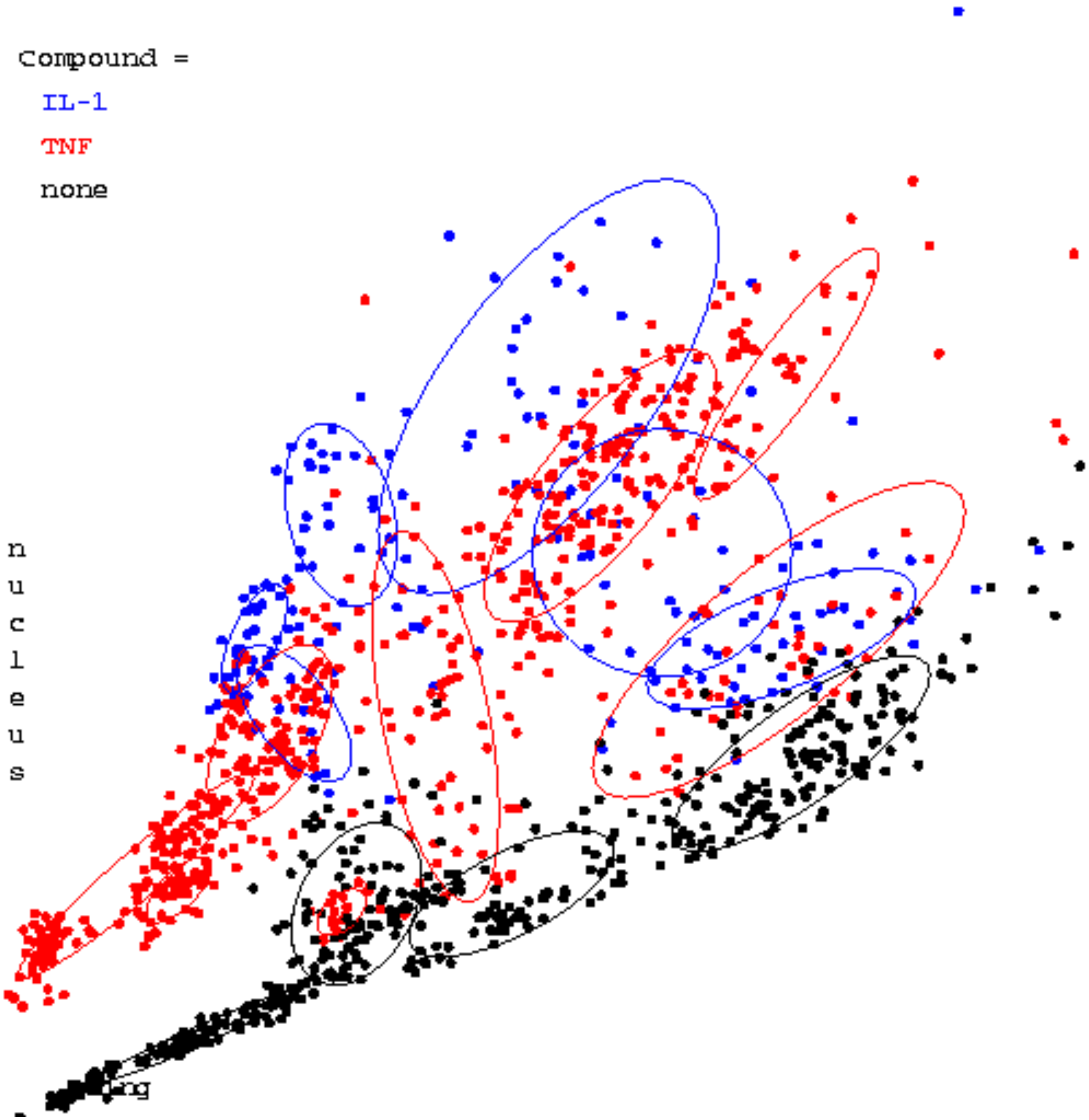


Resulting Density Estimator

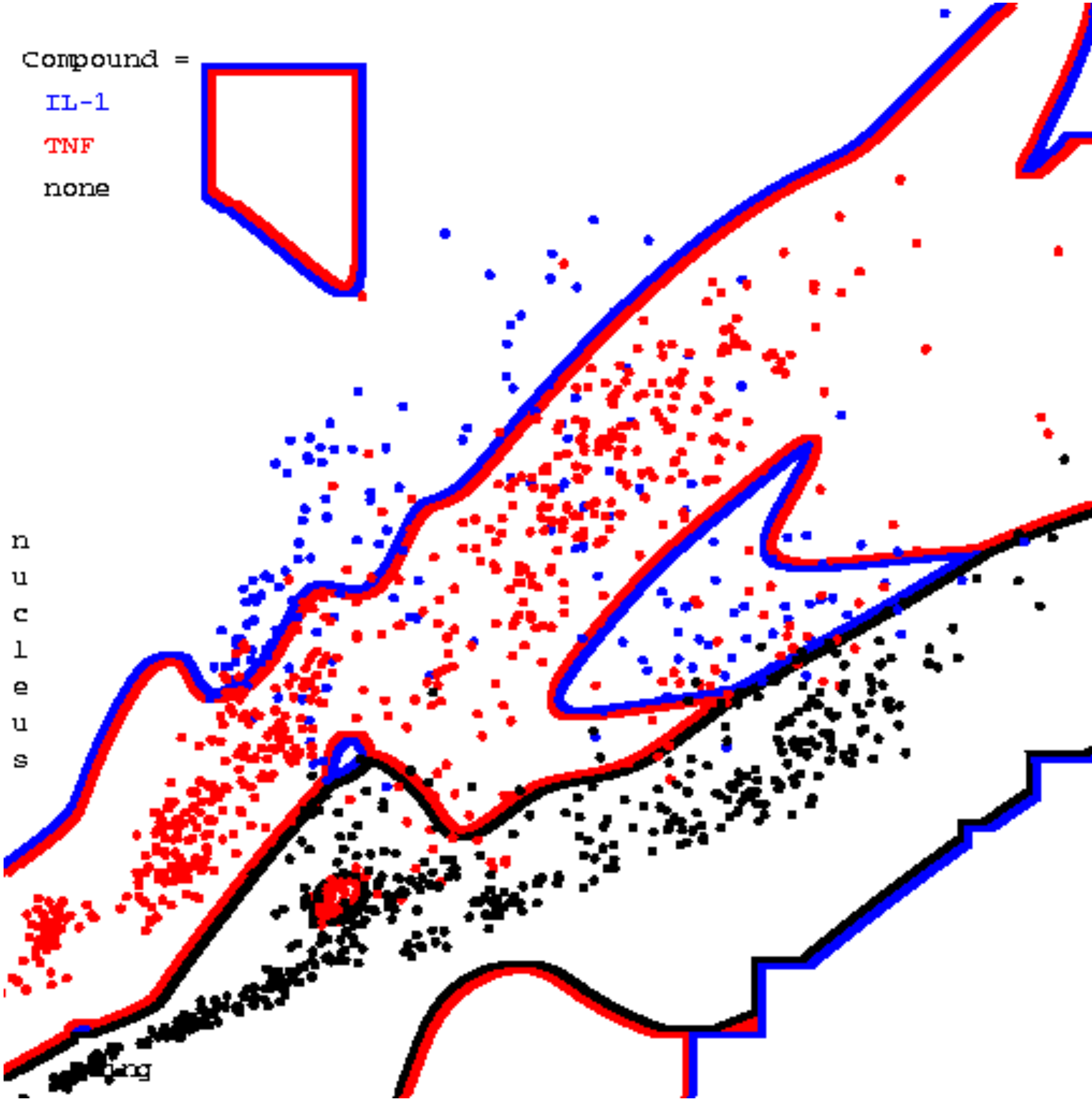


Three classes of assay

(each learned with it’s own mixture model)



Resulting Bayes Classifier



What you need to know…

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

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

    EM algorithm