

CS195-5 : Introduction to Machine Learning

Lecture 19

Greg Shakhnarovich

October 30, 2006

Announcements

Plan for today

Nearest neighbor: extensions

- We can use $k > 1$ nearest neighbors \Rightarrow k -NN classifier
 - Label for \mathbf{x}_0 predicted by majority voting among its k -NN.
- What about regression?

Nearest neighbor: extensions

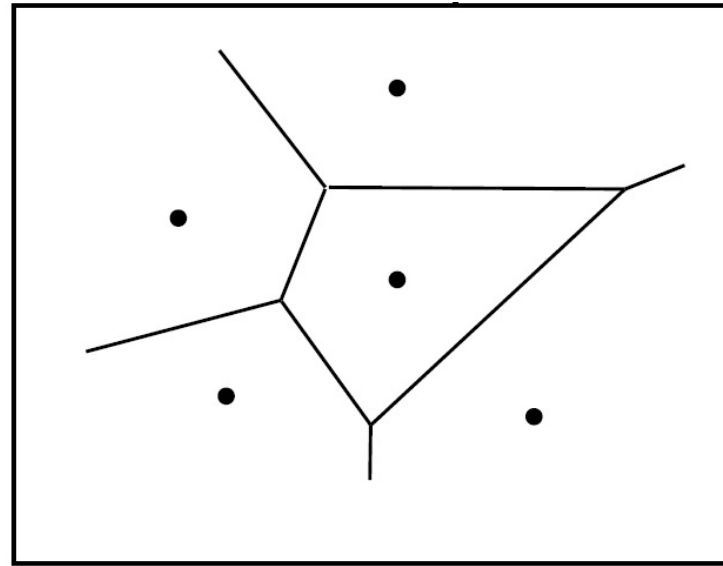
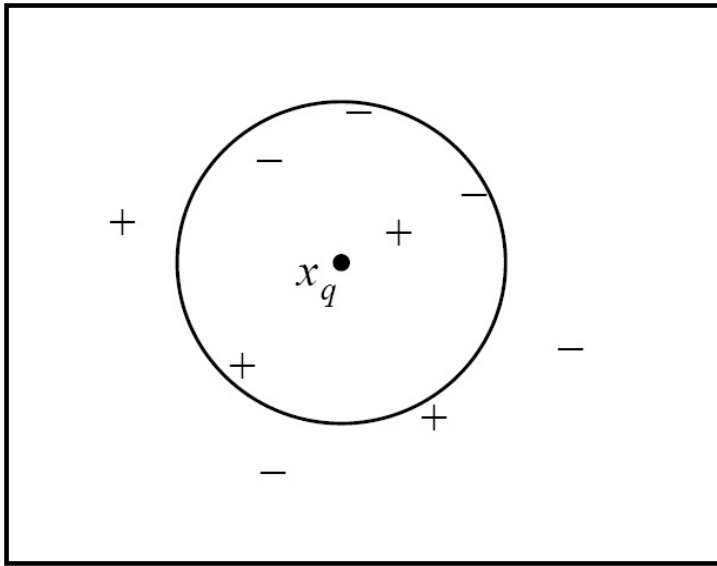
- We can use $k > 1$ nearest neighbors \Rightarrow k -NN classifier
 - Label for \mathbf{x}_0 predicted by majority voting among its k -NN.
- What about regression? Simplest k -NN regression: let $\mathbf{x}'_1, \dots, \mathbf{x}'_k$ be the neighbors, and y'_1, \dots, y'_k their labels.
 - Predict $\hat{y} = \frac{1}{k} \sum_{j=1}^k y'_j$.
- What kind of functions can we estimate in this way?

Nearest neighbor: extensions

- We can use $k > 1$ nearest neighbors \Rightarrow k -NN classifier
 - Label for \mathbf{x}_0 predicted by majority voting among its k -NN.
- What about regression? Simplest k -NN regression: let $\mathbf{x}'_1, \dots, \mathbf{x}'_k$ be the neighbors, and y'_1, \dots, y'_k their labels.
 - Predict $\hat{y} = \frac{1}{k} \sum_{j=1}^k y'_j$.
- What kind of functions can we estimate in this way?
- What is the effect of k ?

Geometry of nearest neighbor

- NN induce *Voronoi tessellation* of the space:



Locally weighted regression

- Idea 1: bring back the kernel.

$$\hat{y} = \frac{1}{Z} \sum_{j=1}^k y'_j K(\mathbf{x}_0, \mathbf{x}'_j),$$

with normalization $Z = \sum_j K(\mathbf{x}_0, \mathbf{x}'_j)$.

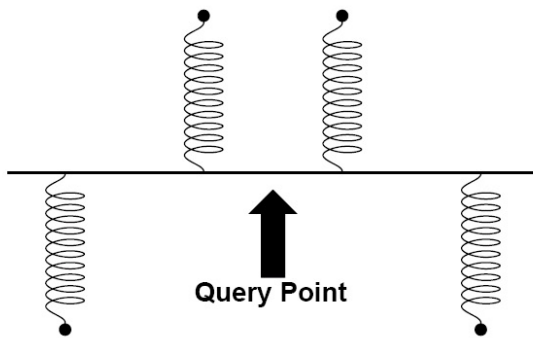


Figure 2: Unweighted averaging using springs.

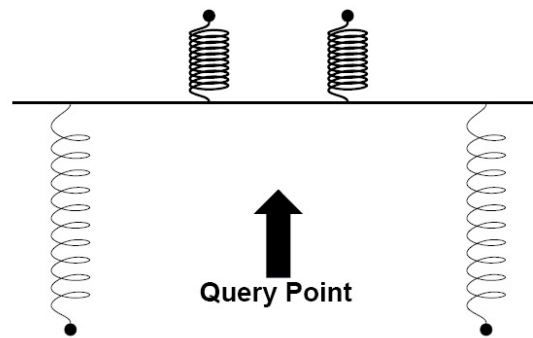
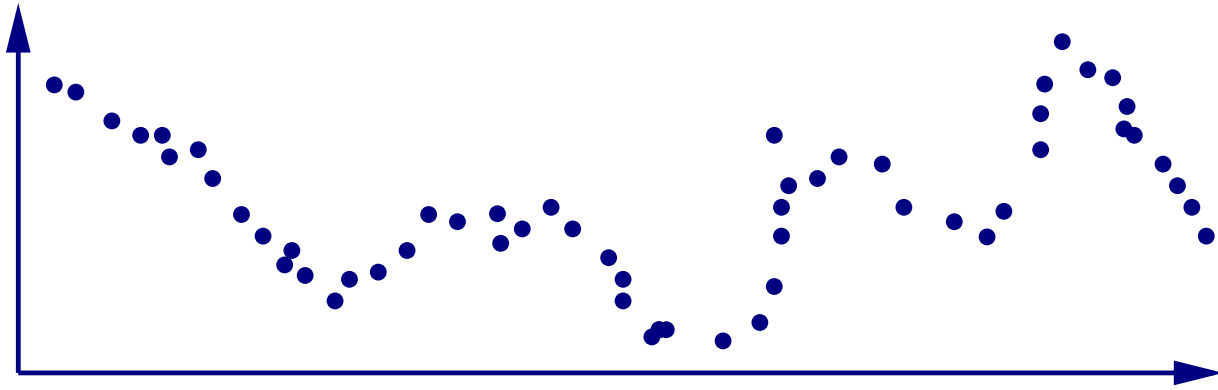


Figure 3: Locally weighted averaging using springs.

from Atkeson et al.

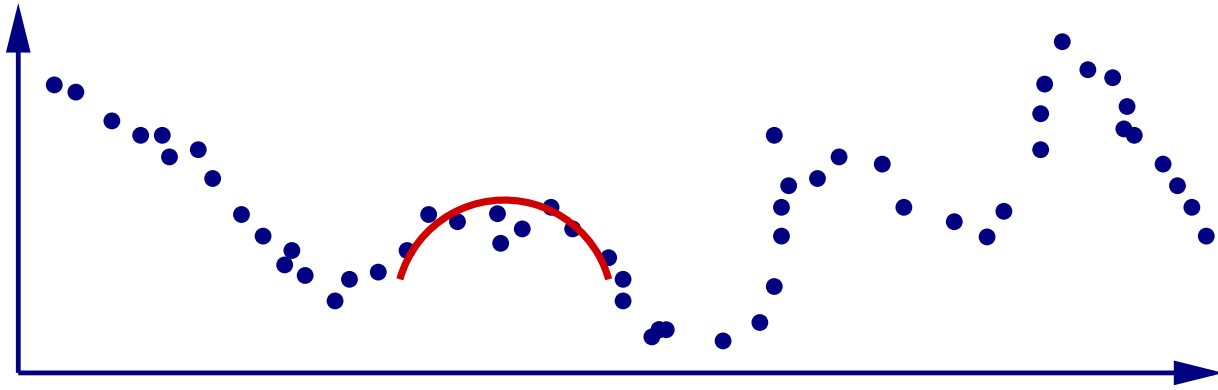
Parametric locally weighted regression

- Idea 2: bring back the parameters.
- Fit a (simple) parametric model to the neighbors of \mathbf{x}_0 .



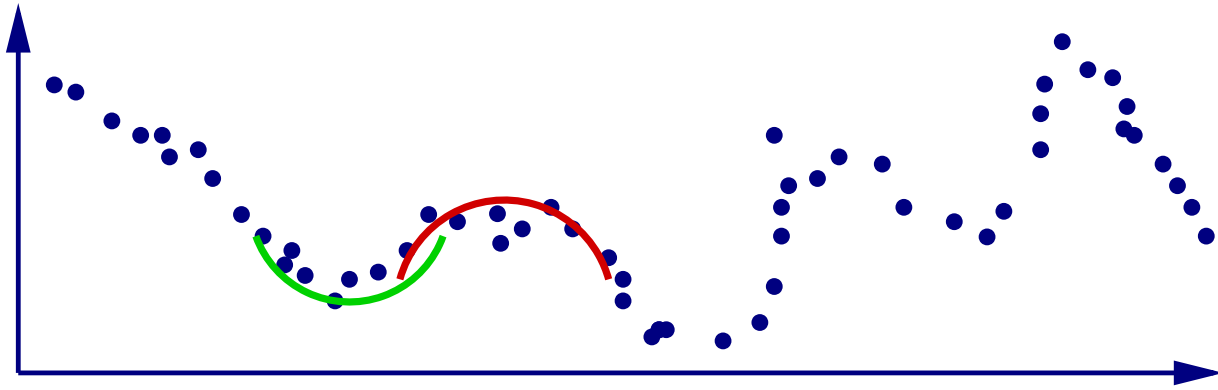
Parametric locally weighted regression

- Idea 2: bring back the parameters.
- Fit a (simple) parametric model to the neighbors of \mathbf{x}_0 .



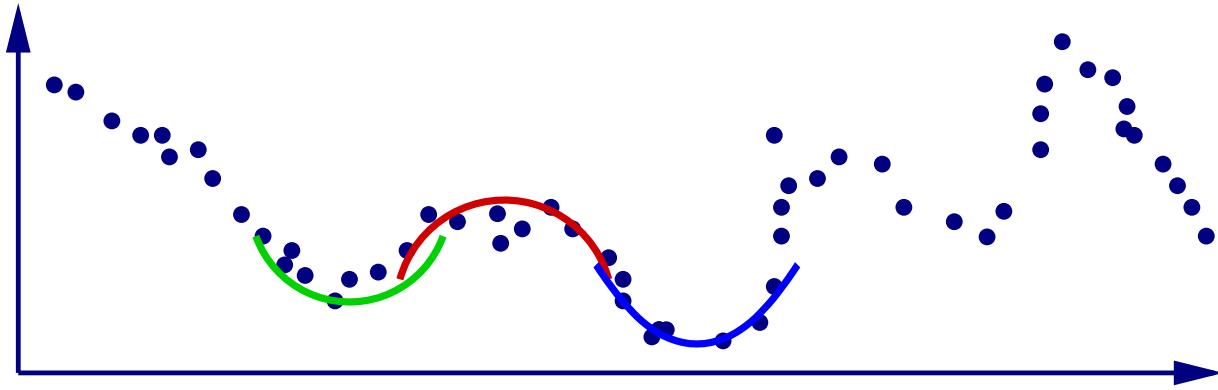
Parametric locally weighted regression

- Idea 2: bring back the parameters.
- Fit a (simple) parametric model to the neighbors of \mathbf{x}_0 .



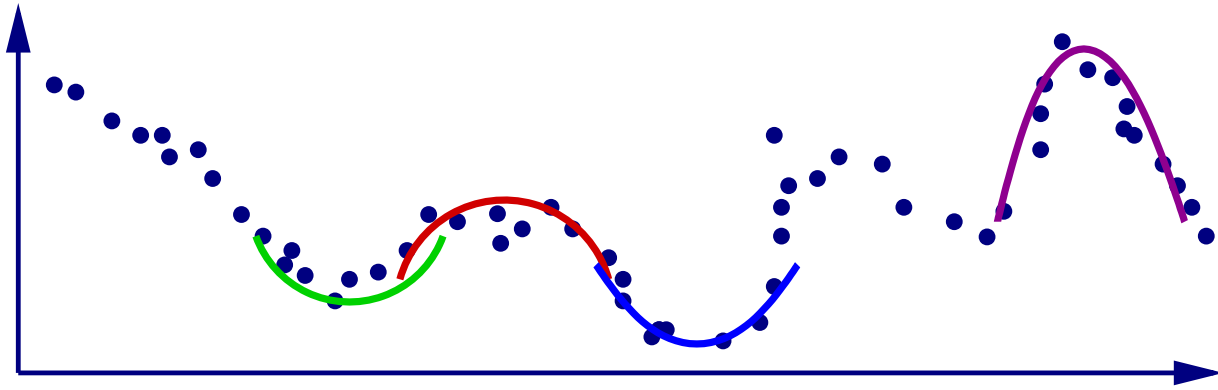
Parametric locally weighted regression

- Idea 2: bring back the parameters.
- Fit a (simple) parametric model to the neighbors of \mathbf{x}_0 .



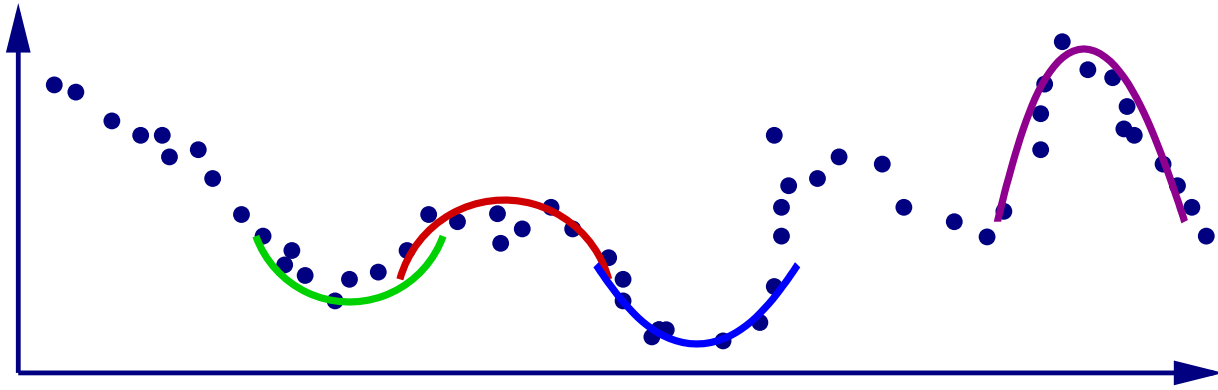
Parametric locally weighted regression

- Idea 2: bring back the parameters.
- Fit a (simple) parametric model to the neighbors of \mathbf{x}_0 .



Parametric locally weighted regression

- Idea 2: bring back the parameters.
- Fit a (simple) parametric model to the neighbors of \mathbf{x}_0 .



- Implicit assumption: the target function is reasonably smooth.

Example: linear LWR

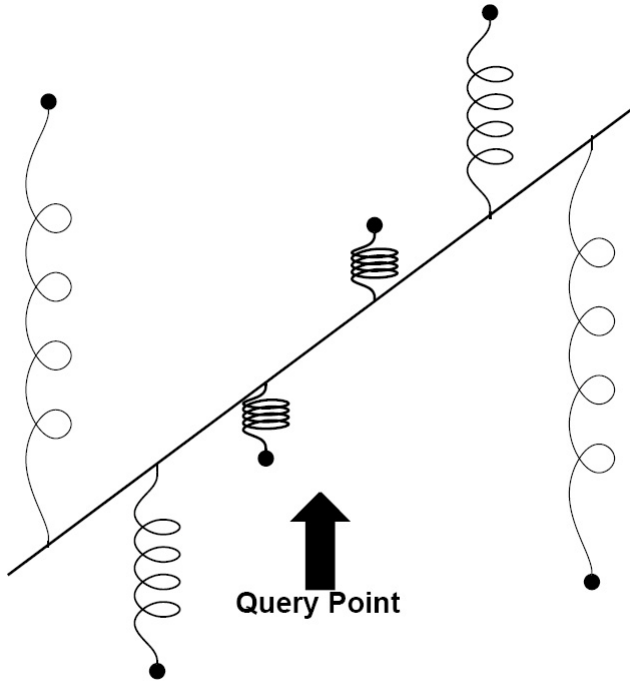


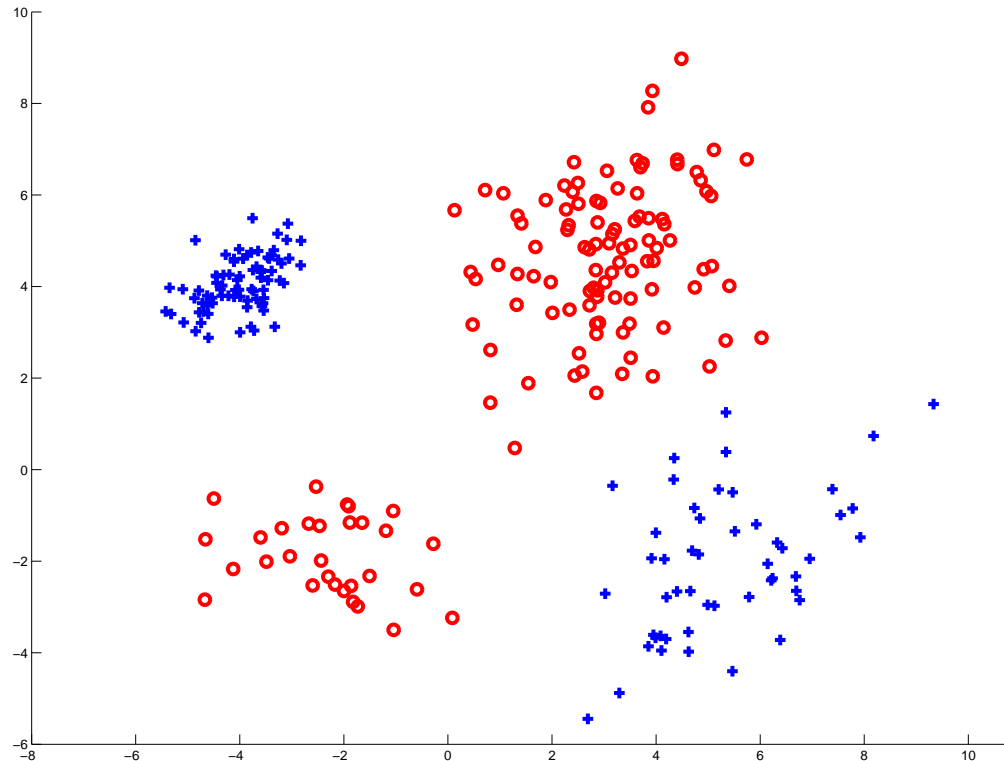
Figure 5: Weighted springs.

from Atkeson et al.
Spring stiffness $\Leftrightarrow K(\mathbf{x}_0, \mathbf{x}_i)$

- What kind of functions can we estimate with this model?

Mixture models

- So far, we have assumed that each class has a single coherent model.
- What if the examples (within the same class) are from a number of distinct “types”?

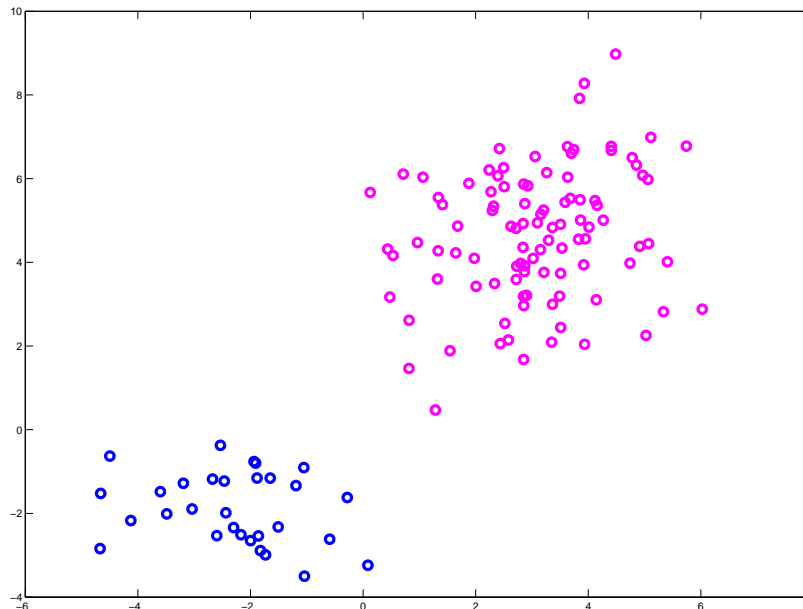


Examples

- Images of the same person under different conditions: with/without glasses, different expressions, different views.
- Images of the same category but different sorts of objects: chairs with/without armrests.
- Multiple topics within the same document.
- Different ways of pronouncing the same phonemes.

Mixture models

- Assumptions:
 - k underlying types (components);
 - y_i is the identity of the component “responsible” for \mathbf{x}_i ;
 - y_i is a *hidden (latent)* variable: never observed.
- A *mixture model*:



$$p(\mathbf{x}; \mathbf{p}) = \sum_{c=1}^k p(y = c)p(\mathbf{x} | y = c).$$

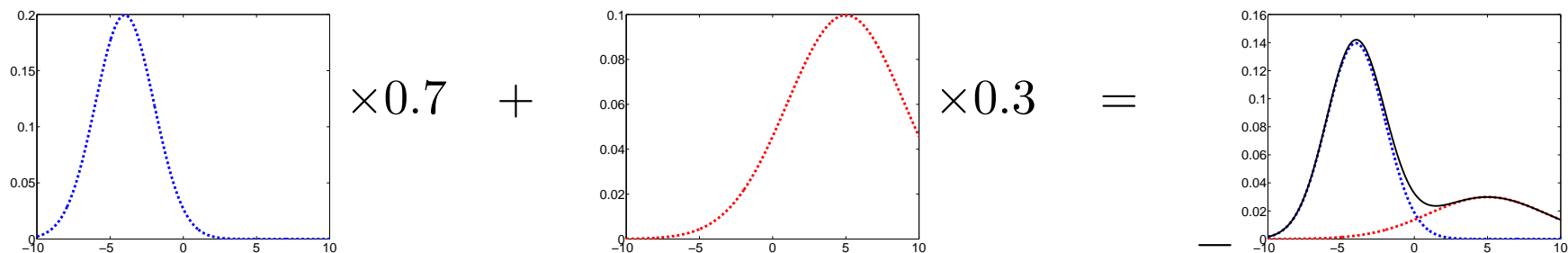
- $p_c \triangleq p(y = c)$ are the *mixing probabilities*
- We need to parametrize the component densities $p(\mathbf{x} | y = c)$.

Parametric mixtures

- Suppose that the parameters of the c -th component are θ_c . Then we can denote $\theta = [\theta_1, \dots, \theta_k]$ and write

$$p(\mathbf{x}; \theta, \mathbf{p}) = \sum_{c=1}^k p_c \cdot p(\mathbf{x}; \theta_c).$$

- Any valid setting of θ and \mathbf{p} , subject to $\sum_{c=1}^k p_c = 1$, produces a valid pdf.
- Example: mixture of Gaussians.

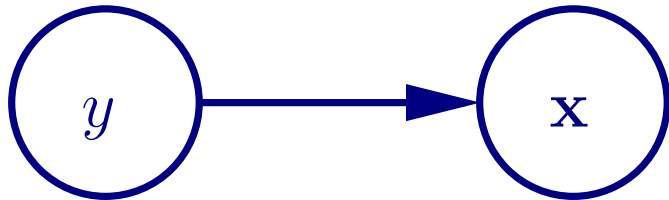


Generative model for a mixture

- The generative process with k -component mixture:
 - The parameters θ_c for each component c are fixed.
 - Draw $y_i \sim [p_1, \dots, p_k]$;
 - Given y_i , draw $\mathbf{x}_i \sim p(\mathbf{x} | y_i)$.

Generative model for a mixture

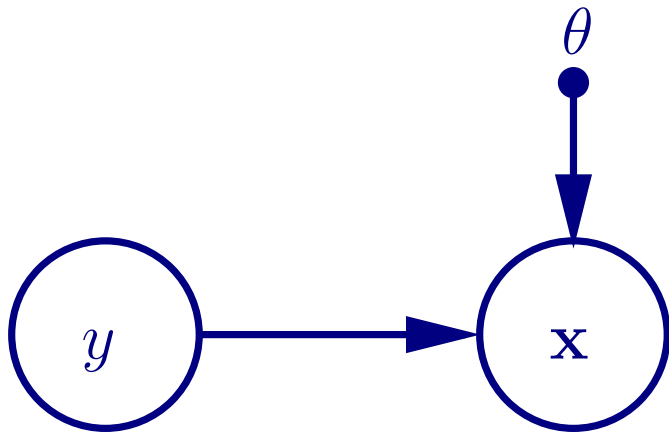
- The generative process with k -component mixture:
 - The parameters θ_c for each component c are fixed.
 - Draw $y_i \sim [p_1, \dots, p_k]$;
 - Given y_i , draw $\mathbf{x}_i \sim p(\mathbf{x} | y_i)$.
 - The *graphical model* representation:



$$p(\mathbf{x}, y) = p(y) \cdot p(\mathbf{x}|y; \theta)$$

Generative model for a mixture

- The generative process with k -component mixture:
 - The parameters θ_c for each component c are fixed.
 - Draw $y_i \sim [p_1, \dots, p_k]$;
 - Given y_i , draw $\mathbf{x}_i \sim p(\mathbf{x} | y_i)$.
 - The *graphical model* representation:

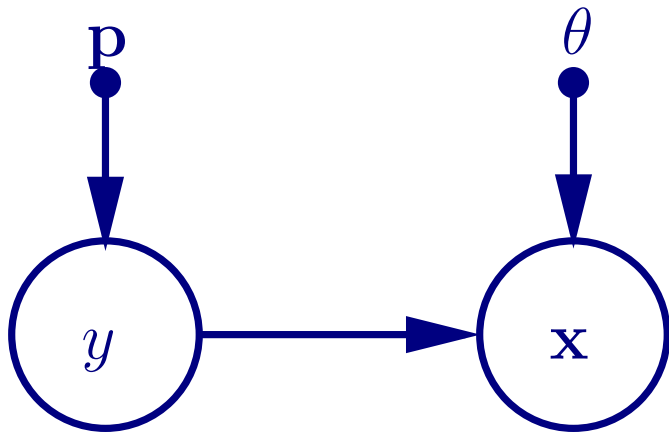


$$p(\mathbf{x}, y) = p(y) \cdot p(\mathbf{x}|y; \theta)$$

- Any data point \mathbf{x}_i could have been generated in k ways.

Generative model for a mixture

- The generative process with k -component mixture:
 - The parameters θ_c for each component c are fixed.
 - Draw $y_i \sim [p_1, \dots, p_k]$;
 - Given y_i , draw $\mathbf{x}_i \sim p(\mathbf{x} | y_i)$.
 - The *graphical model* representation:



$$p(\mathbf{x}, y) = p(y) \cdot p(\mathbf{x}|y; \theta)$$

- Any data point \mathbf{x}_i could have been generated in k ways.

Gaussian mixture models

- If the c -th component is a Gaussian, $p(\mathbf{x} | y = c) = \mathcal{N}(\mathbf{x}; \mu_c, \Sigma_c)$, then

$$p(\mathbf{x}; \theta, \mathbf{p}) = \sum_{c=1}^k p_c \cdot \mathcal{N}(\mathbf{x}; \mu_c, \Sigma_c),$$

where $\theta = [\mu_1, \dots, \mu_k, \Sigma_1, \dots, \Sigma_k]$.

- The graphical model

Mixture density estimation

- Suppose that we do observe $y_i \in \{1, 2\}$ for each $i = 1, \dots, N$.
- Let us introduce a set of binary *indicator variables* $\mathbf{z}_i = [z_{i1}, \dots, z_{ik}]$ where

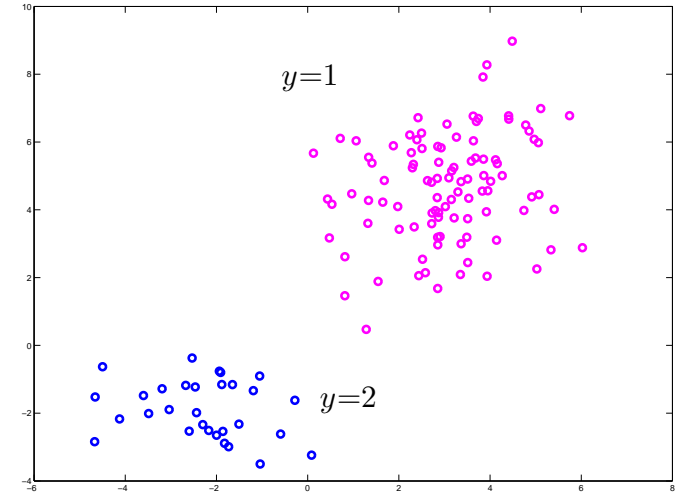
$$z_{ic} = 1 = \begin{cases} 1 & \text{if } y_i = c, \\ 0 & \text{otherwise.} \end{cases}$$

- The count of examples from c -th component:

$$N_c = \sum_{i=1}^N z_{ic}.$$

Mixture density estimation: known labels

- If we know \mathbf{z}_i , the ML estimates of the Gaussian components, just like in class-conditional model, are



$$\hat{p}_c = \frac{N_c}{N},$$

$$\hat{\mu}_c = \frac{1}{N_c} \sum_{i=1}^N z_{ic} \mathbf{x}_i,$$

$$\hat{\Sigma}_c = \frac{1}{N_c} \sum_{i=1}^N z_{ic} (\mathbf{x}_i - \hat{\mu}_c)(\mathbf{x}_i - \hat{\mu}_c)^T.$$

Credit assignment

- When we don't know y , we face a *credit assignment* problem: which component is responsible for \mathbf{x}_i ?
- Suppose for a moment that we do know component parameters $\theta = [\mu_1, \dots, \mu_k, \Sigma_1, \dots, \Sigma_k]$ and mixing probabilities $\mathbf{p} = [p_1, \dots, p_k]$.
- Then, we can compute the posterior of each label using Bayes' theorem (again, just like with classification):

$$\hat{p}(y = c | \mathbf{x}; \theta, \mathbf{p}) = \frac{p_c \cdot p(\mathbf{x}; \mu_c, \Sigma_c)}{\sum_{l=1}^k p_l \cdot p(\mathbf{x}; \mu_l, \Sigma_l)}$$

- We will call $\hat{p}(y = c | \mathbf{x}; \theta, \mathbf{p})$ the *responsibility* of the c -th component for \mathbf{x} .
 - Note: $\sum_{c=1}^k \hat{p}(y = c | \mathbf{x}; \theta, \mathbf{p}) = 1$.

EM: the intuition

- We are interested in the scenario when we know neither θ , \mathbf{p} nor \mathbf{z} .
- The idea: iterate between

Next time

Continue with the EM.