Expectation-Maximization

Dr. Víctor Uc Cetina

Facultad de Matemáticas
Universidad Autónoma de Yucatán

cetina@informatik.uni-hamburg.de
https://sites.google.com/view/victoruccetina

Content

K-means clustering algorithm

- In the clustering problem, we are given a training set $\{x^{(1)}, \ldots, x^{(m)}\}$, $x^{(i)} \in \mathbb{R}^n$.

K-means clustering algorithm

- In the clustering problem, we are given a training set $\{x^{(1)}, \dots, x^{(m)}\}$, $x^{(i)} \in \mathbb{R}^n$.
- We want to group the data into a few cohesive clusters.

K-means clustering algorithm

- In the clustering problem, we are given a training set $\{x^{(1)}, \ldots, x^{(m)}\}$, $x^{(i)} \in \mathbb{R}^n$.
- We want to group the data into a few cohesive clusters.
- This is an **unsupervised learning** problem.

K-means clustering algorithm

The k-means clustering algorithm is as follows:

1. Initialize cluster centroids $\mu_1, \mu_2, \ldots, \mu_k \in \mathbb{R}^n$ randomly.
2. Repeat until convergence {

   - For every i, set

   $$c^{(i)} := \arg\min_j ||x^{(i)} - \mu_j||2.$$

   - For every j, set

   $$\mu_j = \frac{\sum_{i=1}^m 1\{c^{(i)}=j\}x^{(i)}}{\sum_{i=1}^m 1\{c^{(i)}=j\}}.$$

}

K-means clustering algorithm

The inner-loop of the algorithm repeatedly carries out two steps:

1. Assigning each training example $x^{(i)}$ to the closest cluster centroid μ_j, and

K-means clustering algorithm

The inner-loop of the algorithm repeatedly carries out two steps:

1. Assigning each training example $x^{(i)}$ to the closest cluster centroid μ_j, and
2. Moving each cluster centroid μ_j to the mean of the points assigned to it.

K-means clustering algorithm

- k is the number of clusters we want to find.

K-means clustering algorithm

- k is the number of clusters we want to find.
- The cluster centroids μ_j represent our current guesses for the positions of the centers of the clusters.

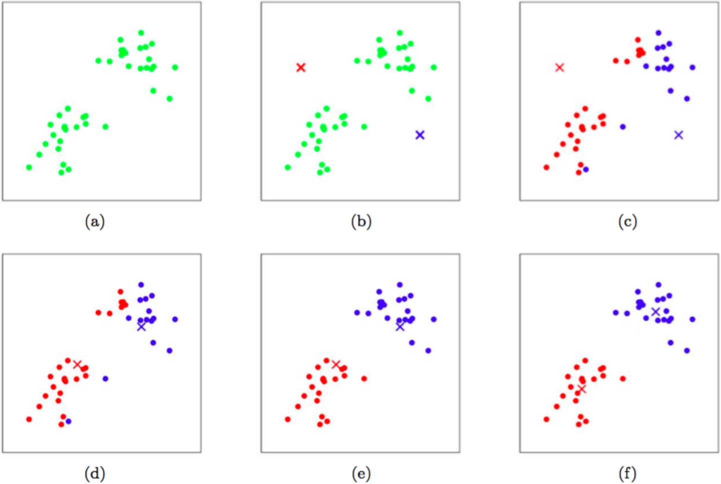
K-means clustering algorithm

- k is the number of clusters we want to find.
- The cluster centroids μ_j represent our current guesses for the positions of the centers of the clusters.
- We could choose k training examples randomly, and set the initial cluster centroids to be equal to the values of these k examples.

K-means clustering algorithm

- k is the number of clusters we want to find.
- The cluster centroids μ_j represent our current guesses for the positions of the centers of the clusters.
- We could choose k training examples randomly, and set the initial cluster centroids to be equal to the values of these k examples.
- Other initialization methods are also possible.

K-means clustering algorithm


(a) (b) (c) (d) (e) (f)

K-means clustering algorithm

Original image

$K = 2$

K-means clustering algorithm

Original image

$K = 3$

K-means clustering algorithm

Original image

$K = 10$

K-means clustering algorithm

$$K = 2$$

K-means clustering algorithm

$$K = 3$$

K-means clustering algorithm

$$K = 10$$

K-means clustering algorithm

- The k-means algorithm is guaranteed to converge in a certain sense. Let us define the distortion function to be:

  $$J(c, \mu) = \sum_{i=1}^{m} ||x^{(i)} - \mu_{c^{(i)}}||^2.$$

K-means clustering algorithm

- The k-means algorithm is guaranteed to converge in a certain sense. Let us define the distortion function to be:

$$J(c, \mu) = \sum_{i=1}^{m} ||x^{(i)} - \mu_{c^{(i)}}||^2.$$

- In theory, it is possible for k-means to oscillate between a few different clusterings that have exactly the same value of J, but this almost never happens in practice.

K-means clustering algorithm

- The k-means algorithm is guaranteed to converge in a certain sense. Let us define the distortion function to be:

$$J(c, \mu) = \sum_{i=1}^{m} ||x^{(i)} - \mu_{c^{(i)}}||^2.$$

- In theory, it is possible for k-means to oscillate between a few different clusterings that have exactly the same value of J, but this almost never happens in practice.

- The distortion function J is a non-convex function, and then the algorithm can get stuck in local minima. Nevertheless, very often k-means will work fine and come up with very good clusterings.

K-means clustering algorithm

- Worried about getting stuck in bad local minima?

K-means clustering algorithm

- Worried about getting stuck in bad local minima?
- One common thing to do is run k-means many times using different random initial values for the cluster centroids μ_j.

Expectation-Maximization

- Lets talk first about Maximum Likelihood (ML) estimation.

Expectation-Maximization

- Lets talk first about Maximum Likelihood (ML) estimation.
- The goal of ML estimation is to find the parameters that maximize the probability of having received certain measurements of a random variable, distributed by some probability density function.

Expectation-Maximization

- Lets talk first about Maximum Likelihood (ML) estimation.
- The goal of ML estimation is to find the parameters that maximize the probability of having received certain measurements of a random variable, distributed by some probability density function.
- Given the density function $p(x|\theta)$ that is governed by a set of parameters θ, where p might be a set of Gaussians and θ could be the means and the covariances.

Expectation-Maximization

- Given also a data set $\{x^{(1)}, \ldots, x^{(m)}\}$ of size m, drawn from this distribution.

Expectation-Maximization

- Given also a data set $\{x^{(1)}, \ldots, x^{(m)}\}$ of size m, drawn from this distribution.
- Then, the resulting density for the samples is

$$p(x|\theta) = \prod_{i=1}^{m} p(x(i)|\theta) = \mathcal{L}(\theta|x).$$

Expectation-Maximization

- Given also a data set $\{x^{(1)}, \ldots, x^{(m)}\}$ of size m, drawn from this distribution.
- Then, the resulting density for the samples is

$$p(x|\theta) = \prod_{i=1}^{m} p(x(i)|\theta) = \mathcal{L}(\theta|x).$$

- The function $\mathcal{L}(\theta|x)$ is called the likelihood of the parameters θ given the data.

Expectation-Maximization

- In the maximum likelihood problem, our goal is to find the parameters θ that maximize $\mathcal{L}$.

Expectation-Maximization

- In the maximum likelihood problem, our goal is to find the parameters θ that maximize $\mathcal{L}$.

- That is, we want to find θ^* where

$$\theta^* = \arg\max_{\theta} \mathcal{L}(\theta|x).$$

Expectation-Maximization

- In the maximum likelihood problem, our goal is to find the parameters θ that maximize $\mathcal{L}$.

- That is, we want to find θ^* where

$$\theta^* = \arg\max_{\theta} \mathcal{L}(\theta|x).$$

- Often we maximize $\log \mathcal{L}(\theta|x)$ instead because it is analytically easier.

Expectation-Maximization

- In the maximum likelihood problem, our goal is to find the parameters θ that maximize $\mathcal{L}$.
- That is, we want to find θ^* where

$$\theta^* = \arg\max_{\theta} \mathcal{L}(\theta|x).$$

- Often we maximize $\log \mathcal{L}(\theta|x)$ instead because it is analytically easier.
- However, for many problems, it is not possible to solve this maximization problem analytically in closed form, and we must resort to more elaborate techniques.

Expectation-Maximization

- The Expectation-Maximization (EM) algorithm is a general method of finding the maximum likelihood estimate of the parameters of a distribution, given a data set.

Expectation-Maximization

- The Expectation-Maximization (EM) algorithm is a general method of finding the maximum likelihood estimate of the parameters of a distribution, given a data set.
- The EM algorithm is useful when the data is incomplete or has missing values.

Expectation-Maximization

There are two main applications of the EM algorithm:

Expectation-Maximization

There are two main applications of the EM algorithm:

1. The first occurs when the data indeed has missing values, due to problems with or limitations of the observation process.

Expectation-Maximization

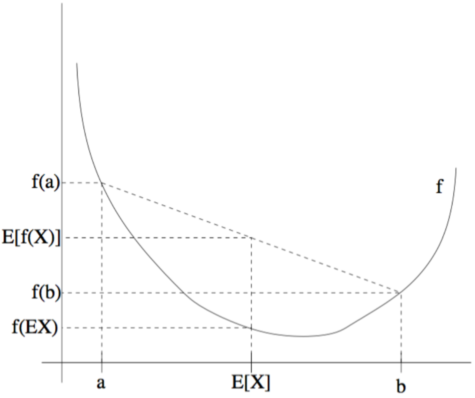
There are two main applications of the EM algorithm:

1. The first occurs when the data indeed has missing values, due to problems with or limitations of the observation process.
2. The second occurs when optimizing the likelihood function is analytically intractable, but the likelihood function can be simplified by assuming the existence of hidden parameters.

Expectation-Maximization

Jensen's inequality

$E[f(X)] \geq f(E[X])$



Expectation-Maximization

- When $f(X)$ is **convex** we have:

$$E\left[f(X)\right] \geq f(E\left[X\right])$$

Expectation-Maximization

- When $f(X)$ is **convex** we have:

$$E\left[f(X)\right] \geq f(E\left[X\right])$$

- When $f(X)$ is **concave** we have:

$$f(E\left[X\right]) \geq E\left[f(X)\right]$$

Expectation-Maximization

Suppose we have the training set $\{x^{(1)}, \ldots, x^{(m)}\}$, $x \in \mathbb{R}^n$.
Consisting of m independent examples.

Expectation-Maximization

Suppose we have the training set $\{x^{(1)}, \ldots, x^{(m)}\},\ x \in \mathbb{R}^n$. Consisting of m independent examples.

We wish to fit the parameters θ of a model $p(x, z)$ to the data where the likelihood is given by

$$\begin{aligned} \ell(\theta) &= \sum_{i=1}^{m} \log p(x;\theta) \\ &= \sum_{i=1}^{m} \log \sum_{z} p(x, z; \theta). \end{aligned}$$

Expectation-Maximization

$$\begin{aligned}\sum_i \log p(x^{(i)};\theta) &= \sum_i \log \sum_{z^{(i)}} p(x^{(i)}, z^{(i)};\theta) \\ &= \sum_i \log \sum_{z^{(i)}} Q_i(z^{(i)}) \frac{p(x^{(i)}, z^{(i)};\theta)}{Q_i(z^{(i)})} \\ &\geq \sum_i \sum_{z^{(i)}} Q_i(z^{(i)}) \log \frac{p(x^{(i)}, z^{(i)};\theta)}{Q_i(z^{(i)})}\end{aligned}$$

Expectation-Maximization

For each i let Q_i be some distribution over the z's , that is $\sum_z Q_i(z) = 1, Q_i(z) \geq 0$.

In the EM algorithm, you repeat until convergence:

(E-step) For each i, set

$$Q_i(z^{(i)}) := p(z^{(i)}|x^{(i)};\theta).$$

(M-step) Set

$$\theta := \arg\max_{\theta} \sum_i \sum_{z^{(i)}} Q_i(z^{(i)}) \log \frac{p(x^{(i)}, z^{(i)}; \theta)}{Q_i(z^{(i)})}.$$

Mixtures of Gaussians

- Again, suppose that we are given a training set $\{x^{(1)}, \ldots, x^{(m)}\},\ x \in \mathbb{R}^n$.

Mixtures of Gaussians

- Again, suppose that we are given a training set $\{x^{(1)}, \ldots, x^{(m)}\},\ x \in \mathbb{R}^n$.
- We wish to model the data by specifying a joint distribution $p(x^{(i)}, z^{(i)}) = p(x^{(i)}|z^{(i)})p(z^{(i)})$.

Mixtures of Gaussians

- Again, suppose that we are given a training set $\{x^{(1)}, \ldots, x^{(m)}\},\ x \in \mathbb{R}^n$.
- We wish to model the data by specifying a joint distribution

  $p(x^{(i)}, z^{(i)}) = p(x^{(i)}|z^{(i)})p(z^{(i)})$.
- Here, $z^{(i)} \sim \mathrm{Multinomial}(\phi)$, where

  $\phi_j \geq 0$,

  $\sum_{j=1}^{k} \phi_j = 1$.

Mixtures of Gaussians

- Again, suppose that we are given a training set $\{x^{(1)}, \ldots, x^{(m)}\},\ x \in \mathbb{R}^n$.
- We wish to model the data by specifying a joint distribution

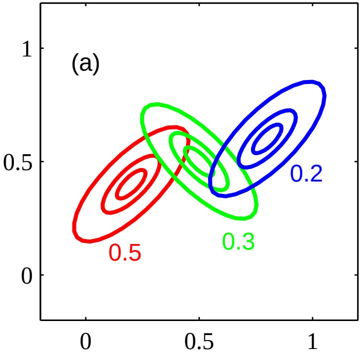
  $p(x^{(i)}, z^{(i)}) = p(x^{(i)}|z^{(i)})p(z^{(i)})$.
- Here, $z^{(i)} \sim \mathrm{Multinomial}(\phi)$, where

  $\phi_j \geq 0$,

  $\sum_{j=1}^{k} \phi_j = 1$.
- The parameter ϕ_j gives $p(z^{(i)} = j)$ and $x^{(i)}|z^{(i)} = j \sim \mathcal{N}(\mu_j, \Sigma_j)$.

Mixtures of Gaussians

$K = 3$ Gaussians



Mixtures of Gaussians

- $p(x^{(i)}, z^{(i)}) = p(x^{(i)}|z^{(i)})p(z^{(i)})$.

Mixtures of Gaussians

- $p(x^{(i)}, z^{(i)}) = p(x^{(i)}|z^{(i)})p(z^{(i)})$.
- We let k denote the number of values that the $z^{(i)}$'s can take on.

Mixtures of Gaussians

- $p(x^{(i)}, z^{(i)}) = p(x^{(i)}|z^{(i)})p(z^{(i)})$.
- We let k denote the number of values that the $z^{(i)}$'s can take on.
- Thus, our model posits that each $x^{(i)}$ was generated by randomly choosing $z^{(i)}$ from $\{1, \ldots, k\}$, and then $x^{(i)}$ was drawn from one of k Gaussians, depending on $z^{(i)}$.

Mixtures of Gaussians

- $p(x^{(i)}, z^{(i)}) = p(x^{(i)}|z^{(i)})p(z^{(i)})$.
- We let k denote the number of values that the $z^{(i)}$'s can take on.
- Thus, our model posits that each $x^{(i)}$ was generated by randomly choosing $z^{(i)}$ from $\{1, \ldots, k\}$, and then $x^{(i)}$ was drawn from one of k Gaussians, depending on $z^{(i)}$.
- This is called the **mixture of Gaussians** model.

Mixtures of Gaussians

- $p(x^{(i)}, z^{(i)}) = p(x^{(i)}|z^{(i)})p(z^{(i)})$.
- We let k denote the number of values that the $z^{(i)}$'s can take on.
- Thus, our model posits that each $x^{(i)}$ was generated by randomly choosing $z^{(i)}$ from $\{1, \ldots, k\}$, and then $x^{(i)}$ was drawn from one of k Gaussians, depending on $z^{(i)}$.
- This is called the **mixture of Gaussians** model.
- Note that the $z^{(i)}$'s are **latent** random variables, meaning that they are hidden/unobserved.

Mixtures of Gaussians

- The parameters of our model are: ϕ, μ and Σ.

Mixtures of Gaussians

- The parameters of our model are: ϕ, μ and Σ.
- To estimate them, we write the likelihood as follows:

$$\begin{aligned}\ell(\phi,\mu,\Sigma) &= \sum_{i=1}^{m} \log p(x^{(i)};\phi,\mu,\Sigma) \\ &= \sum_{i=1}^{m} \log \sum_{z^{(i)}=1}^{k} p(x^{(i)}|z^{(i)};\mu,\Sigma)p(z^{(i)};\phi).\end{aligned}$$

Mixtures of Gaussians

- The parameters of our model are: ϕ, μ and Σ.
- To estimate them, we write the likelihood as follows:

$$\begin{aligned}\ell(\phi, \mu, \Sigma) &= \sum_{i=1}^{m} \log p(x^{(i)}; \phi, \mu, \Sigma) \\ &= \sum_{i=1}^{m} \log \sum_{z^{(i)}=1}^{k} p(x^{(i)} | z^{(i)}; \mu, \Sigma) p(z^{(i)}; \phi).\end{aligned}$$

- However, if we set to zero the derivatives of this formula with respect to the parameters and try to solve, we will see that it is not possible to find the maximum likelihood estimates of the parameters in closed form.

Mixtures of Gaussians

- The random variable $z^{(i)}$ indicates which of the k Gaussians each $x^{(i)}$ had come from.

Mixtures of Gaussians

- The random variable $z^{(i)}$ indicates which of the k Gaussians each $x^{(i)}$ had come from.
- If we knew what the $z^{(i)}$ were, the maximum likelihood problem would have been easy, writing the likelihood as

$$\ell(\phi, \mu, \Sigma) = \sum_{i=1}^{m} \log p(x^{(i)} | z^{(i)}; \mu, \Sigma) + \log p(z^{(i)}; \phi).$$

Mixtures of Gaussians

- The random variable $z^{(i)}$ indicates which of the k Gaussians each $x^{(i)}$ had come from.
- If we knew what the $z^{(i)}$ were, the maximum likelihood problem would have been easy, writing the likelihood as

$$\ell(\phi, \mu, \Sigma) = \sum_{i=1}^{m} \log p(x^{(i)}|z^{(i)}; \mu, \Sigma) + \log p(z^{(i)}; \phi).$$

- Maximizing this with respect to ϕ, μ and Σ gives the parameters:

$$\begin{aligned}
\phi_j &= \tfrac{1}{m} \textstyle\sum_{i=1}^{m} 1\{z^{(i)} = j\}, \\
\mu_j &= \frac{\sum_{i=1}^{m} 1\{z^{(i)}=j\} x^{(i)}}{\sum_{i=1}^{m} 1\{z^{(i)}=j\}}, \\
\Sigma_j &= \frac{\sum_{i=1}^{m} 1\{z^{(i)}=j\}(x^{(i)}-\mu_j)(x^{(i)}-\mu_j)^\top}{\sum_{i=1}^{m} 1\{z^{(i)}=j\}}.
\end{aligned}$$

Mixtures of Gaussians

- However, the $z^{(i)}$ are not known. What can we do?

Mixtures of Gaussians

- However, the $z^{(i)}$ are not known. What can we do?
- We can use the Expectation-Maximization (EM) algorithm that has two steps:

Mixtures of Gaussians

- However, the $z^{(i)}$ are not known. What can we do?
- We can use the Expectation-Maximization (EM) algorithm that has two steps:

  1. In the E-step, it tries to guess the values of the $z^{(i)}$'s.

Mixtures of Gaussians

- However, the $z^{(i)}$ are not known. What can we do?
- We can use the Expectation-Maximization (EM) algorithm that has two steps:

1. In the E-step, it tries to guess the values of the $z^{(i)}$'s.
2. In the M-step, it updates the parameters of our model based on our guesses.

Mixtures of Gaussians

- The EM algorithm would be implemented as follows:

Repeat until convergence {

1. (E-step) For each i, j set

$$w_j^{(i)} := p(z^{(i)} = j | x^{(i)}; \phi, \mu, \Sigma)$$

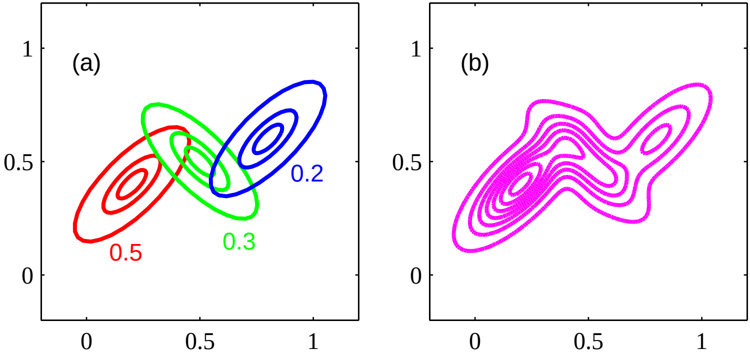
2. (M-step) Update the parameters

$$\begin{aligned}
\phi_j &= \frac{1}{m} \sum_{i=1}^{m} w_j^{(i)}, \\
\mu_j &= \frac{\sum_{i=1}^{m} w_j^{(i)} x^{(i)}}{\sum_{i=1}^{m} w_j^{(i)}}, \\
\Sigma_j &= \frac{\sum_{i=1}^{m} w_j^{(i)} (x^{(i)} - \mu_j)(x^{(i)} - \mu_j)^\top}{\sum_{i=1}^{m} w_j^{(i)}}.
\end{aligned}$$

}

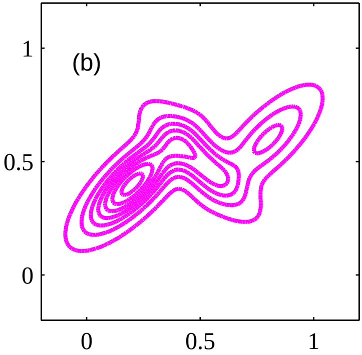
Mixtures of Gaussians

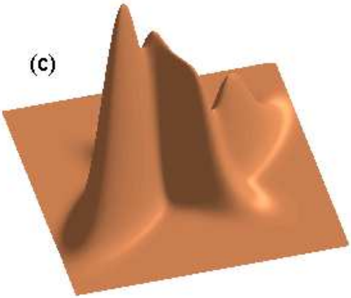
$K = 3$ Gaussians



Mixtures of Gaussians

$K = 3$ Gaussians





Mixtures of Gaussians

- Once we have estimated the parameters ϕ, μ and Σ, we could apply the following Bayes rule to classify a new point $x^{(i)}$:

$$p(z^{(i)} = j | x^{(i)}; \phi, \mu, \Sigma) = \frac{p(x^{(i)} | z^{(i)} = j; \mu, \Sigma) p(z^{(i)} = j; \phi)}{\sum_{l=1}^{k} p(x^{(i)} | z^{(i)} = l; \mu, \Sigma) p(z^{(i)} = l; \phi)}.$$

Mixtures of Gaussians

- Once we have estimated the parameters ϕ, μ and Σ, we could apply the following Bayes rule to classify a new point $x^{(i)}$:

$$p(z^{(i)} = j|x^{(i)}; \phi, \mu, \Sigma) = \frac{p(x^{(i)}|z^{(i)}=j;\mu,\Sigma)p(z^{(i)}=j;\phi)}{\sum_{l=1}^{k} p(x^{(i)}|z^{(i)}=l;\mu,\Sigma)p(z^{(i)}=l;\phi)}.$$

- This algorithm is reminescent of the K-means clustering algorithm, except that instead of having hard assignments $c(i)$, we have soft assignments $w_j^{(i)}$.

Mixtures of Gaussians

- Once we have estimated the parameters ϕ, μ and Σ, we could apply the following Bayes rule to classify a new point $x^{(i)}$:

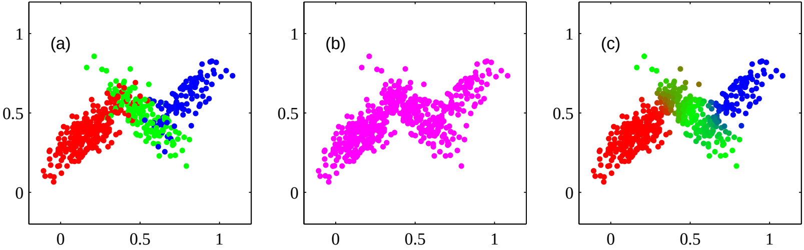
$$p(z^{(i)} = j | x^{(i)}; \phi, \mu, \Sigma) = \frac{p(x^{(i)}|z^{(i)}=j;\mu,\Sigma)p(z^{(i)}=j;\phi)}{\sum_{l=1}^{k} p(x^{(i)}|z^{(i)}=l;\mu,\Sigma)p(z^{(i)}=l;\phi)}.$$

- This algorithm is reminescent of the K-means clustering algorithm, except that instead of having hard assignments $c(i)$, we have soft assignments $w_j^{(i)}$.

- Similarly, to K-means, it is also susceptible to local optima, so reinitializing at several different initial parameters may be a good idea.

Mixtures of Gaussians

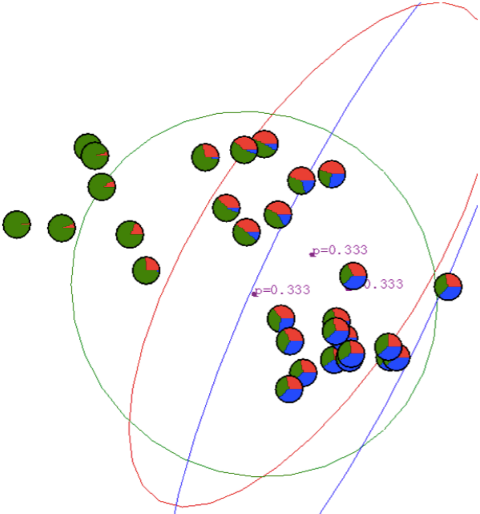
$K = 3$ Gaussians and 500 data points



(a) Complete data; (b) Incomplete data; (c) Final solution

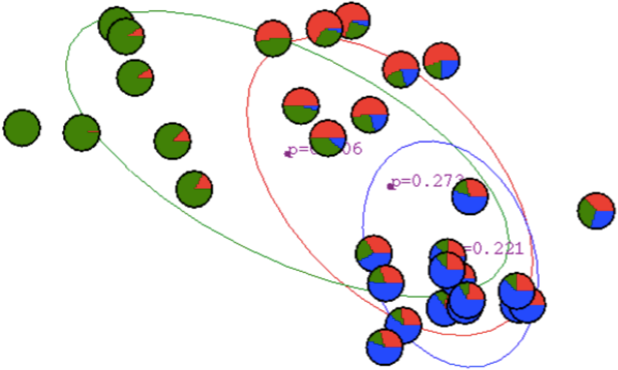
Mixtures of Gaussians

Start



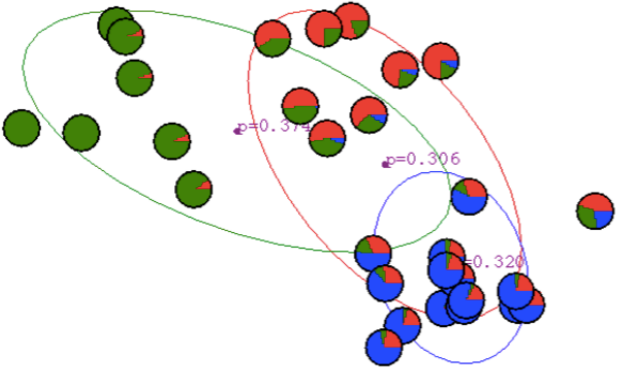
Mixtures of Gaussians

Iteration 1



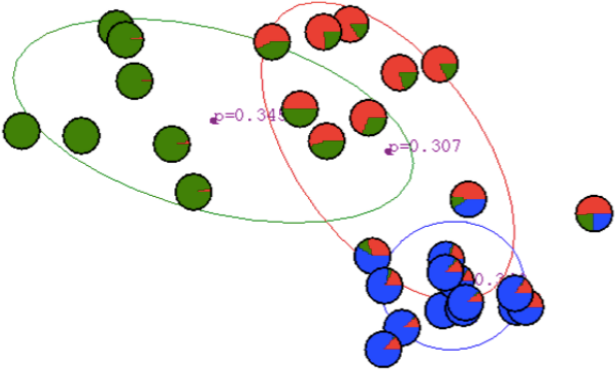
Mixtures of Gaussians

Iteration 2



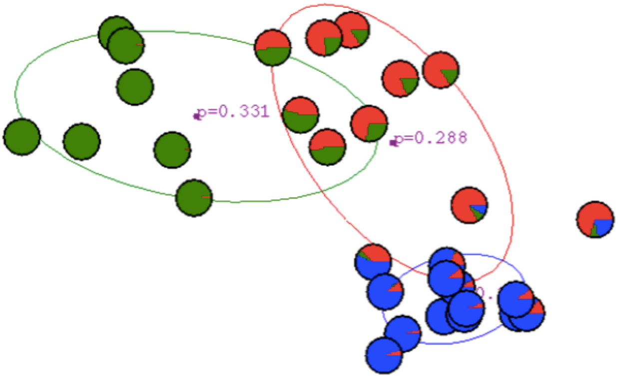
Mixtures of Gaussians

Iteration 3



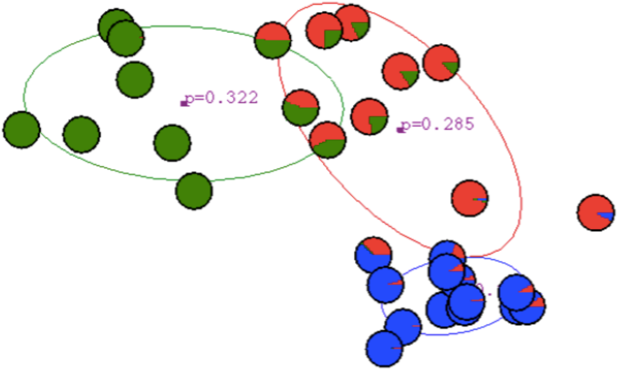
Mixtures of Gaussians

Iteration 4



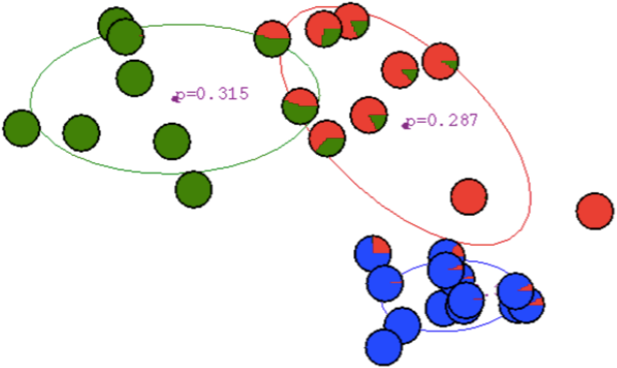
Mixtures of Gaussians

Iteration 5



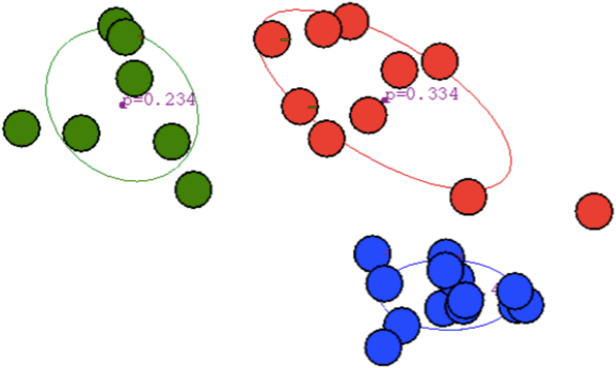
Mixtures of Gaussians

Iteration 6



Mixtures of Gaussians

Iteration 20



Reference

- Moritz Blume. **Expectation Maximization: a Gentle Introduction**. 2002.
- Jeff Bilmes. **A gentle tutorial of the EM algorithm and its application to parameter estimation for Gaussian mixture and hidden Markov models**. International Computer Science Institute. Berkeley. Technical Report TR-97-021. 1998.
- Andrew Ng. **Machine Learning Course Notes**. 2003.
- Christopher Bishop. **Pattern Recognition and Machine Learning**. Springer. 2006.
- Andrew Moore. **Clustering with Gaussian Mixtures**. Slides, CMU. 2001.

Thank you!

Dr. Victor Uc Cetina
cetina@informatik.uni-hamburg.de