Soft Clustering

- So far we have assumed hard clustering:
  - Data point is deterministically assigned to one and only one cluster
- In reality clusters may overlap
- Soft-clustering:
  - Data points are assigned to clusters with probabilities
- To obtain probabilities, we must have some probabilistic models
  - Sometimes referred to as model-based clustering
  - One of the most commonly used model is Gaussian

Side track: Gaussian Bayes Classifier

- We have k classes in our data
- Each class contains data generated from a particular Gaussian distribution
- Data is generated by
  - first randomly select one of the classes according to class prior
  - draw random samples from the Gaussian distribution of that particular class

$$P(\mathbf{x}, y) = P(\mathbf{x} \mid y)P(y)$$

$$P(\mathbf{x} \mid y = i) = \frac{1}{(2\pi)^{d/2}\left|\Sigma_i\right|^{1/2}} e^{-\frac{1}{2}(\mathbf{x}-\boldsymbol{\mu}_i)^T \Sigma_i^{-1}(\mathbf{x}-\boldsymbol{\mu}_i)}$$

Supervised vs Unsupervised Learning

- Now assume we know our data is generated in this way
- And we know the labels (*y*'s), we can estimate the mean and covariance matrix of each class using MLE
- What if the labels are hidden?
- How can we learn the correct model from the incomplete data?

Gaussian Mixture Model (GMM)

$$P(\mathbf{x}) = \sum_{i=1}^{k} P(\mathbf{x}, y = i)$$
$$= \sum_{i=1}^{k} P(\mathbf{x} \mid y = i) P(y = i)$$
$$= \sum_{i=1}^{k} \alpha_i P(\mathbf{x} \mid \theta_i)$$

$\theta_i = \{\mu_i, \Sigma_i\}$

$\underline{\alpha_i}$ = *P(y=i)*: the class prior
Called the mixing parameter

Goal of unsupervised learning:

- Given a set of x's, estimate $\{\alpha_1, \cdots, \alpha_k, \theta_1, \cdots, \theta_k\}$
- Once the model is identified, we can compute p(y=i|x) for i=1, $\cdots$, k, which gives the soft-clustering we want

Maximum Marginal Likelihood

$$\arg\max_{\theta} \prod_{j} P(\mathbf{x}^j) = \arg\max_{\theta} \prod_{j} \sum_{i=1}^{k} P(\mathbf{x}^j, y^j = i)$$

$$= \arg\max_{\theta} \sum_{j=1}^{n} \underbrace{\log \sum_{i=1}^{k} P(\mathbf{x}^j, y^j = i)}$$

log sum is nasty to optimize !

Expectation Maximization (EM)

- A highly used approach for dealing with in-complete data
- It is an iterative algorithm that starts with some initial guesses of the model parameters
- Iteratively performs two main steps:
  - **Expectation (E-step)**: given current model parameters, compute the expectation for the hidden (missing) data
  - **Maximization (M-step)**: re-estimate the parameters assuming that the expected values computed in the E-step are the true values

E-Step for GMM

- If we know $\theta = \{\alpha_1, \cdots, \alpha_k, \theta_1, \cdots, \theta_k\}$
- We can easily compute the probability that point x^j belong to class y=i

$$
\begin{aligned}
p(y=i|\mathbf{x}^j;\theta^{(t)}) &= \frac{p(\mathbf{x}^j|\theta_i^{(t)}) \cdot \alpha_i^{(t)}}{p(\mathbf{x}^j)} \\
&\propto \alpha_i^{(t)} \cdot \frac{1}{(2\pi)^{d/2}|\Sigma_i^{(t)}|^{1/2}} \cdot e^{-\frac{1}{2}(x-\mu_i^{(t)})^T \Sigma_i^{(t)^{-1}} (x-\mu_i^{(t)})}
\end{aligned}
$$

if	$p(y=1\|x, \theta)$	$\propto 0.3$
	$p(y=2\|x, \theta)$	$\propto 0.2$
then	$p(y=1\|x, \theta)$	$= 0.6$
	$p(y=2\|x, \theta)$	$= 0.4$

Same as classification !

M-Step

- If we know the probability of point x^j belongs to y=i, we can then re-estimate θ

$$\alpha_i = \frac{\sum_j P(y^j{=}i|x^j)}{n}$$

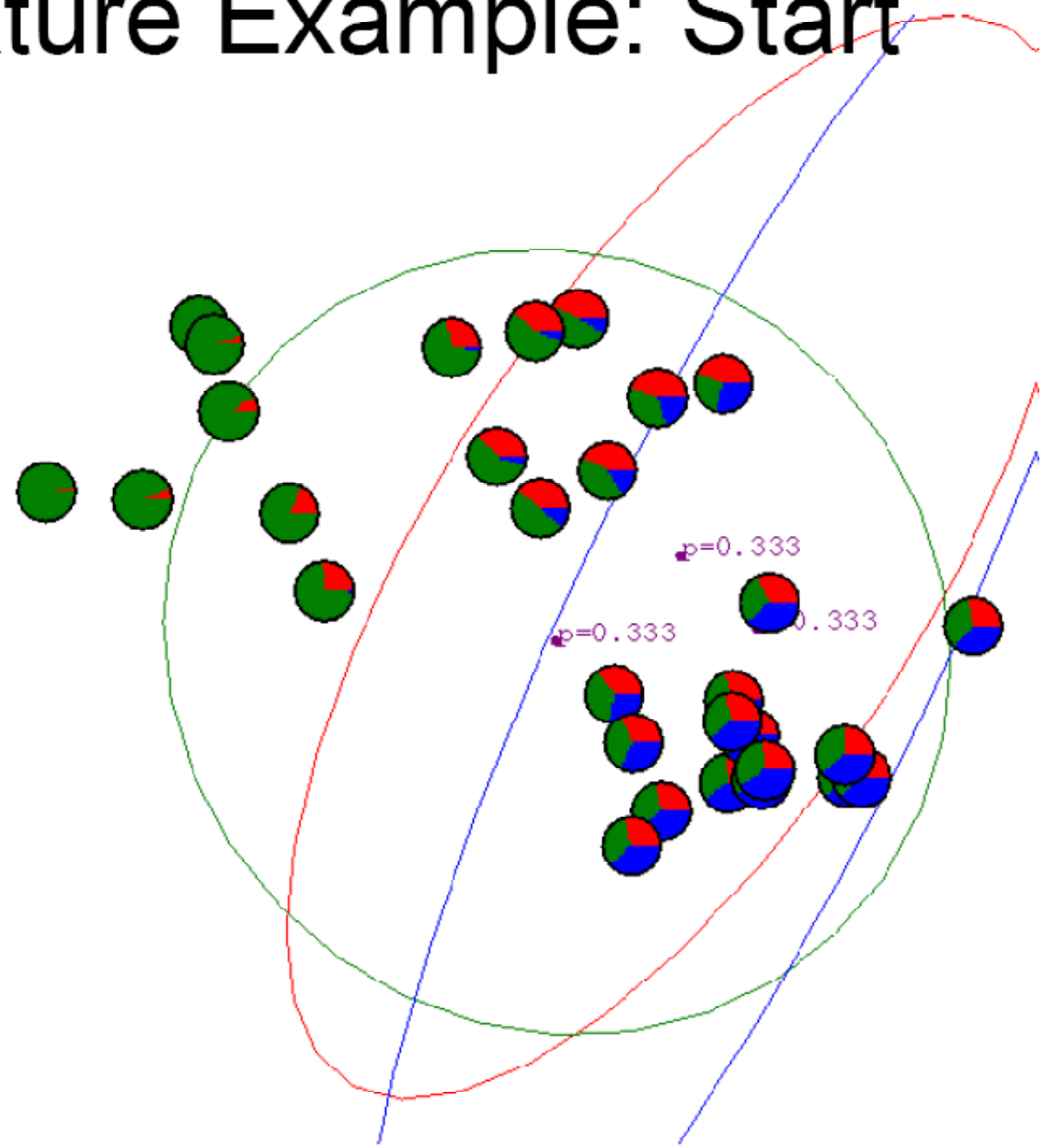
$$\mu_i^{(t+1)} = \frac{\sum_j P(y{=}i|\mathrm{x}^j)\cdot \mathrm{x}^j}{\sum_j P(y^j{=}i|x^j)}$$

$$\Sigma_i^{(t+1)} = \frac{\sum_j P(y^j = i|x^j)\cdot(\mathrm{x}^j - \mu_i^{(t+1)})\cdot(\mathrm{x}^j - \mu_i^{(t+1)})^T}{\sum_j P(y^j = i|x^j)}$$

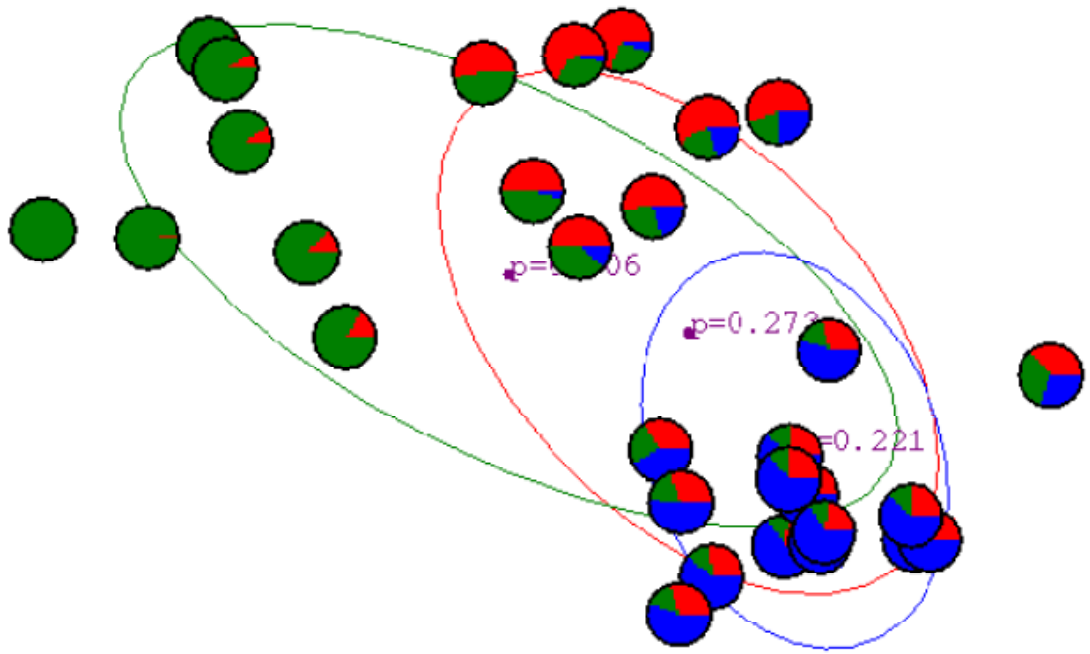
$$\boxed{\begin{array}{ll} \alpha_i = \frac{n_i}{\sum_i n_i} & \hat{\mu}_i = \frac{1}{n_i}\sum_{y^j=i} \mathrm{x}^j \\ \hat{\Sigma}_i = \frac{1}{n_i}\sum_{y^j=i}(\mathrm{x}^j - \hat{\mu}_i)\cdot(\mathrm{x}^j - \hat{\mu}_i)^T & \end{array}}$$

Imagine k copies of each x^j, each with weight $P(y^j{=}i|x^j)$:

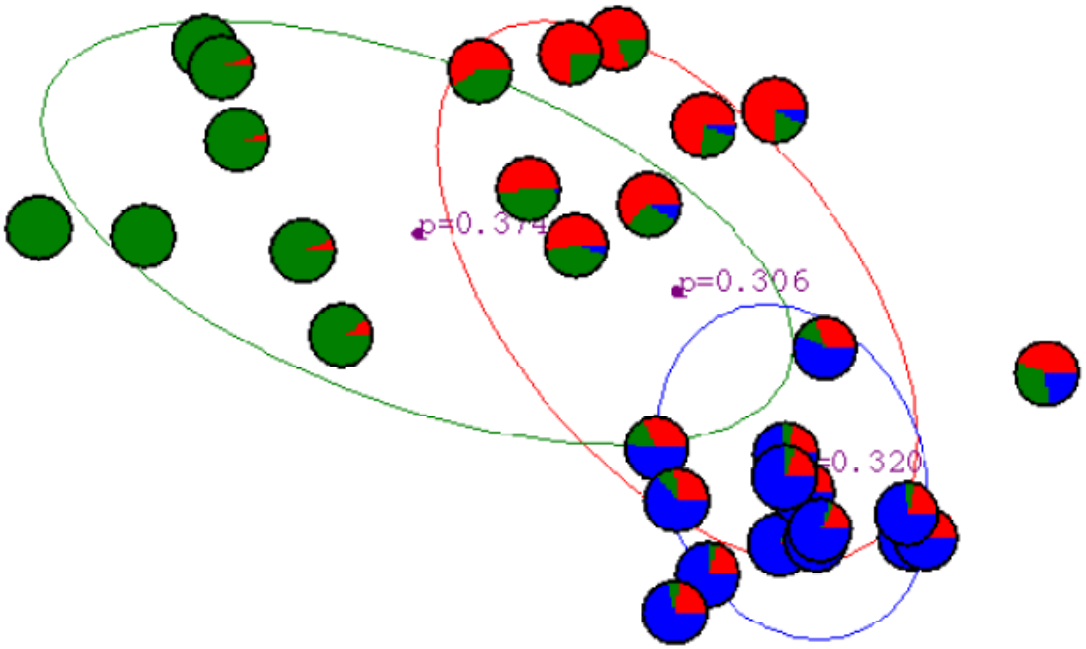
Gaussian Mixture Example: Start



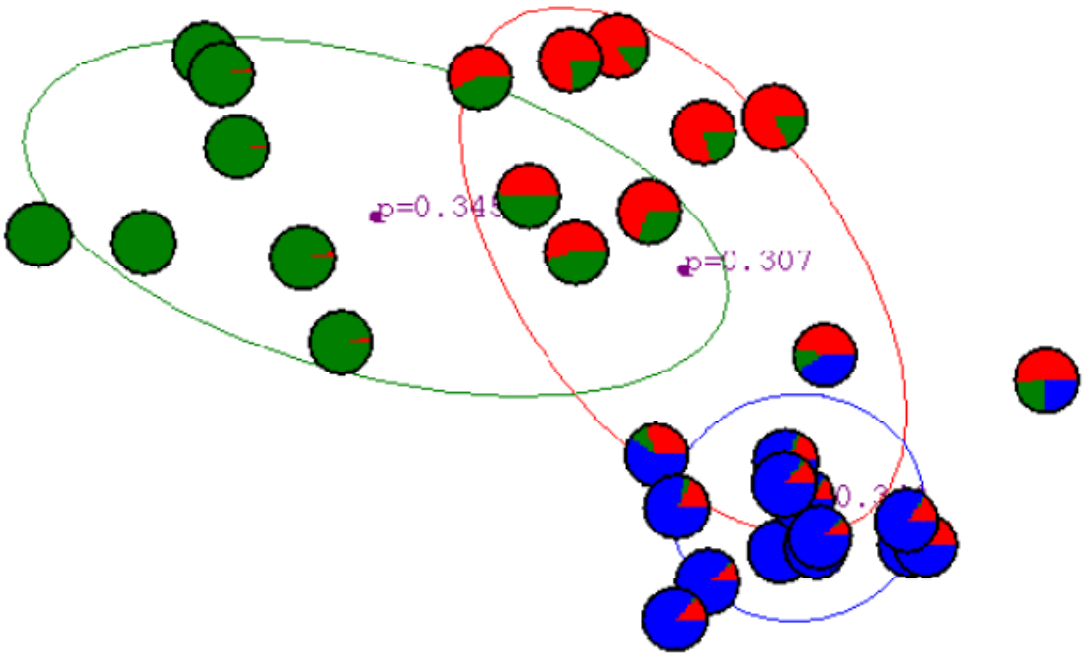
After first iteration



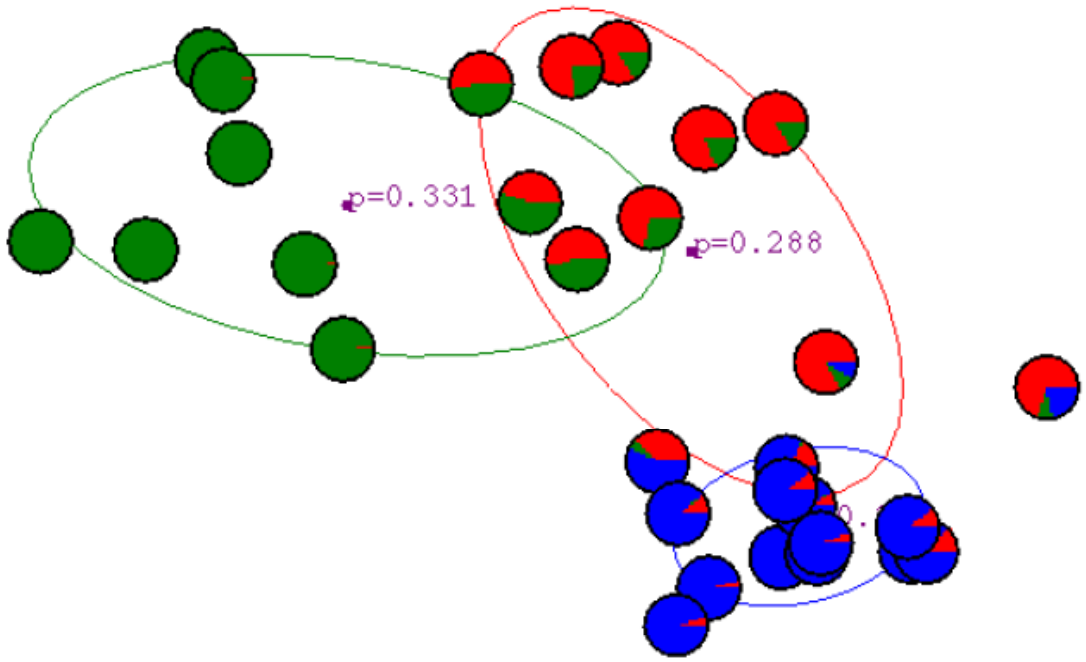
After 2nd iteration



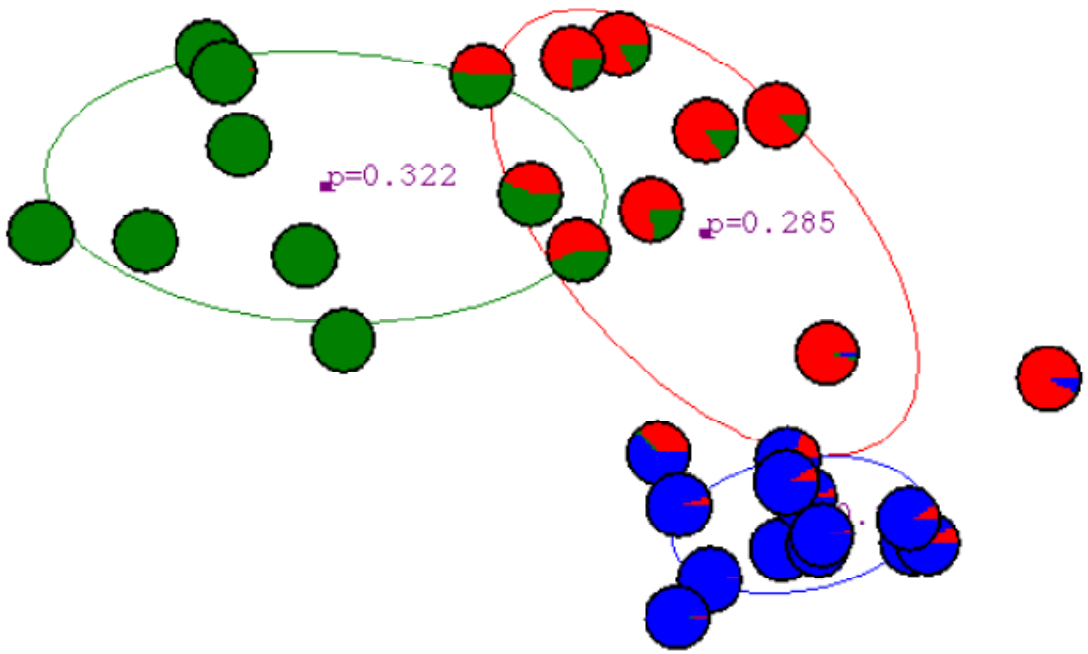
After 3rd iteration



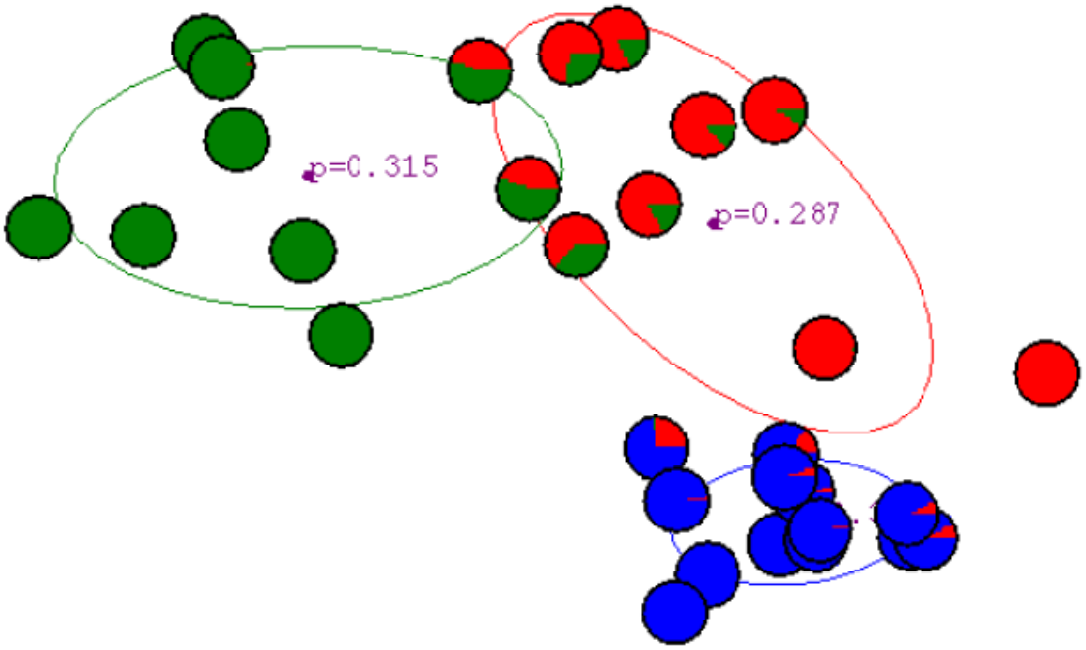
After 4th iteration



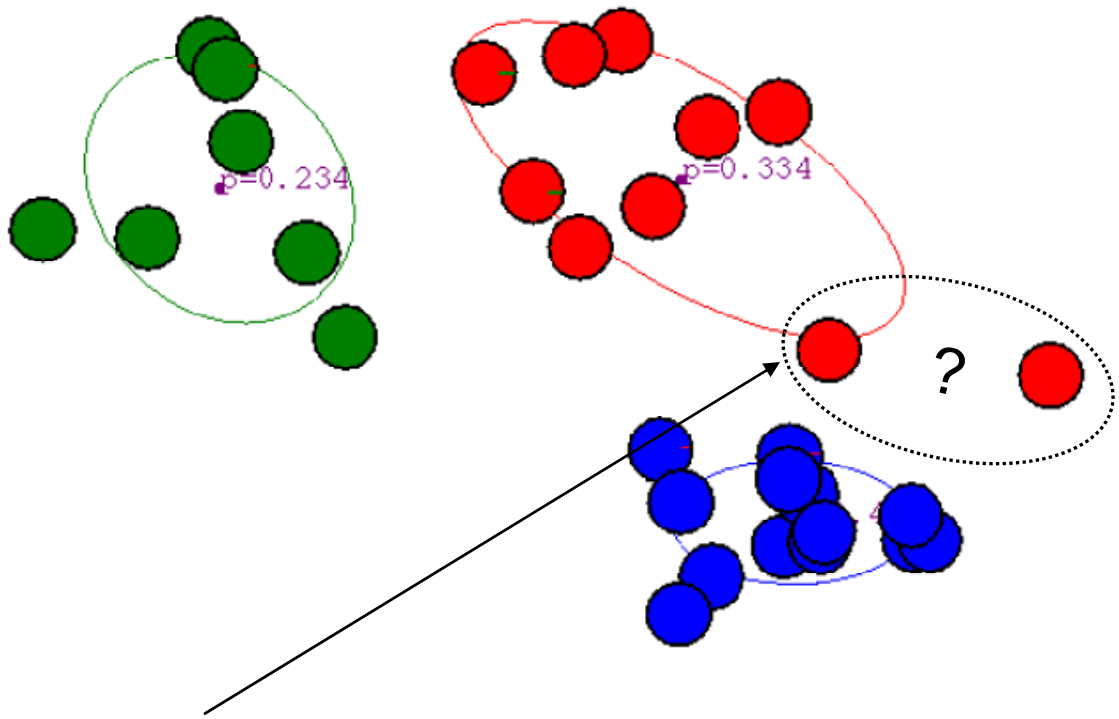
After 5th iteration



After 6th iteration



After 20th iteration



Q: Why are these two points red?

K-Means vs GMM

- we get K-Means if we make following restrictions:
  - Contain only spherical Gaussian (because all dimensions are equally contributing to the Euclidean distance function)
  - Same covariance matrix for all Gaussians
  - Use hard assignment

Behavior of EM

- It is guaranteed to converge
  - Convergence proof is based on the fact that $P(x|\theta)$ must increase or remain same between iterations (not obvious)
  - But $P(x|\theta)$ can never exceed 1 (obvious)
  - So it should always converge
  - In practice it may converge slowly, one can stop early if the change in log-likelihood is smaller than a threshold
- It converges to a local optimum
  - Multiple restart is recommended
  - Output the one with the best log-likelihood