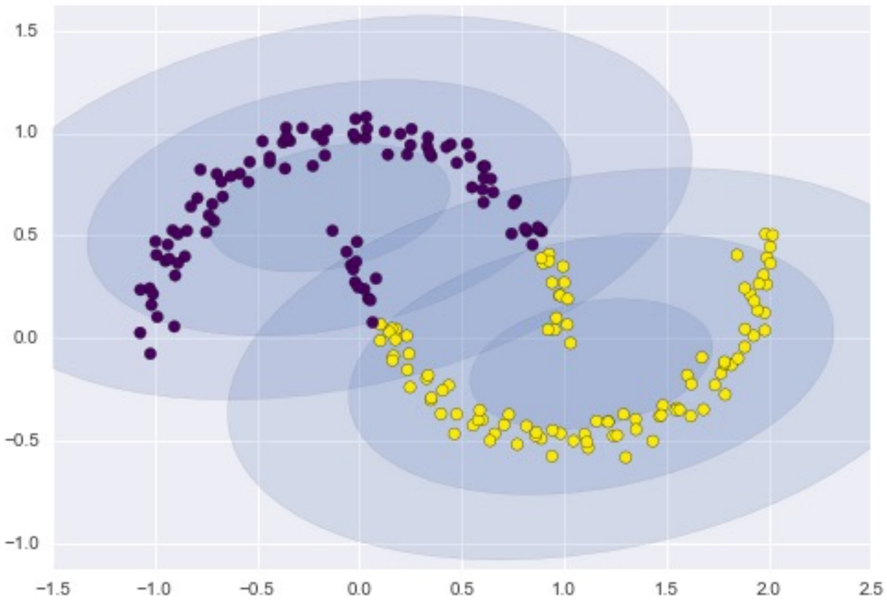
Gaussian Mixture Models (GMM)

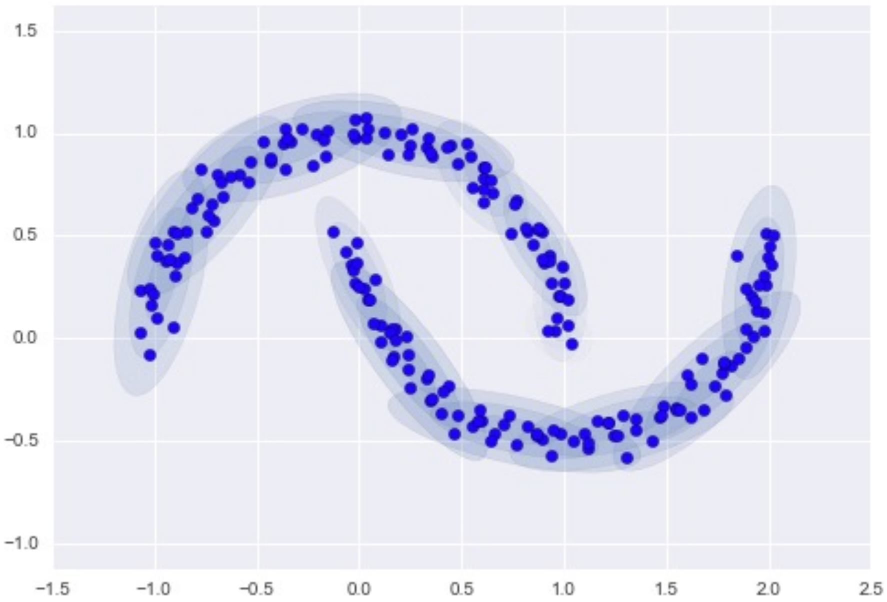
ECE57000: Artificial Intelligence

David I. Inouye

Gaussian mixture models (GMM) can be used for density estimation

1. General density estimation

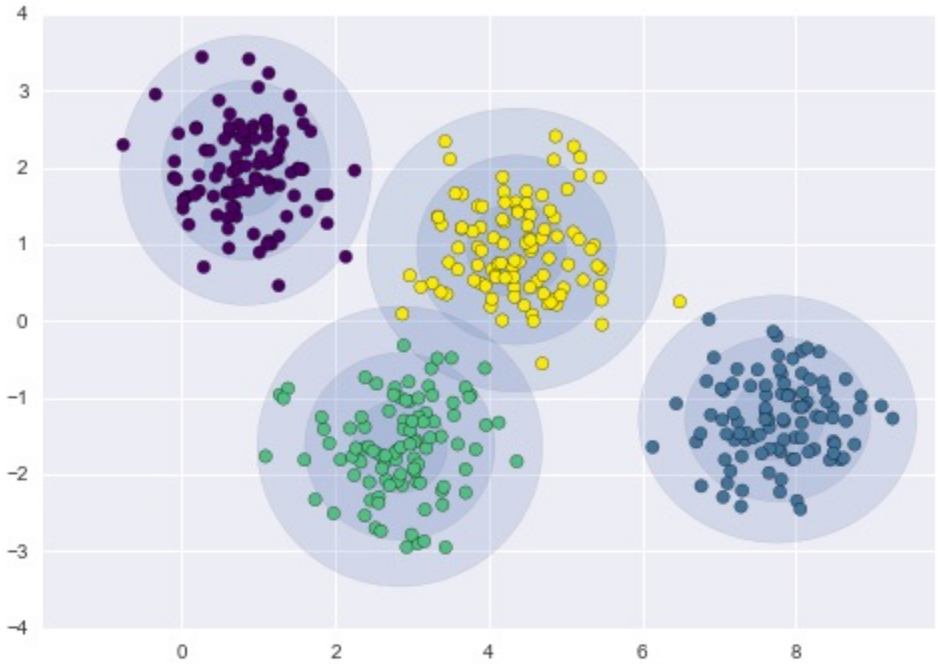




https://jakevdp.github.io/PythonDataScienceHandbook/05.12-gaussian-mixtures.html

Even if each component distribution is independent, the mixture may **not** be independent

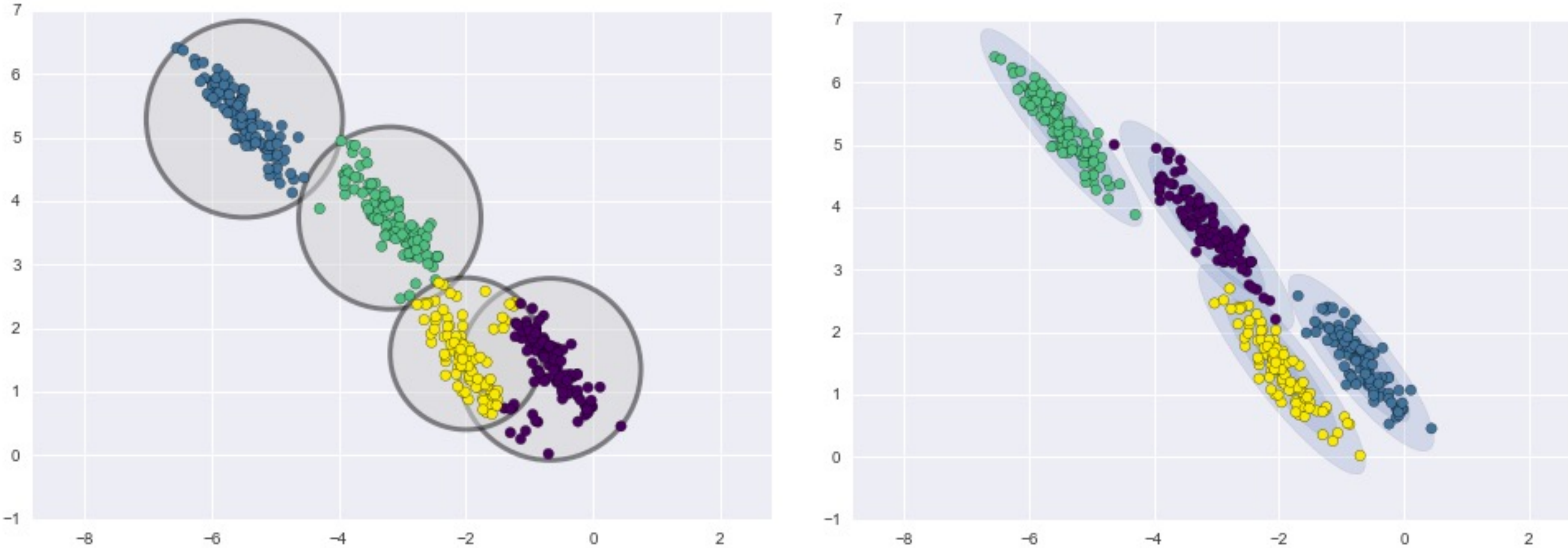
- Each component distribution is spherical (i.e., independent)



https://jakevdp.github.io/PythonDataScienceHandbook/05.12-gaussian-mixtures.html

<u>Gaussian mixture models</u> (GMM) can be used for flexible clustering

2. Flexible clustering



https://jakevdp.github.io/PythonDataScienceHandbook/05.12-gaussian-mixtures.html

Overview of Gaussian Mixture Models

- Introduction
- Mixture model definitions
  - Simple average density
  - Equivalent latent variable formulation
- EM Algorithm for GMMs
  - Expectation step
  - Maximization step
- Derivation of EM algorithm's monotonic increase of log likelihood

Mixture distributions are weighted averages of component distributions

- Mixture distribution
  - Component weights $0 \leq \pi_j, \leq 1$ s.t. $\sum_{j=1}^{k} \pi_j = 1$
  - Component distributions $p_j(x)$
- Simple form of mixture

$$p_{\text{mixture}}(x) = \sum_{j=1}^{k} \pi_j p_j(x)$$

- Exercise: Check that p_{mixture} integrates to 1.

Mixture models can be viewed as <u>latent (or "hidden") variable models</u>

- Simple form of mixture

$$p_{\text{mixture}}(x) = \sum_{j=1}^{k} \pi_j p_j(x)$$

- Let $z \in \{1, \dots, k\}$ be an *auxiliary* **<u>indicator variable</u>**

- Let $p(z = j) = \pi_j$, then the joint density model is:

$$p(x, z) = p(z)p(x|z)$$

- The distribution of x marginalizes over the latent variable z which is equivalent to the mixture above

$$p_{\text{mixture}}(x) \equiv \sum_{z} p(x, z) = \sum_{z} p(z)p(x|z)$$

<u>Gaussian mixture models (GMM)</u> are one of the most common mixture distributions

- Form of Gaussian mixture model

$$p_{\mathrm{GMM}}(x) = \sum_{j=1}^{k} \pi_j p_{\mathcal{N}}(x; \mu_j, \Sigma_j) = \sum_{j=1}^{k} p(z=j) p_{\mathcal{N}}(x; z=j)$$

Machine Learning, Murphy, 2012.

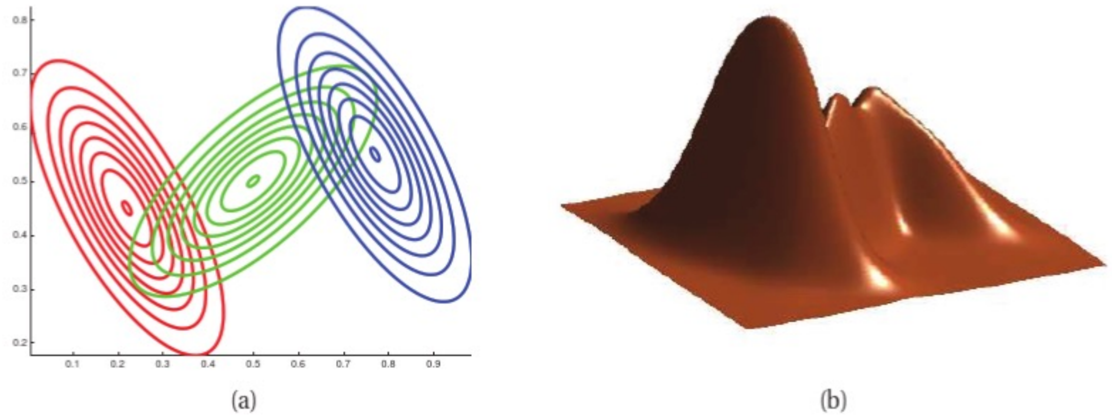


Figure 11.3 A mixture of 3 Gaussians in 2d. (a) We show the contours of constant probability for each component in the mixture. (b) A surface plot of the overall density. Based on Figure 2.23 of (Bishop 2006a). Figure generated by `mixGaussPlotDemo`.

MLE for mixtures is difficult
Reason 1: The algebraic form is more complex

- The mixture log likelihood cannot be simplified

$$\arg\max_{\pi,\mu_j,\Sigma_j} \log \prod_i p_{\text{GMM}}(x_i; \pi, \mu_1, \dots, \mu_k, \Sigma_1, \dots, \Sigma_k)$$

$$\sum_i \log p_{\text{GMM}}(x_i; \pi, \mu_1, \dots, \mu_k, \Sigma_1, \dots, \Sigma_k)$$

$$\sum_i \log \sum_{z_i} \pi_{z_i} p_{\mathcal{N}}\left(x_i \mid \mu_{z_i}, \Sigma_{z_i}\right)$$

$$\sum_i \log \sum_{z_i} p(z_i) p_{\mathcal{N}}(x_i | z_i)$$

- **Cannot exchange log and summation to cancel exp**

MLE for mixtures is difficult
Reason 2: Problem is non-convex
(and could have multiple local optima)

- The intuition is similar to the problem with k-means clustering

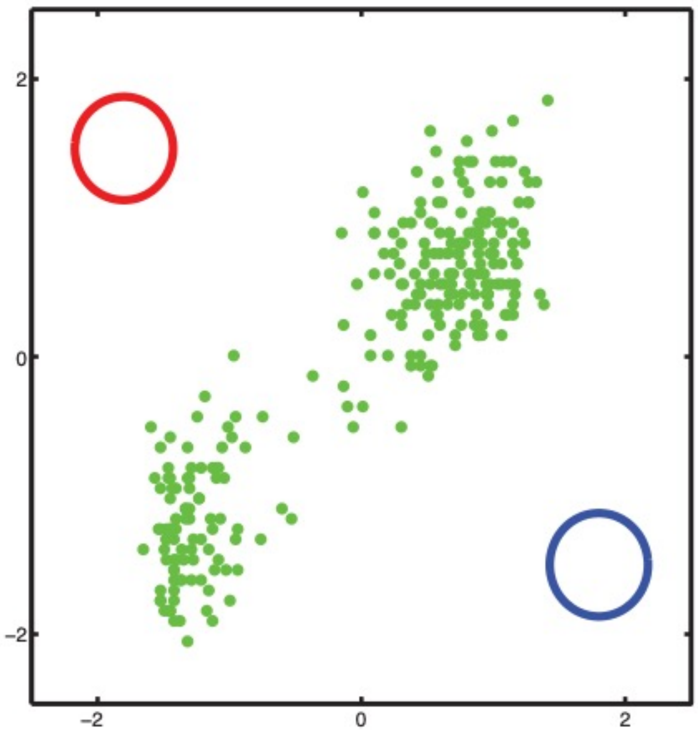
See [ML, Ch. 11, pp. 347-348] for more detailed analysis.

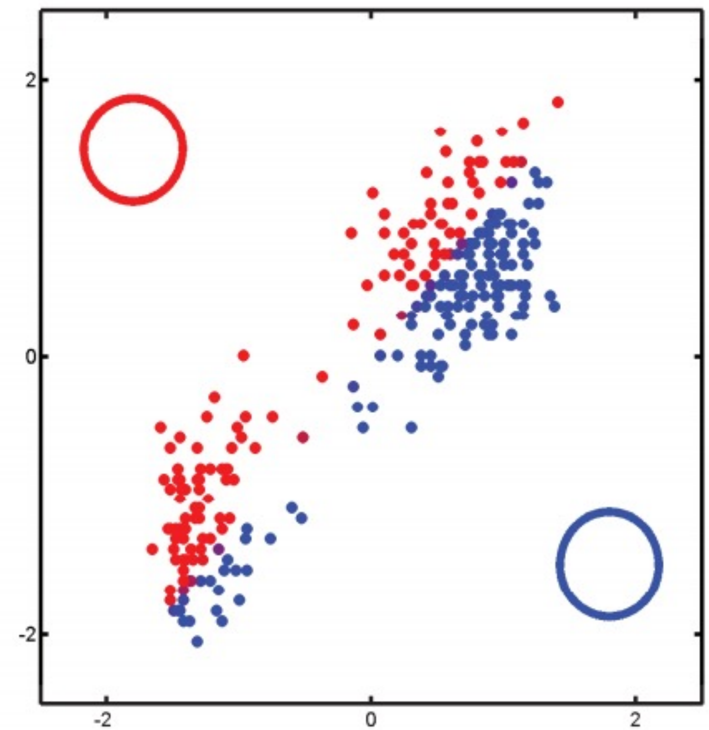
The Expectation-Maximization (EM) can estimate models and is a generalization of k-means

- The EM algorithm for GMM alternates between
  - Probabilistic/soft assignment of points
  - Estimation of Gaussian for each component

- Similar to k-means which alternates between
  - Hard assignment of points
  - Estimation of mean of points in each cluster

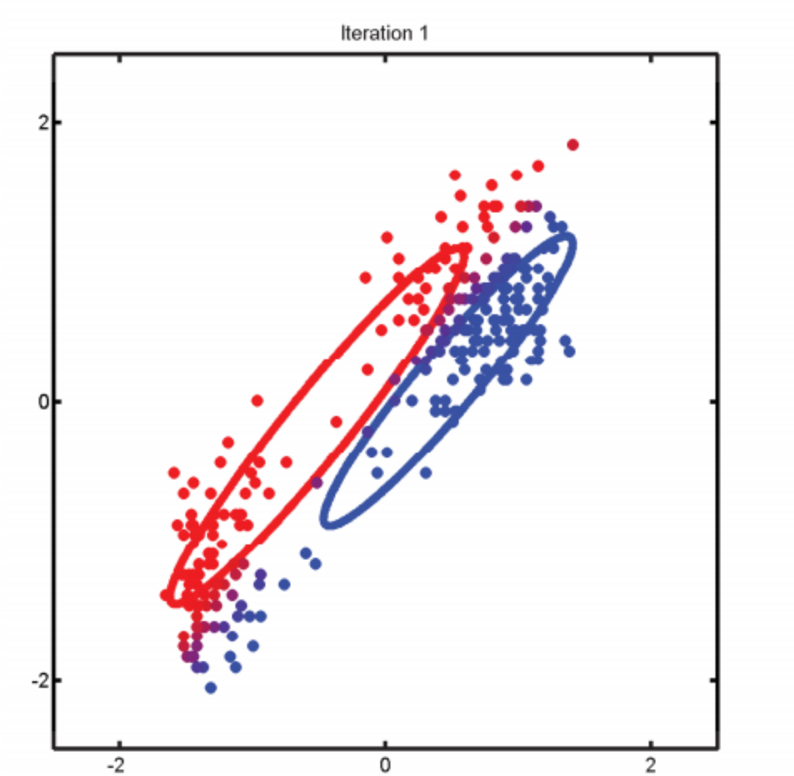
EM Algorithm: Initialization

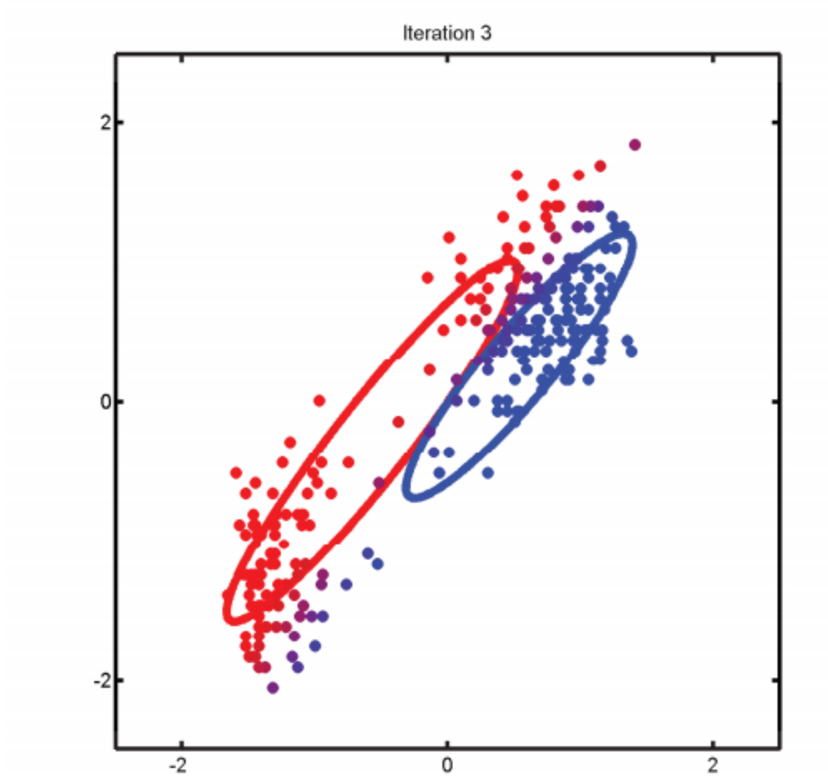




Machine Learning: A probabilistic perspective, Murphy, 2012.

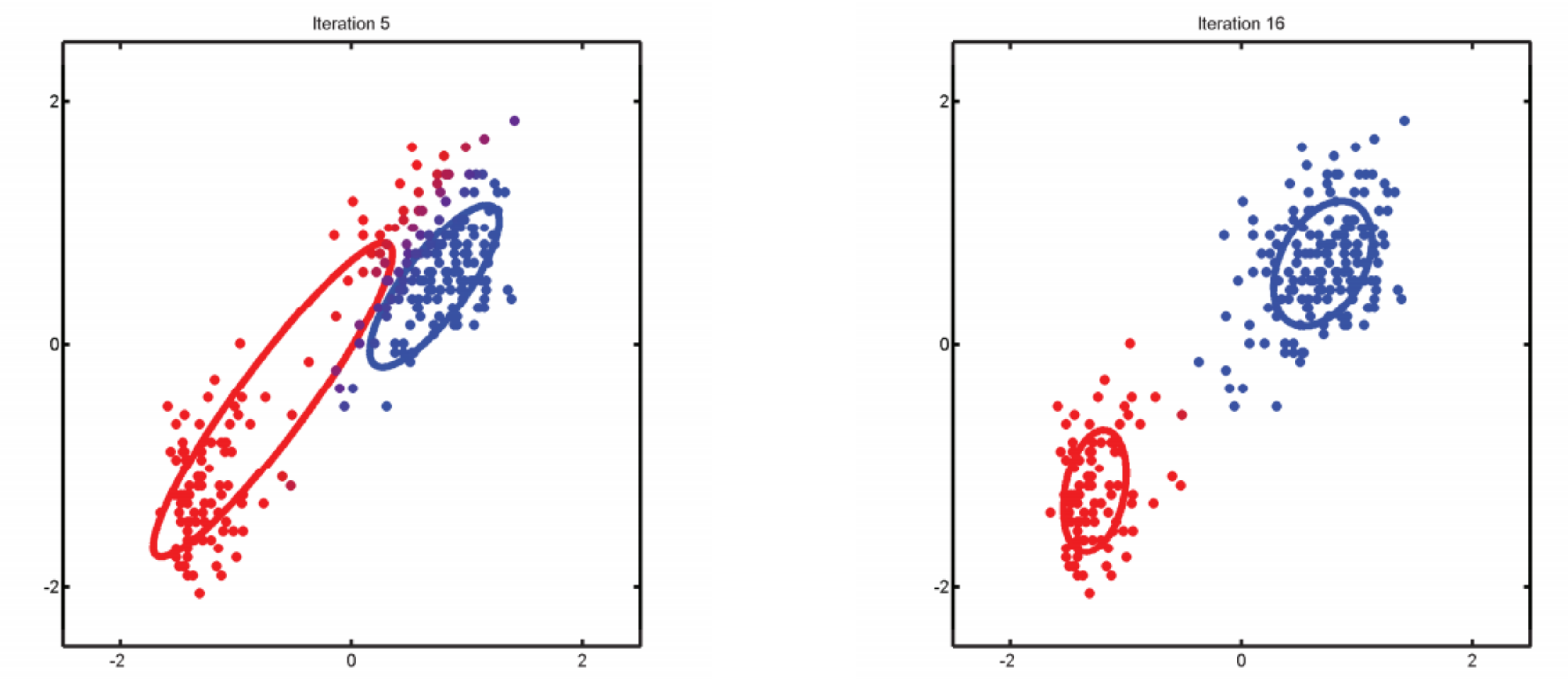
EM Algorithm: Iteration 1 and 3





Machine Learning: A probabilistic perspective, Murphy, 2012.

EM Algorithm: Iteration 5 and 16



Machine Learning: A probabilistic perspective, Murphy, 2012.

EM algorithm for Gaussian mixture models

Expectation step

- Randomly initialize mixture components
- Expectation step (determine soft assignments)
- $r_{ij}^{t} = p(z_i = j | x_i, \theta^{t-1})$
- $= \frac{p\left(z_i, x_i | \theta^{t-1}\right)}{p(x_i | \theta^{t-1})}$
- $= \frac{p\left(z_i | \theta^{t-1}\right) p\left(x_i \mid z_i, \theta^{t-1}\right)}{\sum_{z_i} p(z_i | \theta^{t-1}) p(x_i \mid z_i, \theta^{t-1})}$
- $= \frac{\pi_j p_{\mathcal{N}}\left(x_i \middle| \mu_j^{t-1}, \Sigma_j^{t-1}\right)}{\sum_k \pi_k p_{\mathcal{N}}\left(x_i \middle| \mu_k^{t-1}, \Sigma_k^{t-1}\right)}$

EM algorithm for Gaussian mixture models
Maximization step

- Compute weighted mean and covariance using soft assignments from E step

$$\mu_j^t = \frac{\sum_i r_{ij}\, x_i}{\sum_i r_{ij}}$$

$$\Sigma_j^t = \frac{\sum_i r_{ij}(x_i - \mu_j^t)(x_i - \mu_j^t)^T}{\sum_i r_{ij}}$$

Derivation of EM algorithm

- **Observed/marginal** log likelihood (if z_i is **unknown**)

$$\ell(\theta) = \sum_i \log \sum_{z_i} p(x_i, z_i; \theta)$$

- If z_i were **observed** (i.e., we knew the cluster labels), then optimizing the complete log likelihood is easy
- **Complete** log likelihood (if z_i is **known**)

$$\ell_c(\theta) = \sum_i \log p(x_i, z_i; \theta) = \sum_i \log p(z_i) p_{\mathcal{N}}(x_i \mid z_i)$$

  - For GMMs, this is convex and easy to solve

Derivation of EM algorithm

- Complete log-likelihood

$$\ell_c(\theta) = \sum_i \log p(x_i, z_i | \theta)$$

- Q function: Expected complete log likelihood

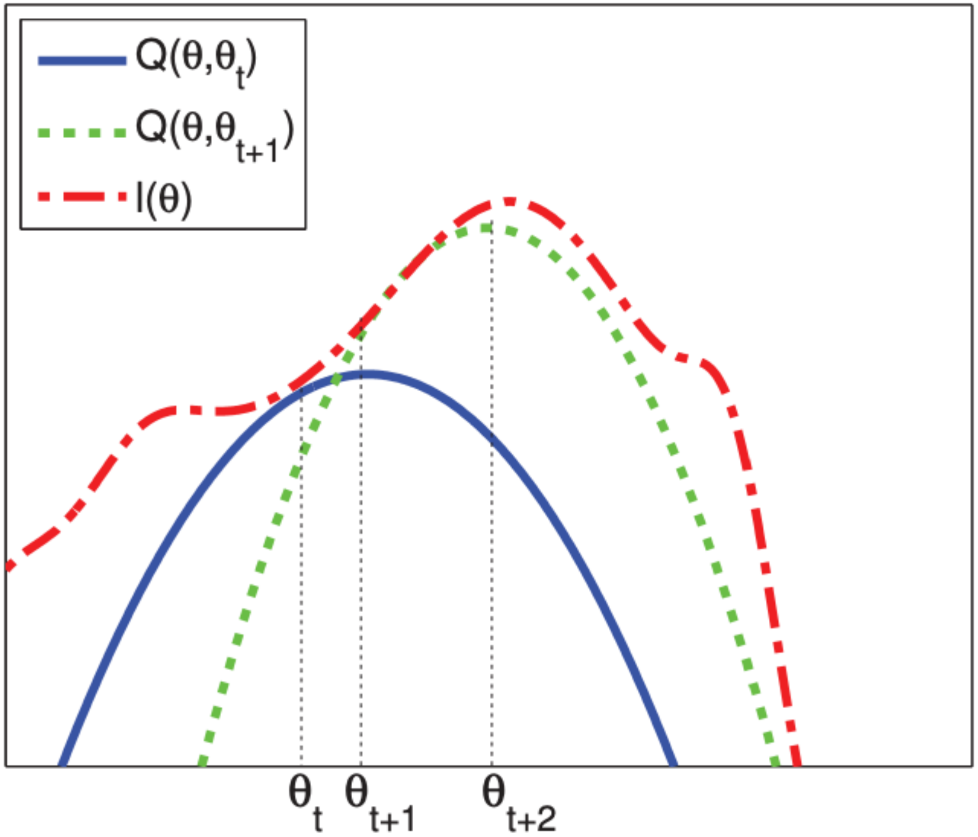
$$Q(\theta; \theta^{t-1}) = Q_{\theta^{t-1}}(\theta) = \mathbb{E}_{\boldsymbol{z}_{\dots}|\boldsymbol{x}_{\dots},\theta^{t-1}}[\ell_c(\theta)]$$

  - **NOTE:** Q is a function of θ **given** the previous parameter value θ^{t-1}
- EM iteration
  - E-step: Form $Q(\theta; \theta^{t-1})$ around θ^{t-1}
  - M-step: $\theta^t = \max_\theta Q(\theta; \theta^{t-1})$
- See 11.4 of [ML] for derivation of EM steps
  - Hint: Write the joint density of x and z as:

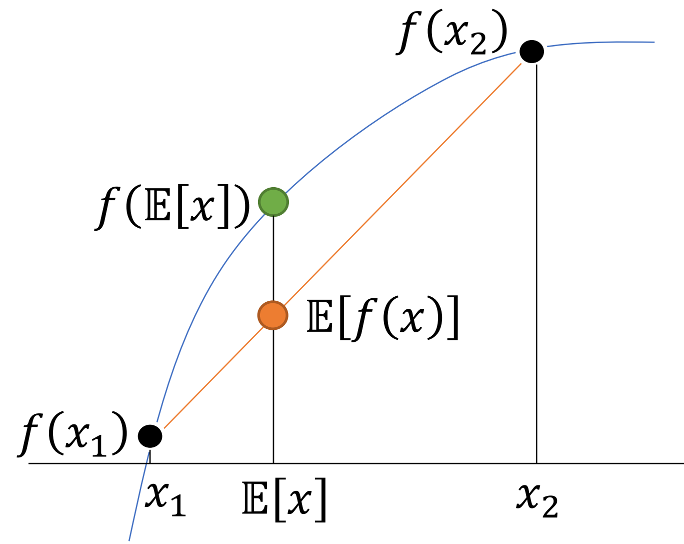
$$p(x_i, z_i | \theta) = \prod_j \left(\pi_j p\left(x_i \middle| \theta_j\right)\right)^{I(z_i = j)}$$

    - $I(z_i = j)$ is an indicator function that is 1 if the inside expression is true or 0 otherwise

EM Theory: EM algorithm is **guaranteed** to increase ***observed*** likelihood, i.e., $\prod_i p_{mixture}(x_i)$



Step 1: Use Jensen's inequality to get concave lower bound



- Jensen's inequality if f is *concave* (e.g., log)

$$f(\mathbb{E}[x]) \geq \mathbb{E}[f(x)]$$

- $\ell(\theta)$
- $= \sum_i \log \sum_{z_i} p(x_i, z_i; \theta)$
- $= \sum_i \log \sum_{z_i} q_i(z_i) \frac{p(x_i,z_i;\theta)}{q_i(z_i)}$
- $= \sum_i \log \mathbb{E}_{q_i}\left[\frac{p(x_i,z_i;\theta)}{q_i(z_i)}\right]$
- $\geq \sum_i \mathbb{E}_{q_i}\left[\log \frac{p(x_i,z_i;\theta)}{q_i(z_i)}\right]$
- $\equiv Q_i(\theta; q)$ for **any** distribution $q = (q_1, \cdots, q_n)$

Just theoretical inequality here.
No algorithm required to ensure this.
Next two steps require EM algorithm.

Step 2: Choose **best** lower bound using the current parameters (for each point x_i)

- $Q_i(\theta, q_i) = \sum_{z_i} q_i(z_i) \log \frac{p(x_i, z_i; \theta)}{q_i(z_i)}$
- $= \sum_{z_i} q_i(z_i) \log \frac{p(x_i\,; \theta) p(z_i\,| x_i\,; \theta)}{q_i(z_i)}$
- $= \sum_{z_i} q_i(z_i) \log \frac{p(z_i\,| x_i\,; \theta)}{q_i(z_i)} + \sum_{z_i} q_i(z_i) \log p(x_i\,;\ \theta)$
- $= \sum_{z_i} q_i(z_i) \log \frac{p(z_i\,| x_i\,; \theta)}{q_i(z_i)} + \log p(x_i\,;\ \theta)$
- $= -\sum_{z_i} q_i(z_i) \log \frac{q_i(z_i)}{p(z_i\,| x_i\,; \theta)} + \log p(x_i\,;\ \theta)$
- $= -KL(q_i(\cdot), p(\cdot | x_i\,; \theta)) + \log p(x_i\,;\ \theta)$
- Maximize $Q_i(\theta, q_i)$ so ideally, $q_i(z_i) = p(z_i | x_i, \theta\ )$ so KL is 0
- Computing $p(z_i | x_i, \theta^t)$ is the E-step in the EM algorithm

Step 2: Lower bound is tight at current parameters θ^t if $q_i^t(z_i) = p(z_i|x_i, \theta^t)$ (E-step)

- The lower bound is **tight** with respect to the observed likelihood:
- $Q(\theta^t, \theta^t) = \sum_i Q_i(\theta^t, q^t)$
- $= \sum_i -KL\left(q_i^t(z_i), p(z_i|x_i\,; \theta^t)\right) + \log p(x_i\,;\ \theta^t)$
- $= \sum_i -KL(p(z_i|x_i\,; \theta^t), p(z_i|x_i\,; \theta^t)) + \log p(x_i\,;\ \theta^t)$
- $= \sum_i \log p(x_i|\theta^t)$
- $= \ell(\theta^t)$
- Last step: KL is 0
- In summary:

$$Q(\theta^t, \theta^t) = \ell(\theta^t)$$

The E-step ensures this equality
where $\mathrm{r}_{i,z_i} = p(z_i|x_i\,; \theta^t)$
And $q_i^t(z_i) = p(z_i|x_i\,; \theta^t)$

Step 3: Maximize the lower bound (M-step)

- We setup the optimization problem to update the parameter based on the lower bound

$$\theta^{t+1} = \arg\max_{\theta} Q(\theta, \theta^t)$$

- By simple definition of maximization, we have:

$$Q(\theta^{t+1}, \theta^t) \geq Q(\theta^t, \theta^t)$$

Putting all the steps together, we can prove monotonic increase of the EM algorithm

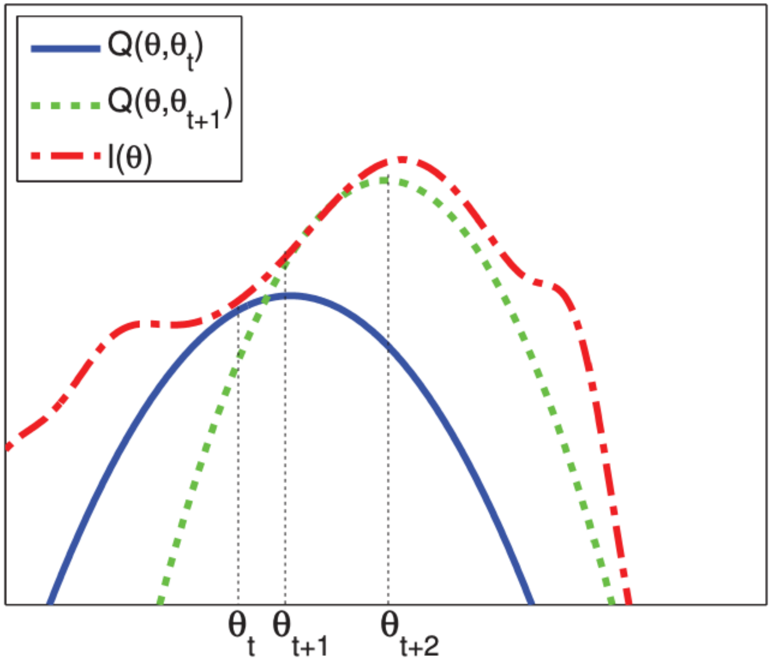
- **Lower bound, maximization, tightness**

$$\ell(\theta^{t+1}) \geq Q(\theta^{t+1}, \theta^t) \geq Q(\theta^t, \theta^t) = \ell(\theta^t)$$

Step 1: Lower bound (Jensen's Inequality)

Step 3: Maximization inequality (M step)

Step 2: Tightness of bound (E step)



See 11.4.7 in [ML] for full derivation of theoretical proof