EM (cont'd) and Hidden Markov Models

Latent variable models

- Latent variable models allow us to hypothesize different types of structures that may underlie the observed data and then recover those structures
- We have understand how to
  - specify them (variables, distributions with parameters)
  - sample from them (as generative models)
  - estimate them from data
- Recall mixture models

$$y \qquad y \in \{1, \ldots, K\} \qquad P(y) = p_y$$

$$\downarrow$$

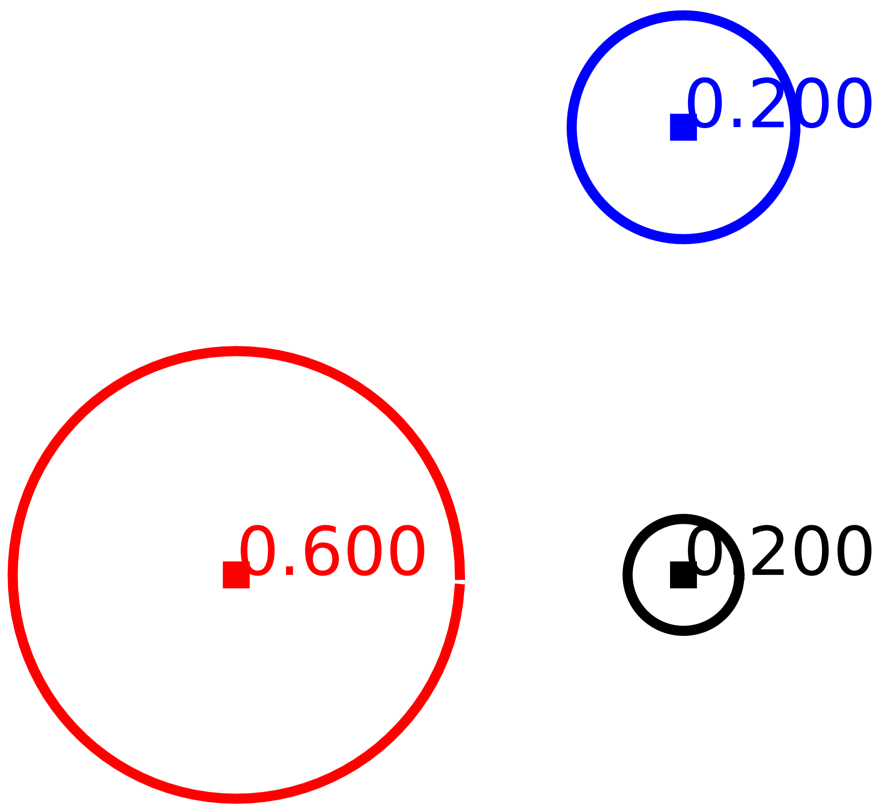
$$x \qquad x \in \mathbb{R}^d \qquad P(x|y) = N(x; \mu^{(y)}, \sigma_y^2 I)$$

Mixture model: generation

- We can sample points from a mixture model in two steps

1) Sample $y \sim \text{Categ}(p_1, \ldots, p_K)$ **which cluster**

2) Sample $x \sim N(x; \mu^{(y)}, \sigma_y^2 I)$ **which point from the chosen cluster**

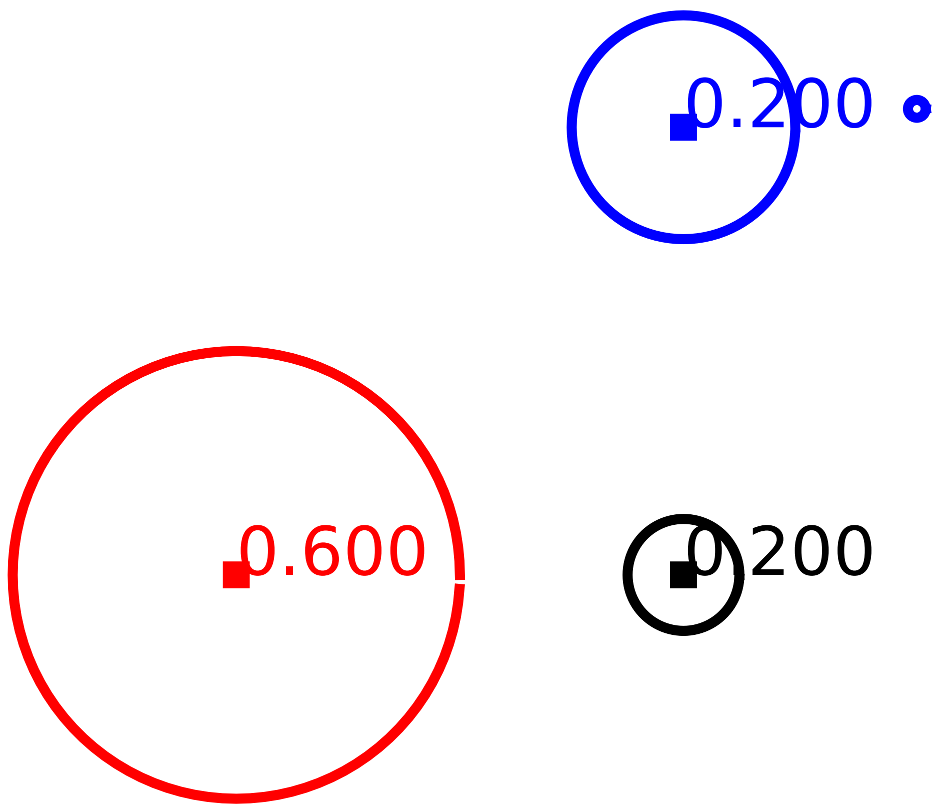


Mixture model: generation

- We can sample points from a mixture model in two steps

1) Sample $y \sim \text{Categ}(p_1, \ldots, p_K)$ **which cluster**

2) Sample $x \sim N(x; \mu^{(y)}, \sigma_y^2 I)$ **which point from the chosen cluster**

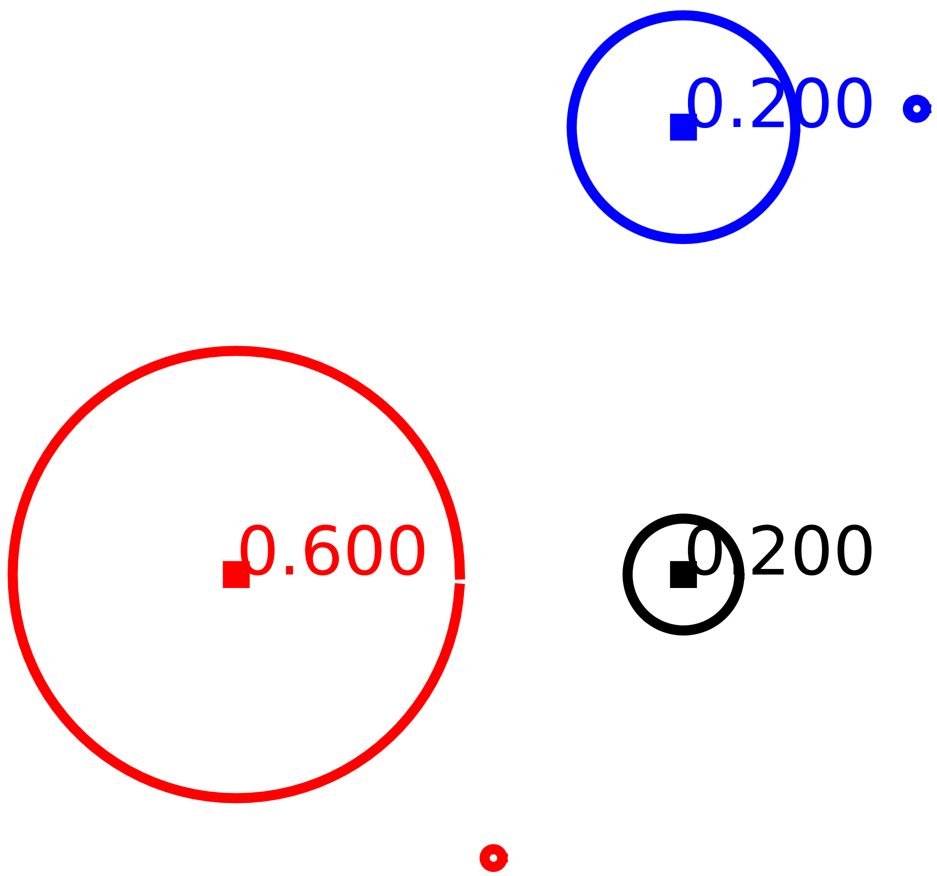


Mixture model: generation

- We can sample points from a mixture model in two steps

1) Sample $y \sim \text{Categ}(p_1, \ldots, p_K)$ **which cluster**

2) Sample $x \sim N(x; \mu^{(y)}, \sigma_y^2 I)$ **which point from the chosen cluster**

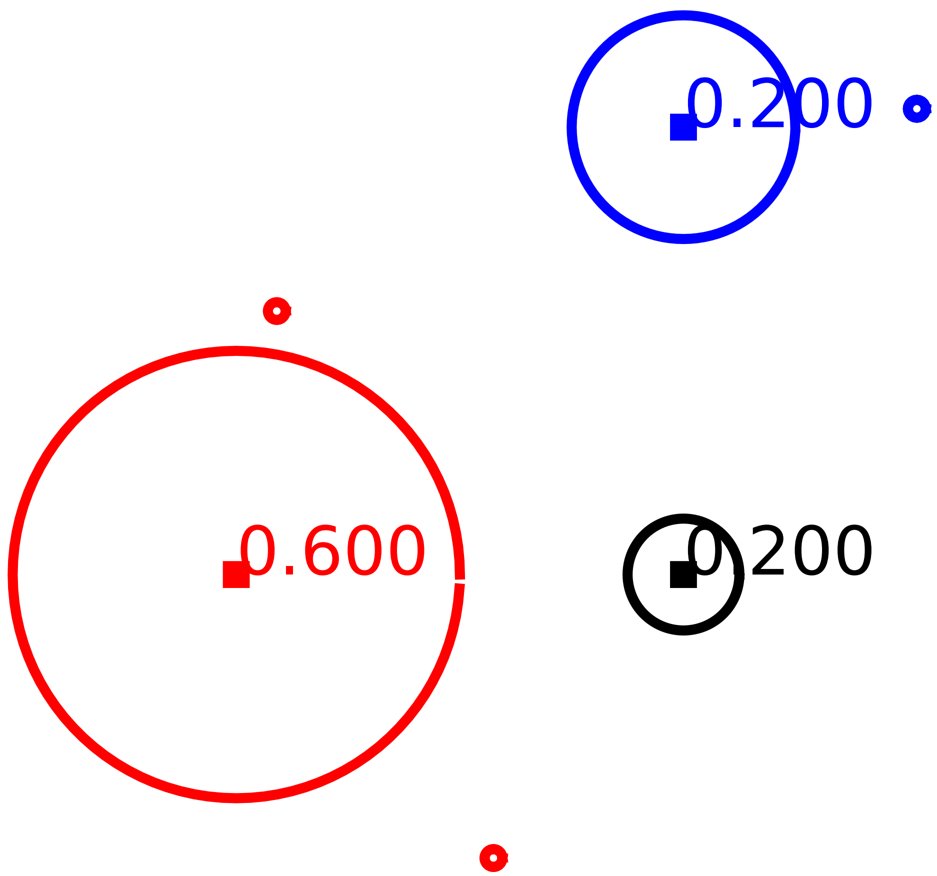


Mixture model: generation

- We can sample points from a mixture model in two steps

1) Sample $y \sim \text{Categ}(p_1, \ldots, p_K)$ **which cluster**

2) Sample $x \sim N(x; \mu^{(y)}, \sigma_y^2 I)$ **which point from the chosen cluster**

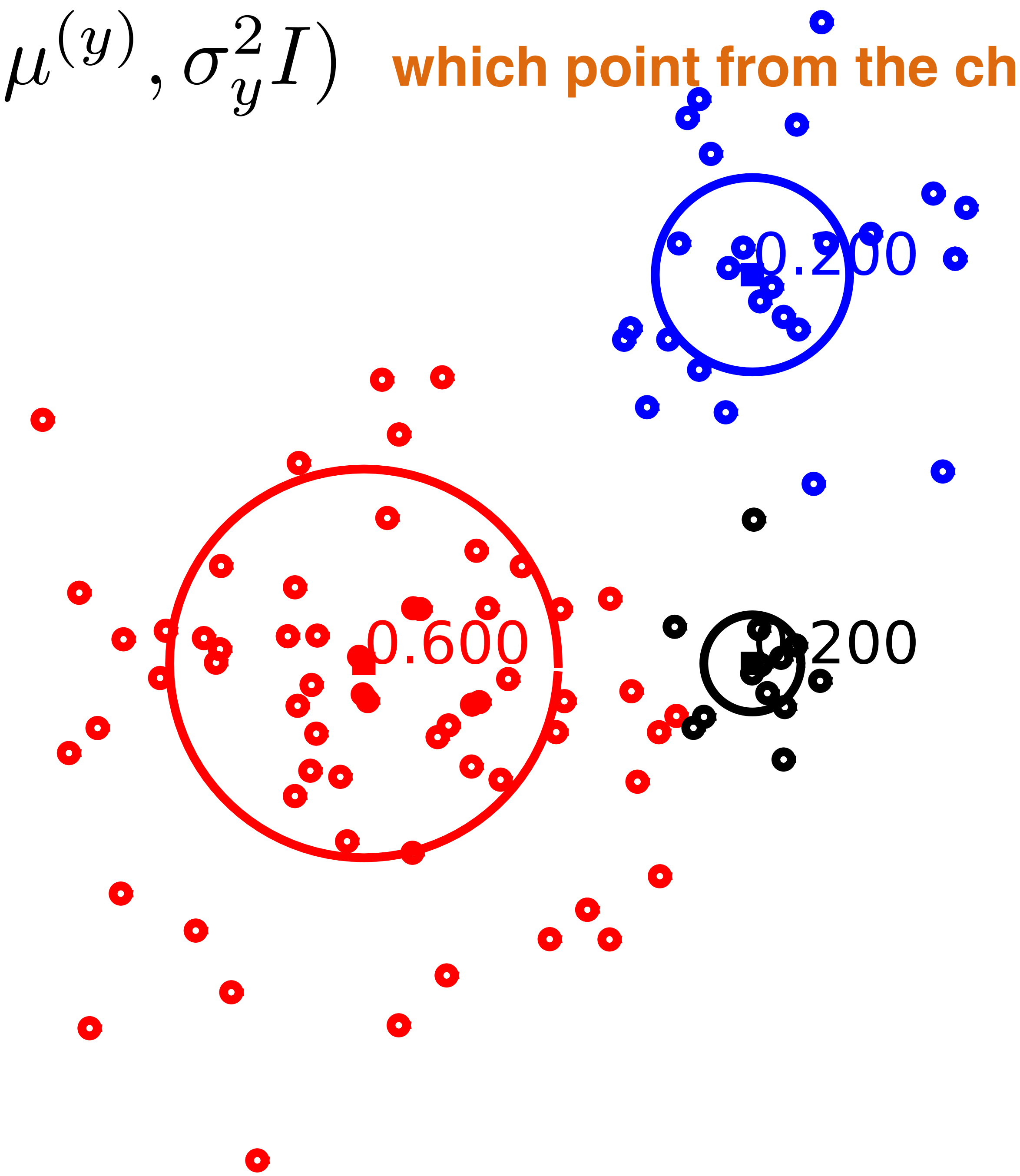


Mixture model: generation

- We can sample points from a mixture model in two steps

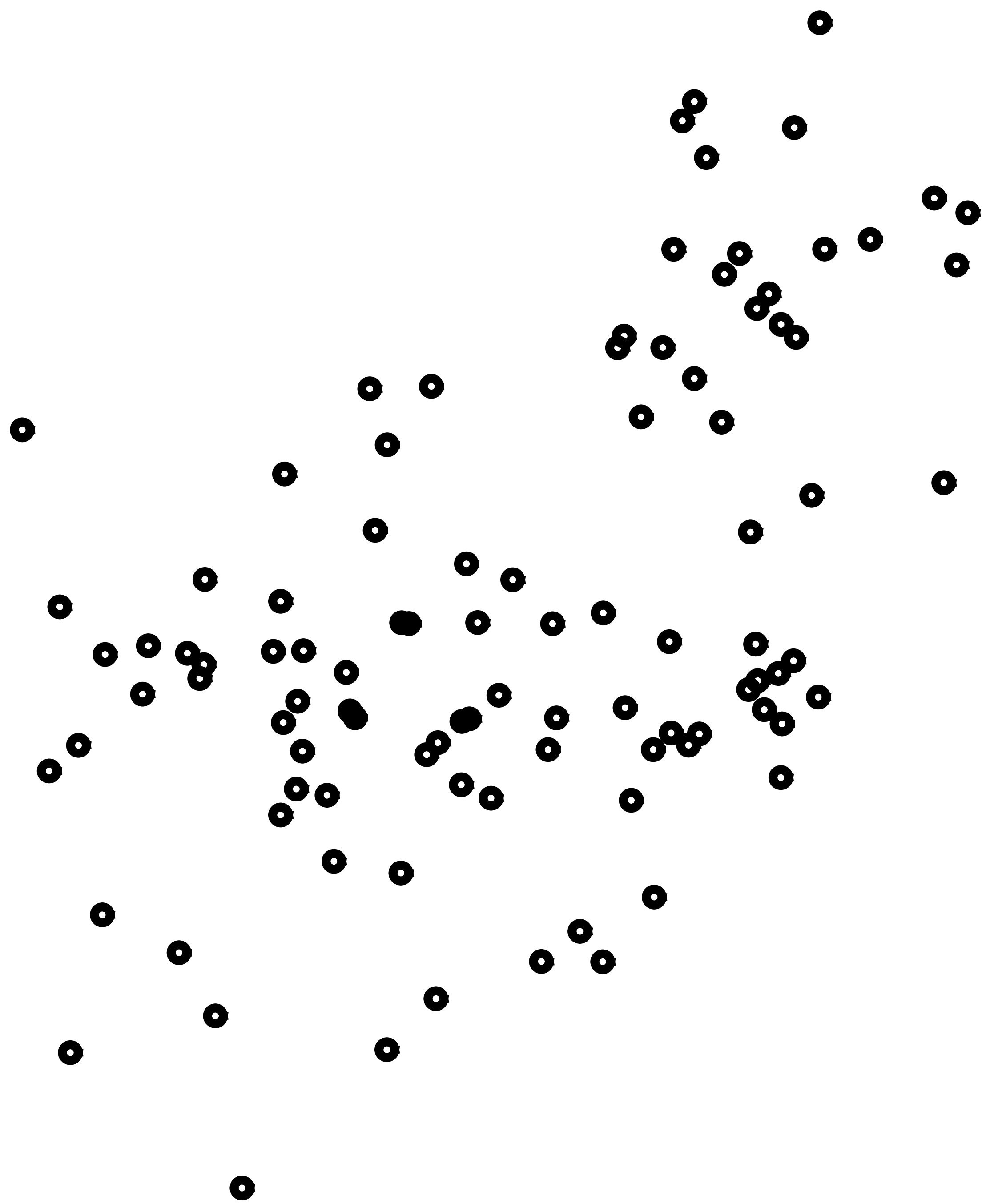
1) Sample $y \sim \text{Categ}(p_1, \ldots, p_K)$ **which cluster**

2) Sample $x \sim N(x; \mu^{(y)}, \sigma_y^2 I)$ **which point from the chosen cluster**



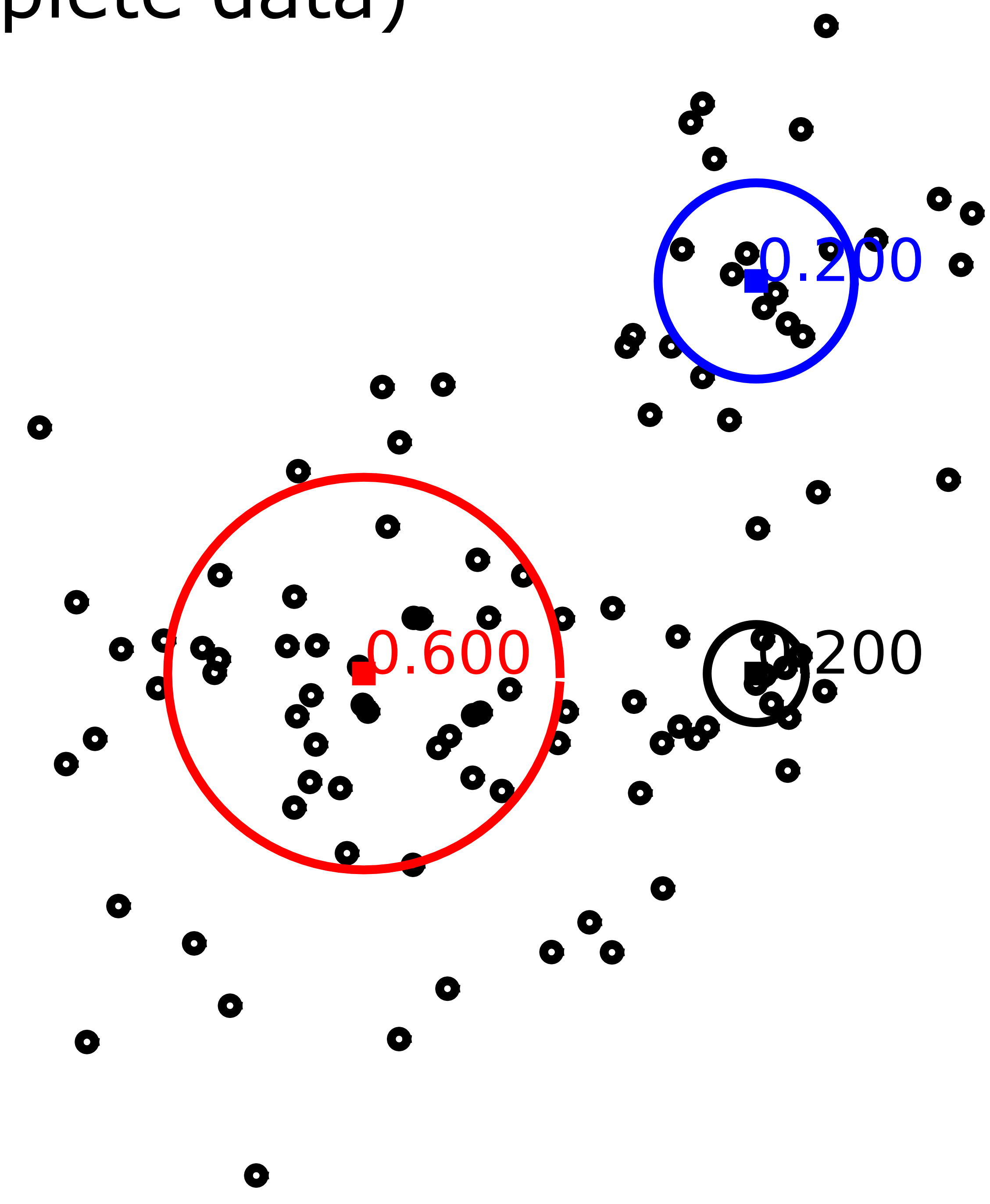
Mixture model: estimation

- But we typically don't get the cluster labels in the observed data, only the resulting points x



Mixture model: estimation

- Our goal is to try to estimate the underlying model (mixing proportions, component Gaussians) from the points alone (incomplete data)



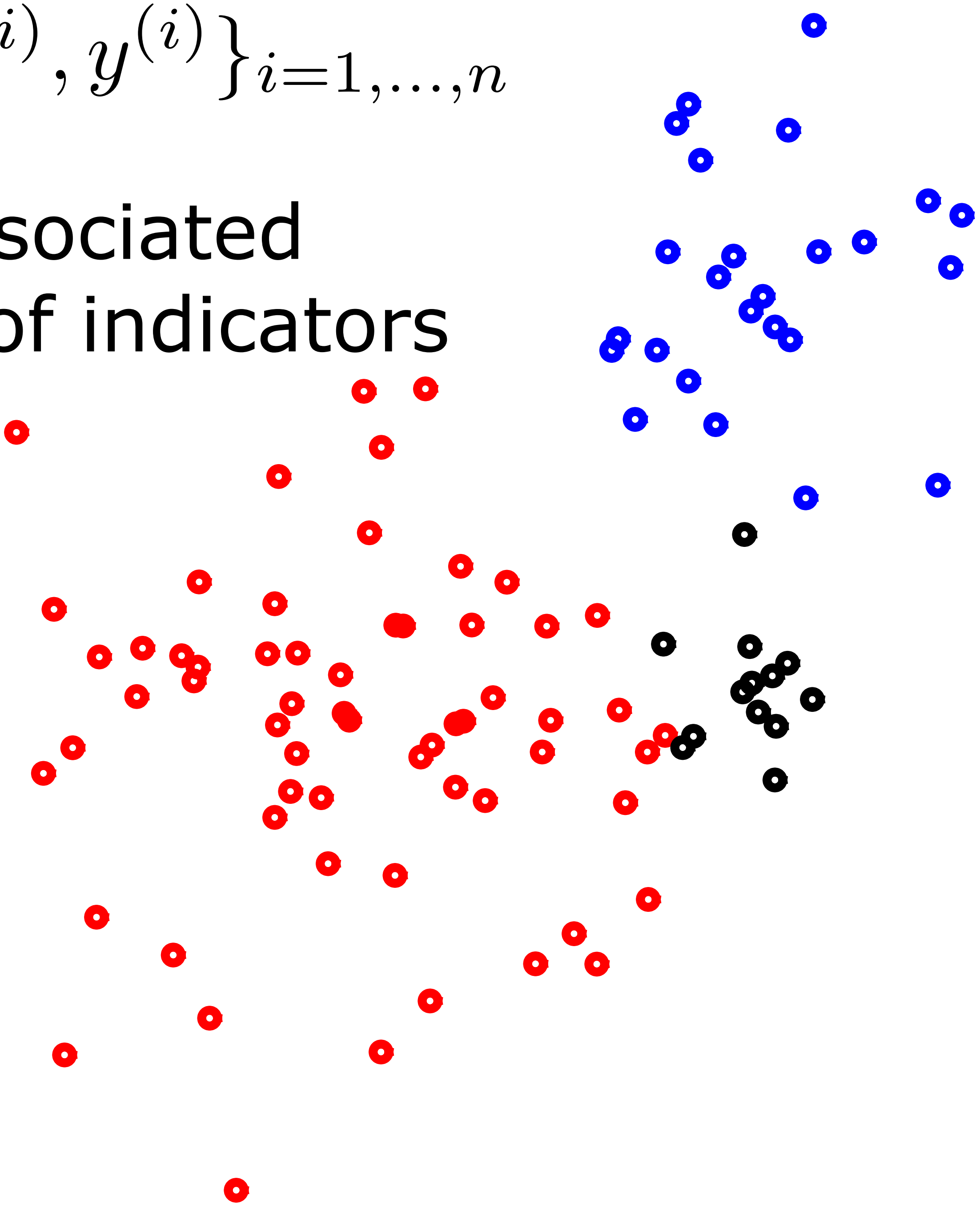
Complete data

- Estimation would be easy if we had complete data, i.e., pairs of cluster ids y and corresponding points x

$$\text{complete data} = \{x^{(i)}, y^{(i)}\}_{i=1,\ldots,n}$$

- We represent the associated cluster ids in terms of indicators

$$\delta(y|i) = 1 \text{ if } y = y^{(i)}$$
$$\text{and } 0 \text{ otherwise}$$



Complete data: mixing proportions

- We maximize the log-likelihood of complete data

$$\sum_{y=1}^{K}\sum_{i=1}^{n}\delta(y|i)\log p_y$$

- Resulting ML estimate

$$\hat{p}_y = \frac{\sum_{i=1}^{n}\delta(y|i)}{n}, \quad y = 1,\ldots,K$$

Complete data: Gaussians

- We maximize the log-likelihood of complete data (separately for each Gaussian)

$$\sum_{i=1}^{n} \delta(y|i) \log N(x^{(i)}; \mu^{(y)}, \sigma_y^2 I)$$

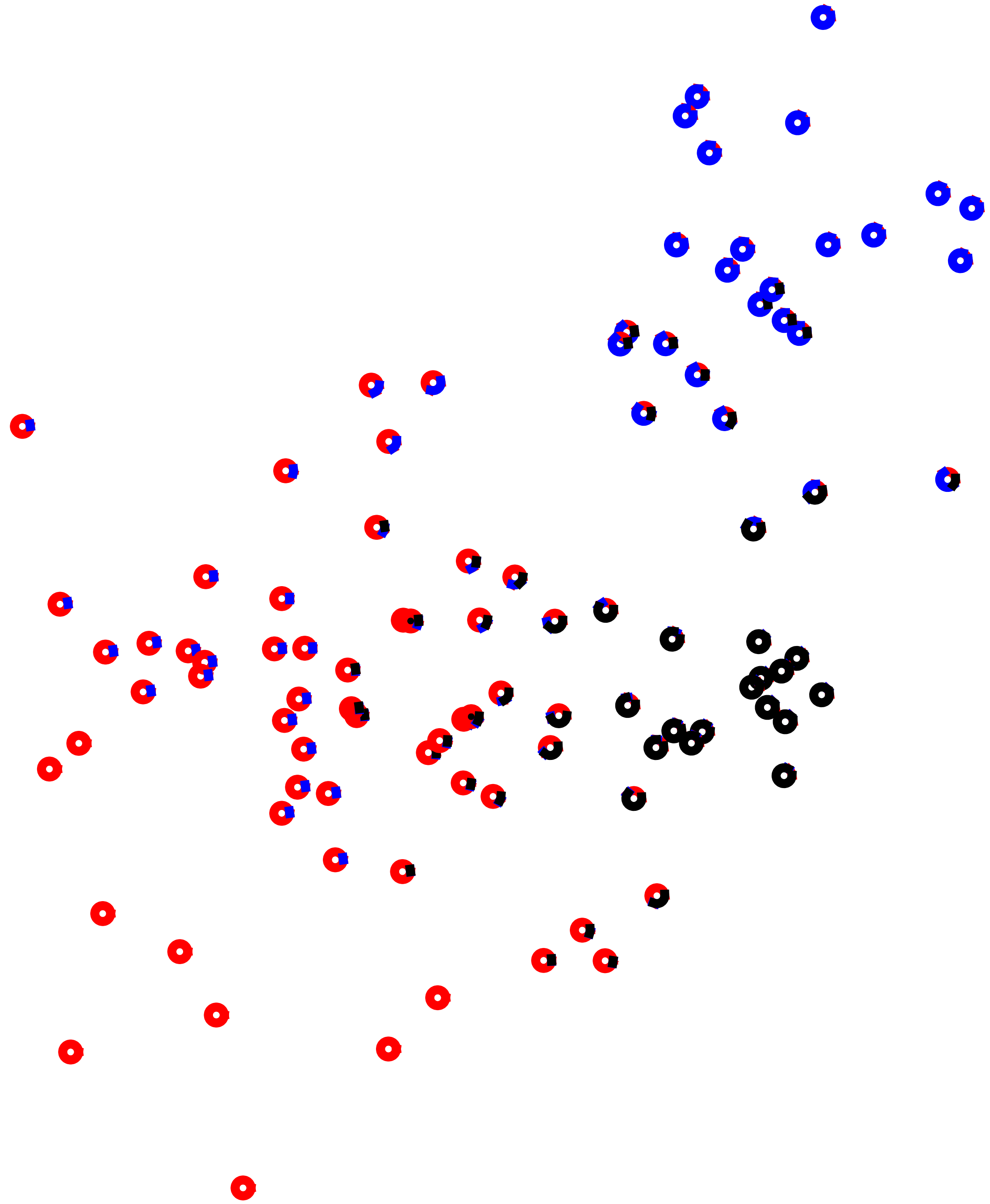
- The resulting ML estimates

$$\hat{\mu}^{(y)} = \frac{\sum_{i=1}^{n} \delta(y|i) x^{(i)}}{\sum_{i=1}^{n} \delta(y|i)}$$

$$\hat{\sigma}_y^2 = \frac{\sum_{i=1}^{n} \delta(y|i) \|x^{(i)} - \hat{\mu}^{(y)}\|^2}{d \sum_{i=1}^{n} \delta(y|i)}$$

Weighted data

- The estimation would be just as easy if instead of true cluster labels y, we had soft assignments



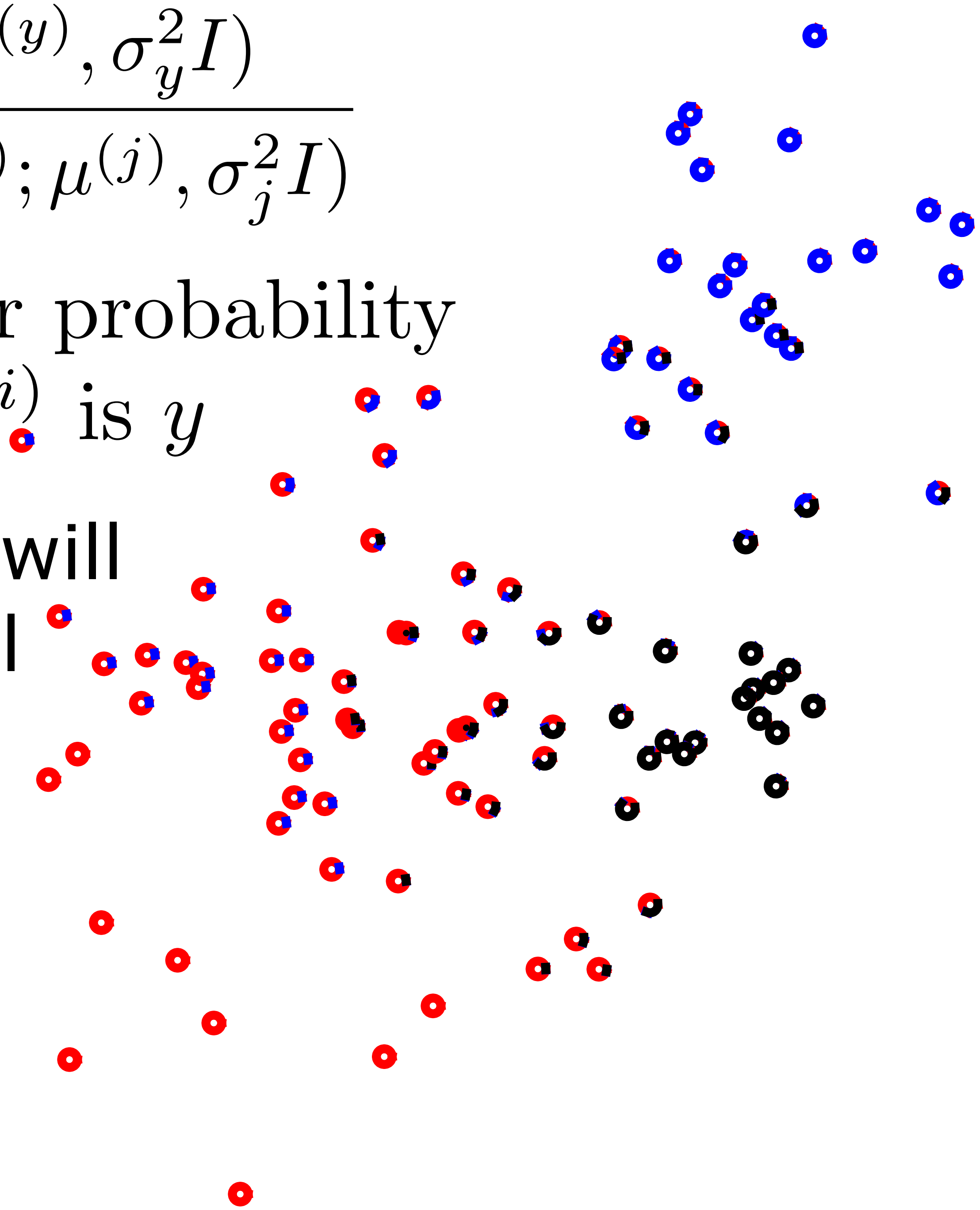
E-step: weighted data

- The estimation would be just as easy if instead of true cluster labels y, we had soft assignments

$$p(y|i) = \frac{p_y N(x^{(i)}; \mu^{(y)}, \sigma_y^2 I)}{\sum_{j=1}^{k} p_j N(x^{(i)}; \mu^{(j)}, \sigma_j^2 I)}$$

$p(y|i)$ is the posterior probability that the unknown $y^{(i)}$ is y

(these assignments will change as the model is updated)



M-step: mixing proportions

- We maximize the expected (weighted) log-likelihood

$$\sum_{y=1}^{K} \sum_{i=1}^{n} p(y|i) \log p_y$$

- The resulting estimate (cf. before)

$$\hat{p}_y = \frac{\sum_{i=1}^{n} p(y|i)}{n}, \quad y = 1, \ldots, K$$

M-step: Gaussians

- We maximize the expected (weighted) log-likelihood (separately for each Gaussian)

$$\sum_{i=1}^{n} p(y|i) \log N(x^{(i)}; \mu^{(y)}, \sigma_y^2 I)$$

- The resulting estimates (cf. before)

$$\hat{\mu}^{(y)} = \frac{\sum_{i=1}^{n} p(y|i) x^{(i)}}{\sum_{i=1}^{n} p(y|i)}$$

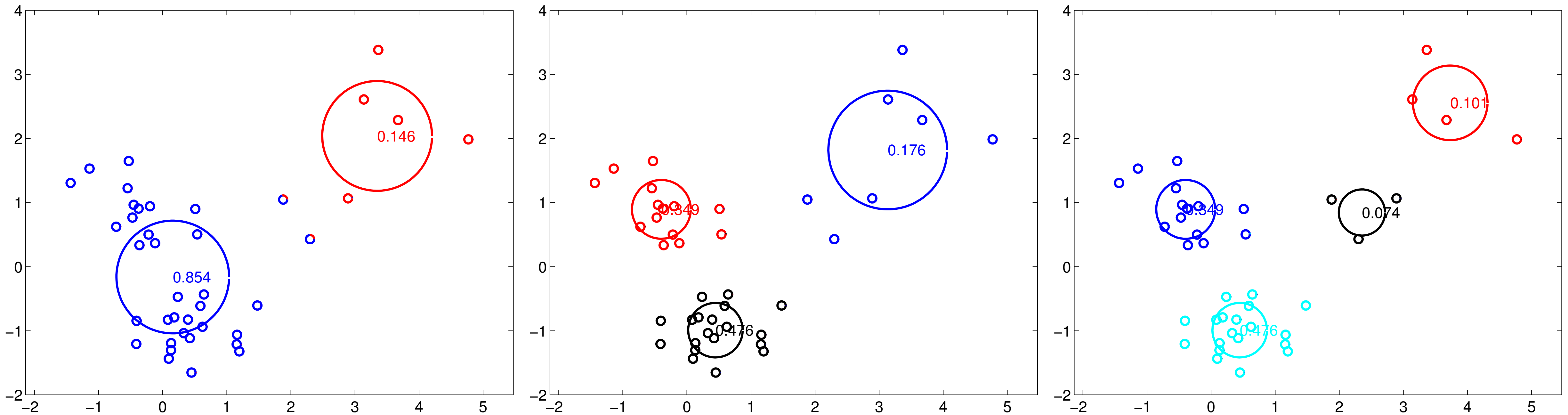
$$\hat{\sigma}_y^2 = \frac{\sum_{i=1}^{n} p(y|i) \|x^{(i)} - \hat{\mu}^{(y)}\|^2}{d \sum_{i=1}^{n} p(y|i)}$$

What did we learn?

- **Modeling**
  - generative models are specified by variables and how the variables are related
  - “latent variables” allow us to specify different underlying structures that we can (try to) uncover
- **Estimation (general)**
  - models can be estimated “in pieces” if we have complete data, i.e., observations with a value assignment for each variable
  - easy estimation extends to weighted complete data
- **Estimation (EM algorithm)**
  - EM algorithm iteratively creates weighted training sets (based on posterior assignments) and updates the model parameters, separately for each piece, based on the weighted data
  - the EM algorithm applies the same to other generative models where y specifies different latent choices (more later)

Model selection

- We can run the EM-algorithm with different numbers of components. Need to specify a criterion for selecting among the different models.



$l(D;\hat{\theta}) = -118.25$ $\quad$ $l(D;\hat{\theta}) = -98.64$ $\quad$ $l(D;\hat{\theta}) = -94.11$

- Basing the selecting on the value of log-likelihood would invariably lead to the largest number of components

Model selection: BIC

- Bayesian Information Criterion (BIC)

complexity penalty

$$BIC(D;\hat{\theta}) = l(D;\hat{\theta}) - \boxed{\frac{dim(\hat{\theta})}{2}\log(n)}$$

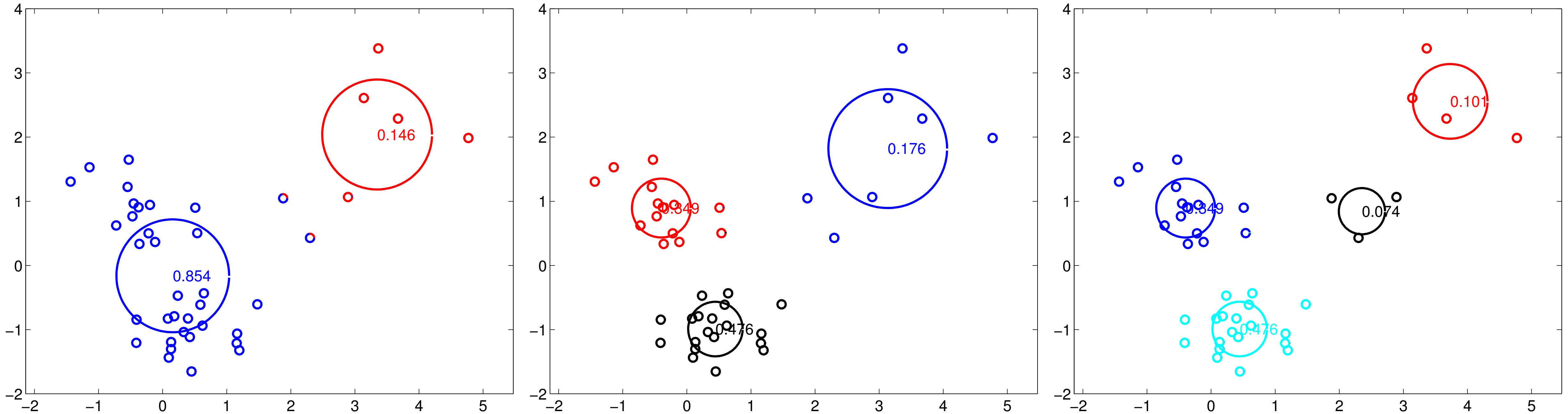
ML parameter estimates

# of parameters in the model

# of training examples

- Many closely related ways to think about “complexity” penalties in model selection criteria
  - how excess parameters help fit the data even in the absence of any signal
  - the cost of communicating the model (likelihood tells us the cost of transmitting the data given the model)
  - fraction of the parameter space that explains the data

Mixture models and BIC



$$l(D;\hat{\theta}) = -118.25 \qquad l(D;\hat{\theta}) = -98.64 \qquad l(D;\hat{\theta}) = -94.11$$

$$BIC(D;\hat{\theta}) = -131.16 \qquad BIC(D;\hat{\theta}) = -118.93 \qquad BIC(D;\hat{\theta}) = -121.78$$

Modeling sequences

- Lots of interesting data come in the form of sequences
  - temporal data (financial, monitoring, speech)
  - languages
  - user behavior
  - bio-sequences
  - etc.
- Our goal is to model such sequences, i.e., specify and learn probability distributions over sequences
  - specification (how to define, parameterize)
  - sampling (understand as a generative model)
  - estimation (learn from data)

Modeling sequences

- Consider a sequence of (e.g., binary) variables

$$y_1 \quad y_2 \quad y_3 \quad y_4 \quad \cdots \qquad\qquad y_i \in V$$

- We wish to specify a probability distribution over their values. By the chain rule, we can always write

$$P(y_1, \ldots, y_n) = P(y_1)P(y_2|y_1)P(y_3|y_2, y_1) \cdots P(y_n|y_{n-1}, \ldots, y_1)$$

  but, without any assumptions, we would need very large probability tables |V|^n

- The 1st order Markov assumption (bigram model):

$$P(y_1, \ldots, y_n) \stackrel{def}{=} P(y_1)P(y_2|y_1)P(y_3|y_2) \cdots P(y_n|y_{n-1})$$

Markov Model

- A 1st order (homogenous) Markov Model requires us to specify two sets of distributions

$$y_1 \longrightarrow y_2 \longrightarrow \cdots \longrightarrow y_{n-1} \longrightarrow y_n$$

- Initial state distribution:

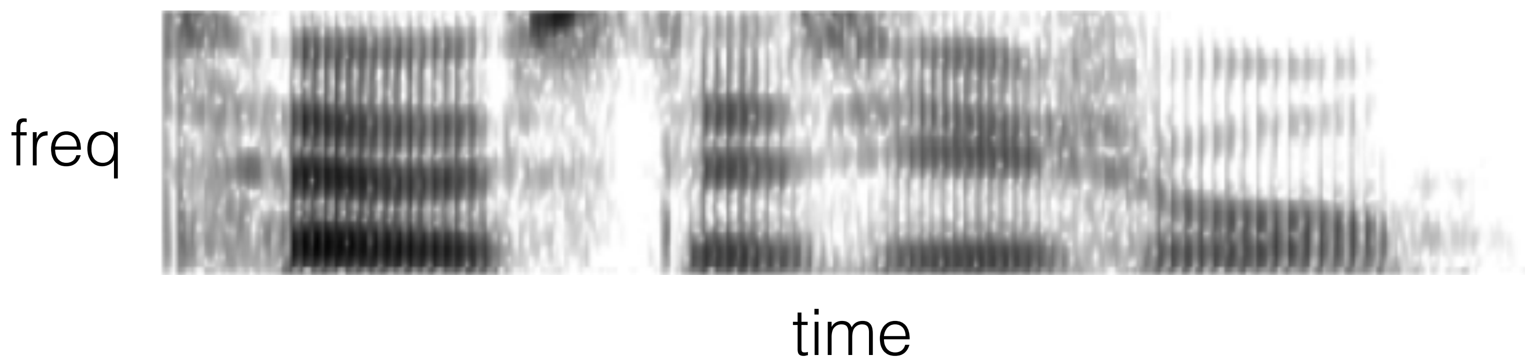
$$P(y_1 = y) = \pi_y, \quad \sum_{y=1}^{K} \pi_y = 1$$

- State transition probabilities:

$$P(y_i = y' | y_j = y) = a_{y,y'}, \quad \sum_{y'=1}^{K} a_{y,y'} = 1 \;\; \forall y$$

Modeling hidden sequences

- Speech recognition



- Handwriting recognition

- Information extraction, part of speech tagging, etc.
- Bio-sequence annotation
- Etc.

Modeling hidden sequences

- Similarly to mixture models, we must connect the latent selections to actual observations

$$\begin{array}{ccccccccc} y_1 & \rightarrow & y_2 & \rightarrow & \cdots & \rightarrow & y_{n-1} & \rightarrow & y_n \\ \downarrow & & \downarrow & & & & \downarrow & & \downarrow \\ x_1 & & x_2 & & \cdots & & x_{n-1} & & x_n \end{array}$$

- If the latent sequence is Markov, and the observations are only tied to the state at the corresponding time, we get a Hidden Markov Model (HMM)

$$P(y_1, \ldots, y_n, x_1, \ldots, x_n) = P(y_1)P(x_1|y_1) \prod_{i=2}^{n} \left[P(y_i|y_{i-1})P(x_i|y_i) \right]$$

HMM

- We need three sets of probabilities to specify a homogeneous HMM

$$\begin{tikzcd}
y_1 \arrow[r] \arrow[d] & y_2 \arrow[r] \arrow[d] & \cdots \arrow[r] & y_{n-1} \arrow[r] \arrow[d] & y_n \arrow[d] \\
x_1 & x_2 & \cdots & x_{n-1} & x_n
\end{tikzcd}$$

$$P(y_1 = y) = \pi_y, \quad \sum_{y=1}^{K} \pi_y = 1$$ **initial state distribution**

$$P(y_i = y' | y_{i-1} = y) = a_{y,y'}, \quad \sum_{y'=1}^{K} a_{y,y'} = 1 \ \ \forall y$$ **state transition**

$$P(x_i = x | y_i = y) = \left\{ \begin{array}{l} N(x; \mu^{(y)}, \sigma_y^2 I) \\ \text{or discrete} \end{array} \right. = b_y(x)$$ **observation model**

HMM

- We need three sets of probabilities to specify a homogeneous HMM

$$
\begin{tikzcd}
y_1 \arrow[r] \arrow[d] & y_2 \arrow[r] \arrow[d] & \cdots \arrow[r] & y_{n-1} \arrow[r] \arrow[d] & y_n \arrow[d] \\
x_1 & x_2 & \cdots & x_{n-1} & x_n
\end{tikzcd}
$$

$$P(y_1 = y) = \pi_y, \quad \sum_{y=1}^{K} \pi_y = 1$$ **initial state distribution**

$$P(y_i = y' | y_{i-1} = y) = a_{y,y'}, \quad \sum_{y'=1}^{K} a_{y,y'} = 1 \;\; \forall y$$ **state transition**

$$P(x_i = x | y_i = y) = \begin{cases} N(x; \mu^{(y)}, \sigma_y^2 I) \\ \text{or discrete} \end{cases} = b_y(x)$$ **observation model**

$$P(y_1, \ldots, y_n, x_1, \ldots, x_n; \theta) = \pi_{y_1} b_{y_1}(x_1) \prod_{i=2}^{n} [a_{y_{i-1}, y_i} b_{y_i}(x_i)]$$

HMM problems to solve

- **Evaluation:** calculate the probability of observed values, i.e., sum over all the underlying state sequences

$$P(x_1, \ldots, x_n) = \sum_{y_1, \ldots, y_n} P(y_1) P(x_1|y_1) \prod_{i=2}^{n} \left[P(y_i|y_{i-1}) P(x_i|y_i) \right]$$

- **Estimation:** EM algorithm for HMMs
  - E-step now needs a little effort…
  - M-step is easy (separate estimation of pieces)

✓ **Prediction:** find the most likely hidden state sequence responsible for the observations (Viterbi)

$$\hat{y}_1, \ldots, \hat{y}_n = \arg \max_{y_1, \ldots, y_n} P(y_1, \ldots, y_n, x_1, \ldots, x_n)$$

Viterbi algorithm

- We can find the most likely hidden sequence using dynamic programming (focusing first on the max value)

$$d_i(y_i) = \max_{y_1,\ldots,y_{i-1}} P(y_1,\ldots,y_i,x_1,\ldots,x_i)$$

- Recursion (should be implemented on a log-scale)

$$d_1(y_1) = P(y_1)P(x_1|y_1)$$

$$d_i(y_i) = \max_{y_{i-1}} \left\{ d_{i-1}(y_{i-1})P(y_i|y_{i-1})P(x_i|y_i) \right\}$$

- The end result of the recursion is that

$$\max_{y_n} d_n(y_n) = \max_{y_1,\ldots,y_n} P(y_1,\ldots,y_n,x_1,\ldots,x_n)$$

Viterbi, backtracking

- We can reconstitute the maximizing sequence by working backwards, instantiating the argmax's

$$\hat{y}_n = \arg\max_{y_n} d_n(y_n)$$

$$\hat{y}_{i-1} = \arg\max_{y_{i-1}} \left\{ d_{i-1}(y_{i-1}) P(\hat{y}_i | y_{i-1}) P(x_i | y_i) \right\}$$

Key things to know

- Mixture models
  - specification, sampling, estimation
  - the EM algorithm (for mixtures)
- Markov Models
  - specification, sampling, ML estimation
- Hidden Markov Models
  - specification, sampling, (estimation qualitatively)
  - Viterbi algorithm for prediction