UNIVERSITY OF THE WITWATERSRAND, JOHANNESBURG

COMS 4030A/7047A Adaptive Computation and Machine Learning

Hima Vadapalli

Semester I, 2022

Last class:
Unsupervised Learning – K Means

Today:
Unsupervised Learning - GMM

Slides based heavily on course material by Eric Eton Andrew Moore and Andrew Ng

Clustering Methods

- Hard Clustering : Clusters do not overlap
  - Element either belongs to cluster or it does not

- Soft Clustering : Clusters may overlap
  - Strength of association between clusters and instances

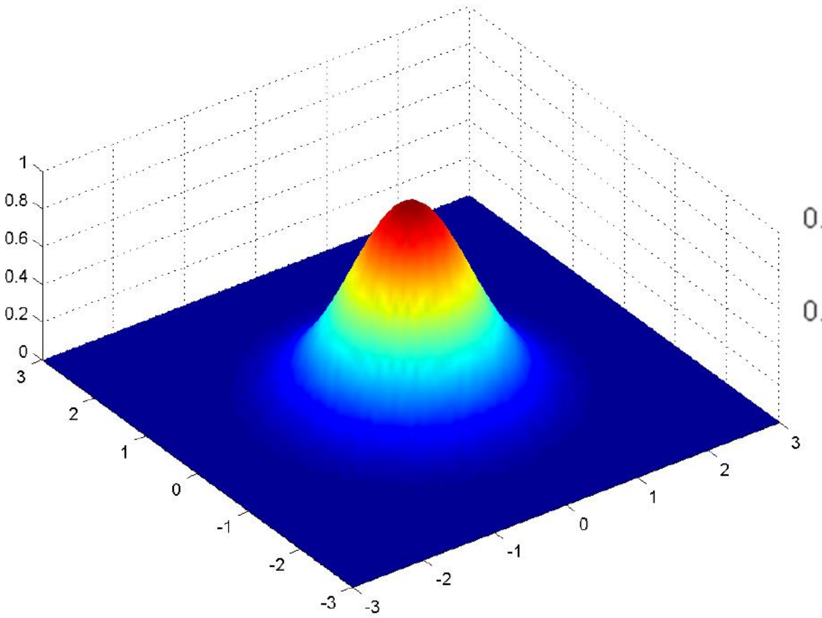
- Mixture models
  - Probabilistically - grounded way of performing soft clustering
  - Each cluster is a generative model (Gaussian or multinomial)
  - Parameters (mean/covariance are unknown)

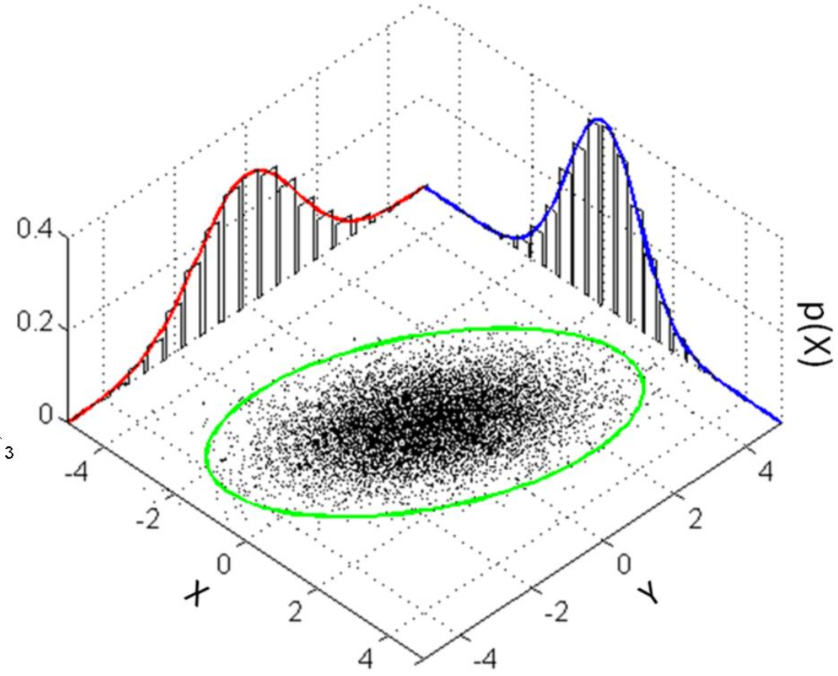
- Expectation – Maximization (EM) algorithm
  - Allows us to estimate all parameters for the mixture of distributions

Gaussian Mixture Models

- Recall the Gaussian distribution:

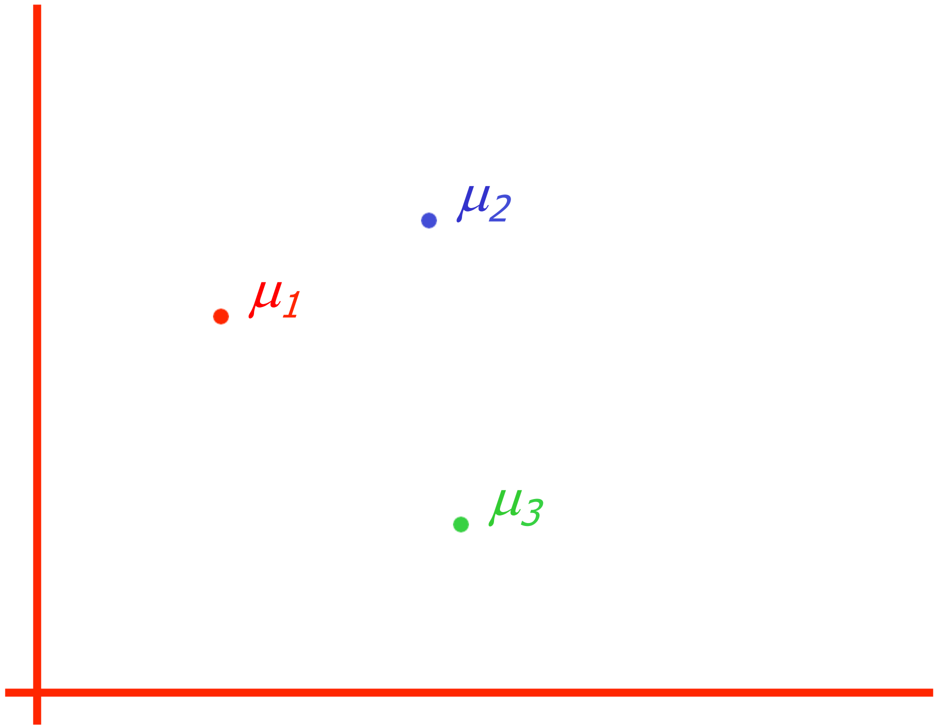
$$P(\mathbf{x} \mid \boldsymbol{\mu}, \boldsymbol{\Sigma}) = \frac{1}{\sqrt{(2\pi)^d |\boldsymbol{\Sigma}|}} \exp\left(-\frac{1}{2}(\mathbf{x}-\boldsymbol{\mu})^\intercal \boldsymbol{\Sigma}^{-1}(\mathbf{x}-\boldsymbol{\mu})\right)$$





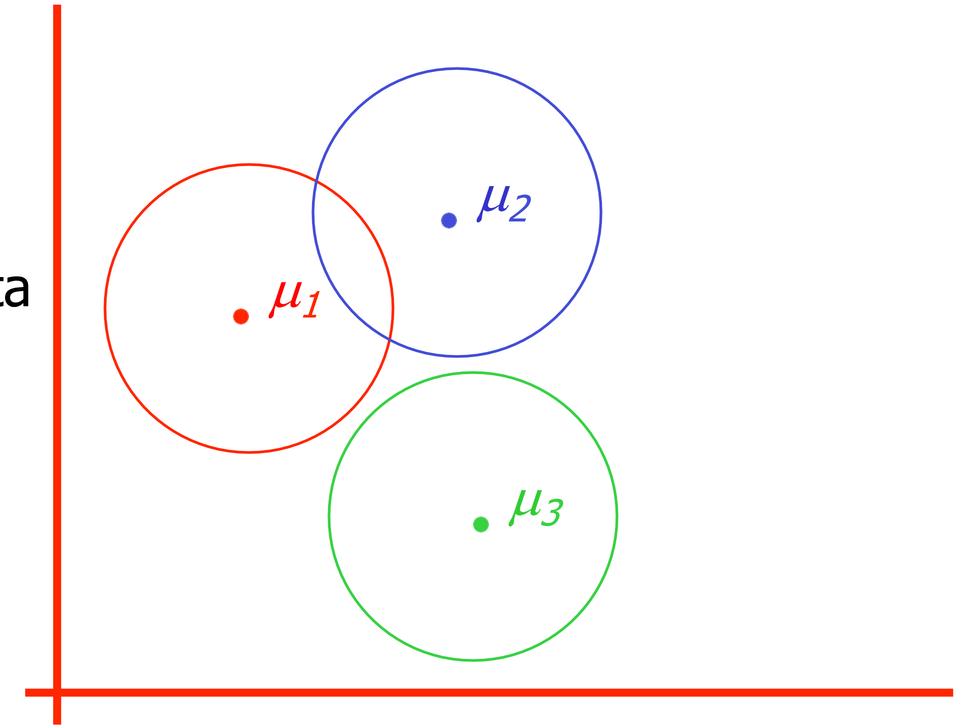
The GMM assumption

- There are k components. The i'th component is called ω_i
- Component ω_i has an associated mean vector μ_I



The GMM assumption

- There are k components. The i'th component is called ω_i
- Component ω_i has an associated mean vector μ_i
- Each component generates data from a Gaussian with mean μ_i and covariance matrix $\sigma^2 \boldsymbol{I}$

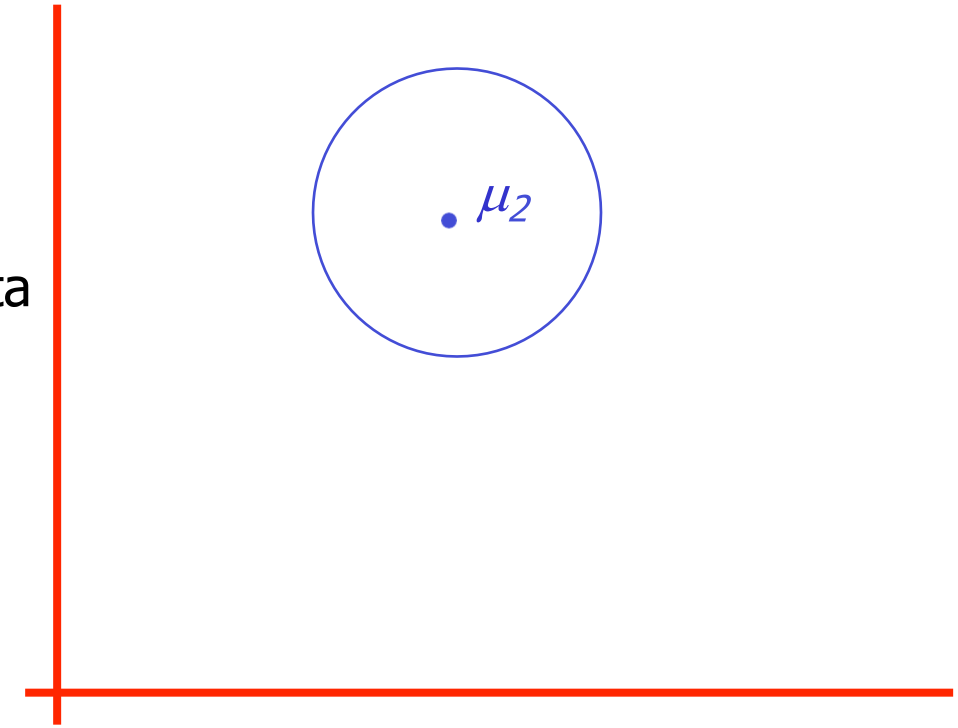


The GMM assumption

- There are k components. The i'th component is called ω_i
- Component ω_i has an associated mean vector μ_i
- Each component generates data from a Gaussian with mean μ_i and covariance matrix $\sigma^2 \boldsymbol{I}$

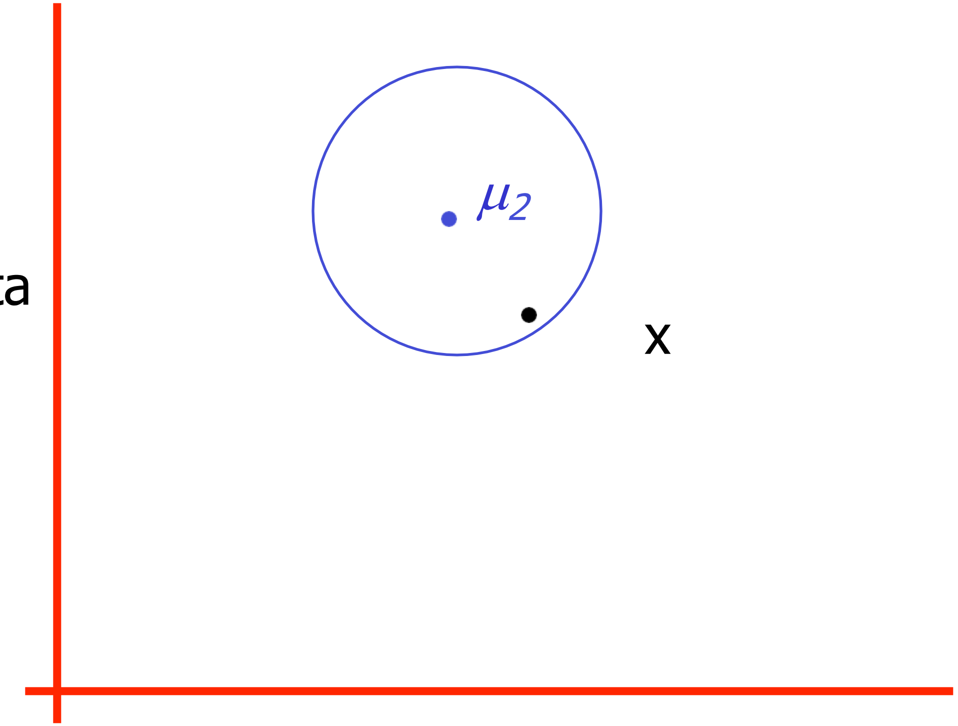
Assume that each datapoint is generated according to the following recipe:

1. Pick a component at random. Choose component i with probability $P(\omega_i)$.



The GMM assumption

- There are k components. The i'th component is called ω_i
- Component ω_i has an associated mean vector μ_i
- Each component generates data from a Gaussian with mean μ_i and covariance matrix $\sigma^2 \boldsymbol{I}$



Assume that each datapoint is generated according to the following recipe:

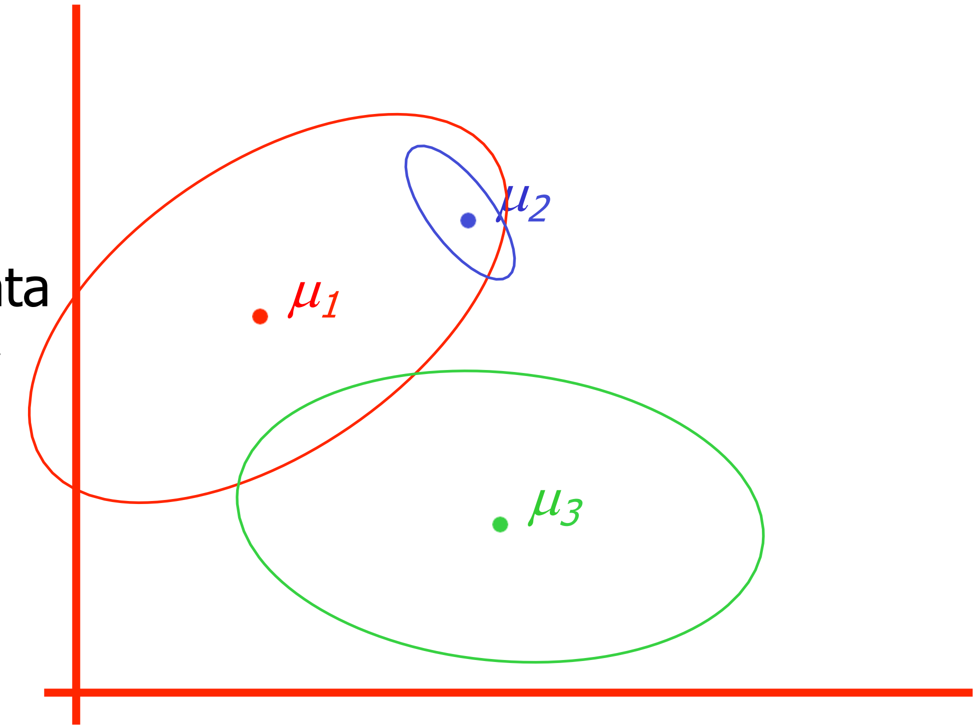
1. Pick a component at random. Choose component i with probability $P(\omega_i)$.
2. Datapoint ~ $N(\mu_i, \sigma^2 \boldsymbol{I})$

The General GMM assumption

- There are k components. The i'th component is called ω_i
- Component ω_i has an associated mean vector μ_i
- Each component generates data from a Gaussian with mean μ_i and covariance matrix Σ_i

Assume that each datapoint is generated according to the following recipe:

1. Pick a component at random. Choose component i with probability $P(\omega_i)$.
2. Datapoint ~ N(μ_i, Σ_i)



E.M. for General GMMs

$p_i(t)$ is shorthand for estimate of $P(\omega_i)$ on t' th iteration

Iterate. On the *t*' th iteration let our estimates be

$$\lambda_t = \{ \mu_1(t), \mu_2(t) \ldots \mu_c(t), \Sigma_1(t), \Sigma_2(t) \ldots \Sigma_c(t), p_1(t), p_2(t) \ldots p_c(t) \}$$

E-step: Compute "expected" clusters of all datapoints

Just evaluate a Gaussian at x_k

$$\mathrm{P}(w_i | x_k, \lambda_t) = \frac{\mathrm{p}(x_k | w_i, \lambda_t)\mathrm{P}(w_i | \lambda_t)}{\mathrm{p}(x_k | \lambda_t)} = \frac{\mathrm{p}(x_k | w_i, \mu_i(t), \Sigma_i(t)) p_i(t)}{\sum_{j=1}^{c} \mathrm{p}(x_k | w_j, \mu_j(t), \Sigma_j(t)) p_j(t)}$$

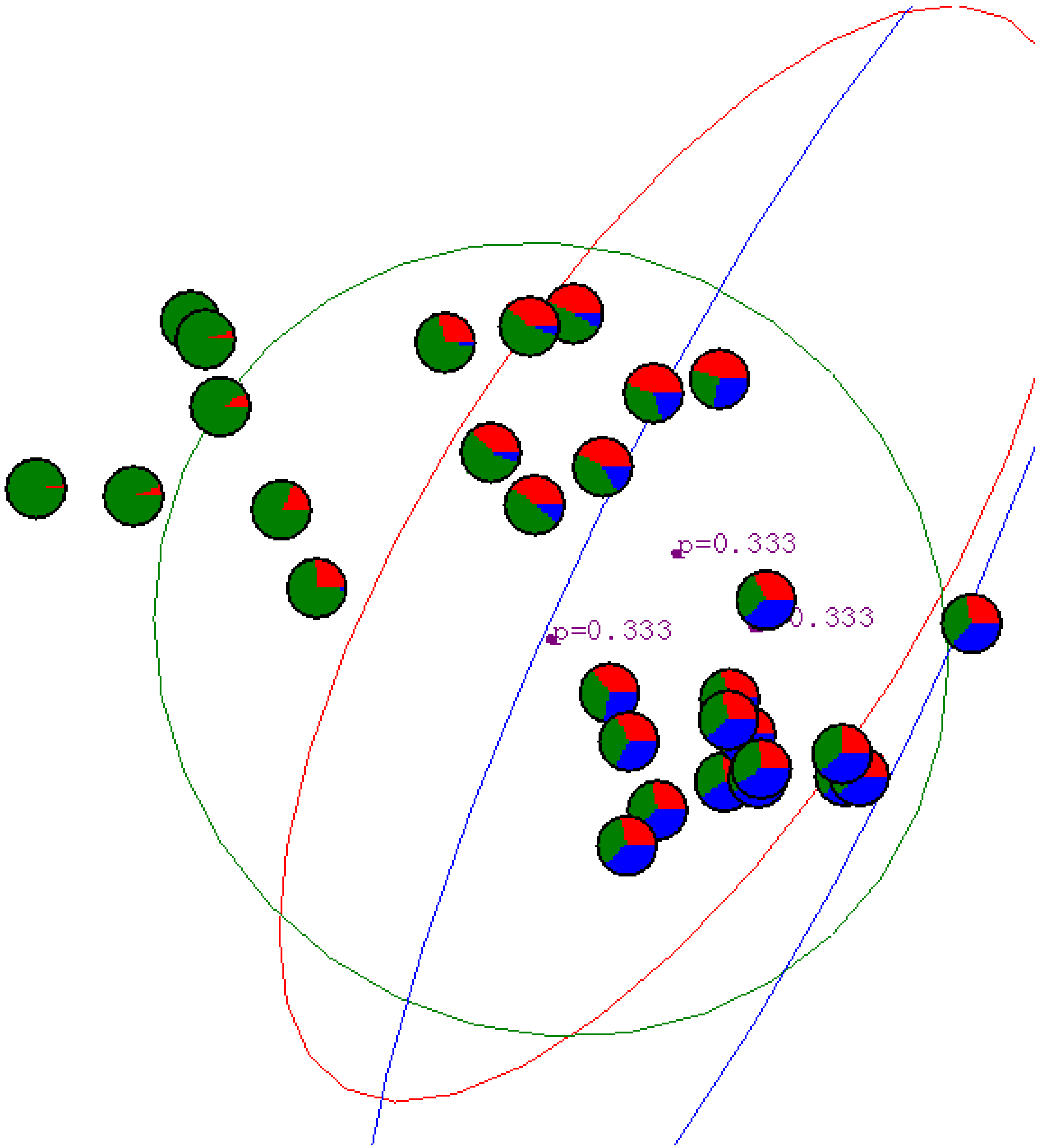
M-step: Estimate **μ, Σ** given our data's class membership distributions

$$\mu_i(t+1) = \frac{\sum_k \mathrm{P}(w_i | x_k, \lambda_t) x_k}{\sum_k \mathrm{P}(w_i | x_k, \lambda_t)} \qquad \Sigma_i(t+1) = \frac{\sum_k \mathrm{P}(w_i | x_k, \lambda_t)[x_k - \mu_i(t+1)][x_k - \mu_i(t+1)]^T}{\sum_k \mathrm{P}(w_i | x_k, \lambda_t)}$$

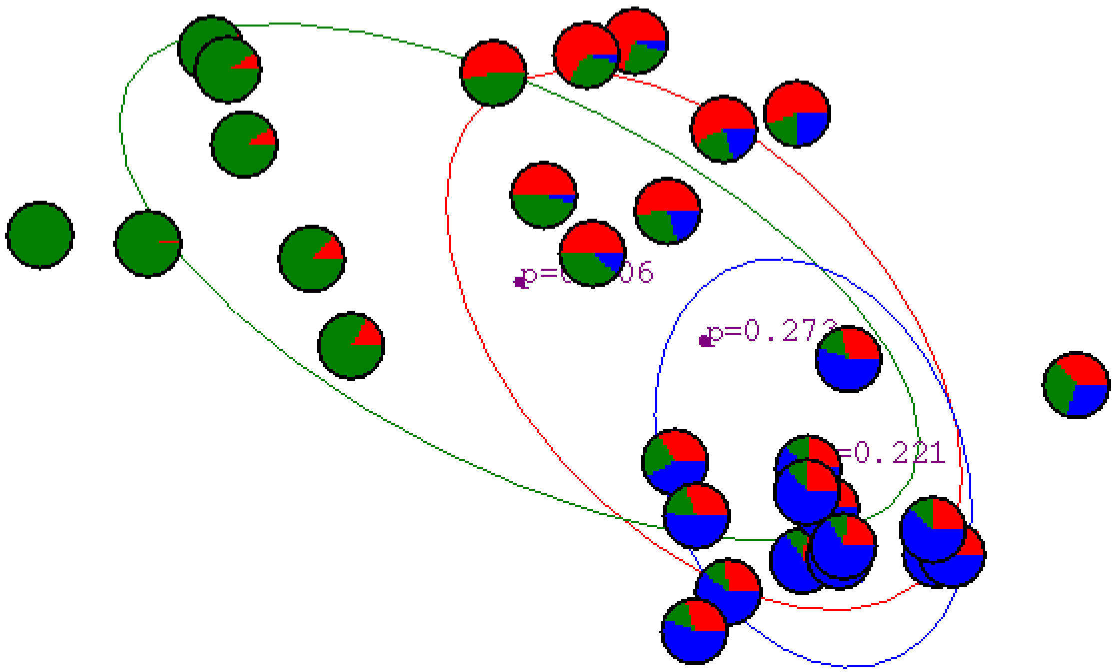
$$p_i(t+1) = \frac{\sum_k \mathrm{P}(w_i | x_k, \lambda_t)}{R}$$

R = #records

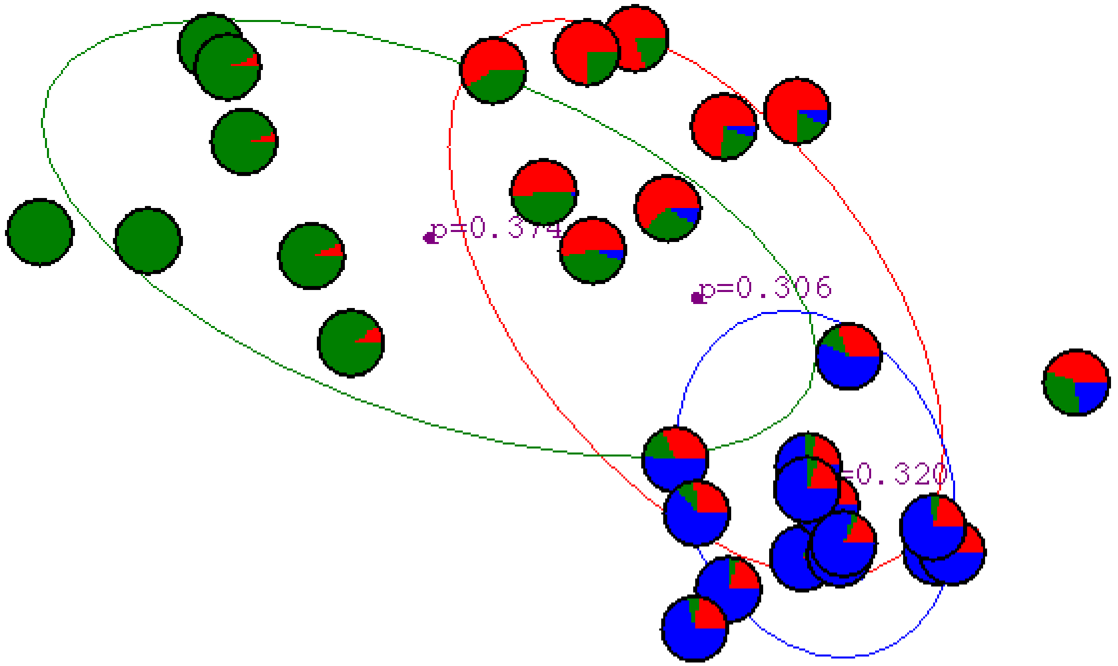
Gaussian Mixture Example: Start



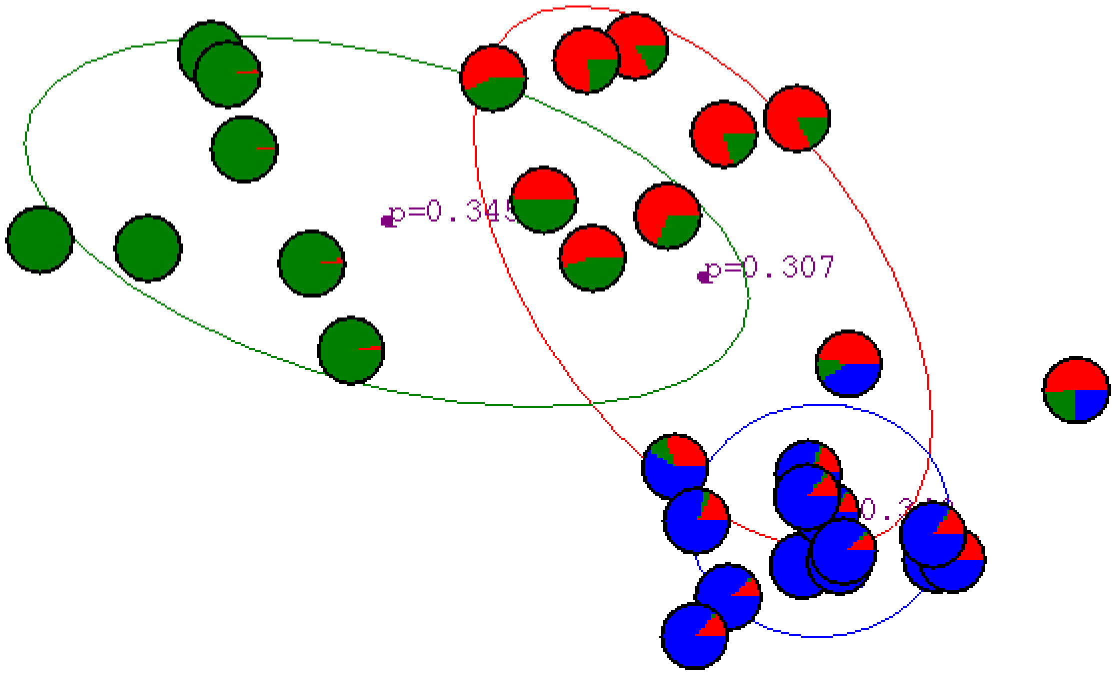
After first iteration



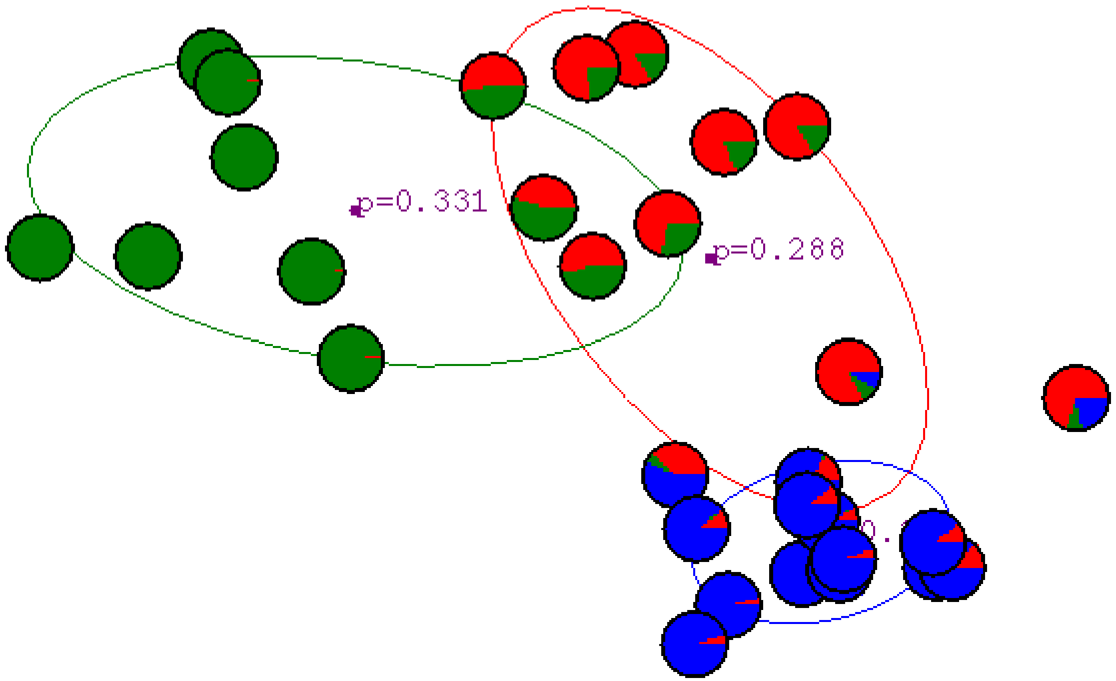
After 2nd iteration



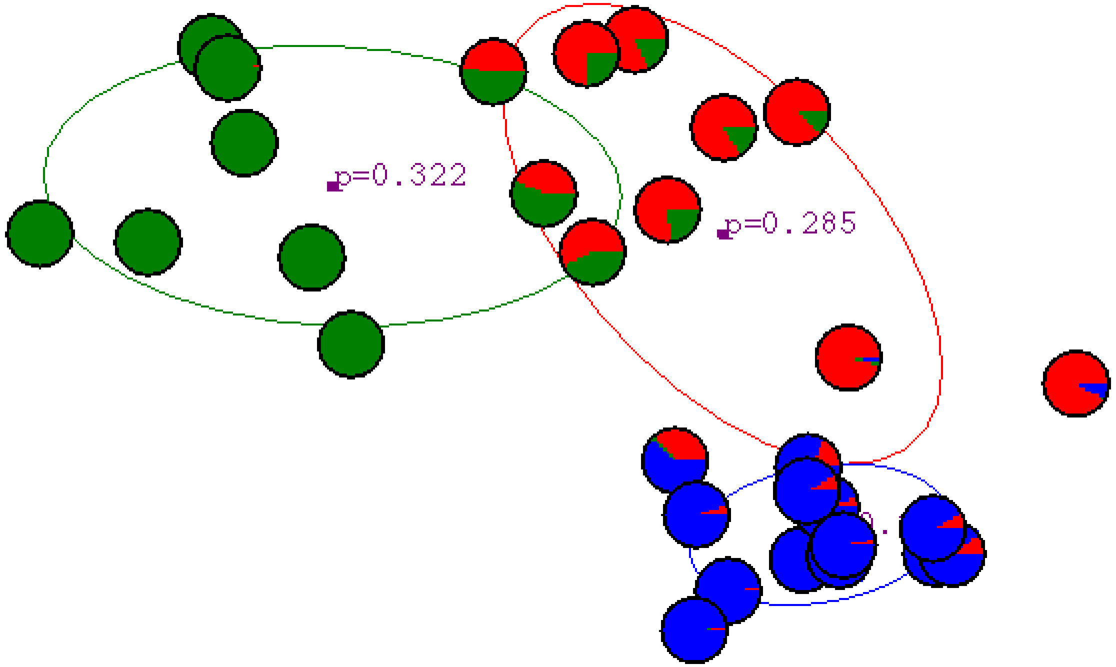
After 3rd iteration



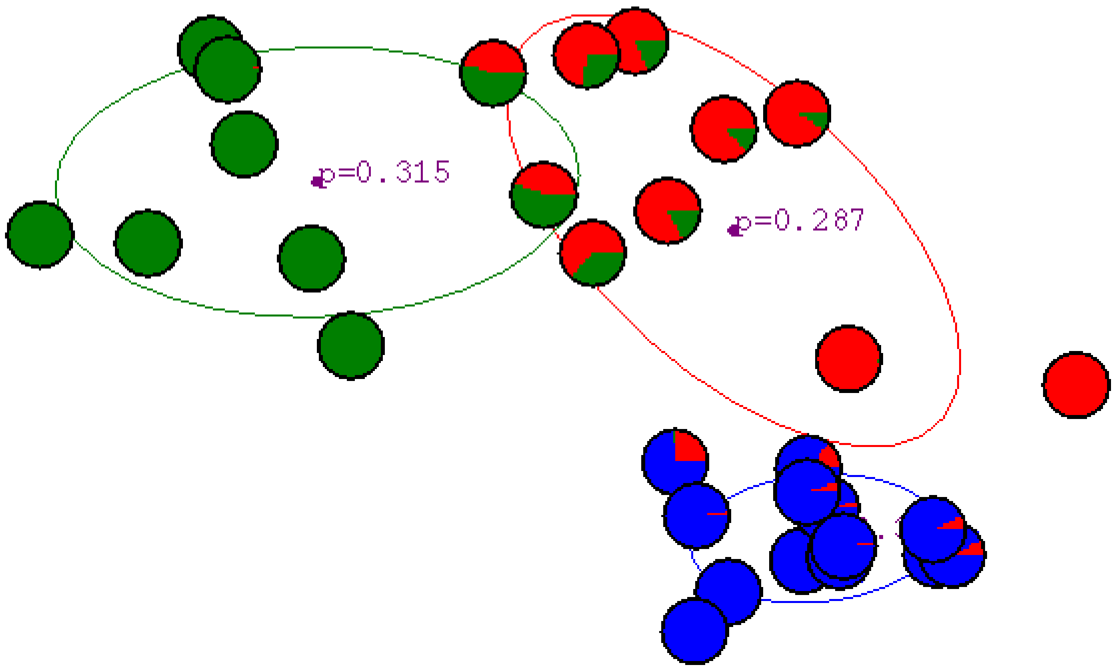
After 4th iteration



After 5th iteration



After 6th iteration



After 20th iteration