Lecture 18

Nov 07 2008

Review

- Clustering
  - Grouping similar objects into clusters
- Hierarchical clustering
  - Agglomerative approach (HAC): iteratively merge similar clusters
  - Different linkage algorithms for computing distances among clusters
- Non hierarchical clustering
  - K-means: start with a set of initial seeds (ceners), iteratively go through reassignment and recentering steps until convergence

More about Kmeans

- It always converges (fast)
- It converges to local optimum
  - Different initial seeds lead to different local optimum, to address this:
    - Many random restart and pick the best wrt MSE
    - Separate initial seeds far apart
- Other problems:
  - It is best suited for cases where clusters are all spherical and similar in size
  - It does not allow an object to partially belong to multiple clusters

Soft vs hard Clustering

- Hard clustering:
  - Data point is deterministically assigned to one and only one cluster
  - But in reality clusters may overlap
- Soft-clustering:
  - Data points are assigned to clusters with certain probabilities

How can we extend Kmeans to make soft clustering

- Given a set of clusters centers $\mu_1, \mu_2, \ldots, \mu_k$, instead of directly assign all data points to their closest clusters, we can assign them **partially based on the distances**

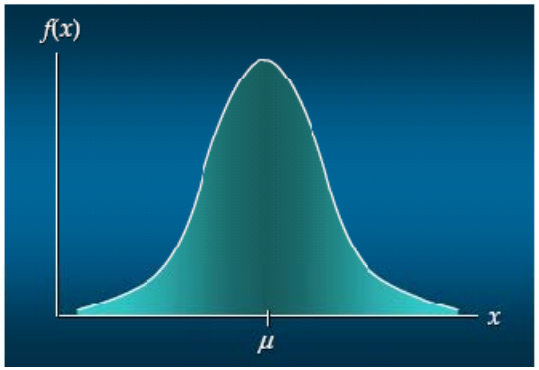
- If each point only partially belongs to a particular cluster, when computing the centroid, should we still use it as if it was fully there?

Gaussian for representing a cluster

- What exactly is a cluster?
  - Intuitively it is a tightly packed ball-shape like thing
- We can use a Gaussian (normal) distribution to describe it
- Let's first review what is a Gaussian distribution

Side track: Gaussian Distribtuion

- Univariate Gaussian distribution:

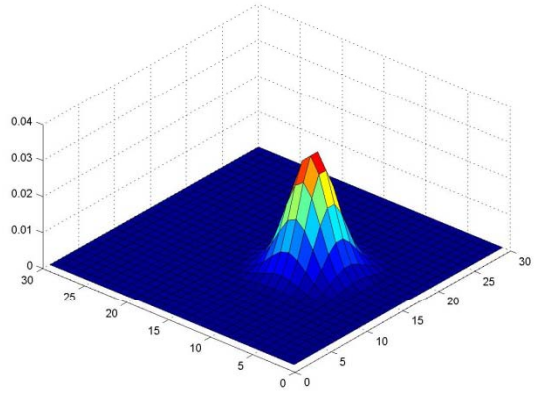


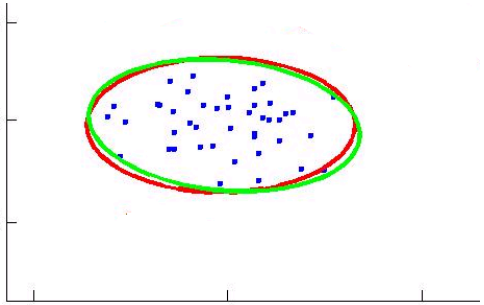
$N(\mu, \sigma^2)$

μ – mean, center of the mass

σ^2 – standard deviation, spread of the mass

- Multivariate Gaussian distribution:





$N(\mu, \Sigma)$

μ – (μ_1, μ_2)

Σ – Covariance matrix

$$\begin{bmatrix} \sigma_1^2 & \sigma_{12} \\ \sigma_{12} & \sigma_2^2 \end{bmatrix}$$

Mixture of Gaussians

- Assume that we have k clusters in our data
- Each cluster contains data generated from a Gaussian distribution
- Overall process of generating data:
  - first randomly select one of the clusters according to a prior distribution of the clusters
  - draw a random sample from the Gaussian distribution of that particular cluster
- Similar to the generative model we have learned in Bayes Classifier, difference?
  - Here we don't know the cluster membership of each data point **(unsupervised)**

Clustering using mixture of Gaussian models

- Given a set of data points, and assume that we know there are k clusters in the data, we need to:
  - Assign the data points to the k clusters (soft assignment)
  - Learn the gaussian distribution parameters for each cluster: μ and Σ

A simpler problem

- If we know the parameters of each Gaussian: (μ_1,Σ_1); (μ_2,Σ_2); ..., (μ_K,Σ_K)
  - we can compute the probability of each data point belonging to each cluster

$$P(x \in C_i \mid x) = \frac{P(x \mid x \in C_i)P(C_i)}{P(x)}$$

$$\propto \alpha_i * \frac{1}{(2\pi)^{d/2}\left|\Sigma_i\right|^{1/2}} \exp[-\frac{1}{2}(x-\mu_i)^T \Sigma_i^{-1}(x-\mu_i)]$$

- The same as in making prediction in Bayes classifier

Another simpler problem

- If we know what points belong to cluster i, we can estimate the gaussian parameters easily:

$$\alpha_i = \frac{n_i}{n} \qquad \hat{\mu}_i = \frac{1}{n_i} \sum_{\mathbf{x}^j \in C_i} \mathbf{x}^j \qquad \hat{\Sigma}_i = \frac{1}{n_i} \sum_{\mathbf{x}^j \in C_i} (\mathbf{x}^j - \hat{\mu}_i)(\mathbf{x}^j - \hat{\mu}_i)^T$$

Cluster prior — Cluster mean — Cluster covariance

- What we have is slightly different –
  - For each data point x^j, we have $P(x^j \in C_i | x^j)$ for i=1,2,…, K

Modifications

Cluster prior

$$\alpha_i = \frac{n_i}{n} \Rightarrow \alpha_i = \frac{1}{n}\sum_{j=1,\cdots,n} P(\mathbf{x}^j \in C_i \mid \mathbf{x}^j)$$

Cluster mean

$$\hat{\mu}_i = \frac{1}{n_i}\sum_{\mathbf{x}^j \in C_i} \mathbf{x}^j \Rightarrow \hat{\mu}_i = \frac{\sum_{j=1,\cdots,n} \mathbf{x}^j P(\mathbf{x}^j \in C_i \mid \mathbf{x}^j)}{\sum_{j=1,\cdots,n} P(\mathbf{x}^j \in C_i \mid \mathbf{x}^j)}$$

Cluster covariance

$$\hat{\Sigma}_i = \frac{1}{n_i}\sum_{\mathbf{x}^j \in C_i} (\mathbf{x}^j - \hat{\mu}_i)(\mathbf{x}^j - \hat{\mu}_i)^T \Rightarrow$$

$$\hat{\Sigma}_i = \frac{\sum_{j=1,\cdots,n} P(\mathbf{x}^j \in C_i \mid \mathbf{x}^j)(\mathbf{x}^j - \hat{\mu}_i)(\mathbf{x}^j - \hat{\mu}_i)^T}{\sum_{j=1,\cdots,n} P(\mathbf{x}^j \in C_i \mid \mathbf{x}^j)}$$

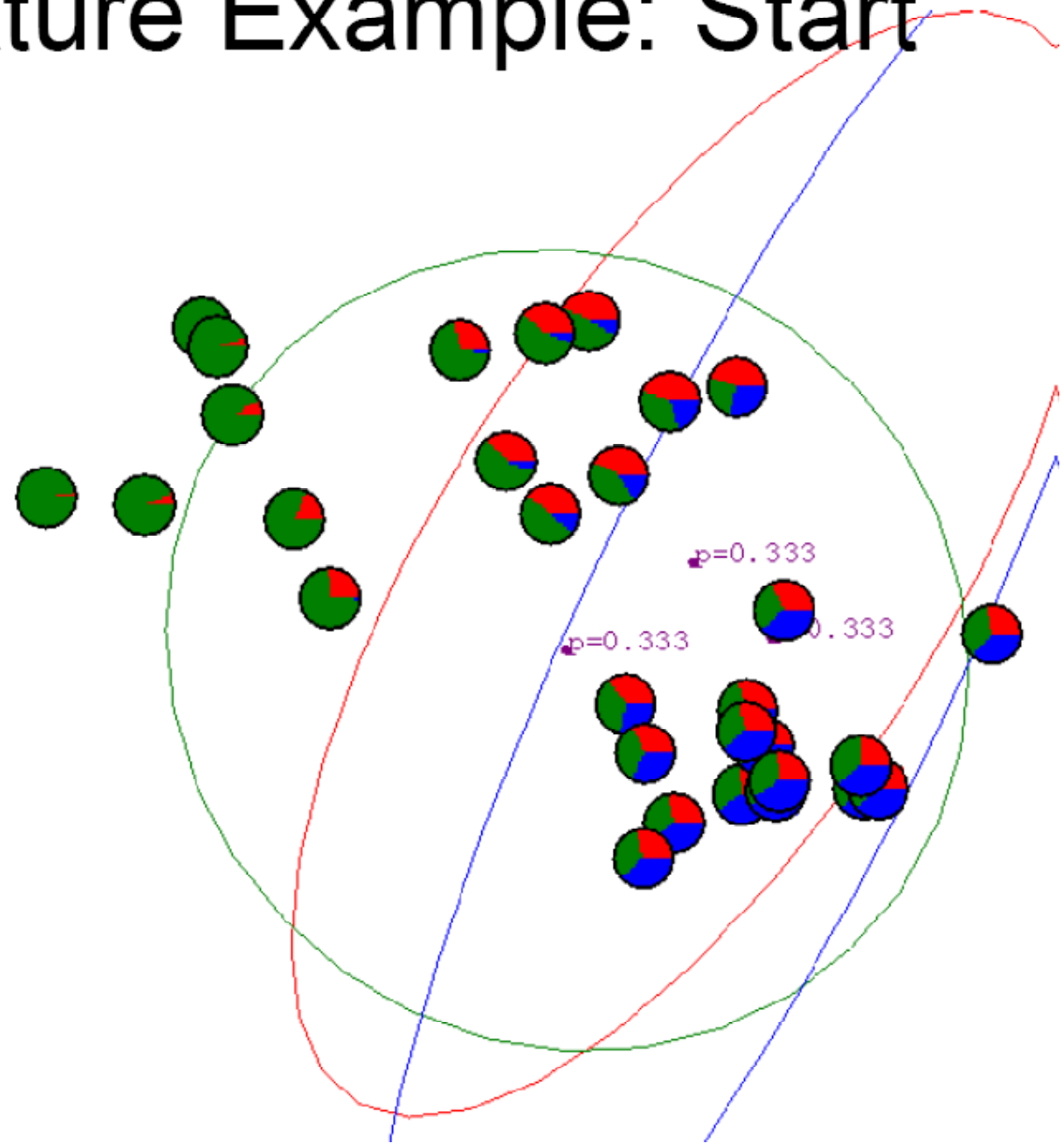
A procedure similar to Kmeans

- Randomly initialize the Gaussian parameters
- Repeat until converge
  1. Compute $P(\mathbf{x}^j \in C_i \mid \mathbf{x}^j)$ for all data points and all clusters

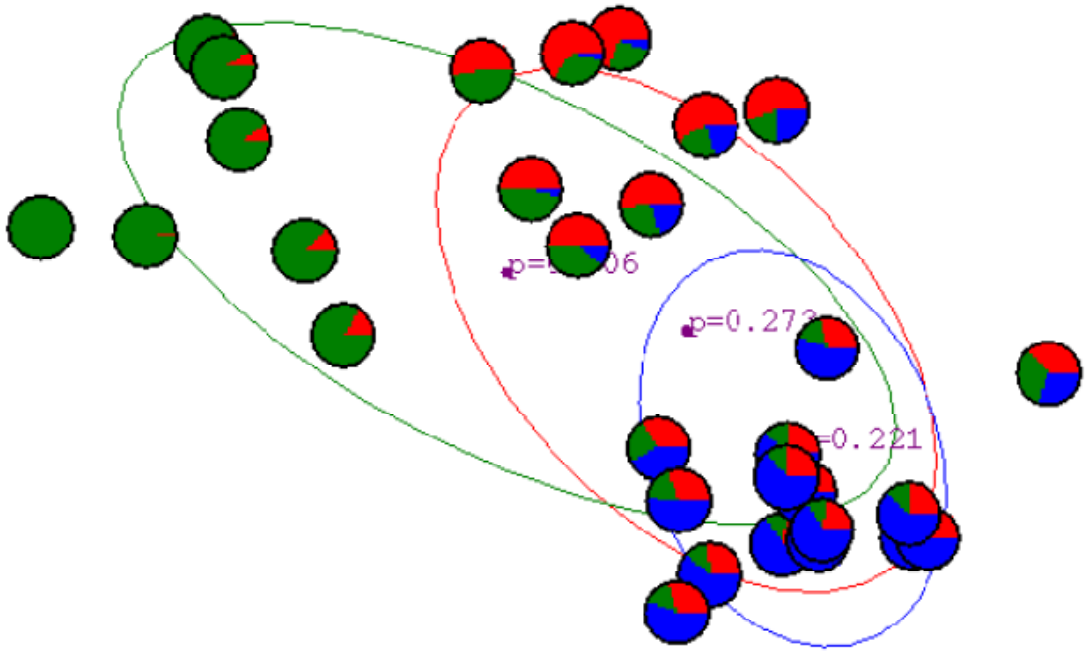
     This is called the E-step for it computes the expected values of the cluster memberships for each data point
  2. Re-compute the parameters of each Gaussian

     This is called the M-step for it performs maximum likelihood estimation of parameters

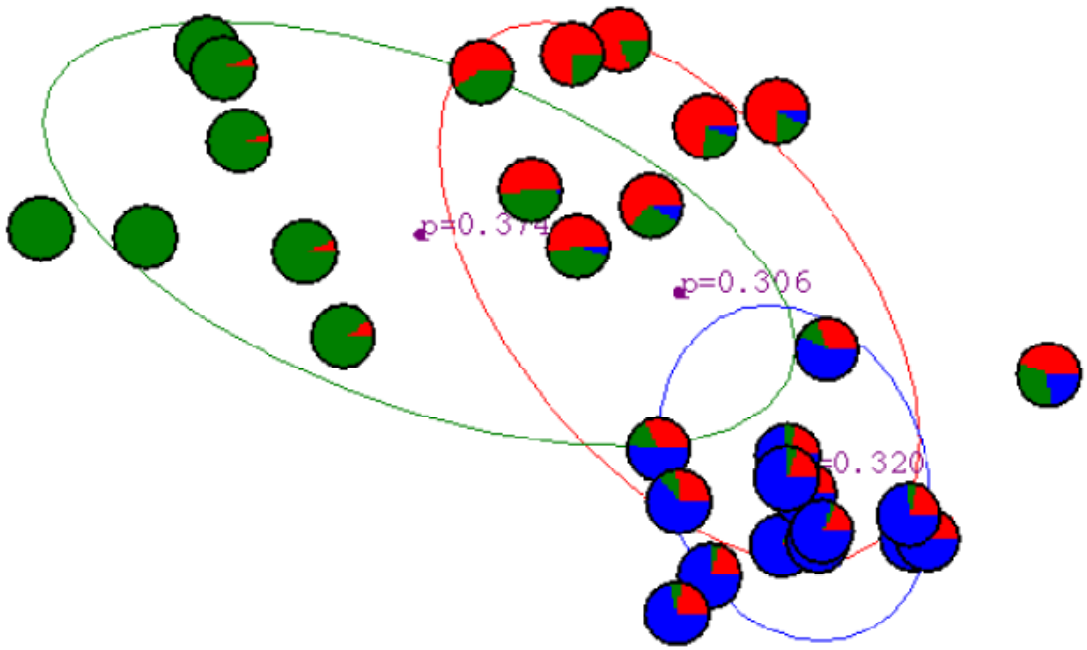
Gaussian Mixture Example: Start



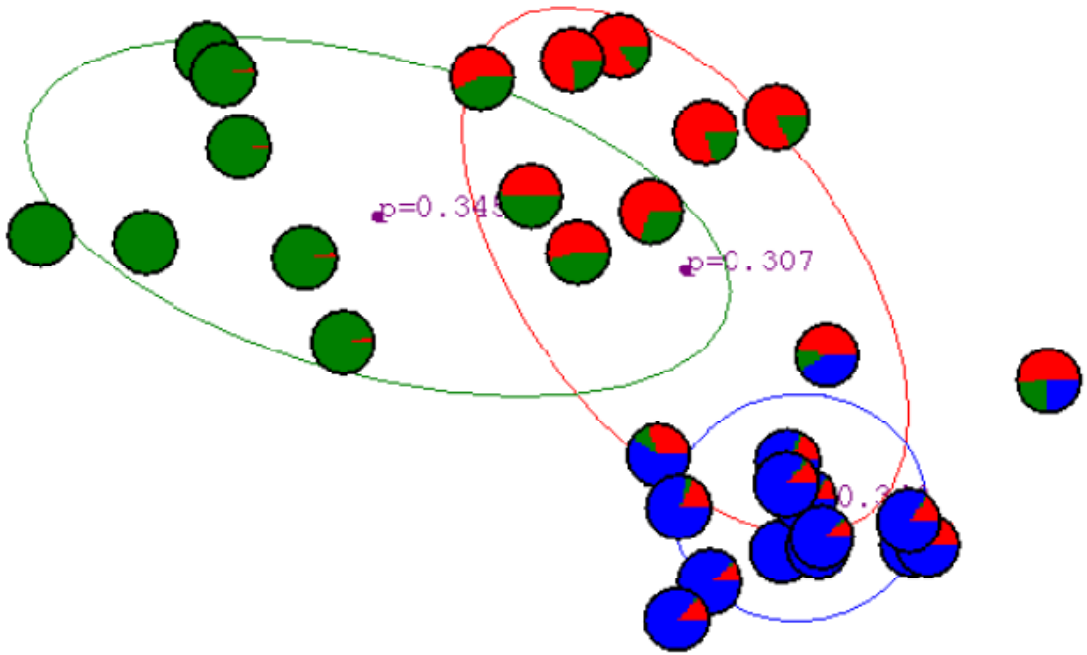
After first iteration



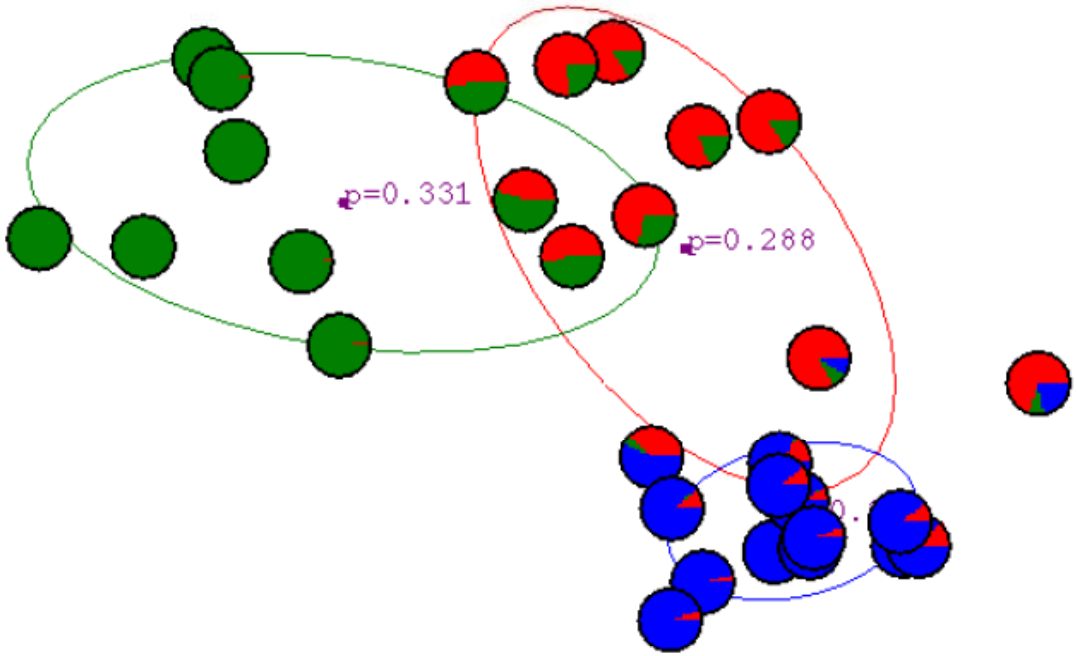
After 2nd iteration



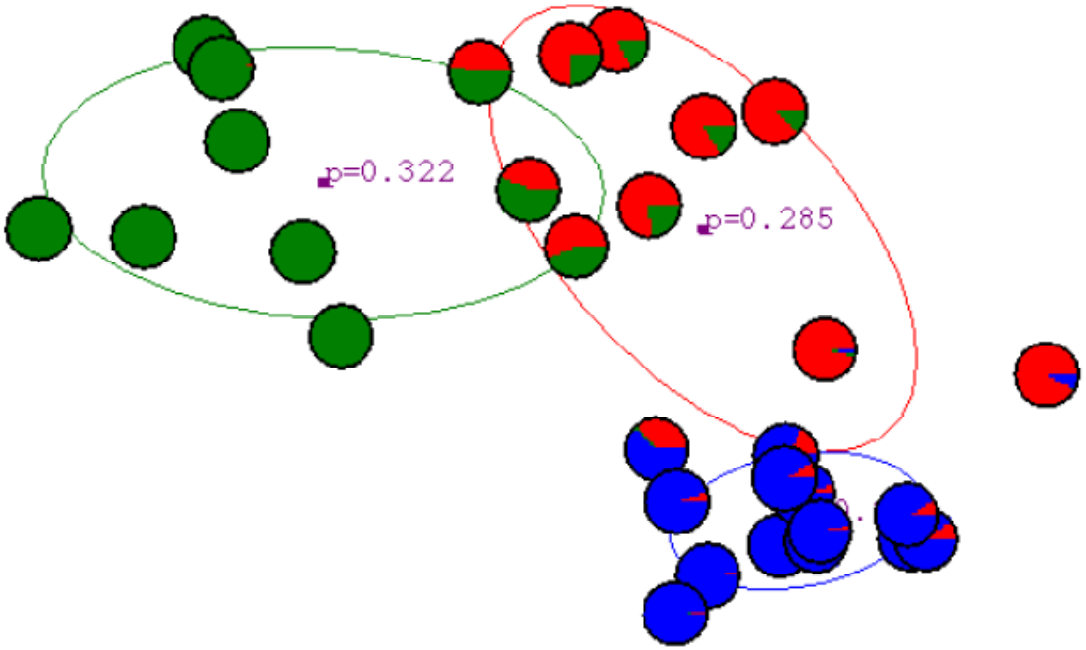
After 3rd iteration



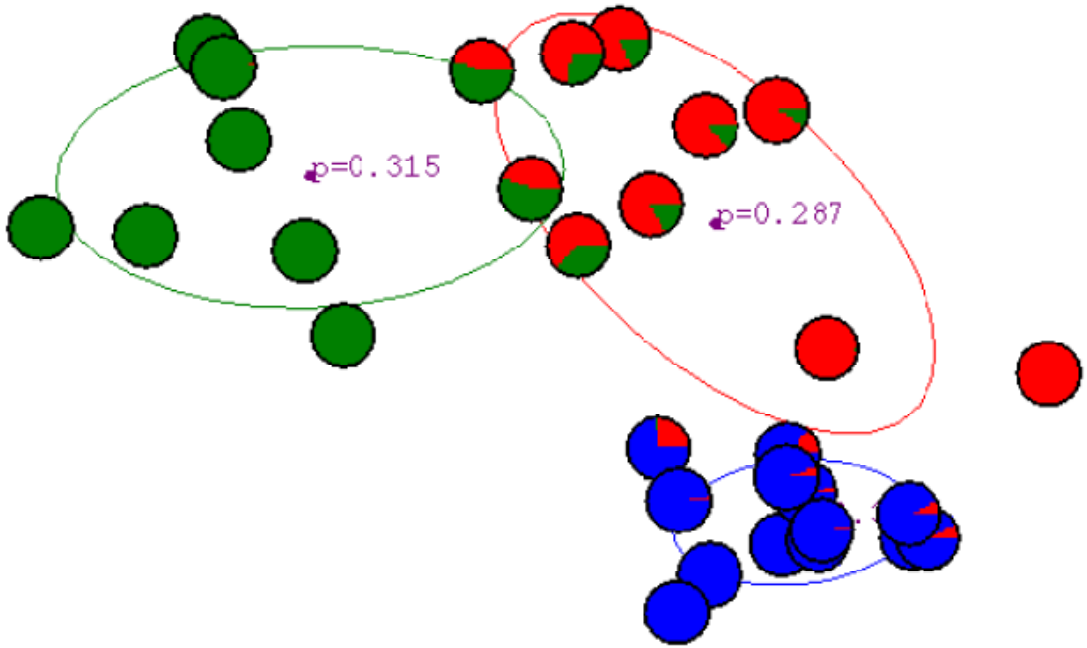
After 4th iteration



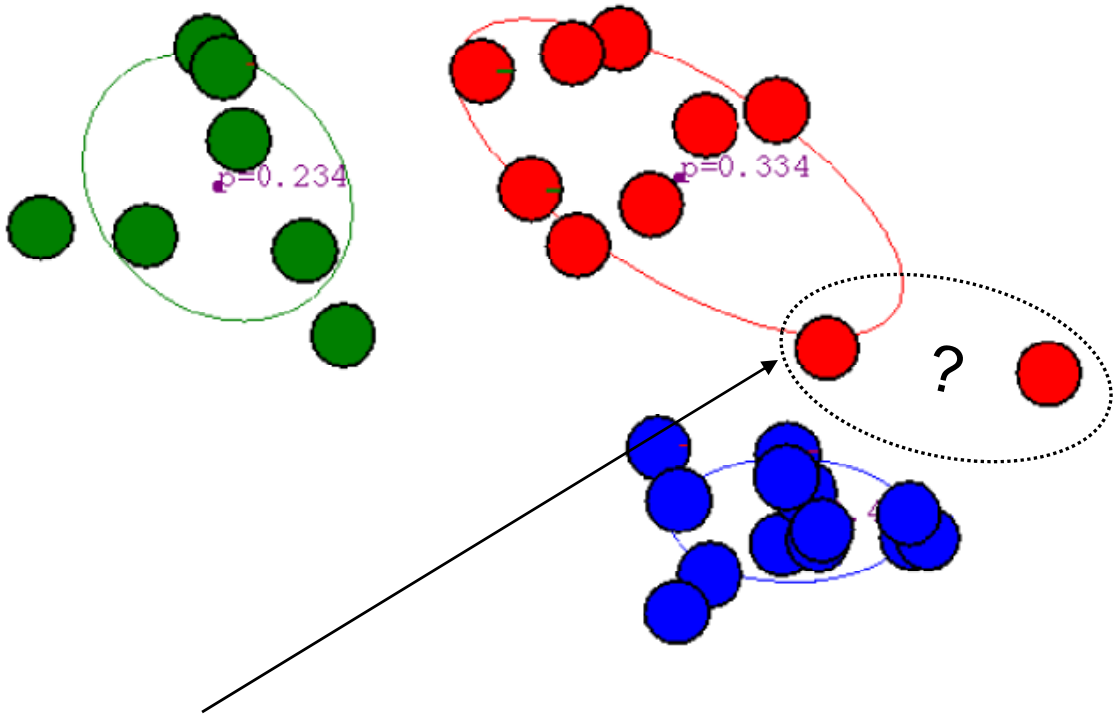
After 5th iteration



After 6th iteration



After 20th iteration



Q: Why are these two points red when they appear to be closer to blue?

K-Means is a Special Case

- we get K-Means if we make following restrictions:
  - All Gaussians have the identity covariance matrix (i.e., spherical Gaussians)
  - Use hard assignment for the E-step to assign data point to its most likely cluster

Behavior of EM

- It is guaranteed to converge
- In practice it may converge slowly, one can stop early if the change in log-likelihood is smaller than a threshold
- Like K-means it converges to a local optimum
  - Multiple restart is recommended