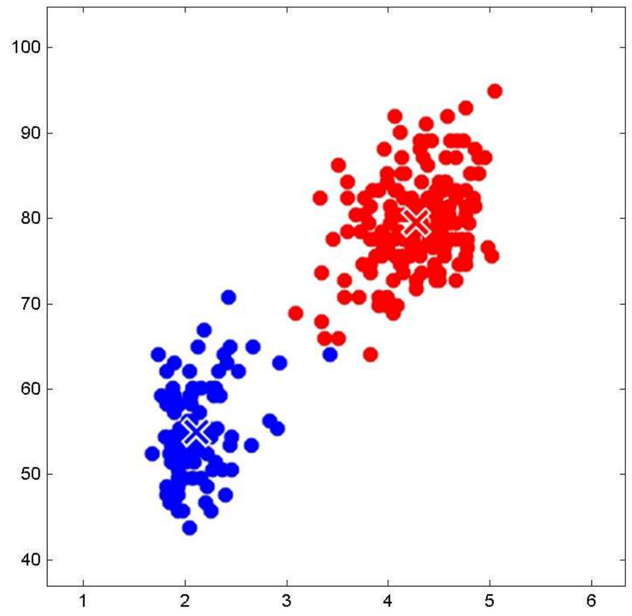
Gaussian Mixture Models

K-Means Clustering

- Algorithm organizes data into clusters such that there is high intra-cluster similarity and low inter-cluster similarity.
    - A datapoint will belong to one cluster, not several – resulting in a specific number of disjoint non-hierarchical clusters.

- Initialized prototypes, then iterated between two phases:
    - Expectation-step: Each data point assigned to nearest prototype
    - Maximization-step: Prototypes updated to be the cluster means



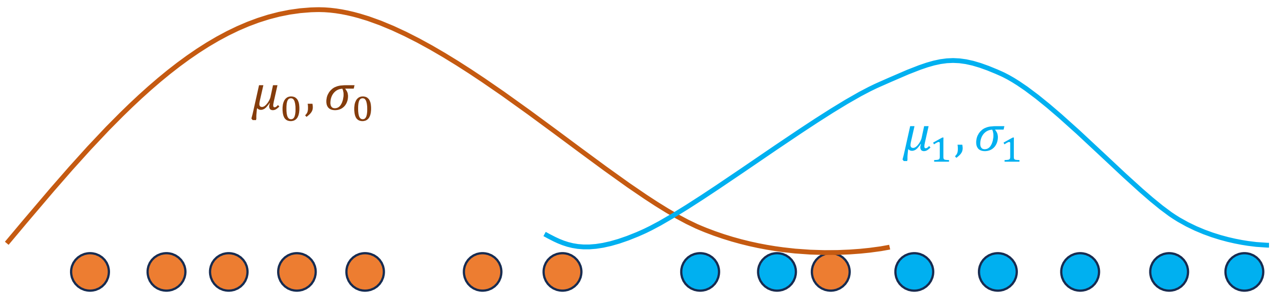
Limitations of K-Means

- Clusters are assumed to be spherical and equally sized, which is not valid in most real-world scenarios.
- K-Means is a hard clustering method, with each data point being assigned to a single cluster.
    - A small shift of a data point can flip it to a different cluster.
    - There is no uncertainty measure or probability that tells us how much a data point is associated with a specific cluster.
- Not clear on how to choose the value of K

- Probable solution is to replace the “hard” clustering of K-means with ‘soft’ probabilistic assignments
- The probability distribution of the data is represented as a Gaussian Mixture Model.

Understanding the Different Approaches

Scenario 1:
We have labels and we would like to model the appearance of the two classes –
Maximum Likelihood Estimation



- Assuming observed data points are generated independently, $D = \{x_1, x_2, \ldots, x_N\}$

$$\text{Likelihood} = p\left(x_1, \ldots, x_N; \mu_i, \sigma_i^2\right) = \prod_{n=1}^{N} \frac{1}{\sqrt{2\pi}\sigma_i} e^{-\frac{(x_n-\mu_i)^2}{2\sigma_i^2}}$$

where i is for Class 0 and Class 1 respectively

$$\text{Likelihood} = p(x_1, \ldots, x_N; \boldsymbol{\mu_i}, \Sigma_i) = \prod_{n=1}^{N} \frac{1}{(2\pi)^{\frac{d}{2}}|\Sigma_i|^{\frac{1}{2}}} e^{\left(-\frac{1}{2}(x_n-\boldsymbol{\mu_i})^T \Sigma_i^{-1} (x_n-\boldsymbol{\mu_i})\right)}$$

Gaussian Density Function

- We start with the Gaussian Density Function,

$$p(x_1, \dots, x_N; \boldsymbol{\mu_i}, \Sigma_i) = \mathcal{N}(\boldsymbol{x}|\boldsymbol{\mu}_i, \Sigma_i) = \prod_{n=1}^{N} \frac{1}{(2\pi)^{\frac{d}{2}}|\Sigma_i|^{\frac{1}{2}}} e^{\left(-\frac{1}{2}(\boldsymbol{x}-\mu_i)^T \Sigma_i^{-1} (\boldsymbol{x}-\mu_i)\right)}$$

- x_n represents our data points,
- d is the number of dimensions of each data sample (# of features)
- μ and Σ are the mean and covariance respectively.

- Say if you have a dataset with 1500 samples and 5 features, $N = 1500, d = 5$

$$x = 1500 \times 5, \quad \mu = 1 \times 5, \quad \Sigma = 5 \times 5 \text{ matrices}$$

Maximum Likelihood

- We set the parameters by maximizing the likelihood function, which we have discussed before is equivalent to maximizing the log likelihood

$$\text{Likelihood} = p(x_1, \dots, x_N; \boldsymbol{\mu}_i, \Sigma_i) = \prod_{n=1}^{N} \frac{1}{(2\pi)^{\frac{d}{2}} |\Sigma_i|^{\frac{1}{2}}} e^{\left(-\frac{1}{2}(x_n - \boldsymbol{\mu}_i)^T \Sigma_i^{-1} (x_n - \boldsymbol{\mu}_i)\right)}$$

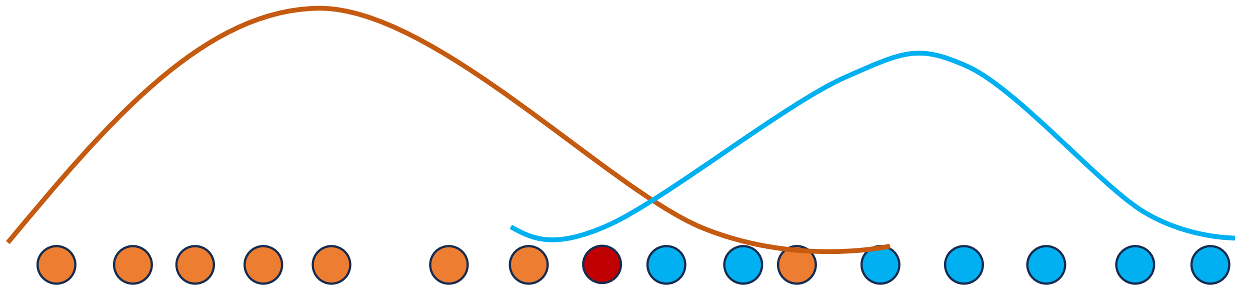
$$\ln p(x_1, \dots, x_N; \boldsymbol{\mu}_i, \Sigma_i) = -\frac{N}{2} \ln |\Sigma_i| - \frac{N}{2} d \ln(2\pi) - \frac{1}{2} \sum_{n=1}^{N} (x_n - \boldsymbol{\mu}_i)^T \Sigma_i^{-1} (x_n - \boldsymbol{\mu}_i)$$

- Maximizing w.r.t. the mean and the covariance gives the sample mean and sample covariance respectively

$$\boldsymbol{\mu}_i = \frac{1}{N} \sum_{n=1}^{N} x_n, \qquad \Sigma_i = \frac{1}{N} \sum_{n=1}^{N} (x_n - \boldsymbol{\mu}_i)(x_n - \boldsymbol{\mu}_i)^T$$

Understanding the Different Approaches

Scenario 2:
Once we have modelled the 2 classes, how do we compute the likelihood that a point belongs to Class 1 or Class 0 – **Probabilistic Inference**



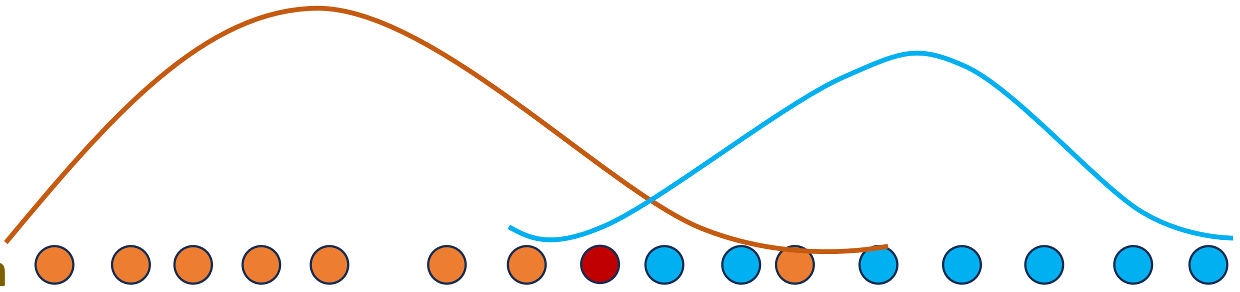
- We compute the likelihood that a particular model generated a sample – Bayes Theorem

$$p(C_0|x_j) = \frac{p(x_j|C_0)p(C_0)}{p(x_j|C_0)p(C_0) + p(x_j|C_1)p(C_1)}, \quad p(C_1|x_j) = \frac{p(x_j|C_1)p(C_1)}{p(x_j|C_0)p(C_0) + p(x_j|C_1)p(C_1)}$$

- Parameters of the Gaussian being known, we can compute the soft assignments of all the points.

Understanding the Different Approaches

Scenario 3:
How can we get the labels and the distribution of the classes both at once?
Expectation Maximization (EM) Algorithm



- We introduce the concept of Gaussian Mixture which is a function comprised of several Gaussians, depending on the number of clusters of our data.

- Each Gaussian is comprised of the following features:
  - Mean μ defining the centre of the Gaussian.
  - Covariance Σ defining the width of the Gaussian.
  - An additional mixing parameter π defining how big or small the Gaussian function will be in terms of probability, $\sum_{k=1}^{K} \pi_k = 1$, where K is the number of clusters.

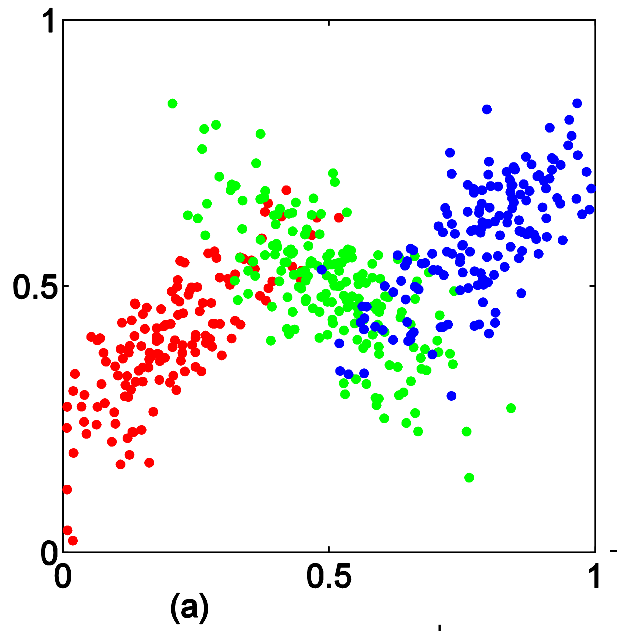
Gaussian Mixtures

- Linear super-position of Gaussians

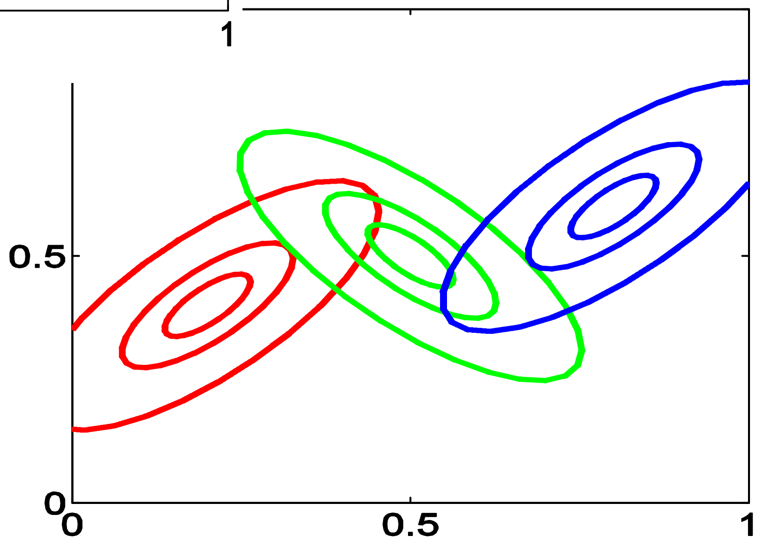
$$p(x) = \sum_{k=1}^{K} \pi_k \mathcal{N}(x|\mu_k, \Sigma_k)$$

- Normalization and positivity require

$$\sum_{k=1}^{K} \pi_k = 1, \qquad 0 \leq \pi_k \leq 1$$



(a)



Understanding Mixing Parameter

- The mixing coefficient for a Gaussian is the overall probability of observing a point x_n that comes from Gaussian k

- Given a data point x_n, what is the probability it comes from a Gaussian distribution k

$$p(x_n) = \sum_{k=1}^{K} \pi_k \mathcal{N}(x_n | \boldsymbol{\mu}_k, \Sigma_k)$$

- To determine the optimal values for $\boldsymbol{\mu}_k, \pi_k, \Sigma_k$ we need to determine the maximum likelihood of the model.

$$p(X) = \prod_{n=1}^{N} p(x_n) = \prod_{n=1}^{N} \sum_{k=1}^{K} \pi_k \mathcal{N}(x_n | \mu_k, \Sigma_k)$$

Understanding Mixing Parameter

- Applying the logarithm to both side of the equation, like before:

$$\ln p(X) = \sum_{n=1}^{N} \ln \sum_{k=1}^{K} \pi_k \mathcal{N}(x_n | \boldsymbol{\mu}_\mathbf{k}, \Sigma_k)$$

- Differentiating the above equation with respect to the parameters $\boldsymbol{\mu}_\mathbf{k}, \pi_k, \Sigma_k$ is not straightforward.

- The iterative method, Expectation – Maximization is used to compute the above equation.

Expectation – Maximization Algorithm

- Step 1: Decide the number of clusters and initialize the mean, covariance, and mixing parameter ($\boldsymbol{\mu}_k, \Sigma_k, \pi_k$) per cluster.

  - The mixing parameter is assumed to be equal for all the clusters in the beginning, say if $K = 3, \pi = \left\{\frac{1}{3}, \frac{1}{3}, \frac{1}{3}\right\}$.

- Step 2: **Expectation Step (E step)** – For each sample data x_i, we calculate the probability that the data point belongs to cluster k. [**Scenario 2**]

$$p(x_{ik}) = \frac{\pi_k \mathcal{N}(x_i | \boldsymbol{\mu}_k, \Sigma_k)}{\sum_{k=1}^{K} \pi_k \mathcal{N}(x_i | \boldsymbol{\mu}_k, \Sigma_k)}$$

Expectation – Maximization Algorithm

- Step 2: The E-step computes the probabilities using the current estimates of the model's parameters. It measures how much the kth Gaussian distribution is responsible for the generation of the ith data point.

- Step 3: **Maximization Step (M step)** – The algorithm uses the measurement obtained for each data point and each Gaussian distribution in the E-step to update the estimates of the model parameters. [**Scenario -1**]

$$\pi_k = \frac{\sum_{i=1}^{N} p(x_{ik})}{N}, \qquad \boldsymbol{\mu}_k = \frac{\sum_{i=1}^{N} p(x_{ik}) x_i}{\sum_{i=1}^{N} p(x_{ik})}, \qquad \Sigma_k = \frac{\sum_{i=1}^{N} p(x_{ik})\,(x_i - \boldsymbol{\mu}_k)^2}{\sum_{i=1}^{N} p(x_{ik})}.$$

Expectation – Maximization Algorithm

- Step 2: Expectation Step (E-step)

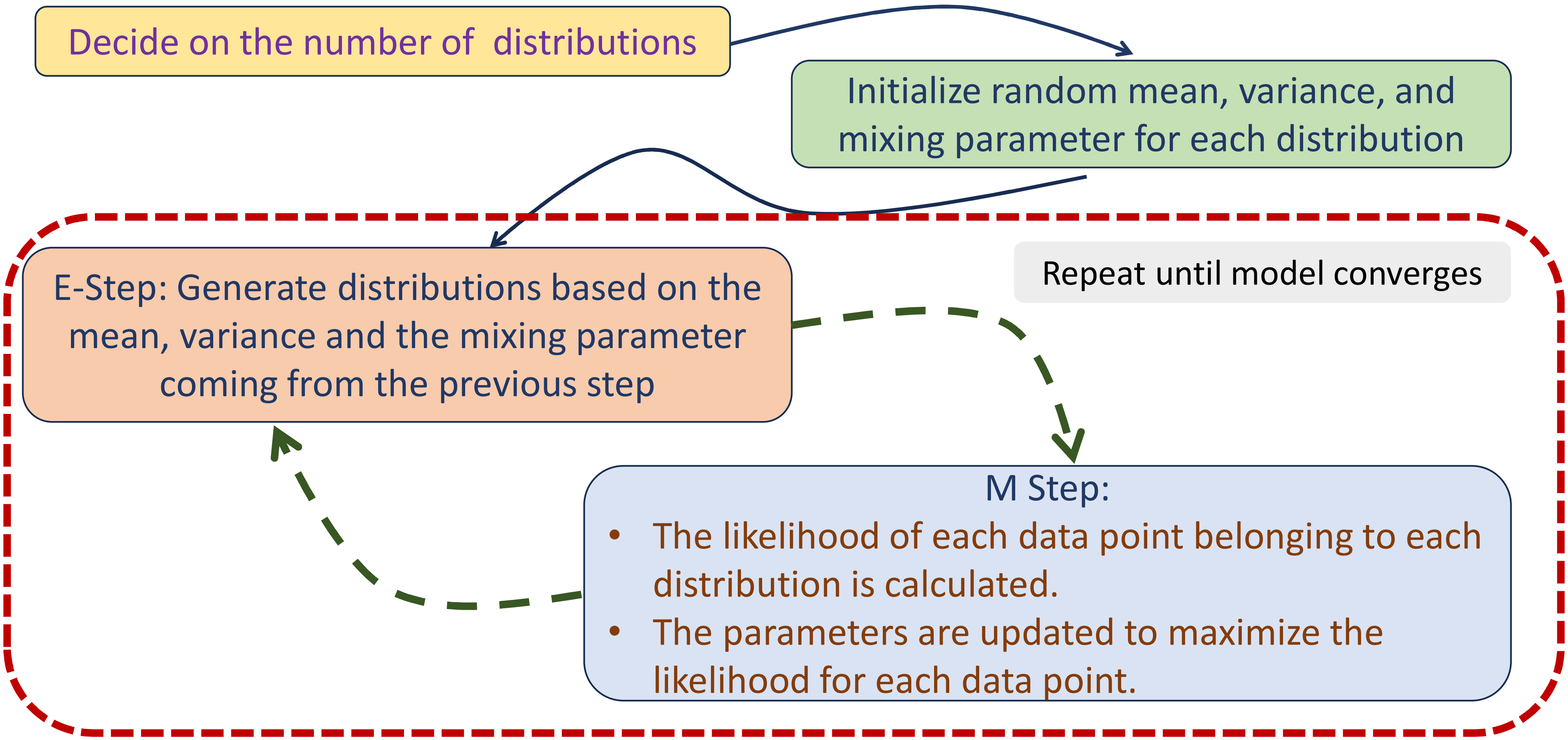
- Step 3: Maximization Step (M- step)

- Step 4: The updated estimates $\boldsymbol{\mu}_{k}, \Sigma_{k}, \pi_{k}$ are used in the next E-step [Step -2] to compute new responsibilities for the data points.

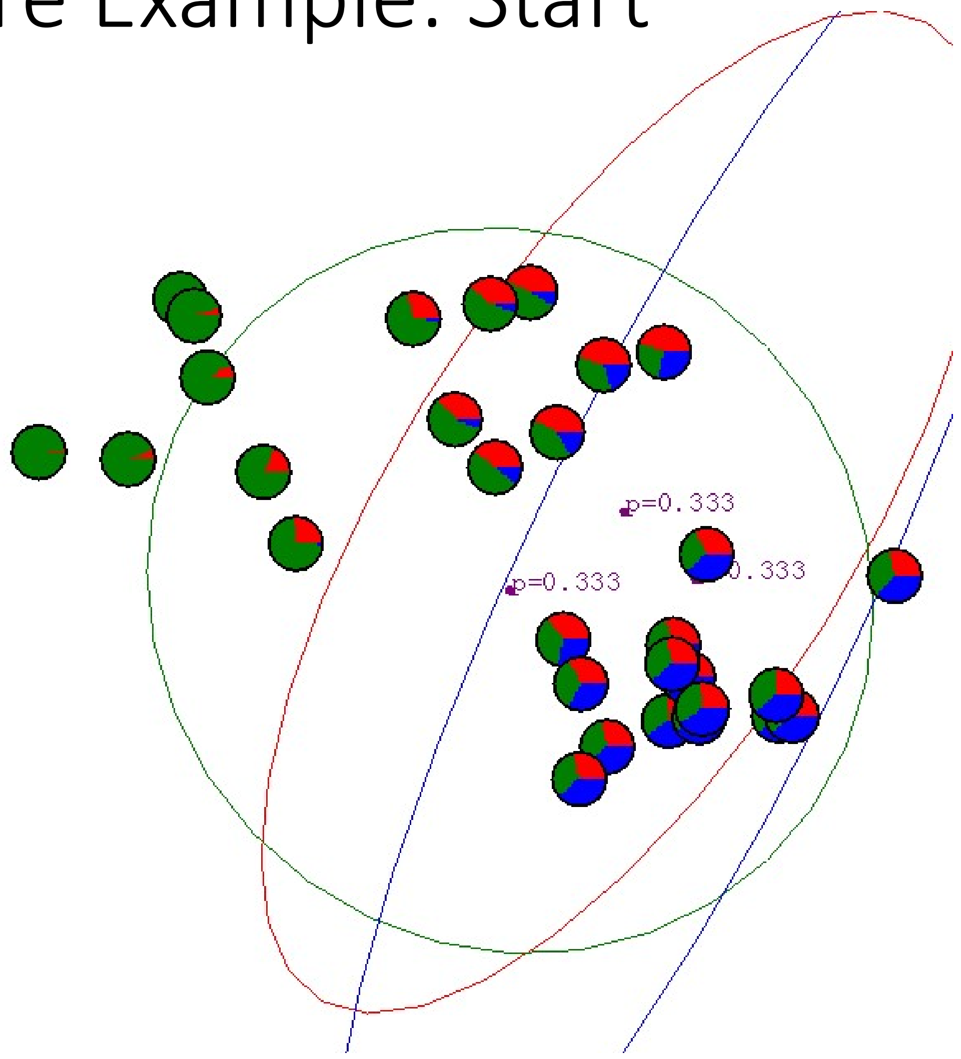
- The process repeats iteratively until convergence, i.e., the model parameters do not change significantly from one iteration to the next.

- Each step increases the log-likelihood of our model

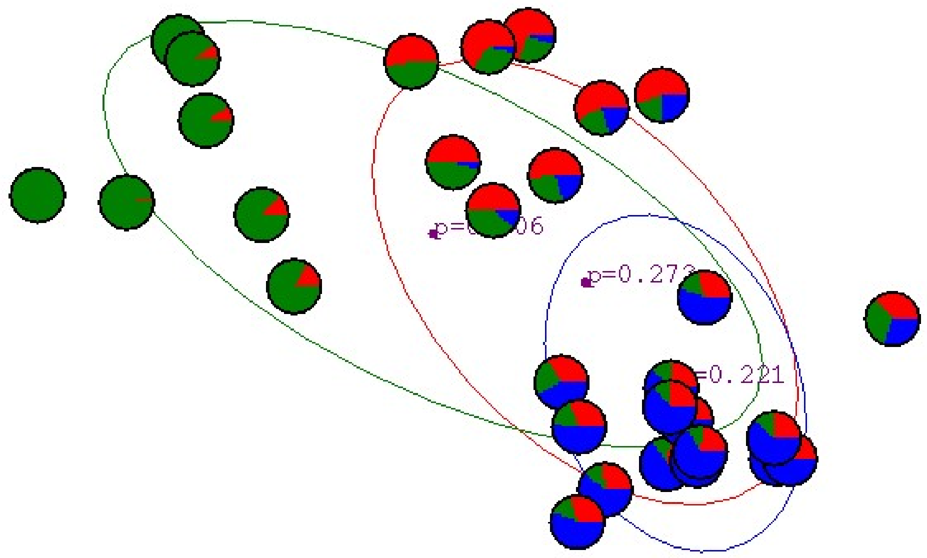
Flowchart


Decide on the number of distributions
Initialize random mean, variance, and mixing parameter for each distribution
E-Step: Generate distributions based on the mean, variance and the mixing parameter coming from the previous step
Repeat until model converges
M Step:
• The likelihood of each data point belonging to each distribution is calculated.
• The parameters are updated to maximize the likelihood for each data point.

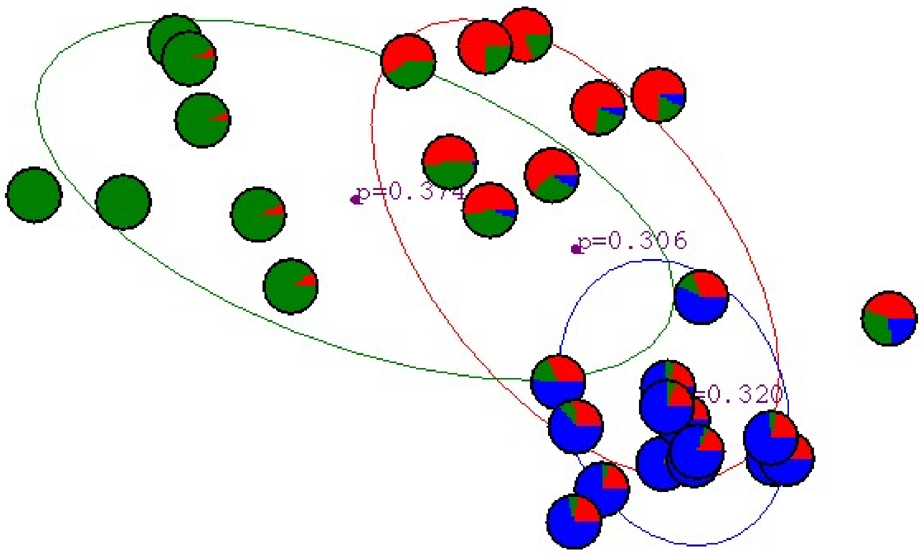
Gaussian Mixture Example: Start



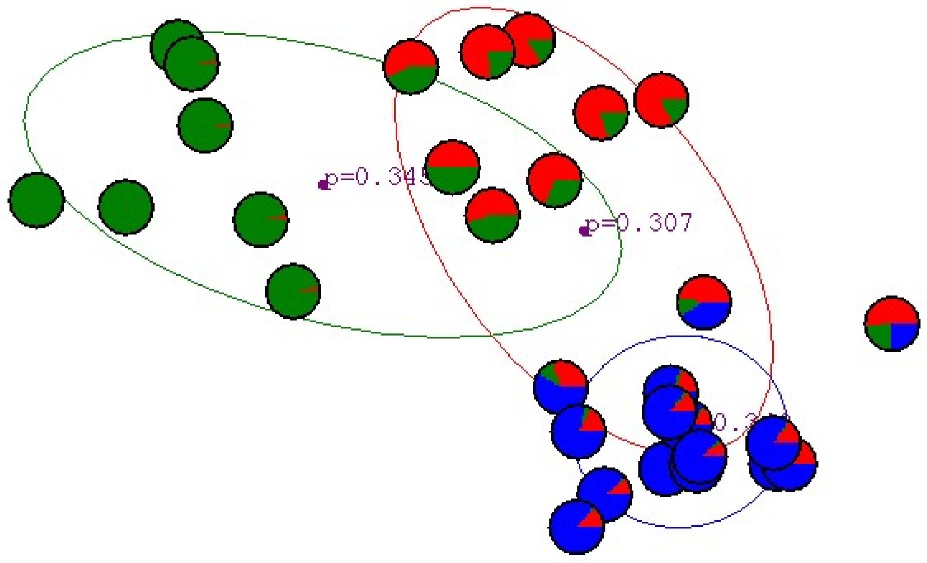
After first iteration



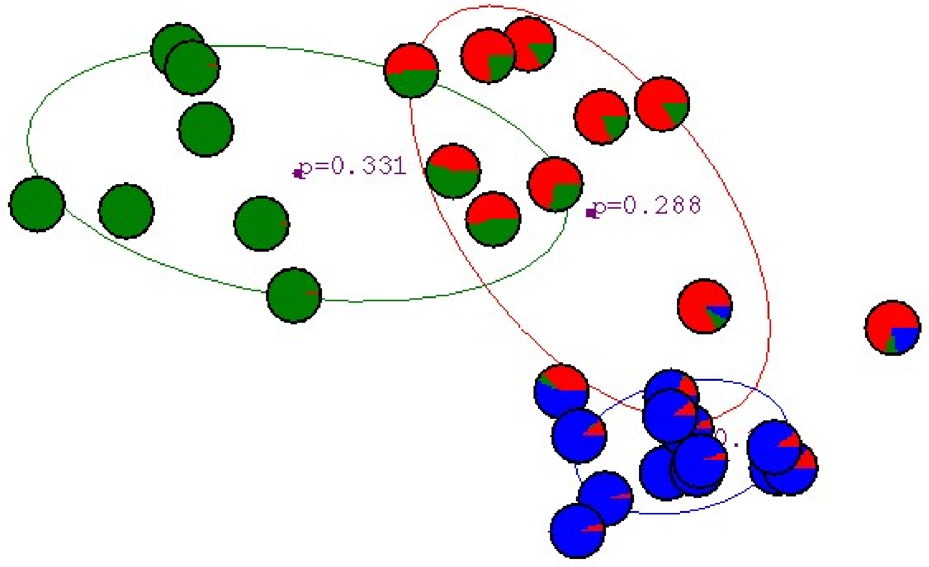
After 2nd iteration



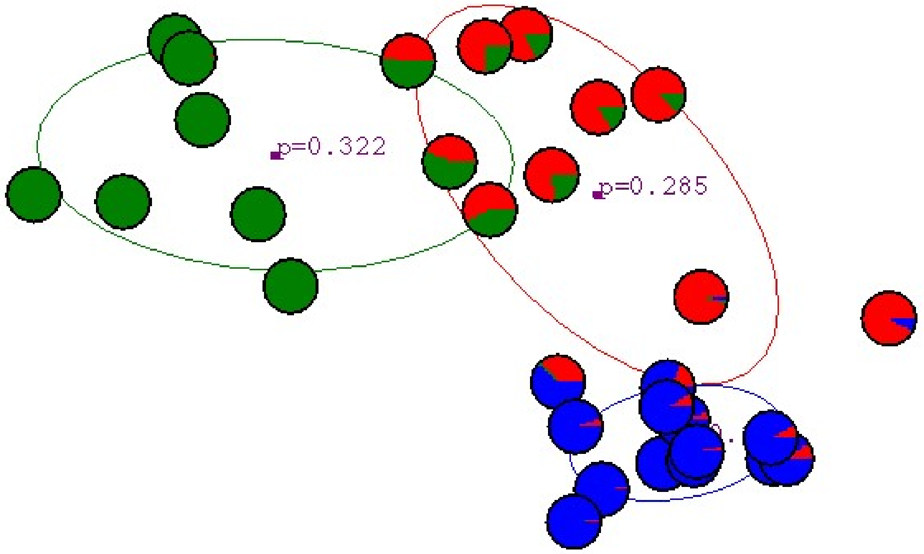
After 3rd iteration



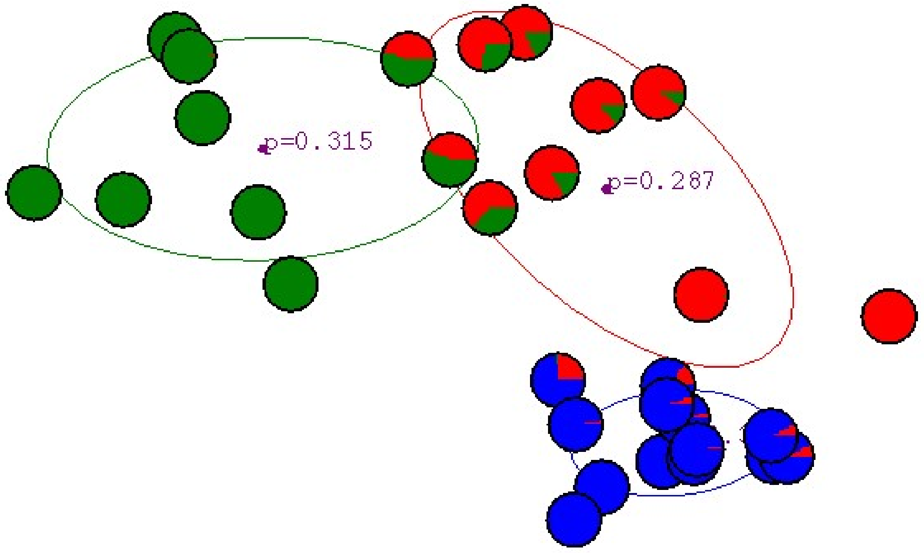
After 4th iteration



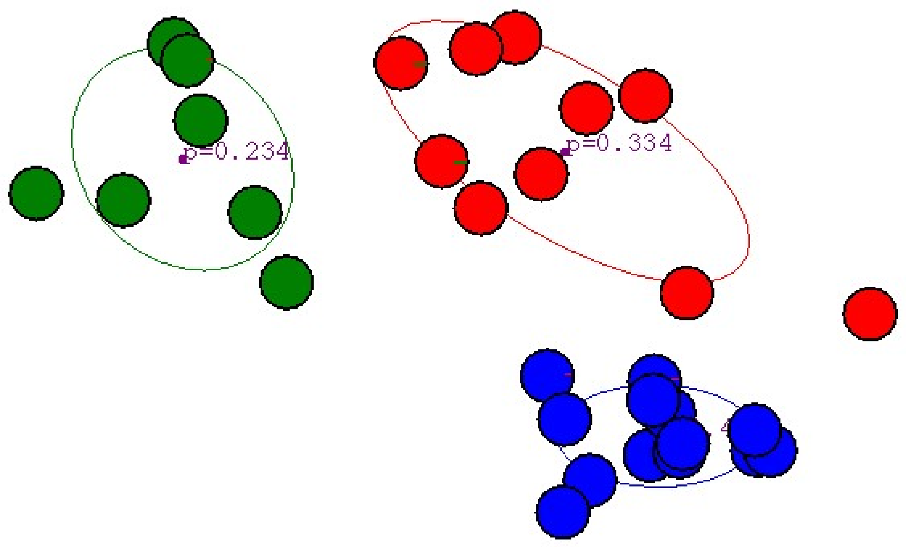
After 5th iteration



After 6th iteration

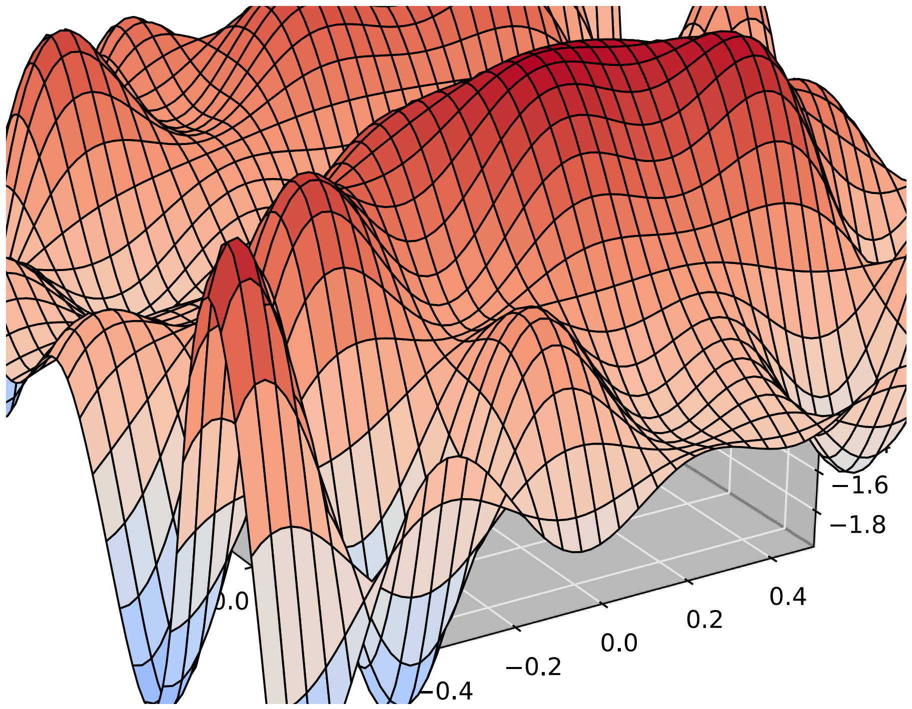


After 20th iteration



Properties of EM

- EM is trying to optimize a nonconvex function
- But EM is a local optimization algorithm
- Typical solution: Random Restarts
  - Just like K-Means, we run the algorithm many times
  - Each time initialize parameters randomly
  - Pick the parameters that give highest likelihood



Pros and Cons for EM

- **Pros:**
  - no learning rate (step-size) parameter
  - automatically enforces parameter constraints
  - very fast for low dimensions
  - each iteration guaranteed to improve likelihood

- **Cons:**
  - can get stuck in local minima
  - Sensitivity to Initialization
  - Computational Complexity
  - Choosing the optimal number of clusters