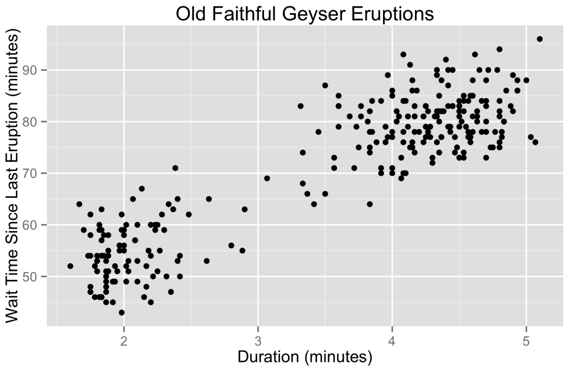
K-Means and Gaussian Mixture Models

David Rosenberg

New York University

June 15, 2015

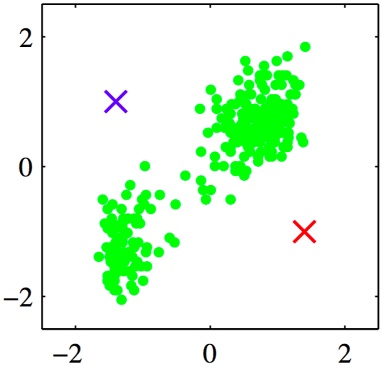
Example: Old Faithful Geyser



- Looks like two clusters.
- How to find these clusters algorithmically?

k-Means: By Example

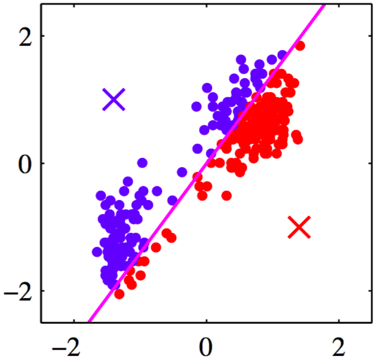
- Standardize the data.
- Choose two cluster centers.



From Bishop's *Pattern recognition and machine learning*, Figure 9.1(a).

k-means: by example

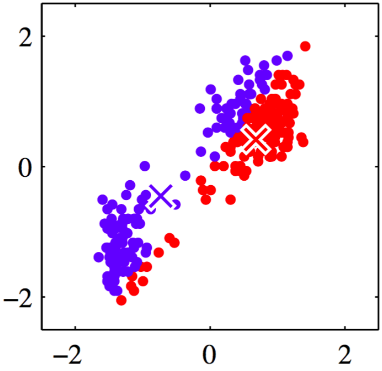
- Assign each point to closest center.



From Bishop's *Pattern recognition and machine learning*, Figure 9.1(b).

k-means: by example

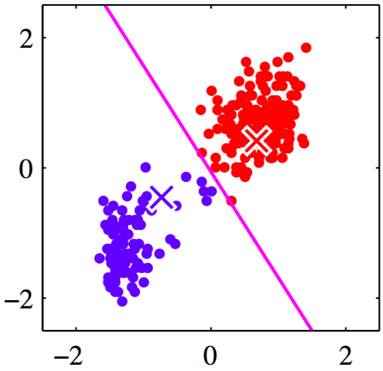
- Compute new class centers.



From Bishop's *Pattern recognition and machine learning*, Figure 9.1(c).

k-means: by example

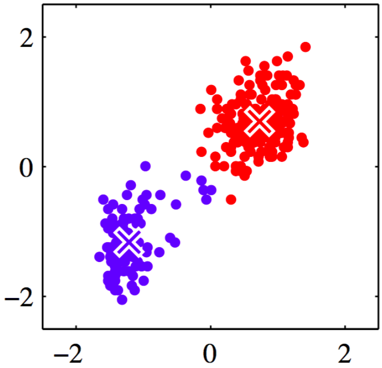
- Assign points to closest center.



From Bishop's *Pattern recognition and machine learning*, Figure 9.1(d).

k-means: by example

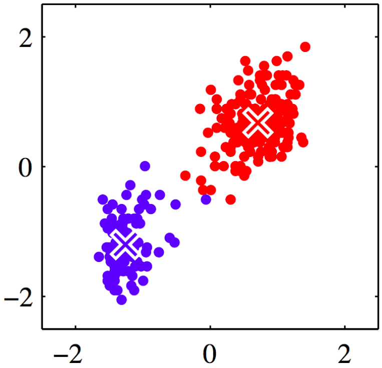
- Compute cluster centers.



From Bishop's *Pattern recognition and machine learning*, Figure 9.1(e).

k-means: by example

- Iterate until convergence.



From Bishop's *Pattern recognition and machine learning*, Figure 9.1(i).

k-means: formalization

- Dataset $\mathcal{D} = \{x_1, \ldots, x_n\} \in \mathbf{R}^d$
- Goal (version 1): Partition data into k clusters.
- Goal (version 2): Partition $\mathbf{R}^d$ into k regions.
- Let $\mu_1, \ldots, \mu_k$ denote cluster centers.

k-means: formalization

- For each x_i, use a **one-hot encoding** to designate membership:

$$r_i = (0, 0, \ldots, 0, 0, 1, 0, 0) \in \mathbf{R}^k$$

- Let

$$r_{ic} = 1(x_i \text{ assigned to cluster } c).$$

- Then

$$r_i = \left(r_{i1}, r_{i2}, \ldots, r_{ik}\right).$$

k-means: objective function

- Find cluster centers and cluster assignments minimizing

$$J(r,\mu)=\sum_{i=1}^{n}\sum_{c=1}^{k} r_{ic}\|x_i-\mu_c\|^2.$$

- Is objective function convex?
- What's the domain of J?
- $r\in\{0,1\}^{n\times k}$, which is not a convex set...
- So domain of J is not convex $\implies$ J is not a convex function
- We should expect local minima.
- Could replace $\|\cdot\|^2$ with something else:
  - e.g. using $\|\cdot\|$ (or any distance metric) gives k-**medoids**.

k-means algorithm

- For fixed r (cluster assignments), minimizing over μ is easy:

$$\begin{aligned} J(r,\mu) &= \sum_{i=1}^{n}\sum_{c=1}^{k} r_{ic}\|x_i-\mu_c\|^2 \\ &= \sum_{c=1}^{k}\underbrace{\sum_{i=1}^{n} r_{ic}\|x_i-\mu_c\|^2}_{=J_c} \\ J_c(\mu_c) &= \sum_{\{i \mid x_i \text{belongs to cluster } c\}} \|x_i-\mu_c\|^2 \end{aligned}$$

- J_c is minimized by

$$\mu_c = \operatorname{mean}\left(\{x_i \mid x_i \text{ belongs to cluster } c\}\right)$$

k-means algorithm

- For fixed μ (cluster centers), minimizing over r is easy:

$$J(r,\mu) \quad = \quad \sum_{i=1}^{n}\sum_{c=1}^{k} r_{ic}\|x_i-\mu_c\|^2$$

- For each i, exactly one of the following terms is nonzero:

$$r_{i1}\|x_i-\mu_1\|^2, r_{i2}\|x_i-\mu_2\|^2, \ldots, r_{ik}\|x_i-\mu_k\|^2$$

- Take

$$r_{ic} = 1(c = \operatorname*{arg\,min}_j \|x_i-\mu_j\|^2)$$

- That is, assign x_i to cluster c with minimum distance

$$\|x_i-\mu_c\|^2$$

k-means algorithm (summary)

- We will use an **alternating minimization** algorithm:
  1. Choose initial cluster centers $\mu = (\mu_1, \ldots, \mu_k)$.
     - e.g. choose k randomly chosen data points
  2. Repeat
     1. For given cluster centers, find optimal cluster assignments:

        $$r_{ic}^{\text{new}} = 1(c = \arg\min_j \|x_i - \mu_j\|^2)$$

     2. Given cluster assignments, find optimal cluster centers:

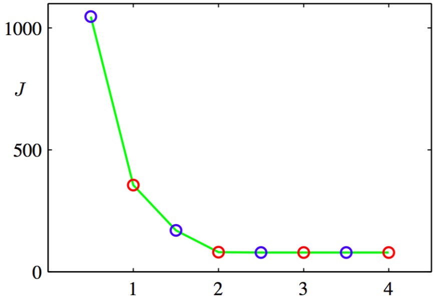
        $$\mu_c^{\text{new}} = \arg\min_{m \in \mathbf{R}^d}; \sum_{\{i \mid r_{ic}=1\}} \|x_i - \mu_c\|^2$$

k-Means Algorithm: Convergence

- Note: Objective value never increases in an update.
  - (Obvious: worst case, everything stays the same)
- Consider the sequence of objective values: $J_1, J_2, J_3, \ldots$
  - monotonically decreasing
  - bounded below by zero
- Therefore, k-**Means objective value converges** to $\inf_t J_t$.
- **Reminder**: This is convergence to a **local** minimum.
- Best to repeat k-means several times, with different starting points

k-Means: Objective Function Convergence

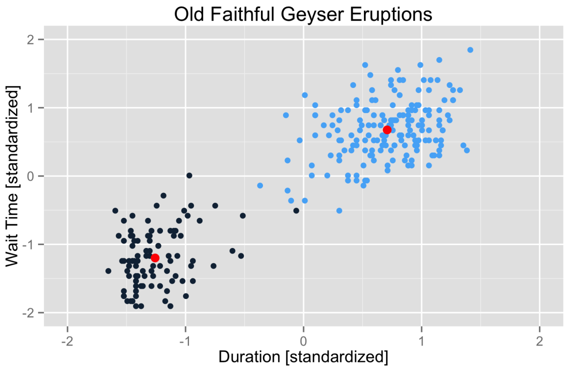
- Blue circles after **"E" step**: assigning each point to a cluster
- Red circles after **"M" step**: recomputing the cluster centers



From Bishop's *Pattern recognition and machine learning*, Figure 9.2.

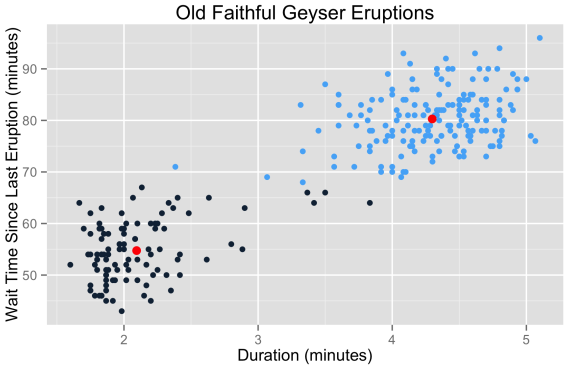
k-Means Algorithm: Standardizing the data

- With standardizing:



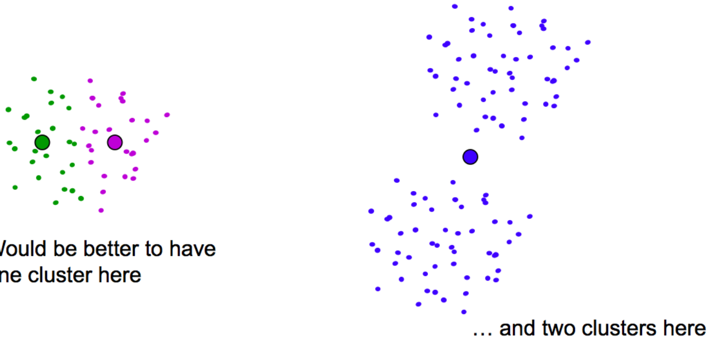
k-Means Algorithm: Standardizing the data

- Without standardizing:



k-Means: Suboptimal Local Minimum

- The clustering for $k = 3$ below is a local minimum, but suboptimal:



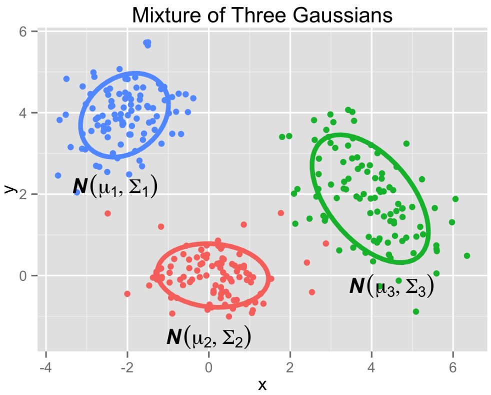
From Sontag's DS-GA 1003, 2014, Lecture 8.

Probabilistic Model for Clustering

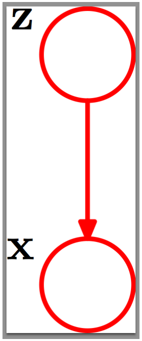
- Let's consider a **generative model** for the data.
- Suppose
  1. There are k clusters.
  2. We have a probability density for each cluster.
- Generate a point as follows
  1. Choose a random cluster $z \in \{1, 2, \ldots, k\}$.
     - $Z \sim \text{Multi}(\pi_1, \ldots, \pi_k)$.
  2. Choose a point from the distribution for cluster Z.
     - $X \mid Z = z \sim p(x \mid z)$.

Gaussian Mixture Model ($k = 3$)

1. Choose $Z \in \{1,2,3\} \sim \text{Multi}\left(\frac{1}{3}, \frac{1}{3}, \frac{1}{3}\right)$.
2. Choose $X \mid Z = z \sim \mathcal{N}(X \mid \mu_z, \Sigma_z)$.



Gaussian Mixture Model: Joint Distribution



- Factorize joint according to Bayes net:

$$\begin{aligned} p(x,z) &= p(z)p(x \mid z) \\ &= \pi_z \mathcal{N}(x \mid \mu_z, \Sigma_z) \end{aligned}$$

- π_z is probability of choosing cluster z.
- $X \mid Z = z$ has distribution $\mathcal{N}(\mu_z, \Sigma_z)$.
- z corresponding to x is the true cluster assignment.

Latent Variable Model

- Back in reality, we observe X, not (X, Z).
- Cluster assignment Z is called a **hidden variable**.

Definition

A **latent variable model** is a probability model for which certain variables are never observed.

- e.g. The Gaussian mixture model is a latent variable model.

Model-Based Clustering

- We observe $X = x$.
- The conditional distribution of the cluster Z given $X = x$ is

$$p(z \mid X = x) = p(x, z)/p(x)$$

- The conditional distribution is a **soft assignment** to clusters.
- A **hard assignment** is

$$z^* = \underset{z \in \{1, \ldots, k\}}{\arg\min}\ \mathbb{P}(Z = z \mid X = x).$$

- So if we have the model, clustering is trival.

Estimating/Learning the Gaussian Mixture Model

- We'll use the common acronym **GMM.**
- What does it mean to "have" or "know" the GMM?
- It means knowing the parameters

$$\begin{aligned}\text{Cluster probabilities}: &\quad \pi = (\pi_1, \dots, \pi_k) \\ \text{Cluster means}: &\quad \mu = (\mu_1, \dots, \mu_k) \\ \text{Cluster covariance matrices}: &\quad \Sigma = (\Sigma_1, \dots \Sigma_k)\end{aligned}$$

- We have a probability model: let's find the MLE.
- Suppose we have data $\mathcal{D} = \{x_1, \dots, x_n\}$.
- We need the model likelihood for $\mathcal{D}$.

Gaussian Mixture Model: Marginal Distribution

- Since we only observe X, we need the **marginal distribution:**

$$\begin{aligned} p(x) &= \sum_{z=1}^{k} p(x,z) \\ &= \sum_{z=1}^{k} \pi_z \mathcal{N}(x \mid \mu_z, \Sigma_z) \end{aligned}$$

- Note that $p(x)$ is a convex combination of probability densities.
- This is a common form for a probability model...

Mixture Distributions (or Mixture Models)

Definition

A probability density $p(x)$ represents a **mixture distribution** or **mixture model**, if we can write it as a **convex combination** of probability densities. That is,

$$p(x) = \sum_{i=1}^{k} w_i p_i(x),$$

where $w_i \geqslant 0$, $\sum_{i=1}^{k} w_i = 1$, and each p_i is a probability density.

- In our Gaussian mixture model, X has a **mixture distribution**.
- More constructively, let S be a set of probability distributions:
  1. Choose a distribution randomly from S.
  2. Sample X from the chosen distribution.
- Then X has a mixture distribution.

Estimating/Learning the Gaussian Mixture Model

- The model likelihood for $\mathcal{D} = \{x_1, \ldots, x_n\}$ is

$$\begin{aligned} L(\pi, \mu, \Sigma) &= \prod_{i=1}^{n} p(x_i) \\ &= \prod_{i=1}^{n} \sum_{z=1}^{k} \pi_z \mathcal{N}(x_i \mid \mu_z, \Sigma_z). \end{aligned}$$

- As usual, we'll take our objective function to be the log of this:

$$J(\pi, \mu, \Sigma) = \sum_{i=1}^{n} \log \left\{ \sum_{z=1}^{k} \pi_z \mathcal{N}(x_i \mid \mu_z, \Sigma_z) \right\}$$

Properties of the GMM Log-Likelihood

- GMM log-likelihood:

$$J(\pi,\mu,\Sigma) = \sum_{i=1}^{n}\log\left\{\sum_{z=1}^{k}\pi_z\mathcal{N}(x_i \mid \mu_z,\Sigma_z)\right\}$$

- Let's compare to the log-likelihood for a single Gaussian:

$$\begin{aligned}&\sum_{i=1}^{n}\log\mathcal{N}(x_i \mid \mu,\Sigma)\\ = \quad & -\frac{nd}{2}\log(2\pi)-\frac{n}{2}\log|\Sigma|-\frac{1}{2}\sum_{i=1}^{n}(x_i-\mu)'\Sigma^{-1}(x_i-\mu)\end{aligned}$$

- For a single Gaussian, the log cancels the exp in the Gaussian density.
  - $\implies$ Things simplify a lot.
- For the GMM, the sum inside the log prevents this cancellation.
  - $\implies$ Expression more complicated. No closed form expression for MLE.

Identifiability Issues for GMM

- Suppose we have found parameters

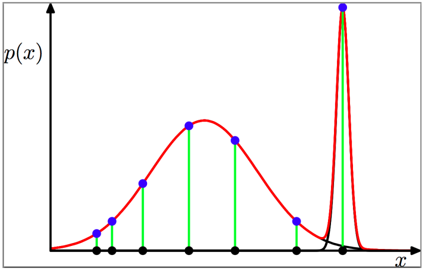
$$\begin{aligned} \text{Cluster probabilities:} \quad & \pi = (\pi_1, \ldots, \pi_k) \\ \text{Cluster means:} \quad & \mu = (\mu_1, \ldots, \mu_k) \\ \text{Cluster covariance matrices:} \quad & \Sigma = (\Sigma_1, \ldots \Sigma_k) \end{aligned}$$

  that are at a local minimum.

- What happens if we shuffle the clusters? e.g. Switch the labels for clusters 1 and 2.
- We'll get the same likelihood. How many such equivalent settings are there?
- Assuming all clusters are distinct, there are $k!$ equivalent solutions.
- Not a problem per se, but something to be aware of.

Singularities for GMM

- Consider the following GMM for 7 data points:



- Let σ^2 be the variance of the skinny component.
- What happens to the likelihood as $\sigma^2 \to 0$?
- In practice, we end up in local minima that do not have this problem.
  - Or keep restarting optimization until we do.
- Bayesian approach or regularization will also solve the problem.

From Bishop's *Pattern recognition and machine learning*, Figure 9.7.

Gradient Descent / SGD for GMM

- What about running gradient descent or SGD on

$$J(\pi, \mu, \Sigma) \quad = \quad -\sum_{i=1}^{n} \log\left\{\sum_{z=1}^{k} \pi_z \mathcal{N}(x_i \mid \mu_z, \Sigma_z)\right\}?$$

- Can be done – but need to be clever about it.
- Each matrix $\Sigma_1, \ldots, \Sigma_k$ has to be positive semidefinite.
- How to maintain that constraint?
    - Rewrite $\Sigma_i = M_i M_i^T$, where M_i is an unconstrained matrix.
    - Then Σ_i is positive semidefinite.
- But we actually prefer positive definite, to avoid singularities.

Cholesky Decomposition for SPD Matrices

Theorem

Every symmetric positive definite matrix $A \in \mathbf{R}^{d \times d}$ has a unique ***Cholesky decomposition****:*

$$A = LL^T,$$

where L a ***lower triangular matrix*** *with positive diagonal elements.*

- A lower triangular matrix has half the number of parameters.
- Symmetric positive definite is better because avoids singularities.
- Requires a non-negativity constraint on diagonal elements.
  - e.g. Use projected SGD method like we did for the Lasso.

MLE for Gaussian Model

- Let's start by considering the MLE for the Gaussian model.
- For data $\mathcal{D} = \{x_1, \ldots, x_n\}$, the log likelihood is given by

$$\sum_{i=1}^{n} \log \mathcal{N}(x_i \mid \mu, \Sigma) = -\frac{nd}{2}\log(2\pi) - \frac{n}{2}\log|\Sigma| - \frac{1}{2}\sum_{i=1}^{n}(x_i - \mu)'\Sigma^{-1}(x_i - \mu).$$

- With some calculus, we find that the MLE parameters are

$$\begin{aligned} \mu_{\text{MLE}} &= \frac{1}{n}\sum_{i=1}^{n} x_i \\ \Sigma_{\text{MLE}} &= \frac{1}{n}\sum_{i=1}^{n}(x_i - \mu_{\text{MLE}})(x_i - \mu_{\text{MLE}})^T \end{aligned}$$

- For GMM, If we knew the cluster assignment z_i for each x_i,
  - we could compute the MLEs for each cluster.

Cluster Responsibilities: Some New Notation

- Denote the probability that observed value x_i comes from cluster j by

$$\gamma_i^j = \mathbb{P}(Z = j \mid X = x_i).$$

- The **responsibility** that cluster j takes for observation x_i.
- Computationally,

$$\begin{aligned}\gamma_i^j &= \mathbb{P}(Z = j \mid X = x_i). \\ &= p(Z = j, X = x_i)/p(x) \\ &= \frac{\pi_j \mathcal{N}(x_i \mid \mu_j, \Sigma_j)}{\sum_{c=1}^k \pi_c \mathcal{N}(x_i \mid \mu_c, \Sigma_c)}\end{aligned}$$

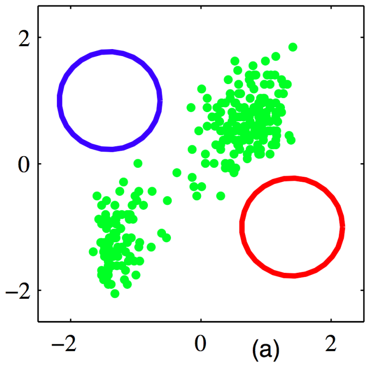
- The vector $\left(\gamma_i^1, \ldots, \gamma_i^k\right)$ is exactly the **soft assignment** for x_i.
- Let $n_c = \sum_{i=1}^n \gamma_i^c$ be the number of points "soft assigned" to cluster c.

EM Algorithm for GMM: Overview

1. Initialize parameters μ, Σ, π.
2. "E step". Evaluate the responsibilities using current parameters:
$$\gamma_i^j = \frac{\pi_j \mathcal{N}(x_i \mid \mu_j, \Sigma_j)}{\sum_{c=1}^k \pi_c \mathcal{N}(x_i \mid \mu_c, \Sigma_c)},$$
for $i = 1, \ldots, n$ and $j = 1, \ldots, k$.
3. "M step". Re-estimate the parameters using responsibilities:
$$\begin{aligned} \mu_c^{\text{new}} &= \frac{1}{n_c} \sum_{i=1}^n \gamma_i^c x_i \\ \Sigma_c^{\text{new}} &= \frac{1}{n_c} \sum_{i=1}^n \gamma_i^c (x_i - \mu_{\text{MLE}})(x_i - \mu_{\text{MLE}})^T \\ \pi_c^{\text{new}} &= \frac{n_c}{n}, \end{aligned}$$
4. Repeat from Step 2, until log-likelihood converges.

EM for GMM

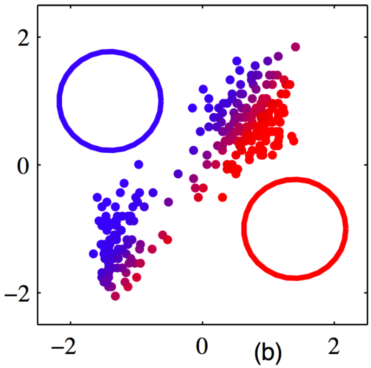
- Initialization



From Bishop's *Pattern recognition and machine learning*, Figure 9.8.

EM for GMM

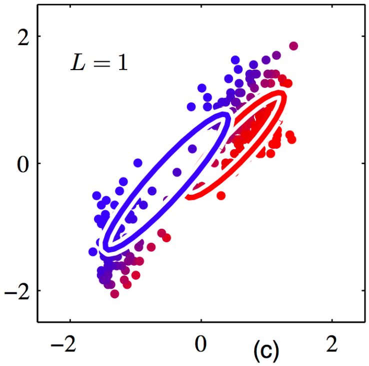
- First soft assignment:



From Bishop's *Pattern recognition and machine learning*, Figure 9.8.

EM for GMM

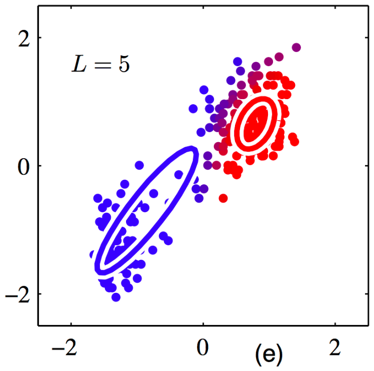
- First soft assignment:



From Bishop's *Pattern recognition and machine learning*, Figure 9.8.

EM for GMM

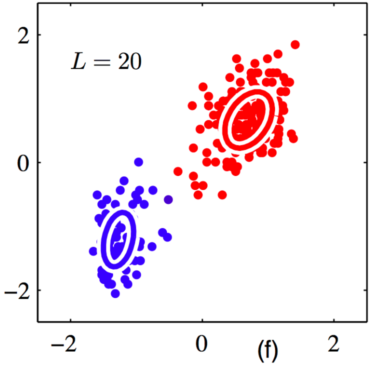
- After 5 rounds of EM:



From Bishop's *Pattern recognition and machine learning*, Figure 9.8.

EM for GMM

- After 20 rounds of EM:



From Bishop's *Pattern recognition and machine learning*, Figure 9.8.

Relation to K-Means

- EM for GMM seems a little like k-means.
- In fact, there is a precise correspondence.
- First, fix each cluster covariance matrix to be $\sigma^2 I$.
- As we take $\sigma^2 \to 0$, the update equations converge to doing k-means.
- If you do a quick experiment yourself, you'll find
  - Soft assignments converge to hard assignments.
  - Has to do with the tail behavior (exponential decay) of Gaussian.

Possible Topics for Next Time

- In last lecture, will give high level view of several topics.
- Possibilities:
  - General EM Algorithm.
  - Bandit problems.
  - LDA / Topic Models
  - Ranking problems.
  - Collaborative Filtering.
  - Generalization bounds.
  - Sequence models (maximum entropy Markov models, HMMs)