Clustering

Anthony Lee & Ben Graham

Outline

Outline

Unsupervised learning

- Much of the first part of the module concerned supervised learning.
  - You have data as well as labels.
  - Want to learn how to associate new data with those labels.
- In unsupervised learning, there are no labels.
  - We want to learn about relationships within the data.
  - Sometimes this is useful also in a supervised setting.
- PCA was an example of unsupervised learning.
  - And can also be used for supervised learning.

Examples

- Generally, given data without labels,
  1. Which items are similar?
  2. What makes them similar?
- Examples
  - Newsgroup data: topic classification / identification
  - MNIST data: which digits are the same?

Clustering

- We will focus on clustering.
  - The assignment of data points to one of a number of clusters.
- Clustering assignments can be
  - hard: point x is in cluster j, or
  - soft: point x is in cluster j with a given probability.

Outline

Centroid-based clustering

- We associate with each cluster a centre.
- The simplest version is known as K-means.
- Let $x_1, \ldots, x_n$ be our data points.
- We want to pick K cluster centres $c_1, \ldots, c_K$ and assign x_i to cluster $z_i \in \{1, \ldots, K\}$ in order to minimize

$$f(c_{1:K}, z_{1:n}) = \sum_{i=1}^{n} d(x_i, c_{z_i}),$$

where d is a distance function.

Global optimization

- We can think of

$$f(c_{1:K}, z_{1:n}) = \sum_{i=1}^{n} d(x_i, c_{z_i}),$$

  as a function to minimize.
- Unfortunately, this is NP-hard in general.
  - NP: non-deterministic polynomial-time.
  - NP-hard: at least as hard as the hardest problems in NP.
  - We know of no deterministic polynomial time algorithms that will solve this problem (and perhaps none exist).
  - K-means: use coordinate descent to find a local minimum.

Coordinate descent

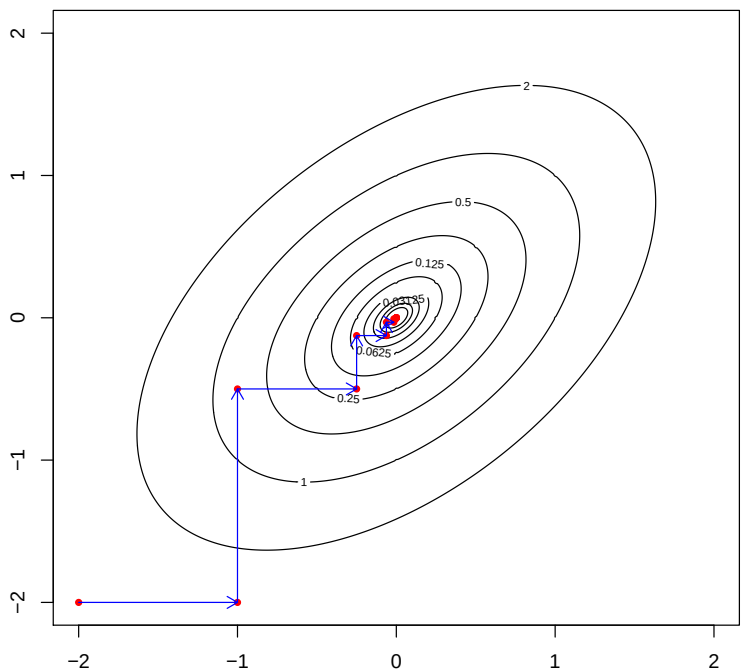
- We previously saw gradient descent to find a local minimum when training ANNs.
- Another strategy is to minimize a function $f(x, y)$ iteratively as follows:
  1. Given (x, y), we find $y' = \arg\min_y f(x, y)$ with x fixed.
  2. Then we find $x' = \arg\min_x f(x, y')$ with y' fixed.
- At each stage, we are making f smaller.
- We are only guaranteed to find a local minimum.
  - In fact, only a point where we cannot decrease f by moves in only one coordinate.

Coordinate descent example

- Consider the function $f(x, y) = x^2 - cxy + y^2$ for $c \in [0, 2)$.
- Then if we start at (x_0, y_0) we will have for $i = 1, 2 \ldots$

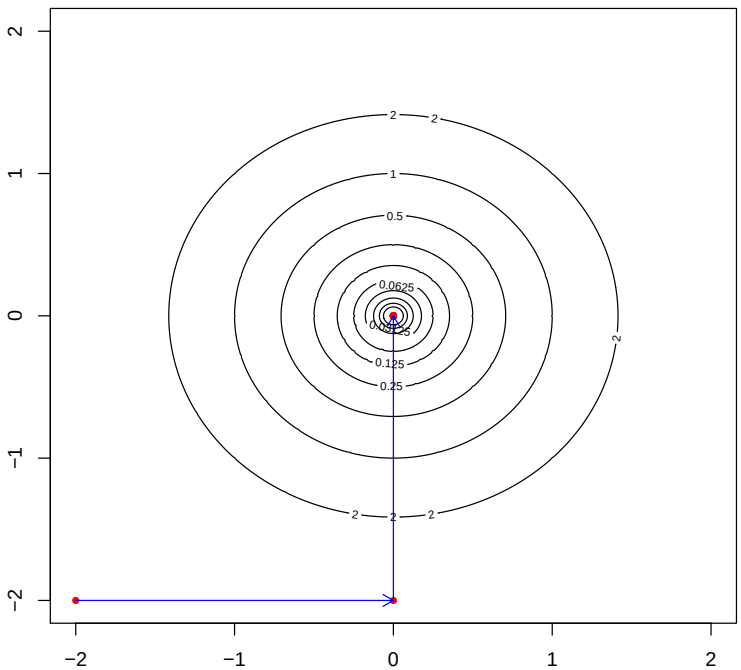
$$x_i = \frac{c}{2} y_{i-1}, \qquad y_i = \frac{c}{2} x_i.$$

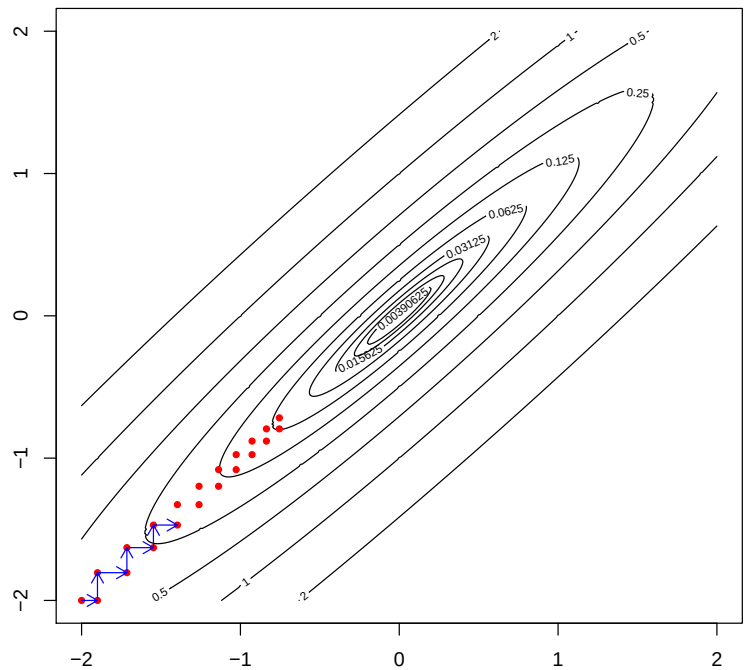
- For $c = 1$:



CD example: $c = 0$ and $c = 1.9$

- The rate at which we get to the minimum depends on c.
- In other words, some problems are harder than others.





K-means

- Recall that

$$f(c_{1:K}, z_{1:n}) = \sum_{i=1}^{n} d(x_i, c_{z_i}).$$

- We will employ coordinate descent.
- To minimize f with $c_{1:K}$ fixed we choose

$$z_i = \arg \min_{j \in \{1,\ldots,K\}} d(x_i, c_j),$$

  i.e., we assign each point to the closest cluster.
- To minimize f with $z_{1:n}$ fixed we choose

$$c_j = \arg \min_{c} \sum_{i:z_i=j} d(x_i, c),$$

  i.e., we make each cluster minimize the sum of the distances to its assigned points.

K-means

- To minimize f with $c_{1:K}$ fixed we assign each point to the closest cluster.
  - This takes $O(nK)$ time as we compute distances to each cluster for each point.
- To minimize f with $z_{1:n}$ fixed when $d(x, y) = \|x - y\|_2^2$, we let c_j be the mean of the points assigned to c_j

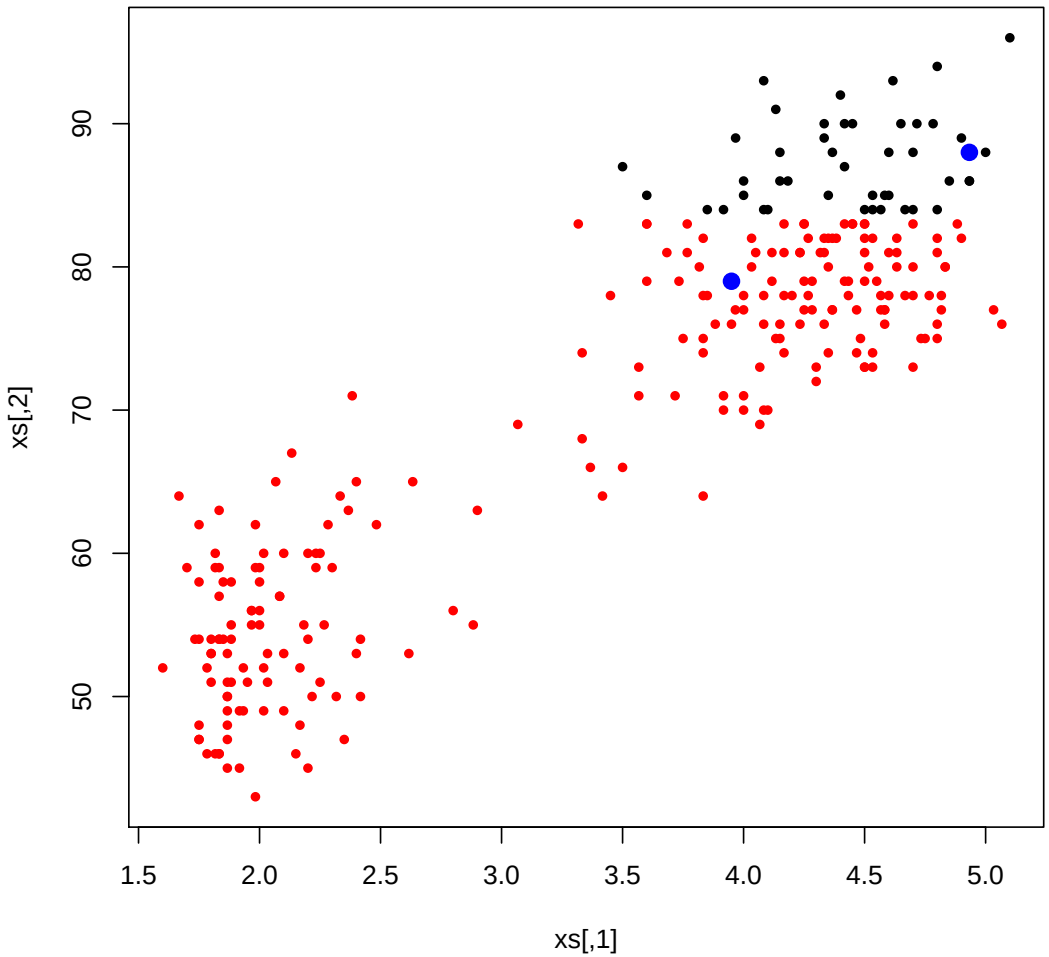
$$c_j = \frac{1}{\sum_{i=1}^{n} \mathbb{I}[z_i = j]} \sum_{i:z_i=j} x_i.$$

  - This takes $O(n)$ time.
- If we used $d(x, y) = \|x - y\|_1$ then c_j would be the median instead. Some people call this K-medians.

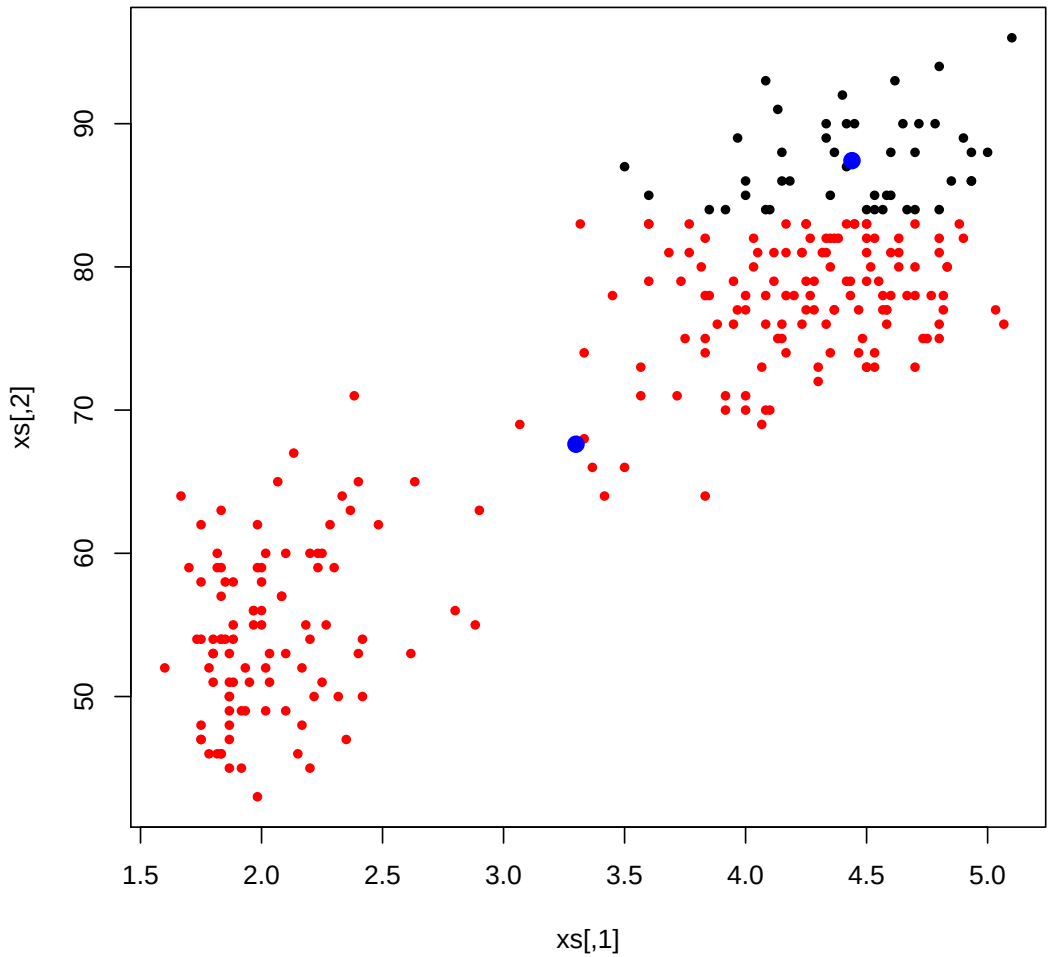
Example: `faithful` data

- Old Faithful geyser, in Yellowstone National Park, Wyoming, USA.
- 272 measurements of eruption length (in minutes) and time to next eruption (in minutes).

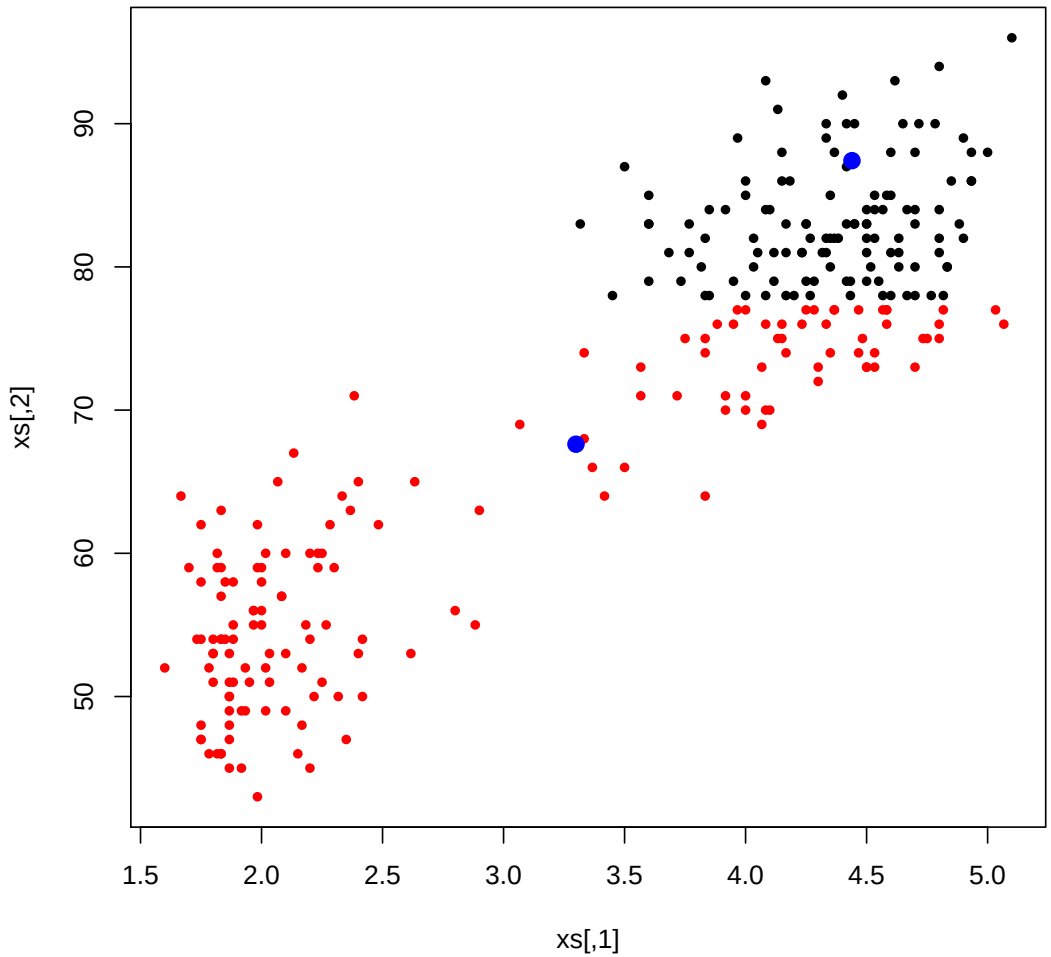
Example: `faithful` data



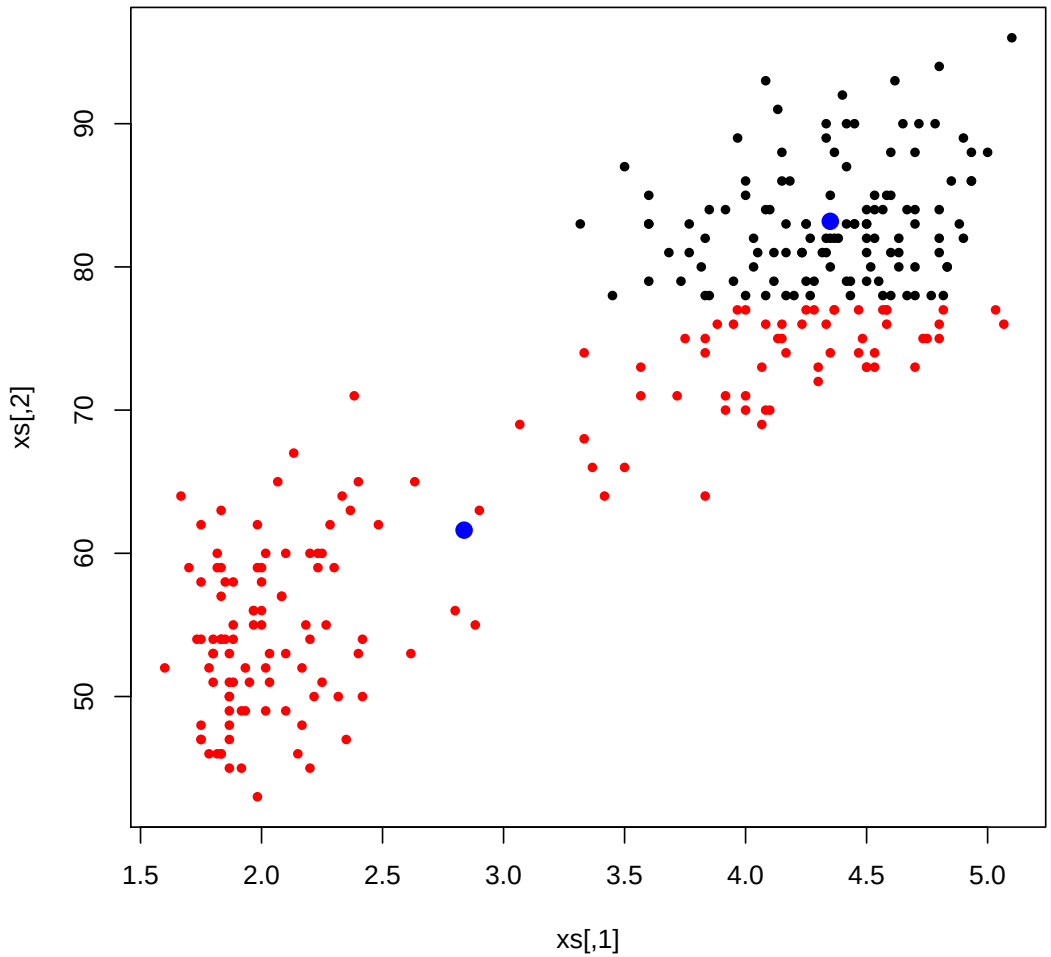
Example: `faithful` data



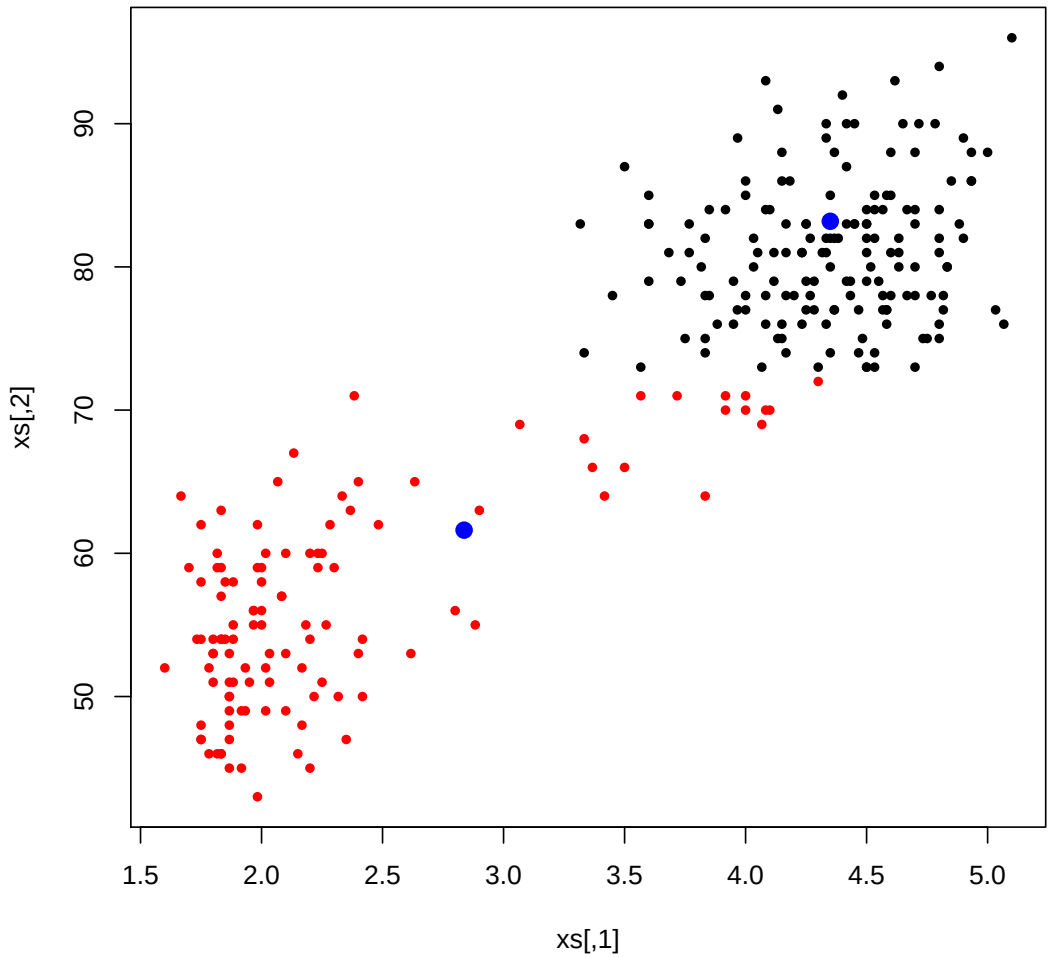
Example: `faithful` data



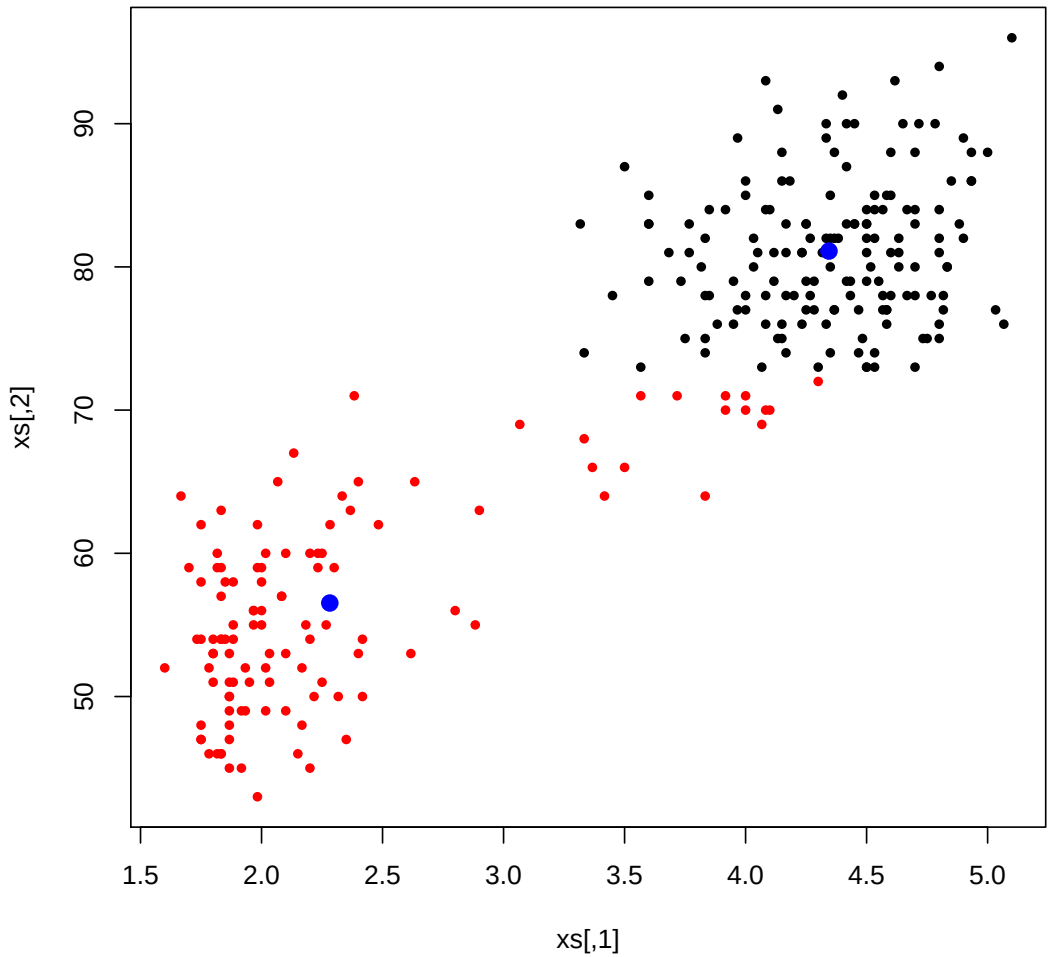
Example: `faithful` data



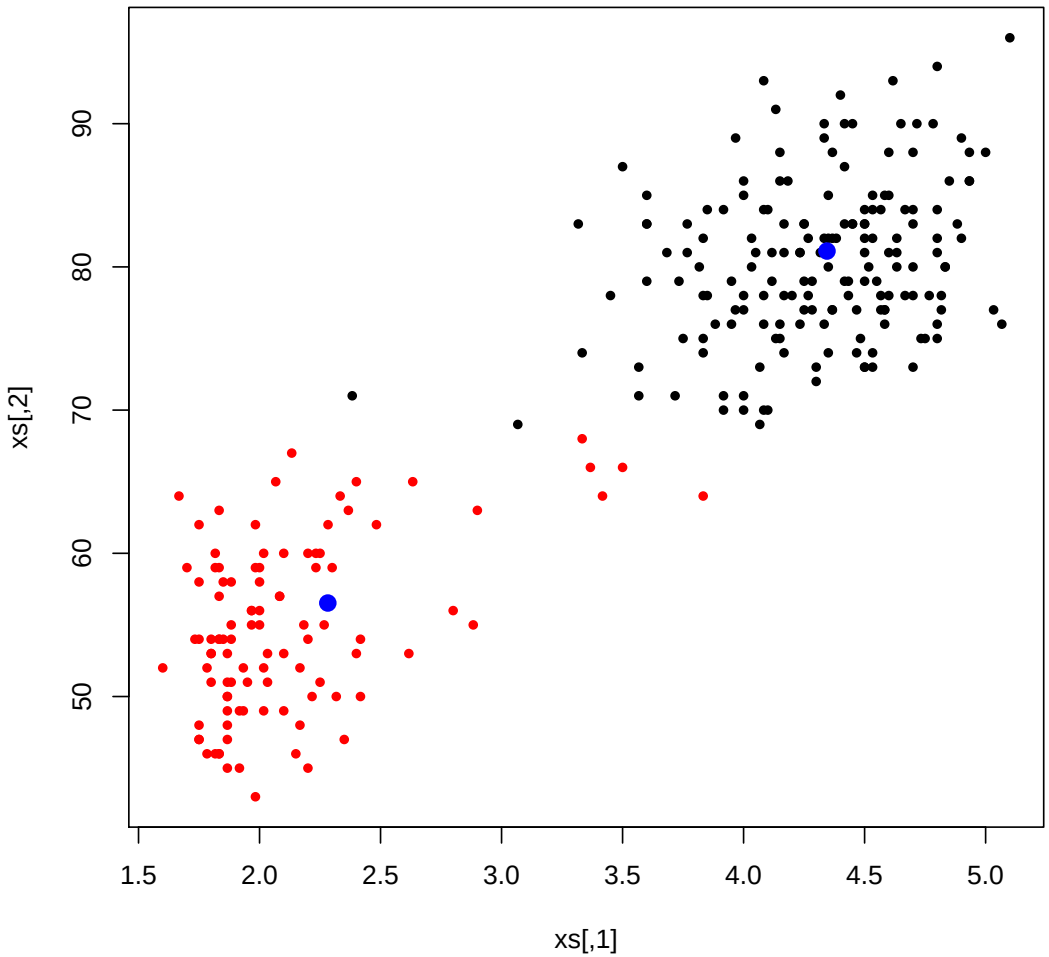
Example: `faithful` data



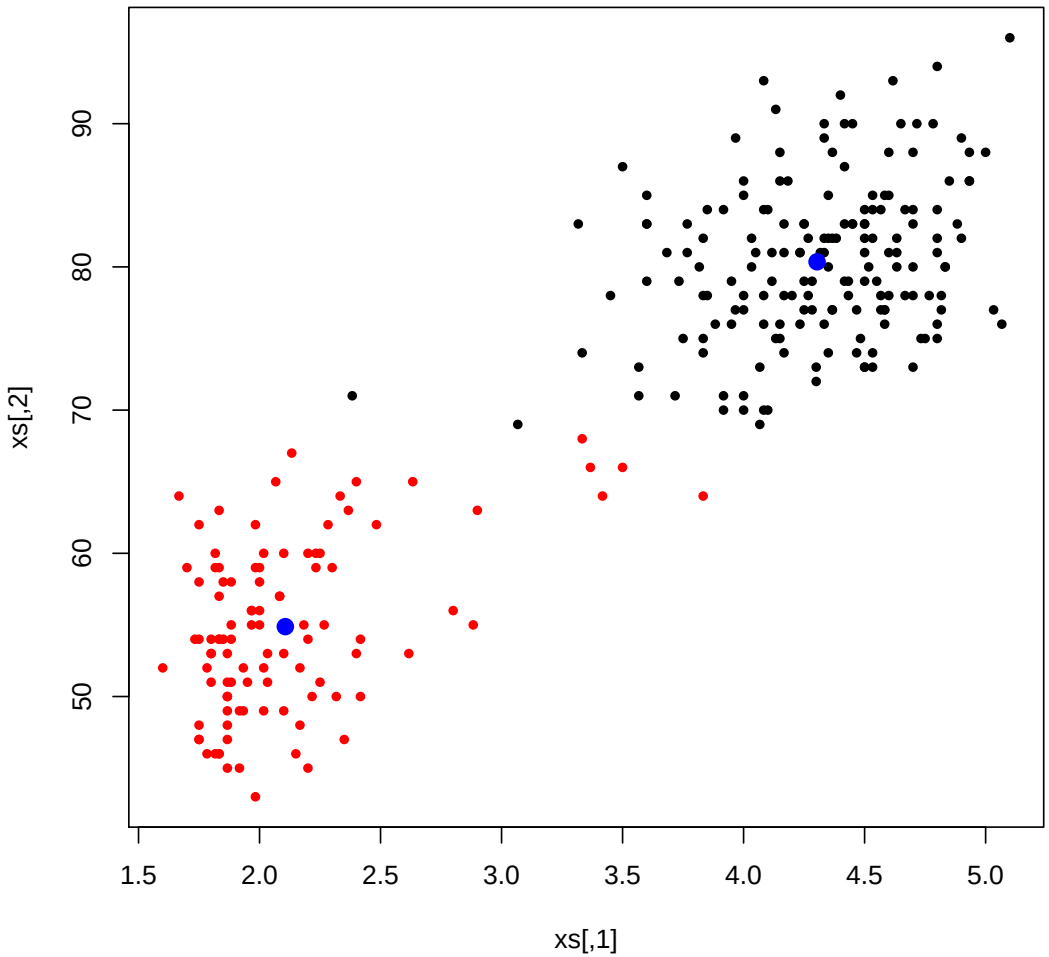
Example: `faithful` data



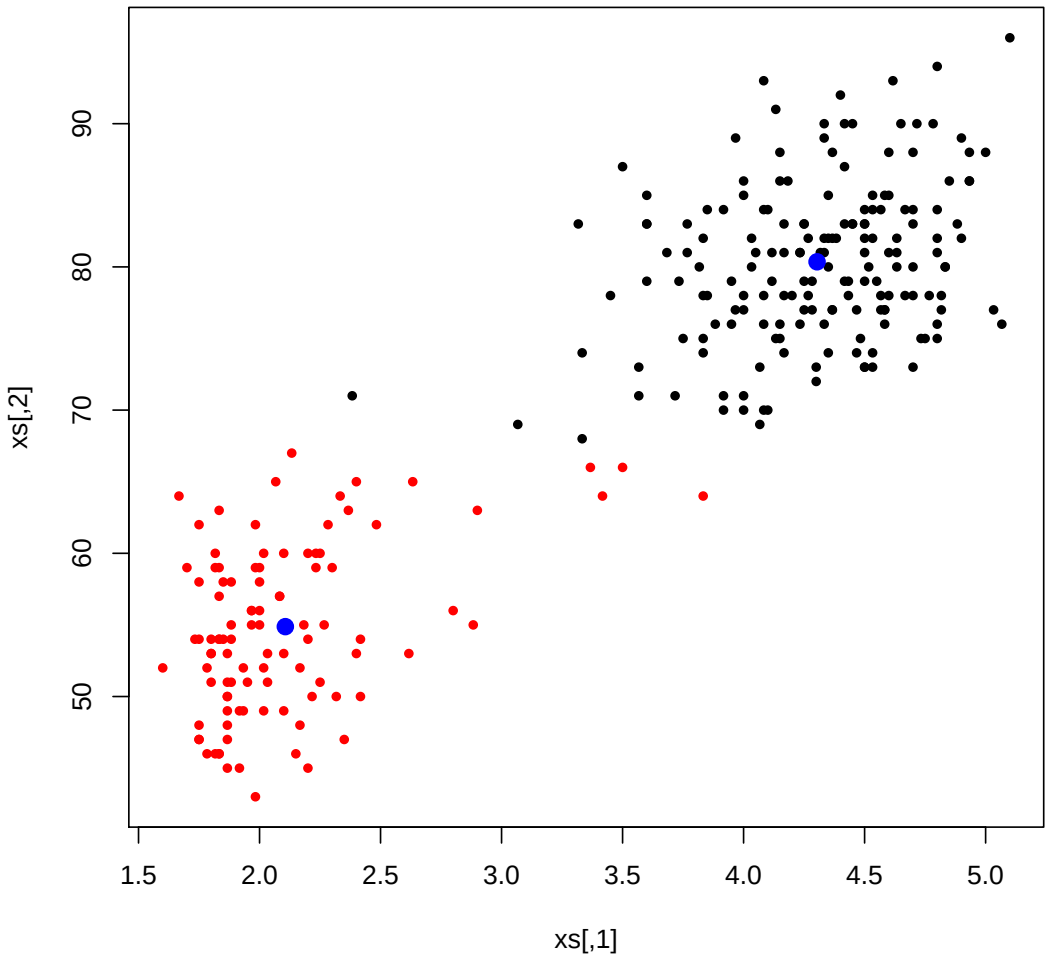
Example: `faithful` data



Example: `faithful` data



Example: `faithful` data



Outline

Soft clustering

- The K-means algorithm is (typically) fast.
- In some situations we may prefer probabilistic statements about clustering.
  - Imagine that two separate phenomena are responsible for geyser eruptions.
  - Would you believe the clustering provided by K-means for the "middle" points?
- In order to make such statements we need to have a way of assigning probabilities.
- In other words, a stochastic model of the data.

Mixture models

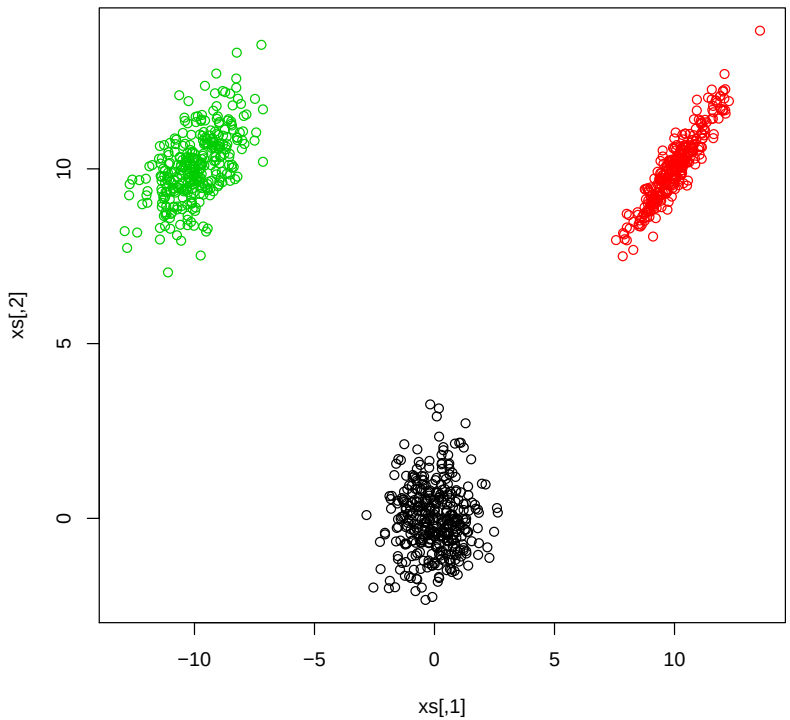
- One popular way to cluster data is to use a mixture model.
- Such a model takes the form

$$p(x) = \sum_{k=1}^{K} w_k p_k(x).$$

- According to the model, data x is generated by
  - selecting a mixture component k with probability w_k,
  - generating x from the distribution associated with p_k.
- In many cases, the distributions p_k come from the same parametric family.
- We are interested in learning $w_1, \ldots, w_K$ and the parameters associated with each component.
- There are then K clusters.
- Note: $p(x, k) = w_k p_k(x)$ satisfies $\sum_{k=1}^{K} p(x, k) = p(x)$.

Example: Gaussian mixture model

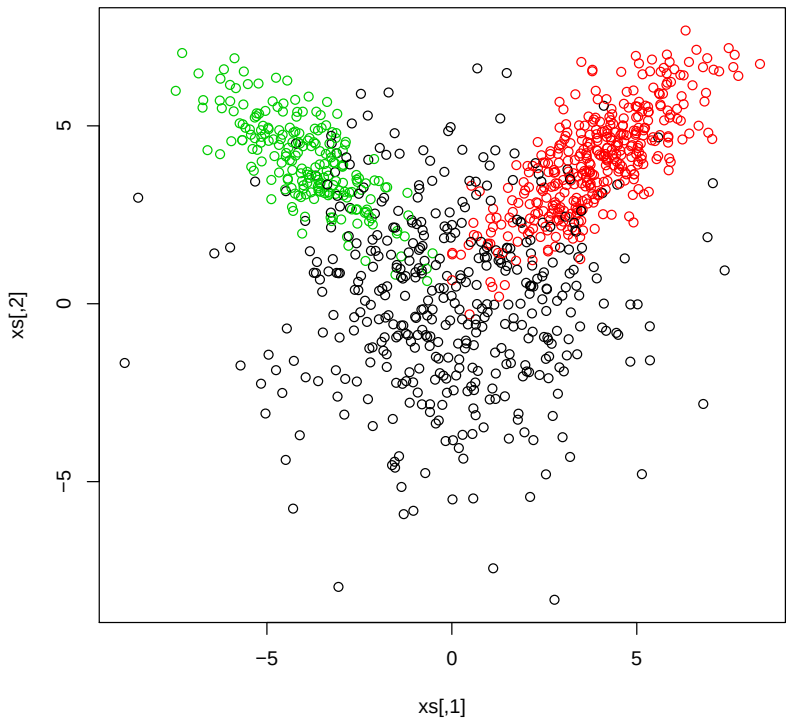
- Let $K = 3$ and $x \in \mathbb{R}^2$, with $p_k(x)$ a multivariate Gaussian with mean μ_k and covariance Σ_k.
- We have $p(x) = \sum_{k=1}^{K} w_k \mathcal{N}(x; \mu_k, \Sigma_k)$.
- Say we have $n = 1000$ i.i.d. points from the mixture model.



- Can we learn $w_{1:K}$, $\mu_{1:K}$ and $\Sigma_{1:K}$?

Example: Gaussian mixture model (cont'd)

- What about now?



- Can we learn $w_{1:K}$, $\mu_{1:K}$ and $\Sigma_{1:K}$?

Maximum likelihood

- One simple method: obtain the maximum likelihood estimate of $\theta = (w_{1:K}, \mu_{1:K}, \Sigma_{1:K})$.
- Problem: how to find the MLE?
- One way to do this is via an algorithm called expectation maximization (Dempster et al. 1997).
  - We only find a local maximum.
- This is common for latent variable models like mixture models.
  - What are the latent variables here?

General mixture model

- We have a mixture model with $\theta = (w_{1:K}, \psi_{1:K})$ defined via

$$p(x|\theta) = \sum_{k=1}^{K} w_k p(x|\psi_k).$$

- We have n i.i.d. data points, so we can write

$$p(x_{1:n}|\theta) = \prod_{i=1}^{n} \sum_{k=1}^{K} w_k p(x_i|\psi_k).$$

- We define the mixture assignments as latent variables, with

$$p(x_{1:n}, z_{1:n}|\theta) = \prod_{i=1}^{n} w_{z_i} p(x_i|\psi_{z_i}).$$

- Check for yourselves that

$$p(x_{1:n}|\theta) = \sum_{z_{1:n} \in \{1,\ldots,K\}^n} p(x_{1:n}, z_{1:n}|\theta).$$

Coordinate ascent

- We will find a "local maximum" by coordinate ascent.
- We will find the maximizing θ for

$$\log p(x_{1:n}|\theta) = \log \left[\sum_{z_{1:n} \in \{1,\ldots,K\}^n} p(x_{1:n}, z_{1:n}|\theta) \right].$$

- What are the coordinates??
- Remember, we do not want to maximize

$$\log p(x_{1:n}, z_{1:n}|\theta),$$

  we want to maximize

$$\log p(x_{1:n}|\theta).$$

- Why log?

Jensen's inequality

- $\varphi(\mathbb{E}[Y]) \geq \mathbb{E}[\varphi(Y)]$ if φ is a concave function.
- A real valued function φ is concave if, for any x and y in $\mathbb{R}$ and $t \in [0, 1]$,

$$\varphi\left(tx + (1-t)y\right) \geq t\varphi(x) + (1-t)\varphi(y).$$

- Therefore let q_i be the probability associated with $Y = y_i$ for $i \in \{1, \ldots, n\}$.
- The definition of a concave function then states that for $n = 2$:

$$\varphi(\mathbb{E}[Y]) = \varphi\left(\sum_{i=1}^{2} q_i y_i\right) \geq \sum_{i=1}^{2} q_i \varphi(y_i) = \mathbb{E}[\varphi(Y)].$$

- Trivially true for $n = 1$.

Jensen's inequality (finite case)

- Assume $\varphi(\mathbb{E}[Y]) \geq \mathbb{E}[\varphi(Y)]$ when Y takes n values.
- Then if Y takes $n+1$ values,

$$\begin{aligned}
\varphi(\mathbb{E}[Y]) &= \varphi\left(\sum_{i=1}^{n+1} q_i y_i\right) = \varphi\left(q_{n+1} y_{n+1} + \sum_{i=1}^{n} q_i y_i\right) \\
&= \varphi\left(q_{n+1} y_{n+1} + [1 - q_{n+1}] \frac{\sum_{i=1}^{n} q_i y_i}{1 - q_{n+1}}\right) \\
&\geq q_{n+1}\varphi(y_{n+1}) + (1 - q_{n+1})\,\varphi\left(\frac{\sum_{i=1}^{n} q_i y_i}{1 - q_{n+1}}\right) \\
&\geq q_{n+1}\varphi(y_{n+1}) + (1 - q_{n+1})\left(\frac{\sum_{i=1}^{n} q_i \varphi(y_i)}{1 - q_{n+1}}\right) \\
&= \sum_{i=1}^{n+1} q_i \varphi(y_i) = \mathbb{E}[\varphi(Y)].
\end{aligned}$$

- So $\varphi(\mathbb{E}[Y]) \geq \mathbb{E}[\varphi(Y)]$ when Y takes any finite number of values.

log is a concave function

- A function is concave if its second derivative is negative.
- We consider $f(x) = \log(x)$ for $x > 0$.
- $f'(x) = \frac{1}{x}$ and $f''(x) = -\frac{1}{x^2}$.
- So $\log(\mathbb{E}[Y]) \geq \mathbb{E}[\log(Y)]$.

Coordinate ascent (cont'd)

- Let q be a p.m.f. with $p(x_{1:n}, z_{1:n}|\theta) > 0 \implies q(z_{1:n}) > 0$.
- We can write

$$\begin{aligned}
\log p(x_{1:n}|\theta) &= \log \left[\sum_{z_{1:n}} p(x_{1:n}, z_{1:n}|\theta) \right] \\
&= \log \left[\sum_{z_{1:n}} q(z_{1:n}) \frac{p(x_{1:n}, z_{1:n}|\theta)}{q(z_{1:n})} \right] \\
&\geq \sum_{z_{1:n}} q(z_{1:n}) \log \left[\frac{p(x_{1:n}, z_{1:n}|\theta)}{q(z_{1:n})} \right] \\
&=: \ell(q, \theta),
\end{aligned}$$

  where we have used $\log(\mathbb{E}_q[Y]) \geq \mathbb{E}_q[\log(Y)]$.
- Note that q represents all the probabilities associated with $z_{1:n}$.
- We will do coordinate ascent on $\ell(q, \theta)$.

Coordinate ascent (cont'd)

- We will maximize $\ell(q, \theta)$ via coordinate ascent.
- Notice that $\ell(q, \theta)$ is maximized for fixed $\theta = \theta^*$ by choosing

$$q(z_{1:n}) = p(z_{1:n}|\theta^*, x_{1:n}) = \frac{p(x_{1:n}, z_{1:n}|\theta^*)}{p(x_{1:n}|\theta^*)}$$

since then

$$\ell(q, \theta^*) = \sum_{z_{1:n}} q(z_{1:n}) \log p(x_{1:n}|\theta^*) = \log p(x_{1:n}|\theta^*).$$

- Important: this means $\max_q \ell(q, \theta^*) = \log p(x_{1:n}|\theta^*)$.
- To optimize θ with $q(z_{1:n}) = p(z_{1:n}|\theta^*, x_{1:n})$ we maximize

$$\ell(q, \theta) = \sum_{z_{1:n}} p(z_{1:n}|\theta^*, x_{1:n}) \log \left[\frac{p(x_{1:n}, z_{1:n}|\theta)}{p(z_{1:n}|\theta^*, x_{1:n})} \right].$$

Coordinate ascent (cont'd)

- We must maximize as a function of θ

$$\begin{aligned} & \sum_{z_{1:n}} p(z_{1:n}|\theta^*, x_{1:n}) \log \left[\frac{p(x_{1:n}, z_{1:n}|\theta)}{p(z_{1:n}|\theta^*, x_{1:n})} \right] \\ = & \sum_{z_{1:n}} p(z_{1:n}|\theta^*, x_{1:n}) \left[\log p(x_{1:n}, z_{1:n}|\theta) - \log p(z_{1:n}|\theta^*, x_{1:n}) \right]. \end{aligned}$$

- This amounts to maximizing

$$\sum_{z_{1:n}} p(z_{1:n}|\theta^*, x_{1:n}) \log p(x_{1:n}, z_{1:n}|\theta) = \mathbb{E}_{Z_{1:n}|\theta^*, x_{1:n}} \left[\log p(x_{1:n}, Z_{1:n}|\theta) \right]$$

- So we can do both ascent steps in one go by finding

$$\arg\max_{\theta} \mathbb{E}_{Z_{1:n}|\theta^*, x_{1:n}} \left[\log p(x_{1:n}, Z_{1:n}|\theta) \right],$$

where θ^* is the previous value of θ.

Expectation maximization

- We didn't use the mixture model framework in our derivation.
- EM is useful for many types of latent variable models.
- We end up iteratively maximizing the expectation of the complete log likelihood $\log p(x, Z|\theta)$ where the expectation is w.r.t. Z distributed according to its conditional distribution given θ_{old} and x:

$$\theta_{\text{new}} = \arg\max_{\theta} \mathbb{E}_{Z|\theta_{\text{old}},x} \left[\log p(x, Z|\theta)\right].$$

- Make sure this is clear: we are maximizing a function of θ, since

$$\mathbb{E}_{Z|\theta_{\text{old}},x} \left[\log p(x, Z|\theta)\right]$$

  is a function of θ.
- "Expectation part" tells us which function of θ to maximize.

Comparison with K-means I

- This is for ease of interpretation and to aid your memory.
- To define the expectation of interest, we find the distribution of the cluster assignments given θ and the data.
  - In K-means, we just assigned each point to its closest cluster.
- To maximize the expectation of interest, we update the cluster parameters given the distribution over the cluster assignments.
  - In K-means, we did the same thing, except the "distribution" was a hard assignment.
- So EM can be viewed as a soft version of K-means.
- We will return to this comparison later.

EM for mixtures: γ (E-step)

- Some of the steps in an EM algorithm are the same for any type of mixture.
- To find $p(z_{1:n}|\theta, x_{1:n})$ we write

$$p(z_{1:n}|\theta, x_{1:n}) = \frac{p(x_{1:n}, z_{1:n}|\theta)}{p(x_{1:n}|\theta)} \propto \prod_{i=1}^{n} p(x_i, z_i|\theta).$$

- Hence we have

$$\Pr(z_i = k|\theta, x_i) = \frac{w_k p(x_i|\psi_k)}{\sum_{j=1}^{K} w_j p(x_i|\psi_j)} =: \gamma_{ik}.$$

- γ_{ik} is the probability that x_i is associated with cluster k.
- These values define the q that maximizes $\ell(q, \theta)$ for fixed θ.

EM for mixtures: $w_{1:K}$ (M-step I)

- Some of the steps in an EM algorithm are the same for any type of mixture.
- To maximize $w_{1:K}$ we want to maximize

$$\begin{aligned} & \mathbb{E}_{Z_{1:n}|\theta^*,x}\left[\log p(x_{1:n}, Z_{1:n}|\theta)\right] \\ = & \sum_{z_{1:n}} \sum_{i=1}^{n} \gamma_{iz_i} \log p(x_i, z_i|\theta_{z_i}) \\ = & \sum_{i=1}^{n} \sum_{k=1}^{K} \gamma_{ik} \log p(x_i, k|\theta_k) \\ = & \sum_{i=1}^{n} \sum_{k=1}^{K} \gamma_{ik} \left[\log(w_k) + \log p(x_i|\psi_k)\right]. \end{aligned}$$

- Hence we need to maximize $\sum_{i=1}^{n} \sum_{k=1}^{K} \gamma_{ik} \log(w_k)$ subject to $\sum_{k=1}^{K} w_k = 1$.

EM for mixtures: $w_{1:K}$ (cont'd)

- To maximize $\sum_{i=1}^{n} \sum_{k=1}^{K} \gamma_{ik} \log(w_k)$ under the constraint that $\sum_{k=1}^{K} w_k = 1$ we can use a Lagrange multiplier.
- We define the objective function

$$f(w_{1:K}) = \sum_{i=1}^{n} \sum_{k=1}^{K} \gamma_{ik} \log(w_k) + \lambda \left(1 - \sum_{k=1}^{K} w_k\right).$$

- We then have

$$\frac{\partial f}{\partial w_k} = \sum_{i=1}^{n} \frac{\gamma_{ik}}{w_k} - \lambda = 0 \implies w_k = \frac{\sum_{i=1}^{n} \gamma_{ik}}{\lambda},$$

  and this implies that $w_k = \sum_{i=1}^{n} \gamma_{ik}/n$ for each $k \in \{1, \ldots, K\}$
- Mixture probabilities are determined by current allocations γ.

EM for mixtures: $\psi_{1:K}$ (M-step II)

- To maximize w.r.t. ψ, the model $p(x, z|\psi_k)$ matters.
- We need to maximize

$$\sum_{i=1}^{n}\sum_{k=1}^{K}\gamma_{ik}\log p(x_i|\psi_k).$$

- Consider the GMM with $\psi = (\mu, \Sigma)$ and $p(x|\psi) = \mathcal{N}(x; \mu, \Sigma)$.
- It turns out this is maximized when for each $k \in \{1, \ldots, K\}$

$$\mu_k = \frac{\sum_{i=1}^{n}\gamma_{ik}x_i}{\sum_{i=1}^{n}\gamma_{ik}},$$

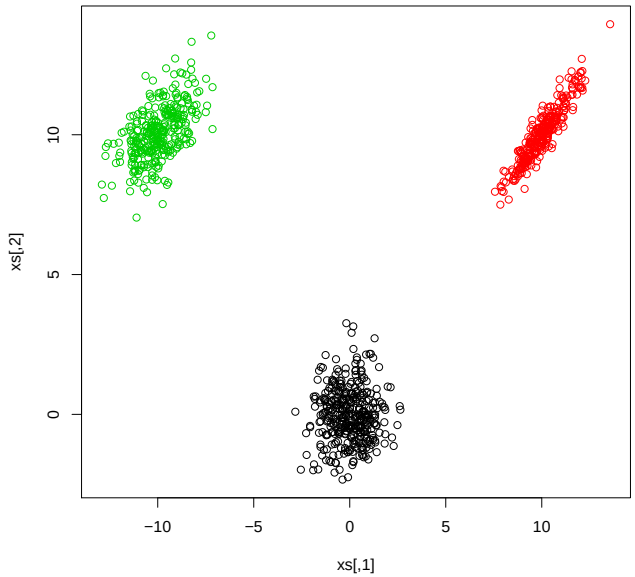
where μ_k and each x_i are vectors, and

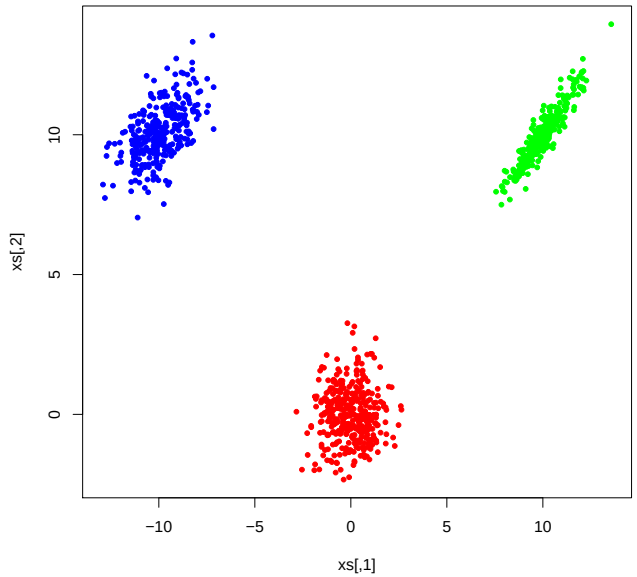
$$\Sigma_k = \frac{\sum_{i=1}^{n}\gamma_{ik}(x_i-\mu_k)(x_i-\mu_k)^T}{\sum_{i=1}^{n}\gamma_{ik}} = \frac{\sum_{i=1}^{n}\gamma_{ik}x_ix_i^T}{\sum_{i=1}^{n}\gamma_{ik}} - \mu_k\mu_k^T.$$

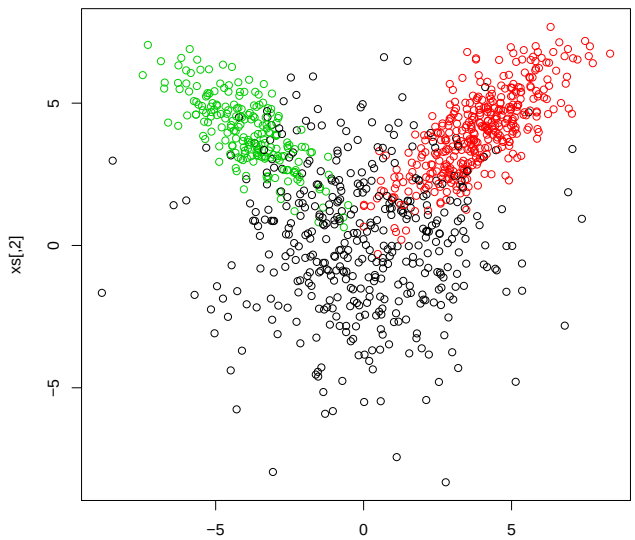
Comparison with K-means II

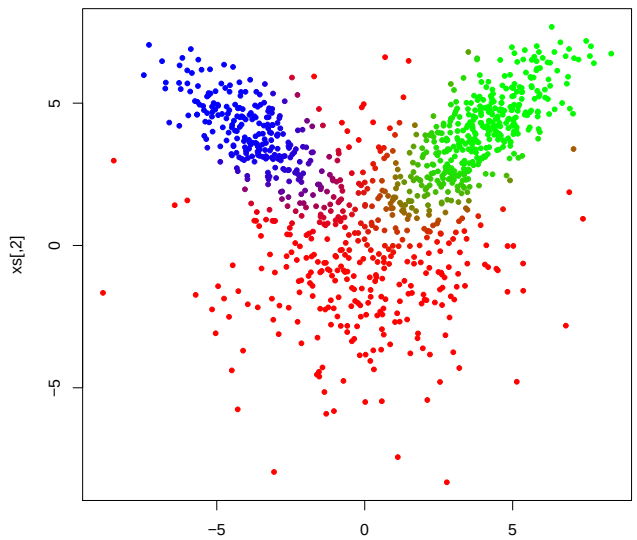
- This is for ease of interpretation and to aid your memory.
- To define the expectation of interest, we find the distribution of the cluster assignment for each data point given θ.
  - In K-means, we just assigned each point to its closest cluster.
- To maximize the expectation of interest, we update the cluster parameters given the distribution over the cluster assignments.
  - In K-means, we did the same thing, except the "distribution" was a hard assignment.
  - In many cases, we will also update some parameters via taking a mean.
- So EM can be viewed as a soft version of K-means.

Examples revisited

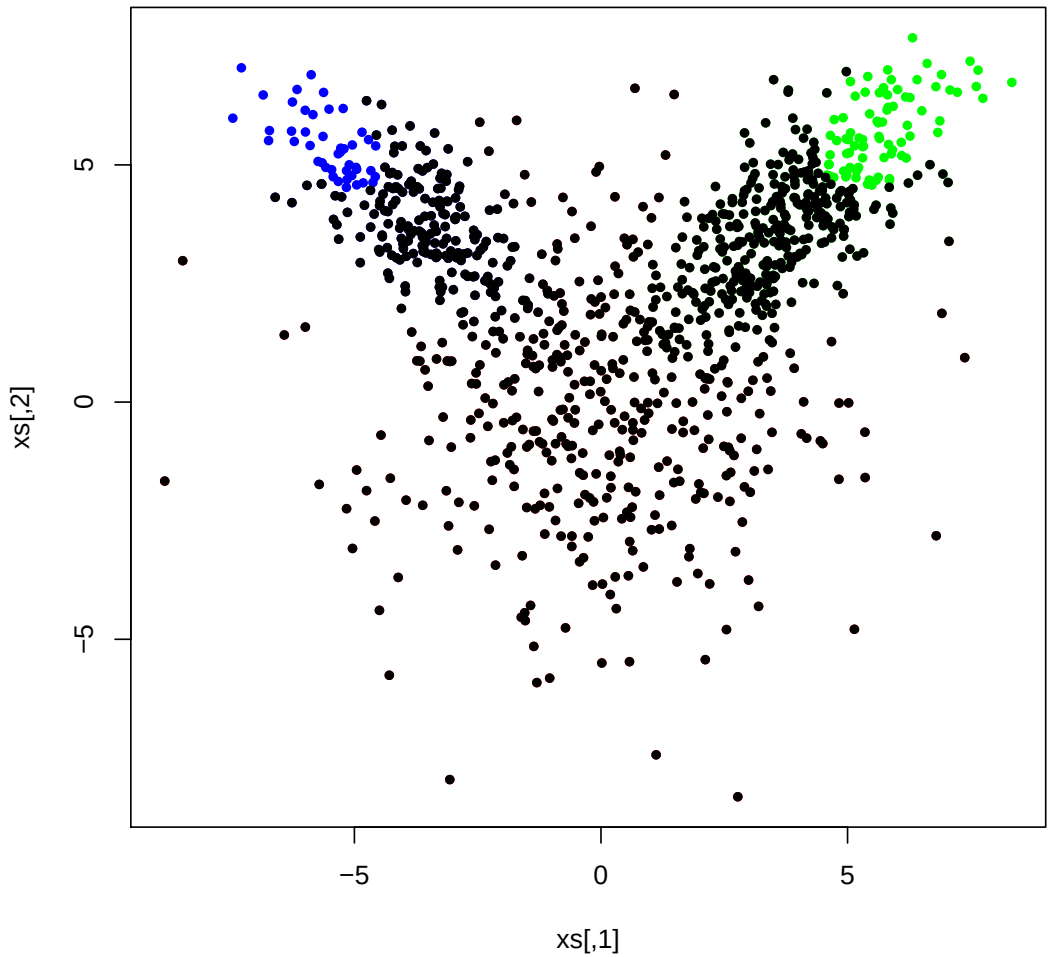




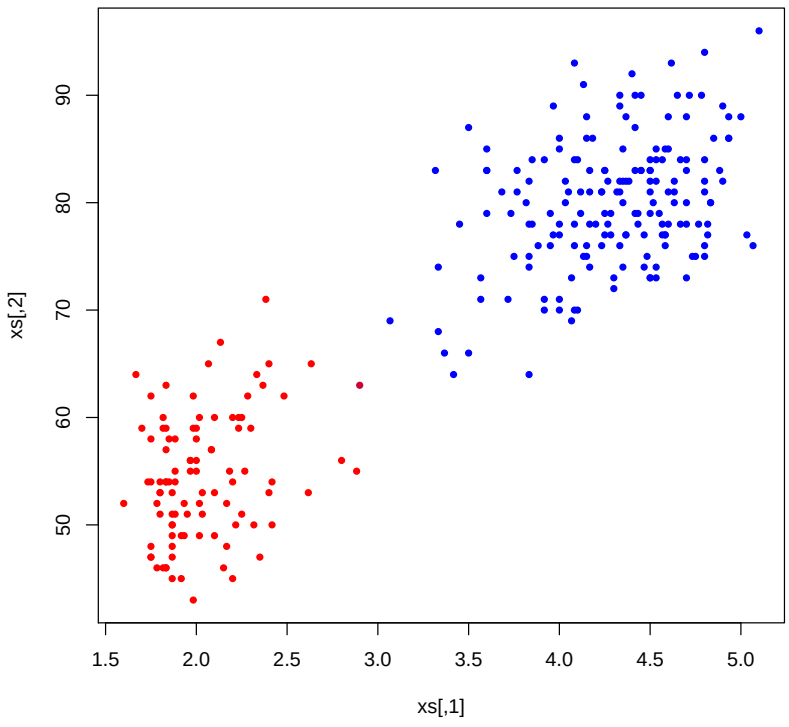


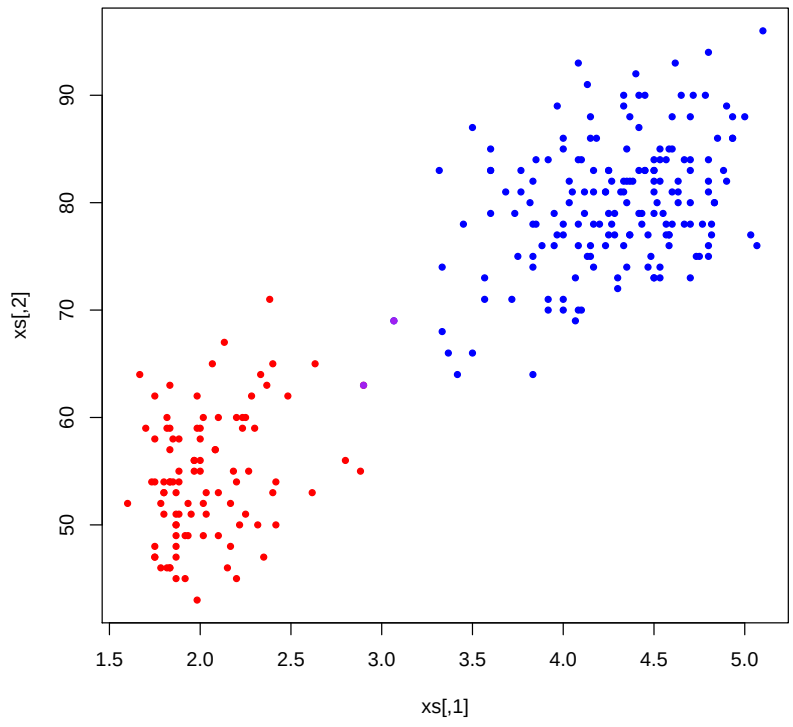


Points with Pr > 0.01 from the middle cluster



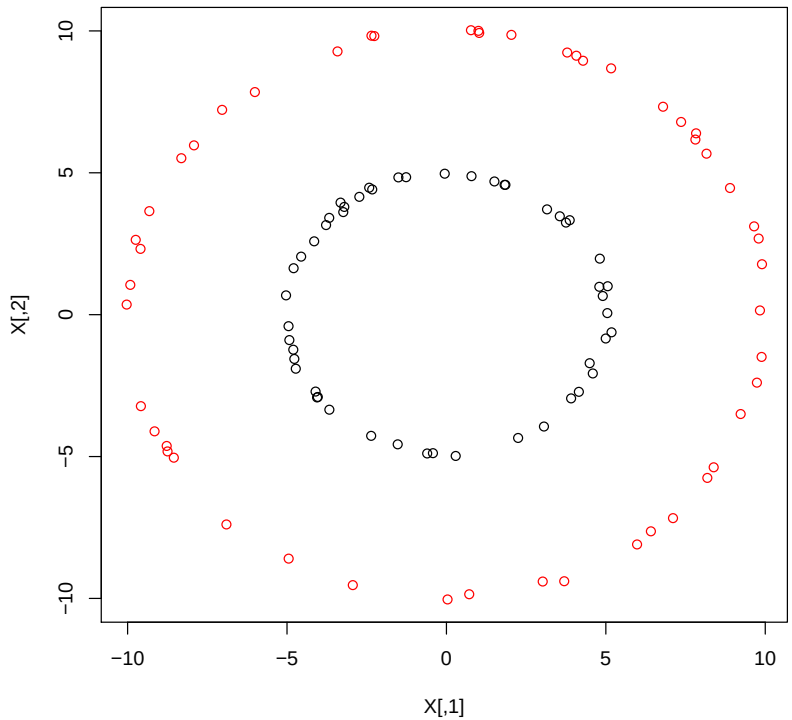
Old faithful





More than just centres

- You can encode a lot in a probabilistic model.
- What would K-means do with this?



Revisiting K-means

- The K-means algorithm can be obtained as a limiting special case.
- We let $\psi = \mu$ and $p(x|\psi) = \mathcal{N}(x; \mu, \epsilon I)$.
  - As $\epsilon \to 0$ then γ_{ik} approaches either 0 or 1.
  - This is like hard clustering.
- This is a Gaussian mixture model where Σ is a known parameter.

Mixtures of Bernoullis

- We can model $x \in \{0,1\}^p$ via independent Bernoulli distributions.
- We have $\psi = \mu$ with

$$p(x|\mu) = \prod_{j=1}^{p} \mu_j^{x_j}(1-\mu_j)^{1-x_j}.$$

- We need to maximize

$$\sum_{i=1}^{n}\sum_{k=1}^{K} \gamma_{ik} \log p(x_i|\mu_k).$$

- To maximize this (Assignment 3) we set

$$\mu_k = \frac{\sum_{i=1}^{n} \gamma_{ik} x_i}{\sum_{i=1}^{n} \gamma_{ik}}.$$

Example: newgroups

- We are modeling each document i as a binary string $x_i \in \{0,1\}^p$.
- A document x is modeled as a sample from the distribution defined by $\mu_k \in [0,1]^p$ where

$$p(x|\mu) = \prod_{j=1}^{p} \mu_j^{x_j}(1-\mu_j)^{1-x_j}.$$

- We don't know μ and furthermore we have many documents $x_{1:n}$ which come from potentially different μ's.
- For $x_i \in \{0,1\}^p$ the mixture model is

$$p(x_i|\theta) = \sum_{k=1}^{K} w_k p(x_i|\mu_k).$$

- Clustering tries to learn γ and $w_{1:K}$ and $\mu_{1:K}$.

Example: newgroups (cont'd)

Words associated with 8 largest values of μ in each cluster

1	2	3	4
team	windows	fact	question
games	email	world	god
players	help	case	fact
baseball	problem	course	problem
hockey	computer	question	university
season	system	government	course
win	software	problem	christian
league	program	sate	help

Sample documents from each cluster

1. "case fans hockey league space system team win"

   "league president research"

   "hockey world"
   "course games hit team"
2. "course display drive nasa space"

   "phone university windows"

   "program system"
   "email files problem"
3. "bible children computer earth god human israel jesus religion solar system world"

   "case christian fact government human israel rights university war world"

   "case games government medicine nasa power world"
   "car course disease doctor fact health insurance law president problem question world"
4. "world"

   "help power"

   "aids government research"
   "children evidence government president"

Numbers of clusters

- Clusters do not always correspond to what we want them to.
- If you have labels, you can try with more clusters (e.g. $K = 100$) and then determine a relationship between clusters and labels.
- This also applies to K-means.
- With K-means on MNIST data, $K = 200$ and "voting" for cluster interpretation, I obtained
  - Training success rate of 90.66%.
  - Test success rate of 91.18%.
- There are a number of ways people pick the number of clusters.
  - One choice is to try and maximize "stability" using cross-validation.