Machine Learning:
a methodology survey with practical examples

Giampiero Salvi
KTH CSC TMH
giampi@kth.se

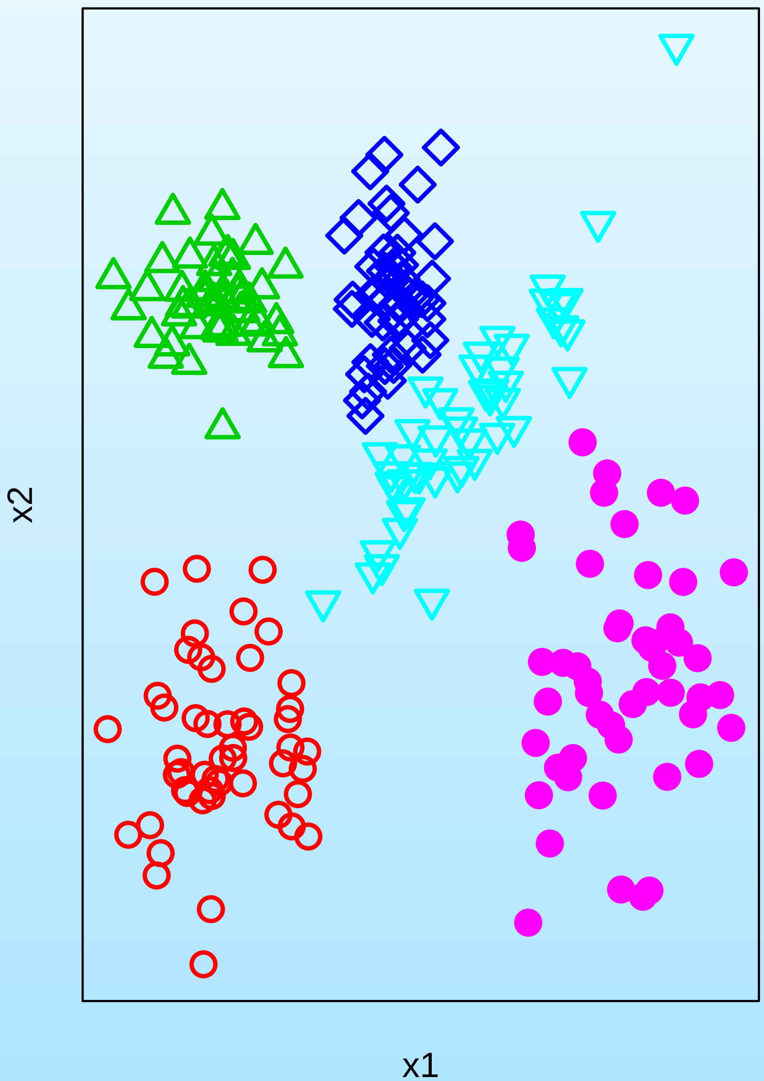
Outline

- What is learning?
- Parametric methods
- Non-parametric methods
- Stochastic methods
- Non-metric methods (skip)
- Universal principles
- Unsupervised learning
- Examples

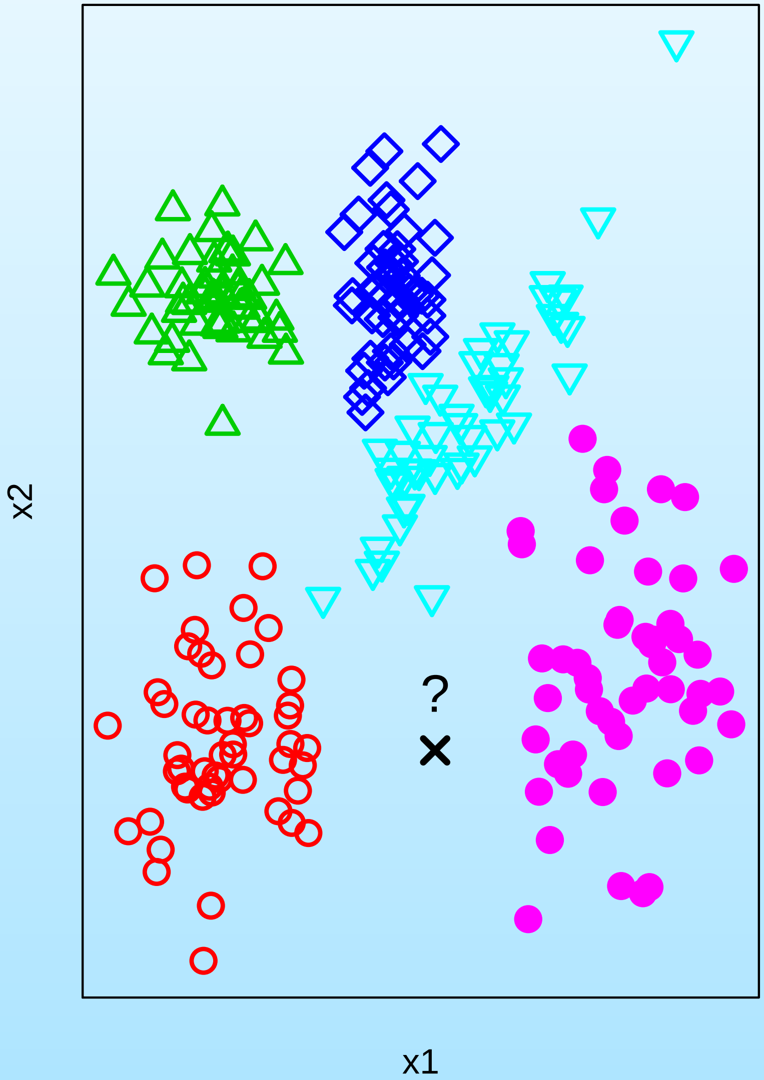
What is learning?

- the process of acquiring *knowledge* from *experience*
- focus on observations that can be described in terms of measurable quantities
  - an observation corresponds to a point $\mathbf{x} \in \mathbb{R}^d$
- given a set of observations $\mathcal{D} = \{\mathbf{x}_i\}$ say something about its structure or about a new observation $\mathbf{x}$

Classification

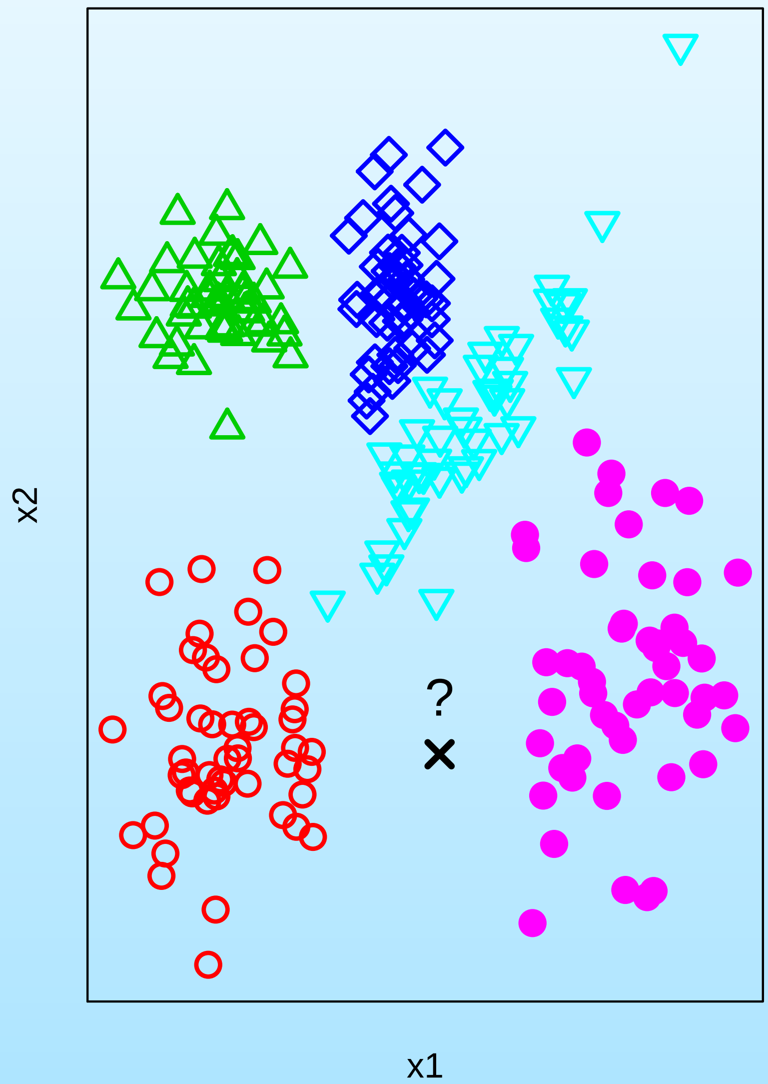


Classification

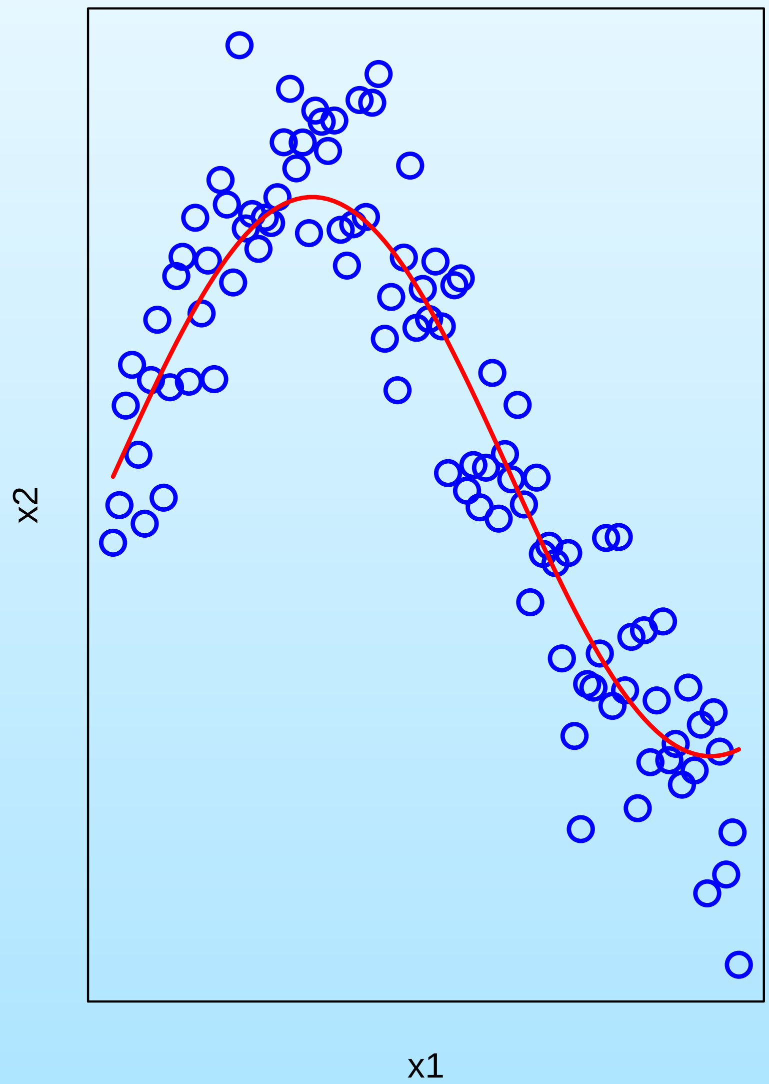


Supervised learning

Classification

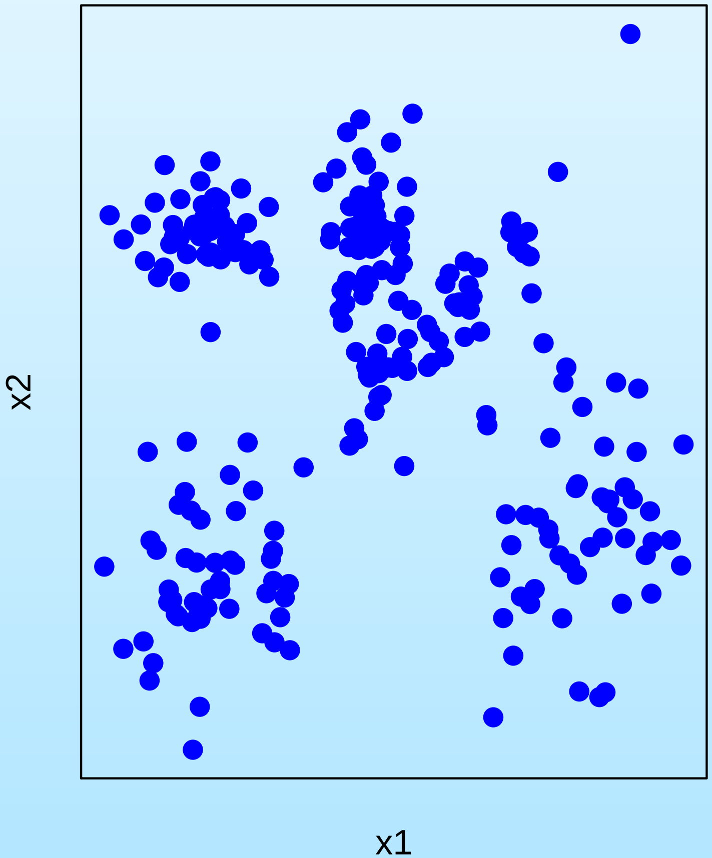


Regression

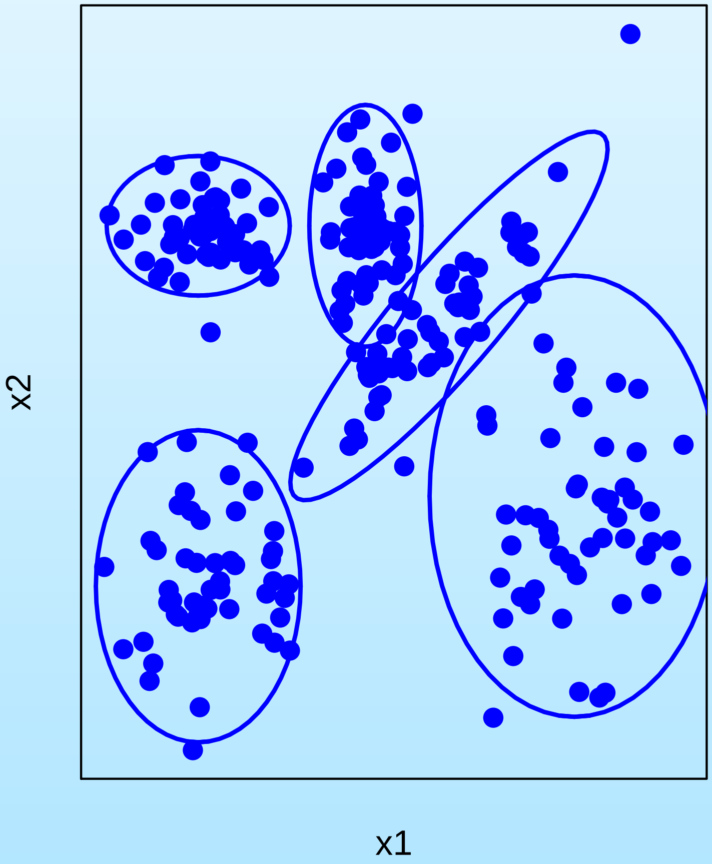


Unsupervised learning

Clustering

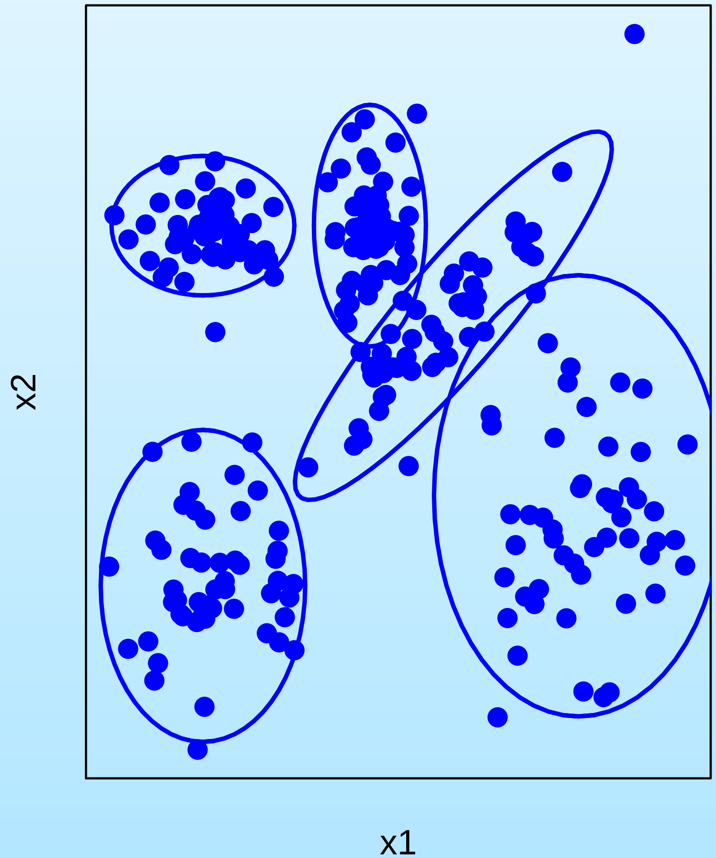


Clustering

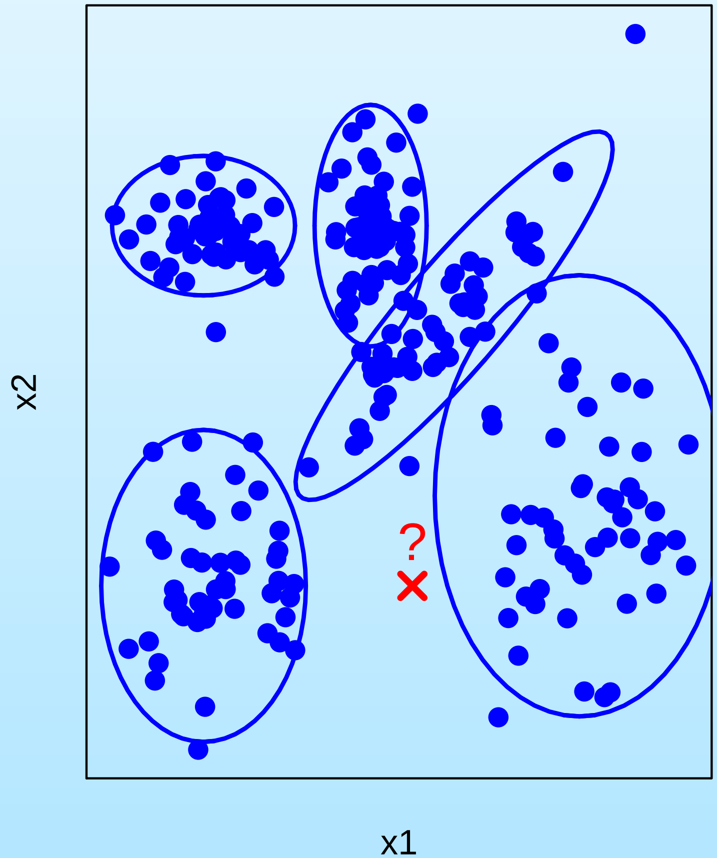


Unsupervised learning

Clustering



Classification

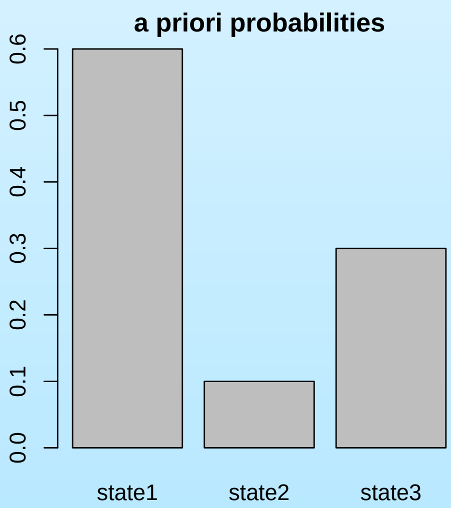


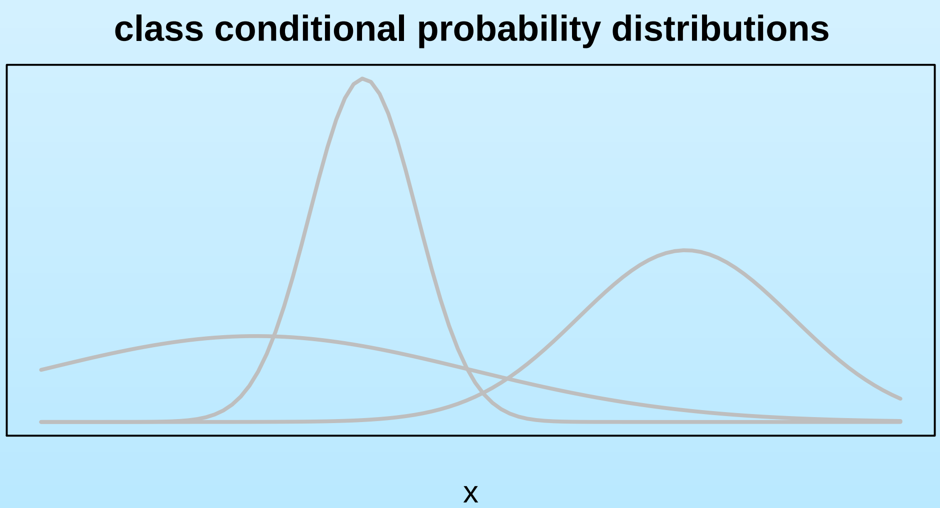
The theory behind

- **parametric methods**
  - probabilistic assumption on the generation of the data $\mathcal{D} = \{\mathbf{x}_i\}$
  - known functional shape of probability distributions, but unknown parameters
- **non parametric**
  - the shape of the distribution is not known
  - no probabilistic assumption at all (heuristics)

The probabilistic model

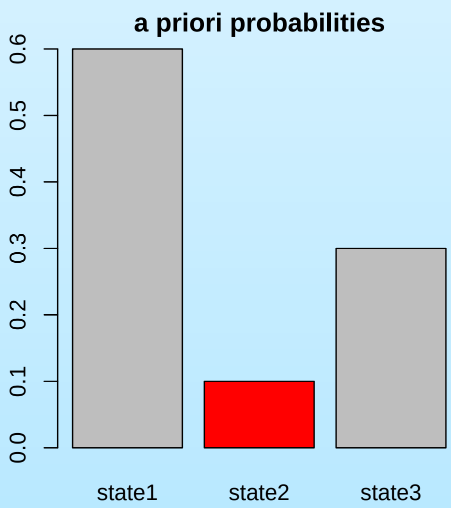
- **Nature assumes one of c states ω_j with *a priori* probability $P(\omega_j)$**
- **When in state ω_j, nature emits observations $\mathbf{x}$ with distribution $p(\mathbf{x}|\omega_j)$**

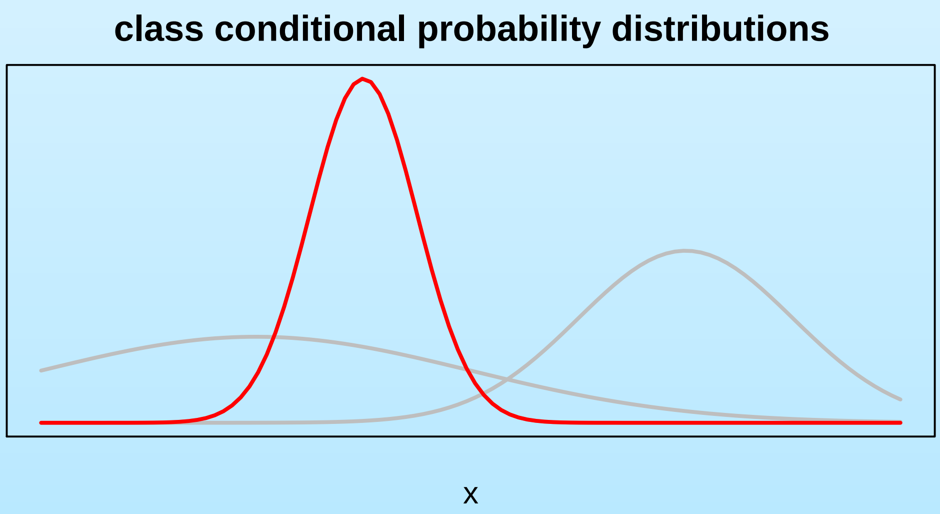




The probabilistic model

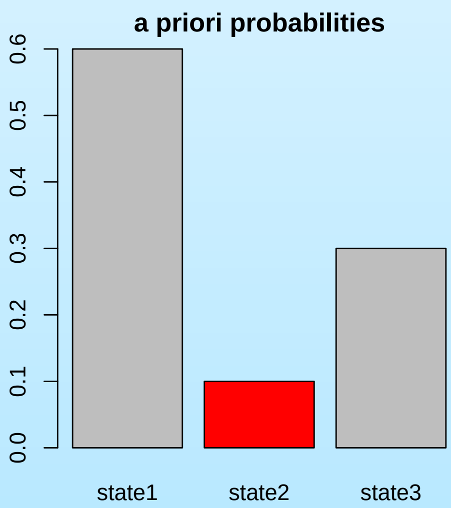
- **Nature assumes one of c states ω_j with *a priori* probability $P(\omega_j)$**
- **When in state ω_j, nature emits observations $\mathbf{x}$ with distribution $p(\mathbf{x}|\omega_j)$**

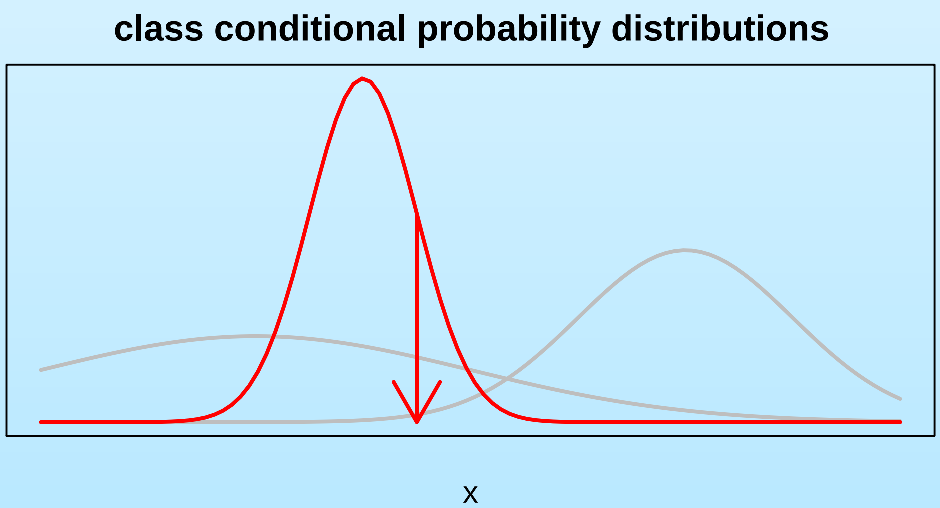




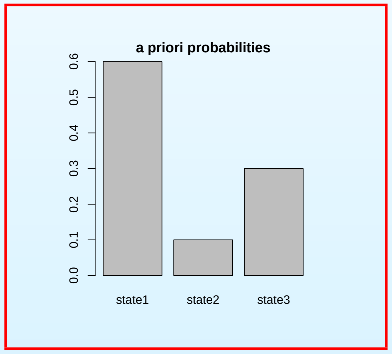
The probabilistic model

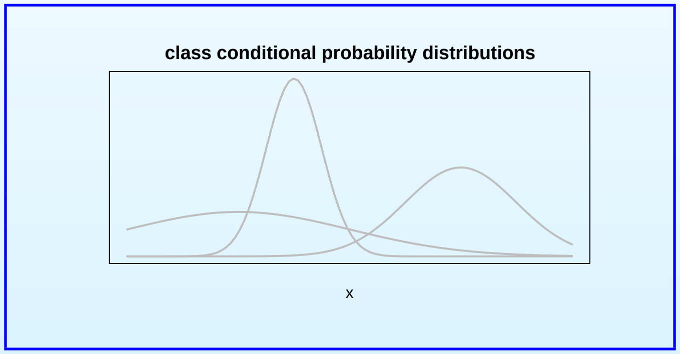
- **Nature assumes one of c states ω_j with *a priori* probability $P(\omega_j)$**
- **When in state ω_j, nature emits observations $\mathbf{x}$ with distribution $p(\mathbf{x}|\omega_j)$**





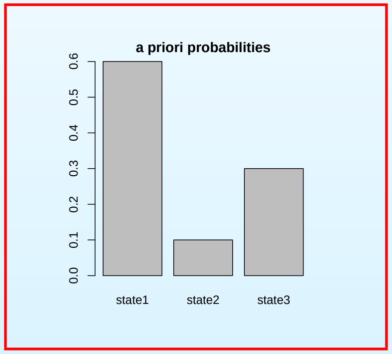
Bayes decision theory

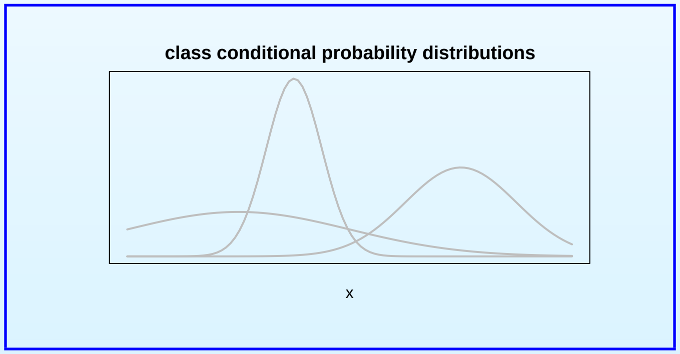




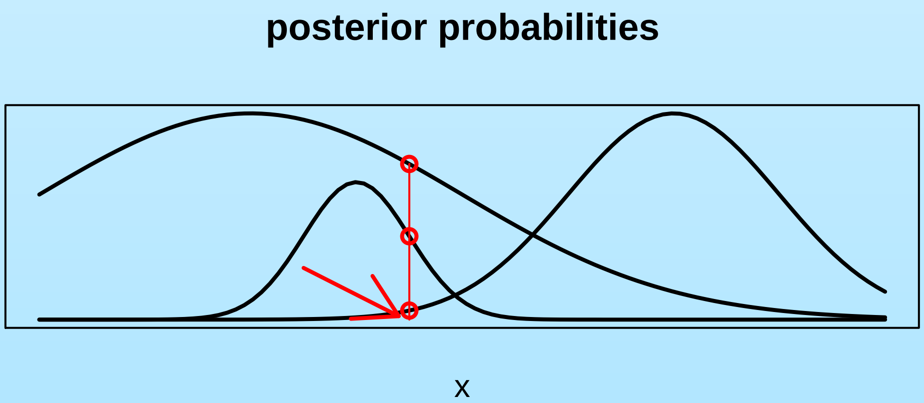
$$P(\omega_j|\mathbf{x}) = \frac{p(\mathbf{x}|\omega_j)\ P(\omega_j)}{p(\mathbf{x})}$$

Bayes decision theory





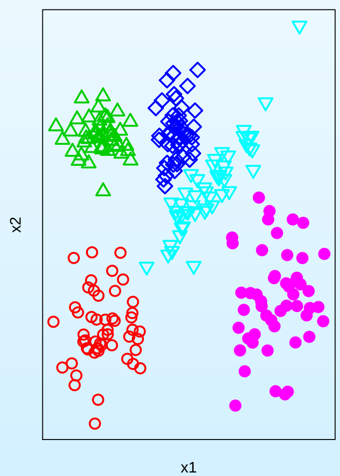
$$P(\omega_j|\mathbf{x}) = \frac{p(\mathbf{x}|\omega_j)\ P(\omega_j)}{p(\mathbf{x})}$$



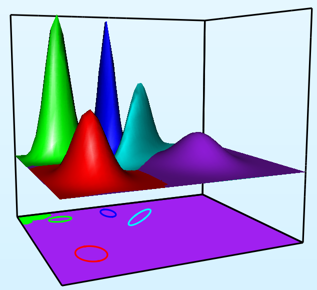
Outline

- What is learning?
- **Parametric methods**
- Non-parametric methods
- Stochastic methods
- Non-metric methods (skip)
- Universal principles
- Unsupervised learning
- Examples

Parameter estimation

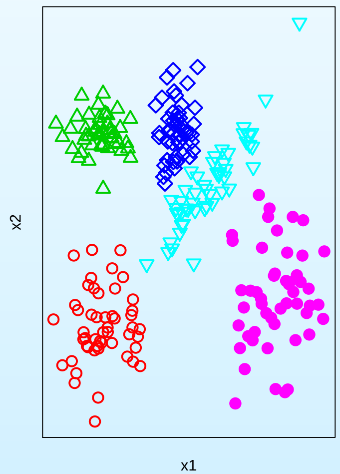


$\longrightarrow$

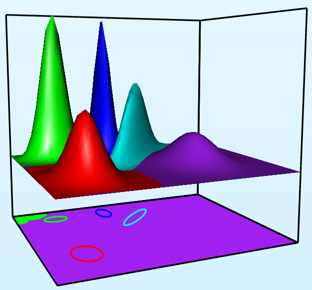


- **ideally:** $p(\mathbf{x}|\omega_j)$ **i.e.** $p(\mathbf{x}|\theta_j)$ **in reality:** $p(\mathbf{x}|\hat{\theta}_j)$ **or** $p(\mathbf{x}|\mathcal{D})$

Parameter estimation



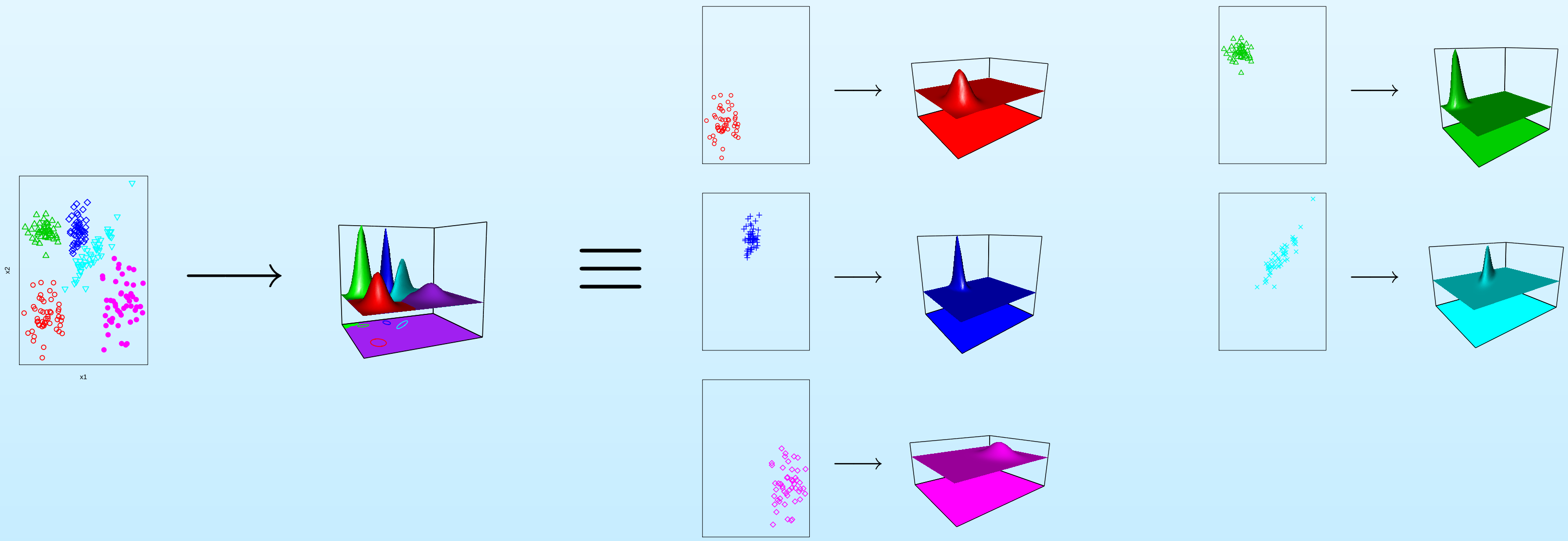
$\longrightarrow$



- **ideally:** $p(\mathbf{x}|\omega_j)$ **i.e.** $p(\mathbf{x}|\theta_j)$ **in reality:** $p(\mathbf{x}|\hat{\theta}_j)$ **or** $p(\mathbf{x}|\mathcal{D})$
- **Assumptions:**
  - **samples from class** ω_i **do not influence estimate for class** $\omega_j, \; i \neq j$
  - **samples from the same class are independent and identically distributed (i.i.d.)**

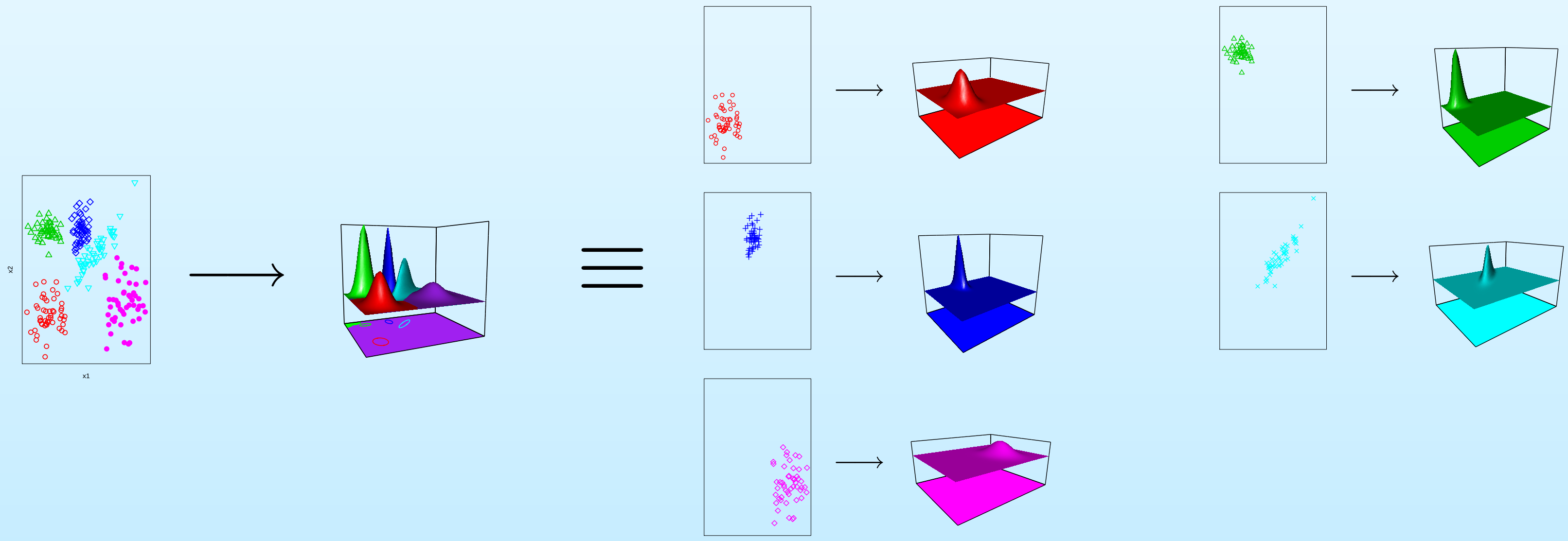
Parameter estimation (cont.)

- **class independence assumption:**



Parameter estimation (cont.)

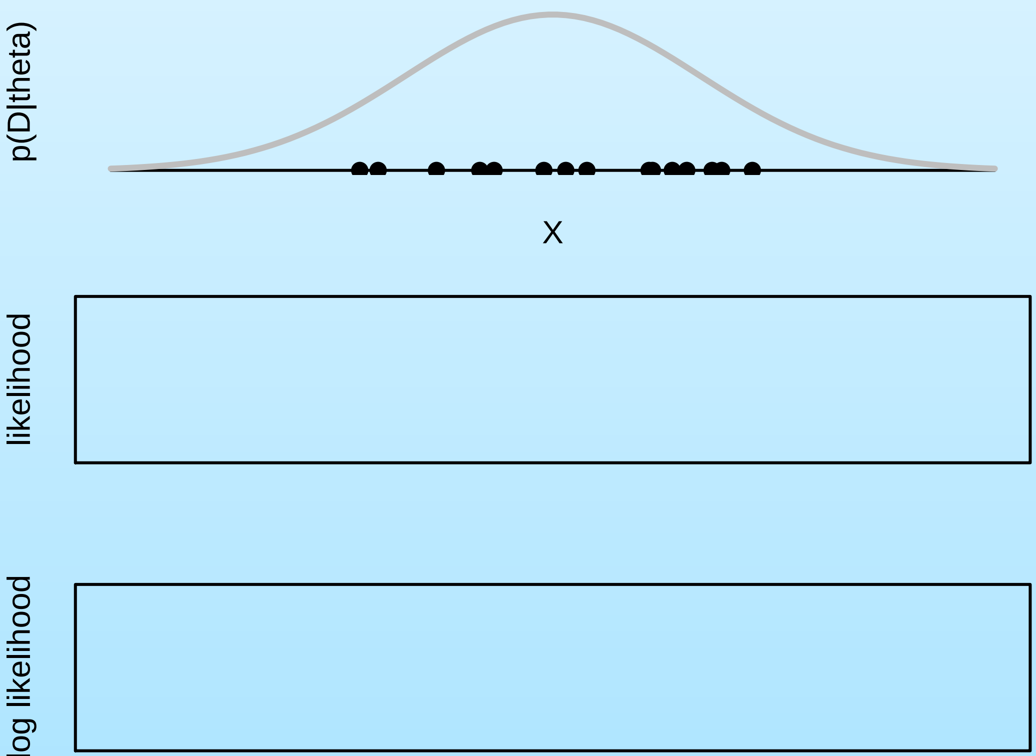
- **class independence assumption:**



- **Maximum likelihood estimation**
- **Bayesian estimation**

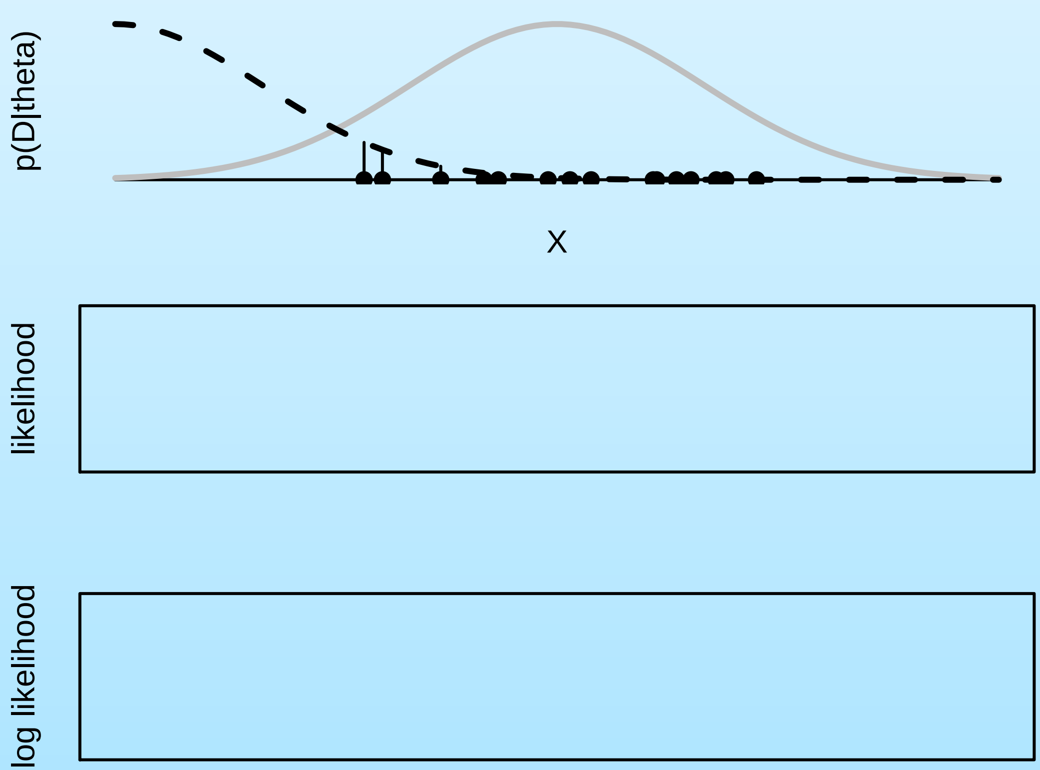
Maximum likelihood estimation

- **Find parameter vector** $\hat{\theta}$ **that maximises** $p(\mathcal{D}|\theta)$ **with** $\mathcal{D} = \{\mathbf{x}_1, \ldots, \mathbf{x}_n\}$
- **i.i.d.** $\rightarrow p(\mathcal{D}|\theta) = \prod_{k=1}^{n} p(\mathbf{x}_k|\theta)$



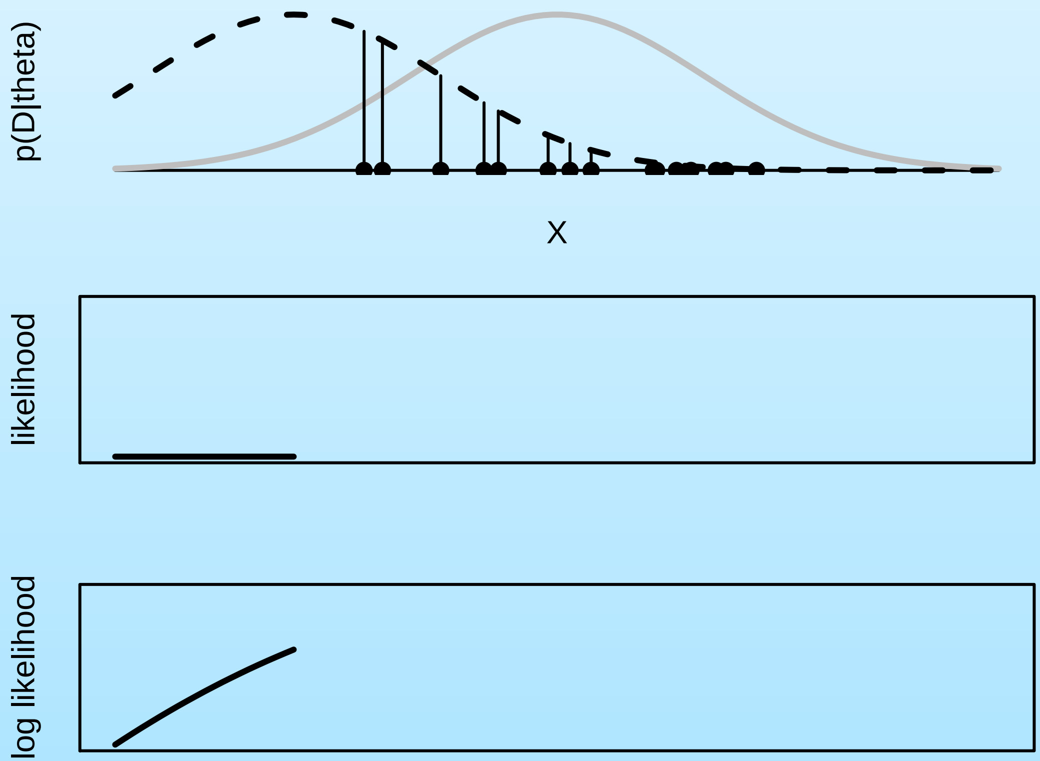
Maximum likelihood estimation

- **Find parameter vector** $\hat{\theta}$ **that maximises** $p(\mathcal{D}|\theta)$ **with** $\mathcal{D} = \{\mathbf{x}_1, \ldots, \mathbf{x}_n\}$
- **i.i.d.** $\rightarrow p(\mathcal{D}|\theta) = \prod_{k=1}^{n} p(\mathbf{x}_k|\theta)$



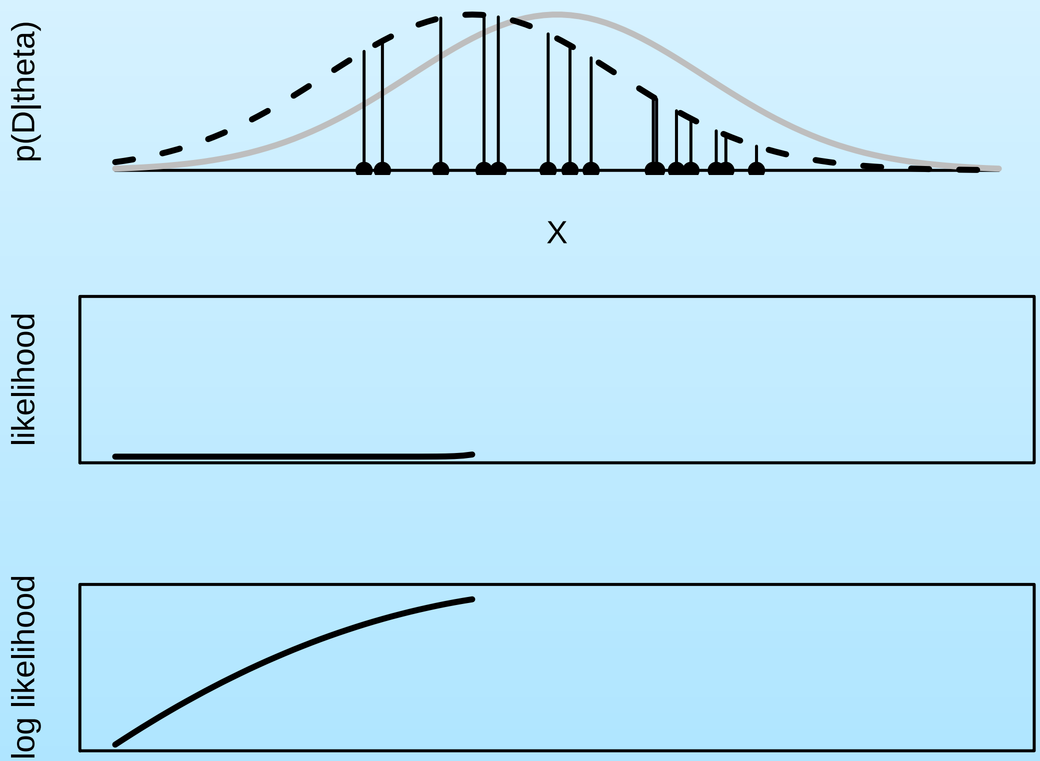
Maximum likelihood estimation

- **Find parameter vector** $\hat{\theta}$ **that maximises** $p(\mathcal{D}|\theta)$ **with** $\mathcal{D} = \{\mathbf{x}_1, \ldots, \mathbf{x}_n\}$
- **i.i.d.** $\rightarrow p(\mathcal{D}|\theta) = \prod_{k=1}^{n} p(\mathbf{x}_k|\theta)$



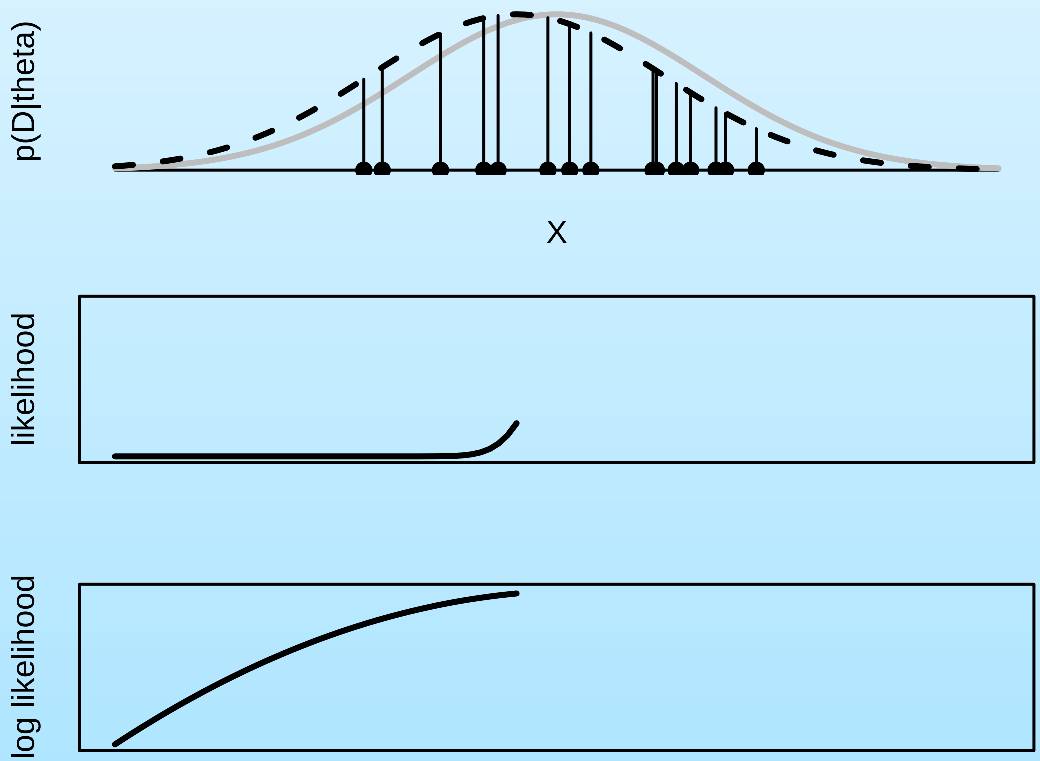
Maximum likelihood estimation

- **Find parameter vector $\hat{\theta}$ that maximises $p(\mathcal{D}|\theta)$ with $\mathcal{D} = \{\mathbf{x}_1, \ldots, \mathbf{x}_n\}$**
- **i.i.d.** $\rightarrow p(\mathcal{D}|\theta) = \prod_{k=1}^{n} p(\mathbf{x}_k|\theta)$



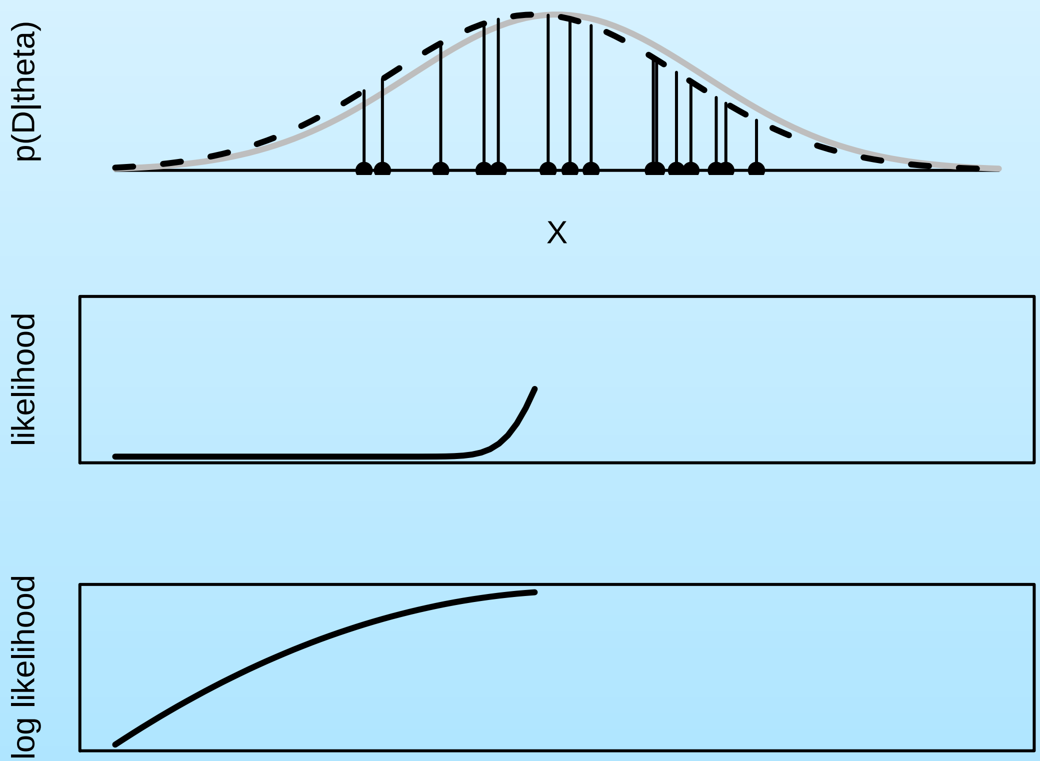
Maximum likelihood estimation

- **Find parameter vector** $\hat{\theta}$ **that maximises** $p(\mathcal{D}|\theta)$ **with** $\mathcal{D} = \{\mathbf{x}_1, \ldots, \mathbf{x}_n\}$
- **i.i.d.** $\rightarrow p(\mathcal{D}|\theta) = \prod_{k=1}^{n} p(\mathbf{x}_k|\theta)$



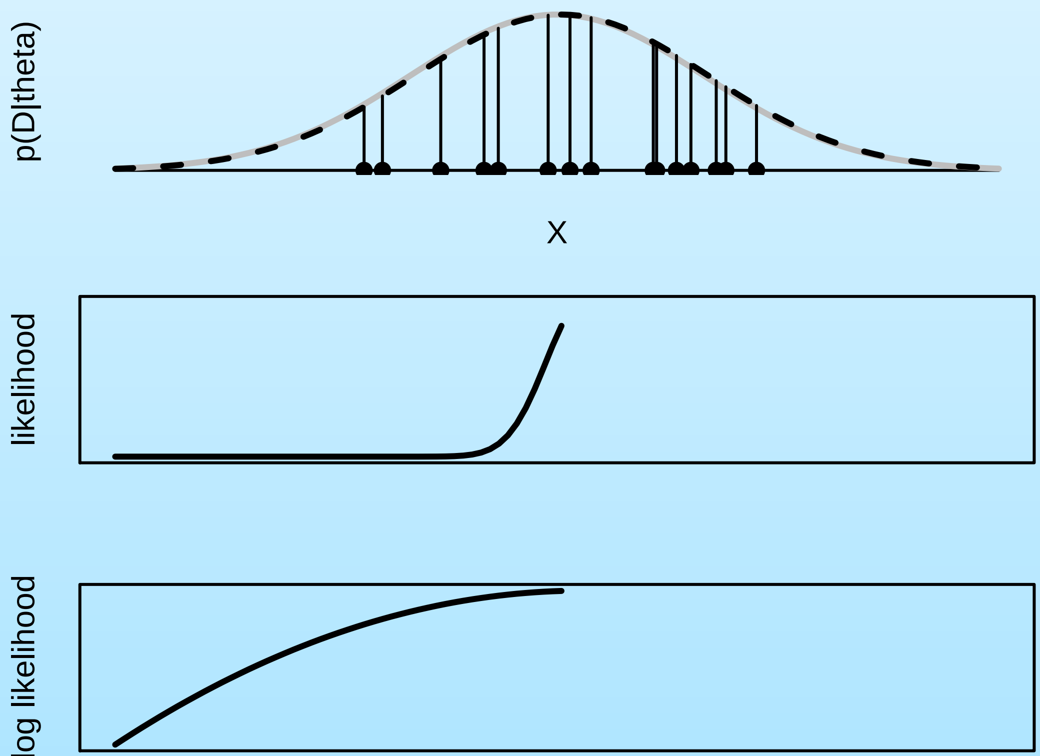
Maximum likelihood estimation

- **Find parameter vector** $\hat{\theta}$ **that maximises** $p(\mathcal{D}|\theta)$ **with** $\mathcal{D} = \{\mathbf{x}_1, \ldots, \mathbf{x}_n\}$
- **i.i.d.** $\rightarrow p(\mathcal{D}|\theta) = \prod_{k=1}^{n} p(\mathbf{x}_k|\theta)$



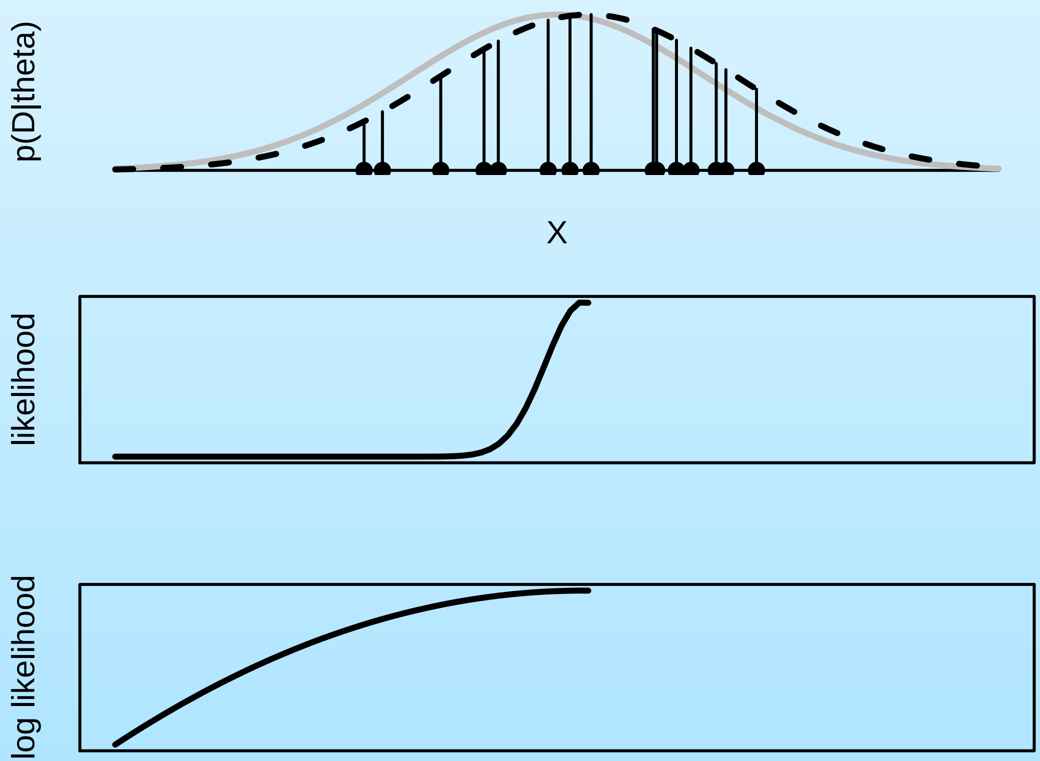
Maximum likelihood estimation

- **Find parameter vector** $\hat{\theta}$ **that maximises** $p(\mathcal{D}|\theta)$ **with** $\mathcal{D} = \{\mathbf{x}_1, \ldots, \mathbf{x}_n\}$
- **i.i.d.** $\rightarrow p(\mathcal{D}|\theta) = \prod_{k=1}^{n} p(\mathbf{x}_k|\theta)$



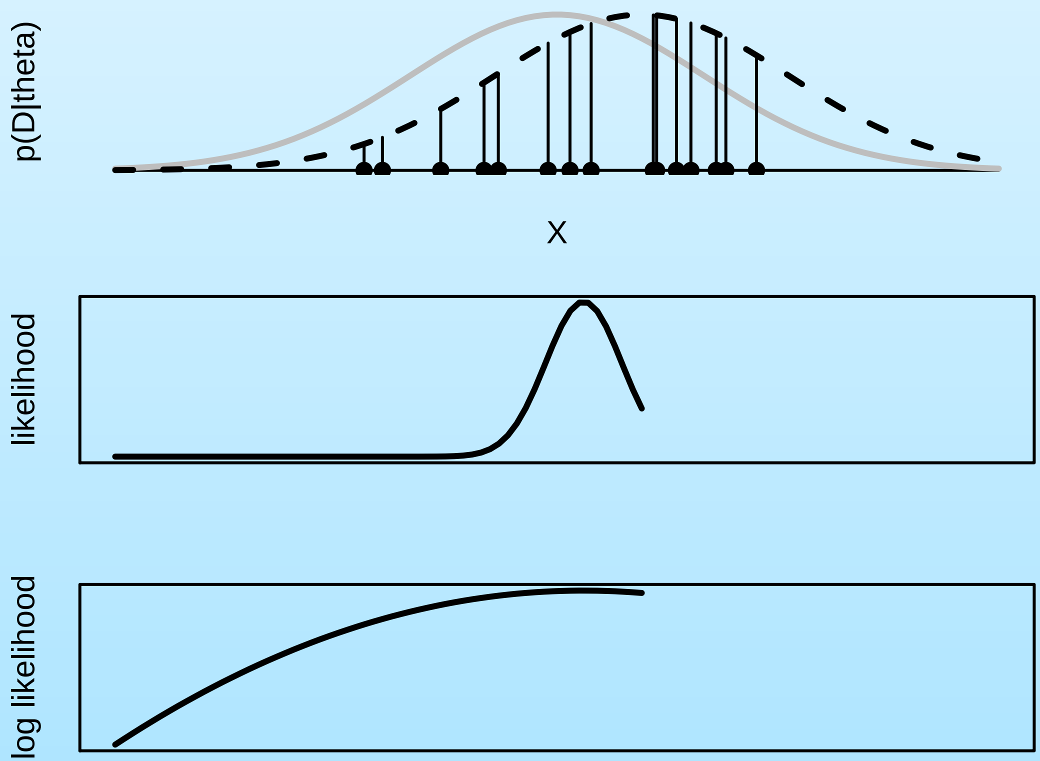
Maximum likelihood estimation

- **Find parameter vector** $\hat{\theta}$ **that maximises** $p(\mathcal{D}|\theta)$ **with** $\mathcal{D} = \{\mathbf{x}_1, \ldots, \mathbf{x}_n\}$
- **i.i.d.** $\rightarrow p(\mathcal{D}|\theta) = \prod_{k=1}^{n} p(\mathbf{x}_k|\theta)$



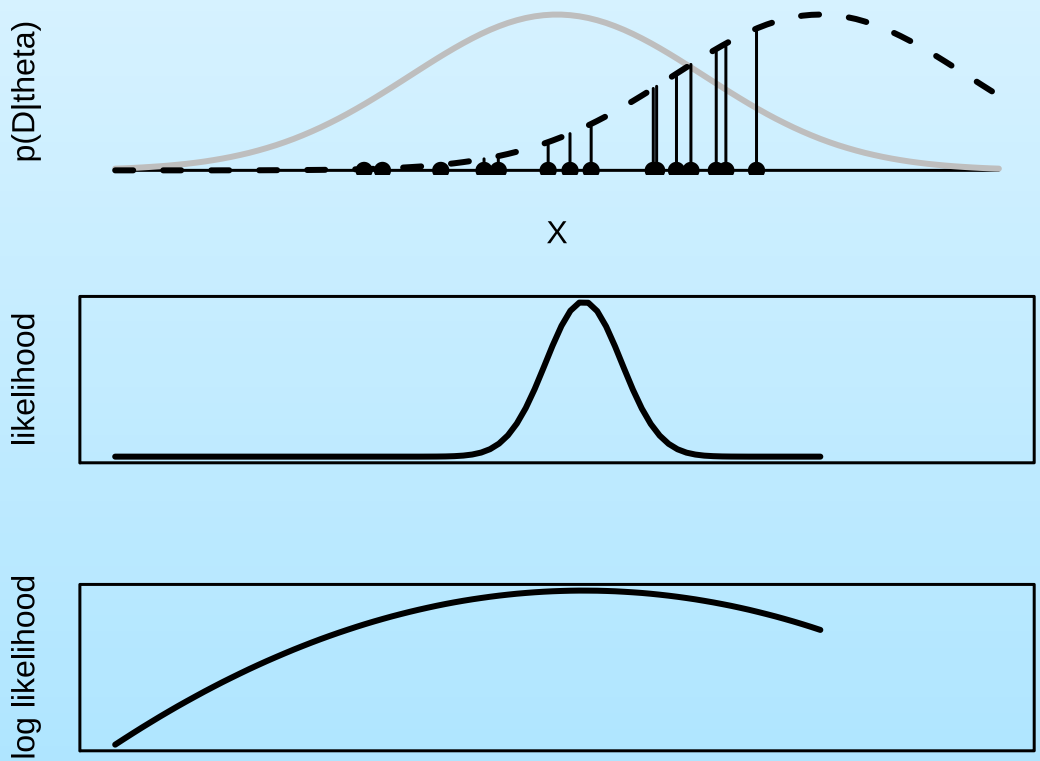
Maximum likelihood estimation

- **Find parameter vector** $\hat{\theta}$ **that maximises** $p(\mathcal{D}|\theta)$ **with** $\mathcal{D} = \{\mathbf{x}_1, \ldots, \mathbf{x}_n\}$
- **i.i.d.** $\rightarrow p(\mathcal{D}|\theta) = \prod_{k=1}^{n} p(\mathbf{x}_k|\theta)$



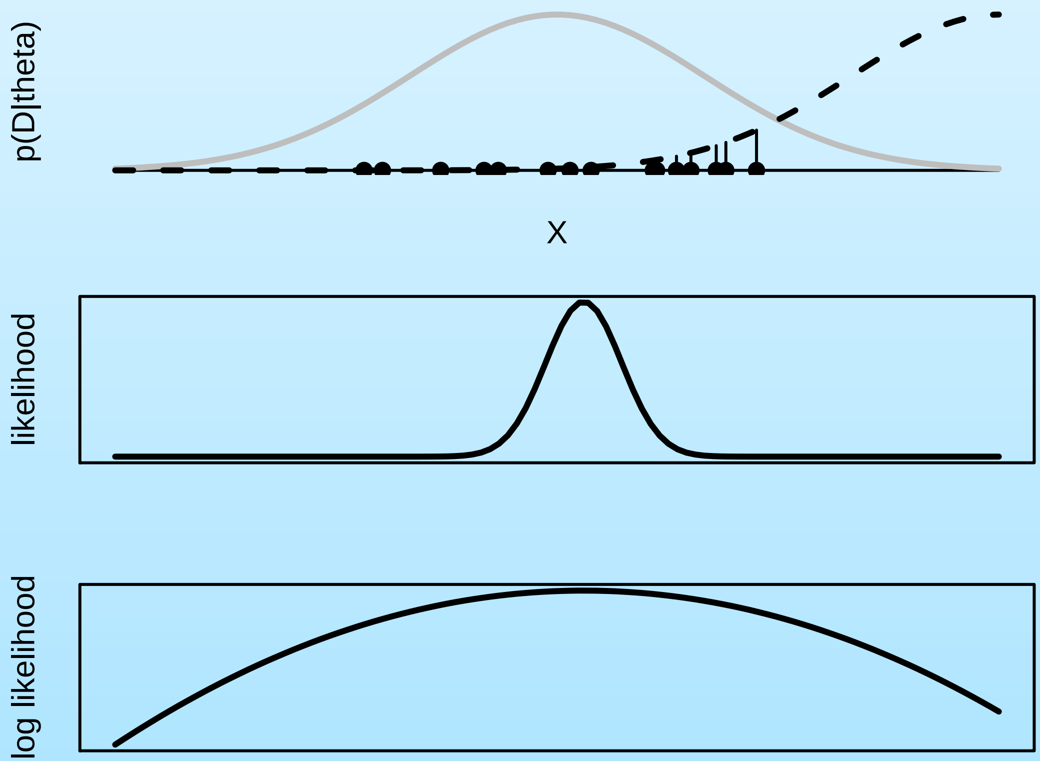
Maximum likelihood estimation

- **Find parameter vector** $\hat{\theta}$ **that maximises** $p(\mathcal{D}|\theta)$ **with** $\mathcal{D} = \{\mathbf{x}_1, \ldots, \mathbf{x}_n\}$
- **i.i.d.** $\rightarrow p(\mathcal{D}|\theta) = \prod_{k=1}^{n} p(\mathbf{x}_k|\theta)$



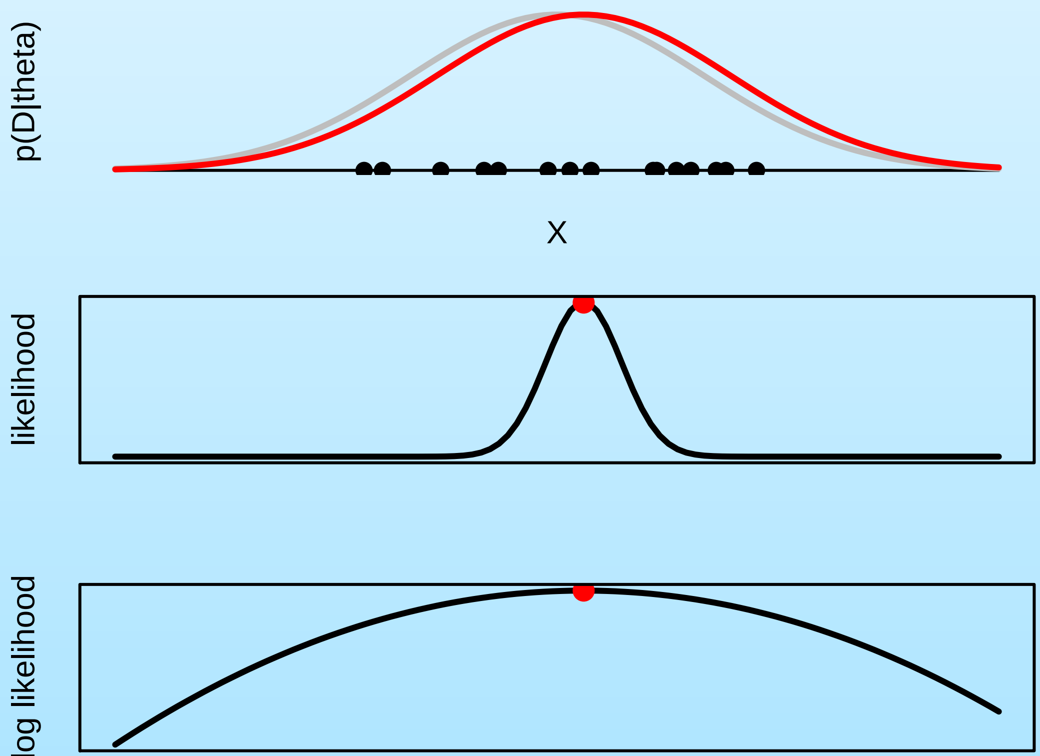
Maximum likelihood estimation

- **Find parameter vector $\hat{\theta}$ that maximises $p(\mathcal{D}|\theta)$ with $\mathcal{D} = \{\mathbf{x}_1, \ldots, \mathbf{x}_n\}$**
- **i.i.d. $\rightarrow p(\mathcal{D}|\theta) = \prod_{k=1}^{n} p(\mathbf{x}_k|\theta)$**



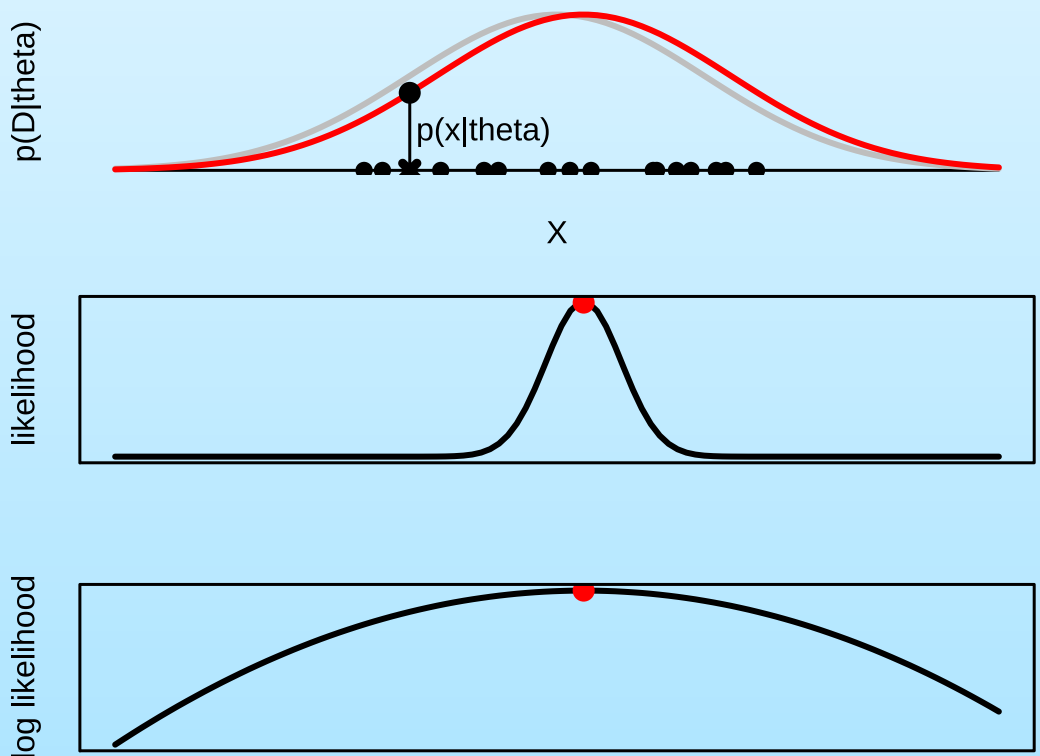
Maximum likelihood estimation

- **Find parameter vector** $\hat{\theta}$ **that maximises** $p(\mathcal{D}|\theta)$ **with** $\mathcal{D} = \{\mathbf{x}_1, \ldots, \mathbf{x}_n\}$
- **i.i.d.** $\rightarrow p(\mathcal{D}|\theta) = \prod_{k=1}^{n} p(\mathbf{x}_k|\theta)$



Maximum likelihood estimation

- **Find parameter vector** $\hat{\theta}$ **that maximises** $p(\mathcal{D}|\theta)$ **with** $\mathcal{D} = \{\mathbf{x}_1, \ldots, \mathbf{x}_n\}$
- **i.i.d.** $\rightarrow p(\mathcal{D}|\theta) = \prod_{k=1}^{n} p(\mathbf{x}_k|\theta)$



Bayesian estimation

- Consider θ as a random variable
- characterise θ with the posterior distribution $p(\theta|\mathcal{D})$ given the data
- using Bayes formula, the posterior can be computed from the likelihood $p(\mathcal{D}|\theta)$ and the prior $p(\theta)$

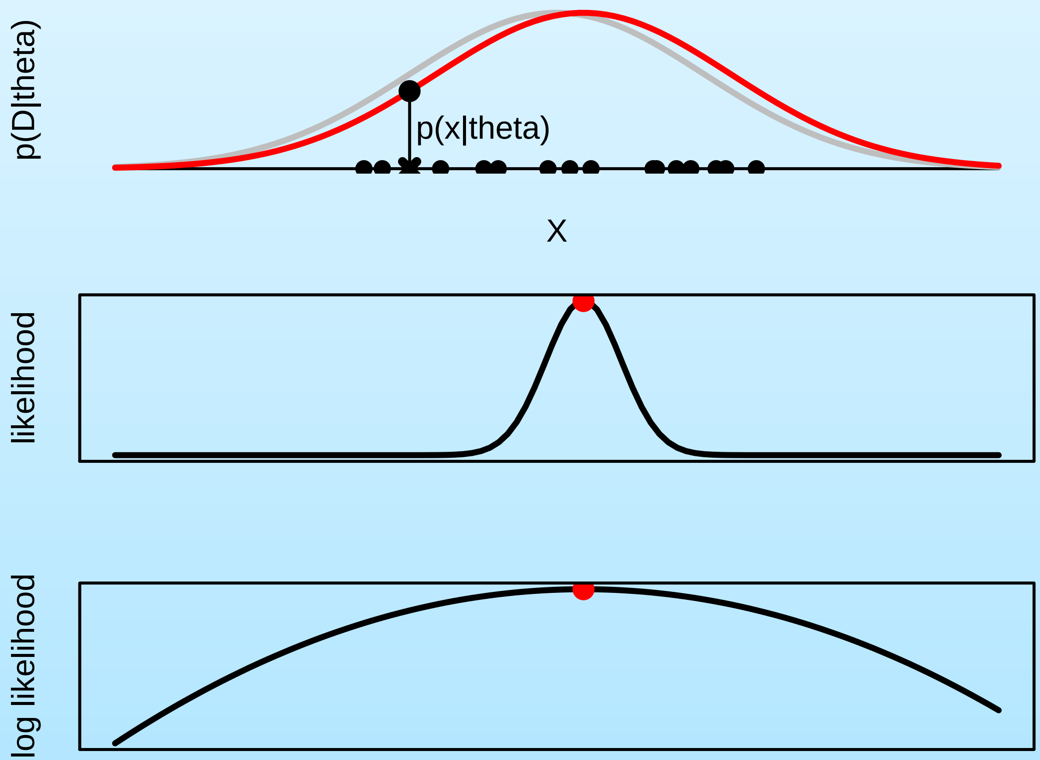
$$p(\theta|\mathcal{D}) = \frac{p(\mathcal{D}|\theta)p(\theta)}{\int p(\mathcal{D}|\theta)p(\theta)d\theta}$$

- **ML:** $\mathcal{D} \longrightarrow \hat{\theta}$ **Bayes:** $\mathcal{D}, p(\theta) \longrightarrow p(\theta|\mathcal{D})$

Bayesian estimation (cont.)

- **we can compute $p(\mathbf{x}|\mathcal{D})$ instead of $p(\mathbf{x}|\hat{\theta})$**
  - **integrate the join density $p(\mathbf{x}, \theta|\mathcal{D}) = p(\mathbf{x}|\theta)p(\theta|\mathcal{D})$**

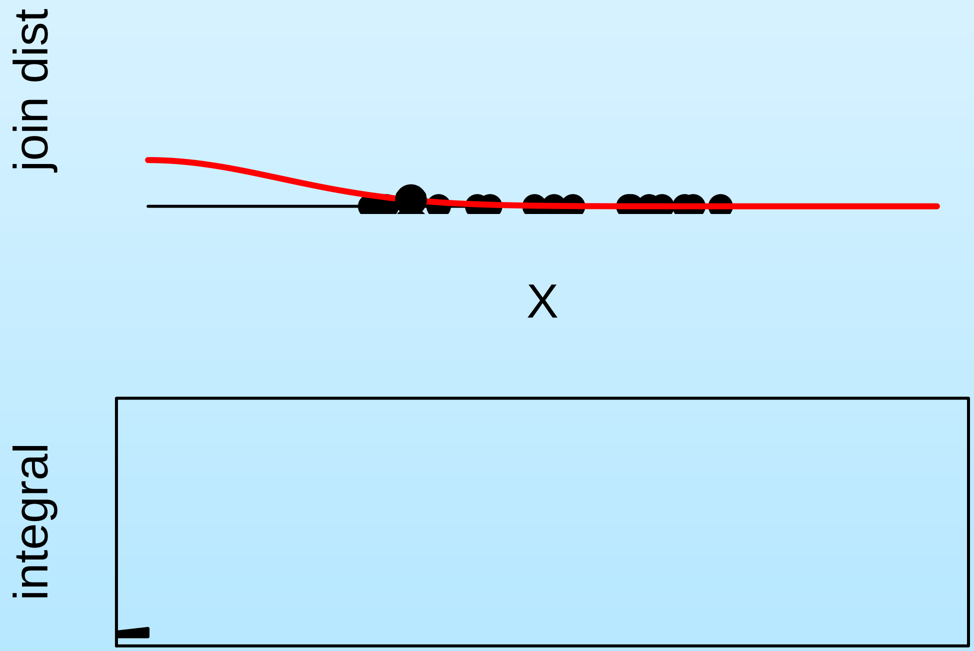
$p(\mathbf{x}|\hat{\theta})$



Bayesian estimation

- **we can compute** $p(\mathbf{x}|\mathcal{D})$ **instead of** $p(\mathbf{x}|\hat{\theta})$
  - **integrate the join density** $p(\mathbf{x},\theta|\mathcal{D}) = p(\mathbf{x}|\theta)p(\theta|\mathcal{D})$

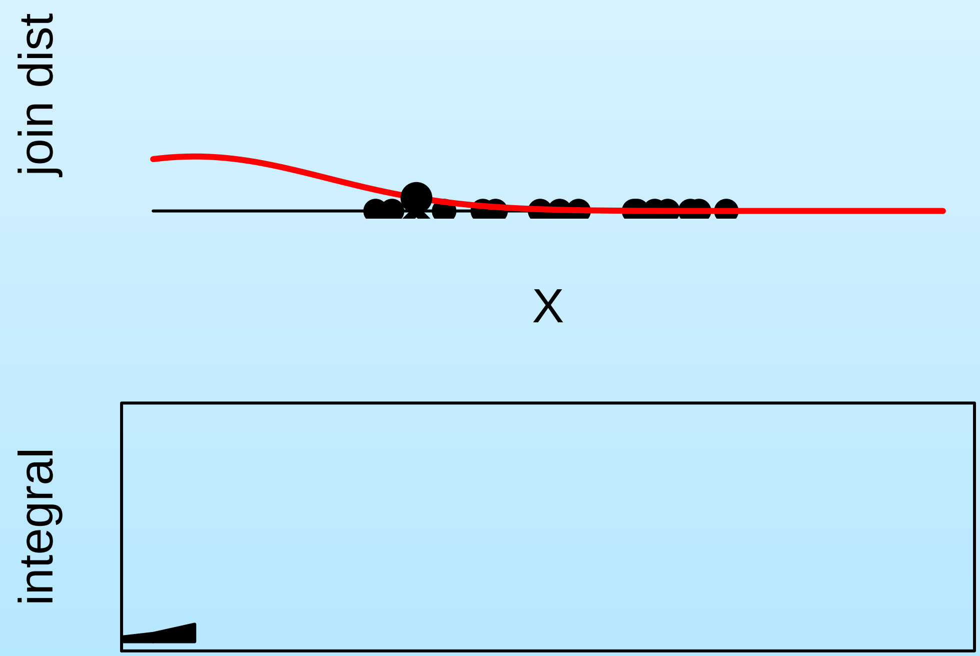
$$p(\mathbf{x}|\mathcal{D}) = \int p(\mathbf{x}|\theta)p(\theta|\mathcal{D})d\theta$$



Bayesian estimation

- **we can compute $p(\mathbf{x}|\mathcal{D})$ instead of $p(\mathbf{x}|\hat{\theta})$**
  - **integrate the join density $p(\mathbf{x}, \theta|\mathcal{D}) = p(\mathbf{x}|\theta)p(\theta|\mathcal{D})$**

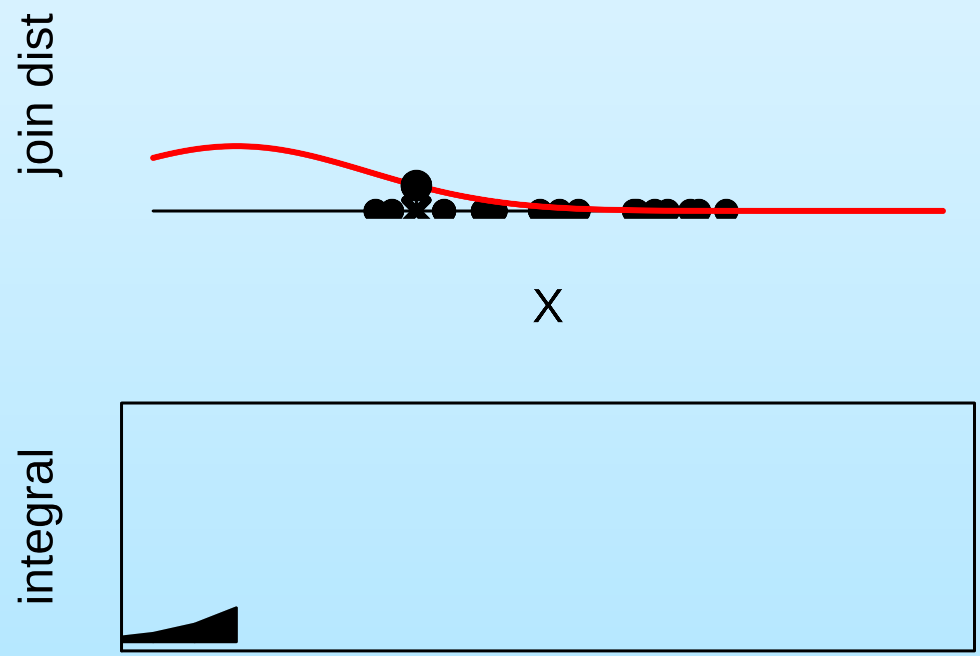
$$p(\mathbf{x}|\mathcal{D}) =$$
$$\int p(\mathbf{x}|\theta)p(\theta|\mathcal{D})d\theta$$



Bayesian estimation

- **we can compute $p(\mathbf{x}|\mathcal{D})$ instead of $p(\mathbf{x}|\hat{\theta})$**
  - **integrate the join density $p(\mathbf{x},\theta|\mathcal{D}) = p(\mathbf{x}|\theta)p(\theta|\mathcal{D})$**

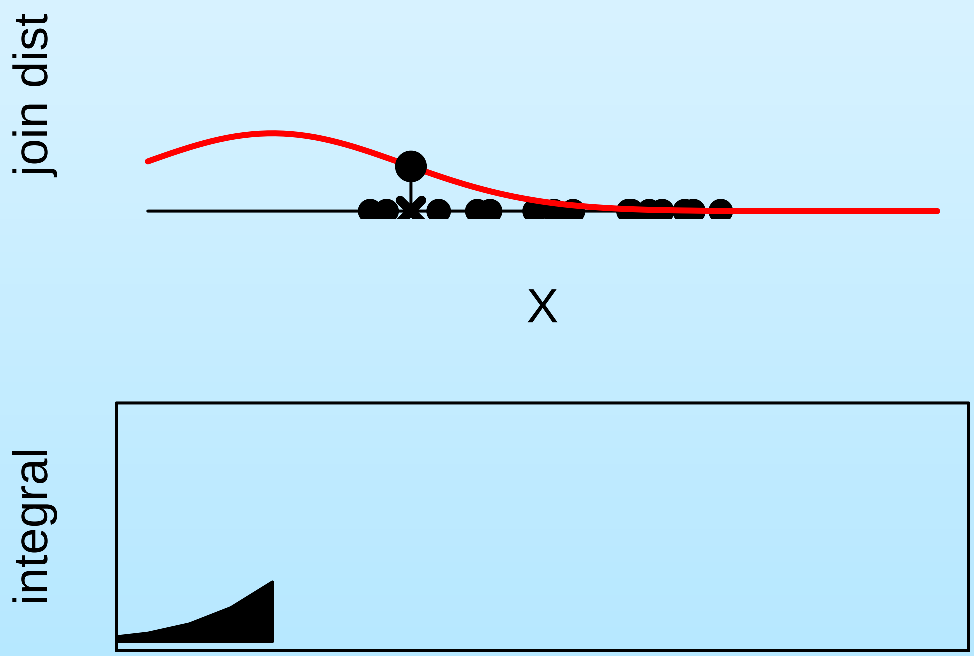
$$p(\mathbf{x}|\mathcal{D}) = \int p(\mathbf{x}|\theta)p(\theta|\mathcal{D})d\theta$$



Bayesian estimation

- **we can compute $p(\mathbf{x}|\mathcal{D})$ instead of $p(\mathbf{x}|\hat{\theta})$**
  - **integrate the join density $p(\mathbf{x}, \theta|\mathcal{D}) = p(\mathbf{x}|\theta)p(\theta|\mathcal{D})$**

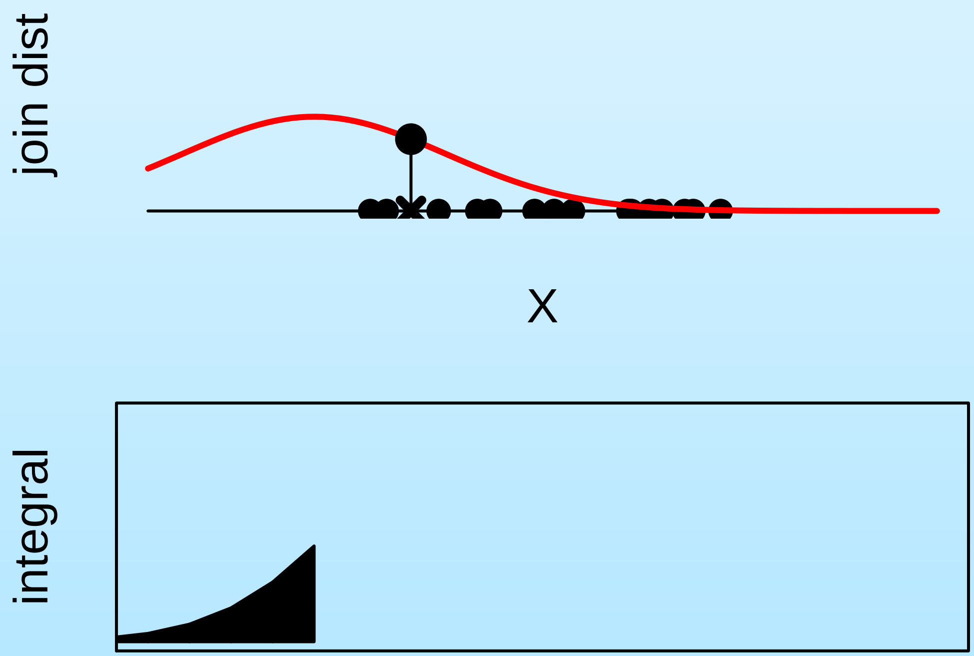
$$p(\mathbf{x}|\mathcal{D}) = \int p(\mathbf{x}|\theta)p(\theta|\mathcal{D})d\theta$$



Bayesian estimation

- **we can compute $p(\mathbf{x}|\mathcal{D})$ instead of $p(\mathbf{x}|\hat{\theta})$**
  - **integrate the join density $p(\mathbf{x},\theta|\mathcal{D}) = p(\mathbf{x}|\theta)p(\theta|\mathcal{D})$**

$$p(\mathbf{x}|\mathcal{D}) = \int p(\mathbf{x}|\theta)p(\theta|\mathcal{D})d\theta$$

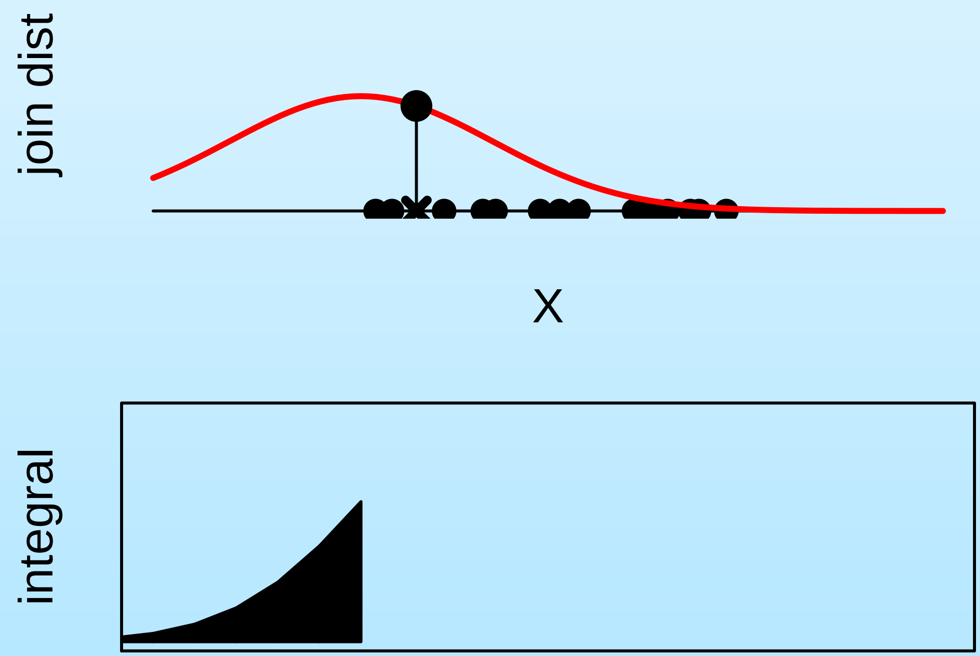


Bayesian estimation

- **we can compute $p(\mathbf{x}|\mathcal{D})$ instead of $p(\mathbf{x}|\hat{\theta})$**
  - **integrate the join density $p(\mathbf{x}, \theta|\mathcal{D}) = p(\mathbf{x}|\theta)p(\theta|\mathcal{D})$**

$$p(\mathbf{x}|\mathcal{D}) =$$

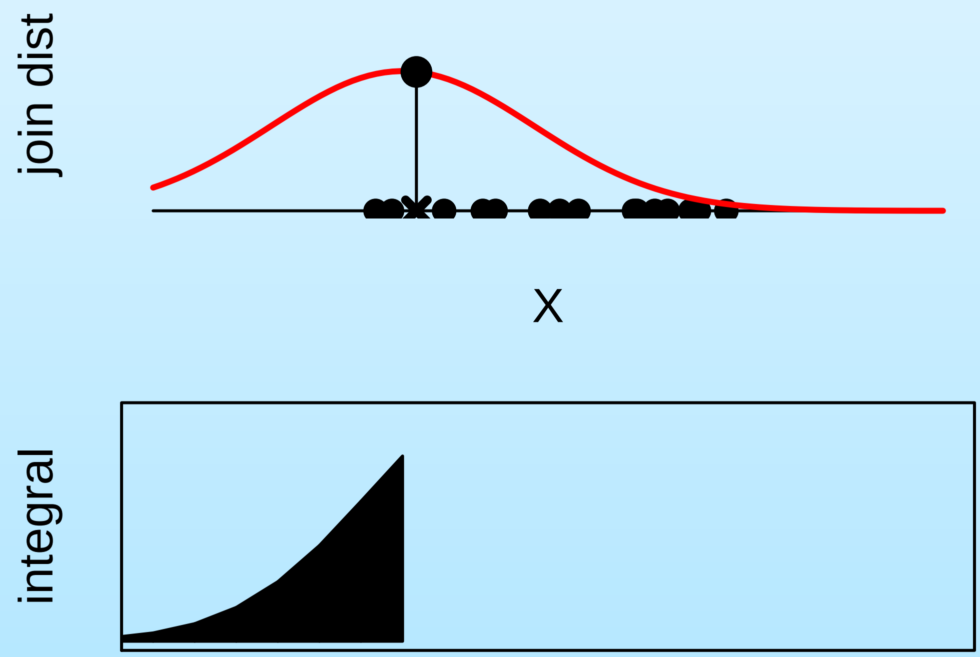
$$\int p(\mathbf{x}|\theta)p(\theta|\mathcal{D})d\theta$$



Bayesian estimation

- **we can compute $p(\mathbf{x}|\mathcal{D})$ instead of $p(\mathbf{x}|\hat{\theta})$**
  - **integrate the join density $p(\mathbf{x},\theta|\mathcal{D}) = p(\mathbf{x}|\theta)p(\theta|\mathcal{D})$**

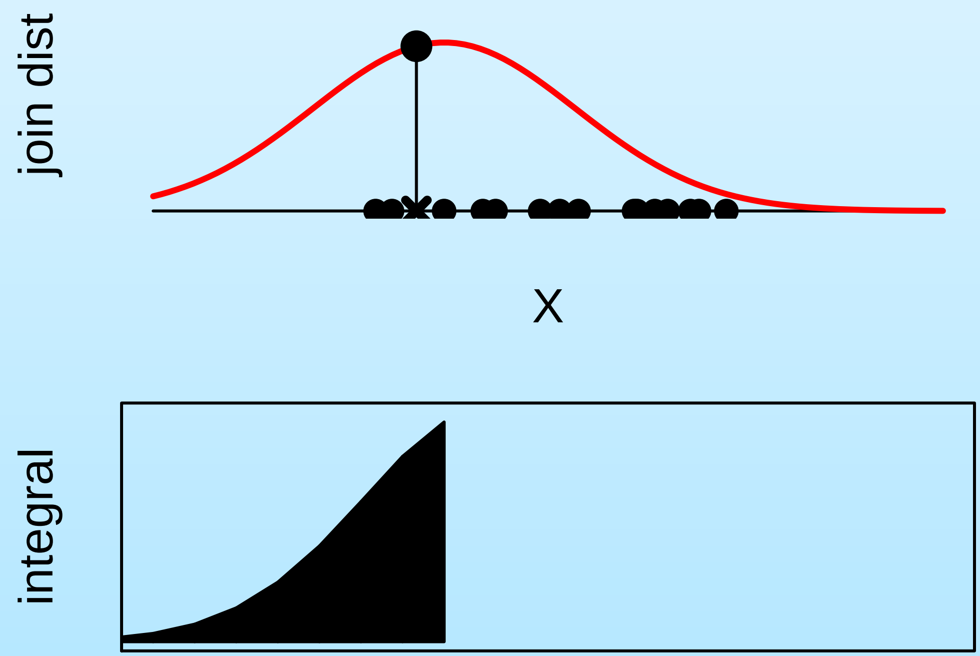
$$p(\mathbf{x}|\mathcal{D}) = \int p(\mathbf{x}|\theta)p(\theta|\mathcal{D})d\theta$$



Bayesian estimation

- **we can compute $p(\mathbf{x}|\mathcal{D})$ instead of $p(\mathbf{x}|\hat{\theta})$**
  - **integrate the join density $p(\mathbf{x}, \theta|\mathcal{D}) = p(\mathbf{x}|\theta)p(\theta|\mathcal{D})$**

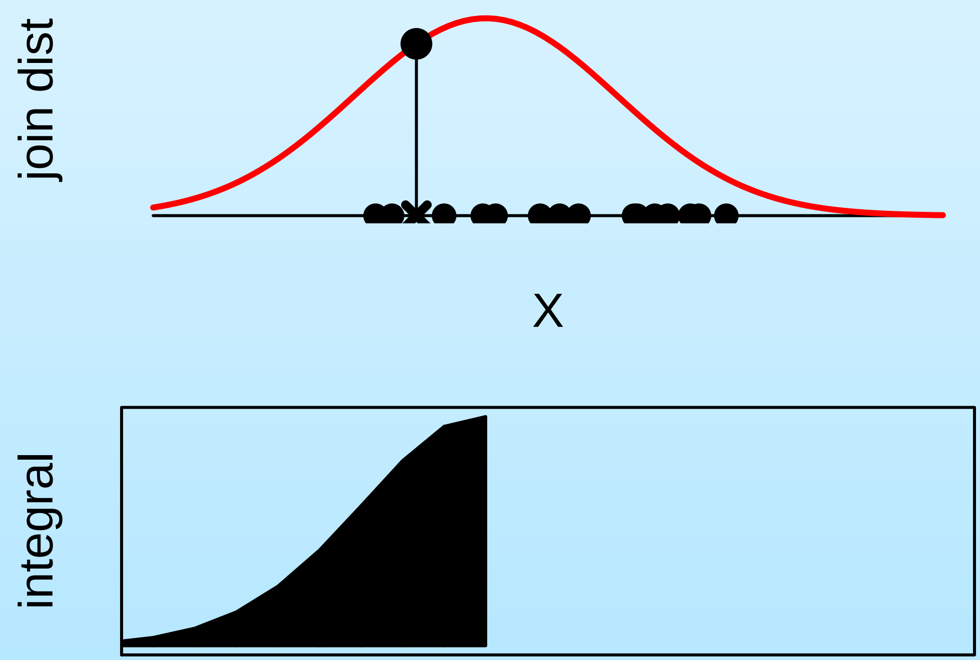
$$p(\mathbf{x}|\mathcal{D}) =$$
$$\int p(\mathbf{x}|\theta)p(\theta|\mathcal{D})d\theta$$



Bayesian estimation

- **we can compute $p(\mathbf{x}|\mathcal{D})$ instead of $p(\mathbf{x}|\hat{\theta})$**
  - **integrate the join density $p(\mathbf{x}, \theta|\mathcal{D}) = p(\mathbf{x}|\theta)p(\theta|\mathcal{D})$**

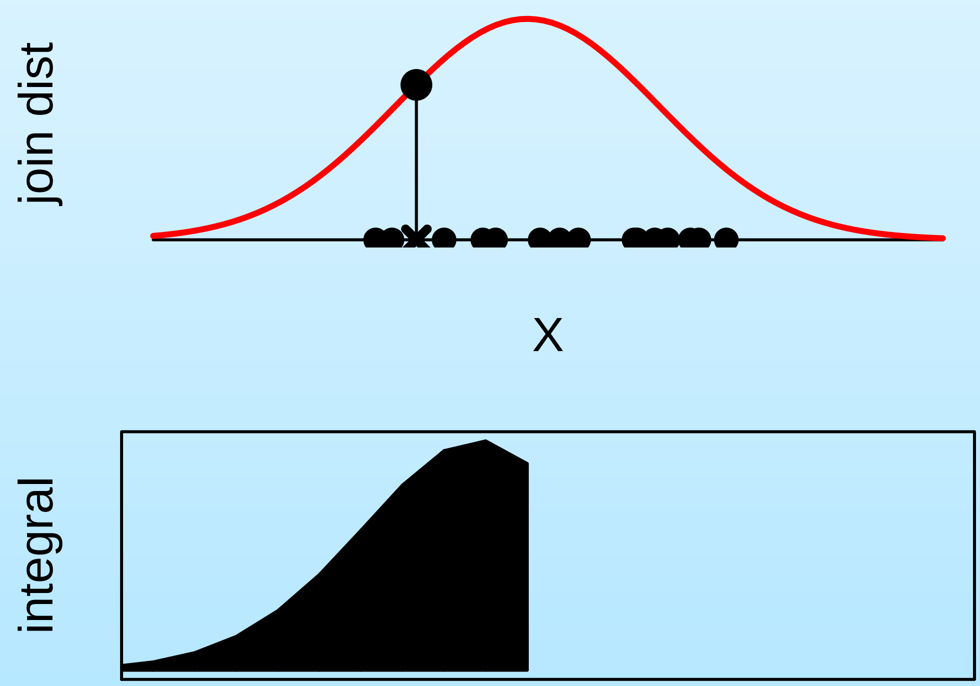
$$p(\mathbf{x}|\mathcal{D}) =$$
$$\int p(\mathbf{x}|\theta)p(\theta|\mathcal{D})d\theta$$



Bayesian estimation

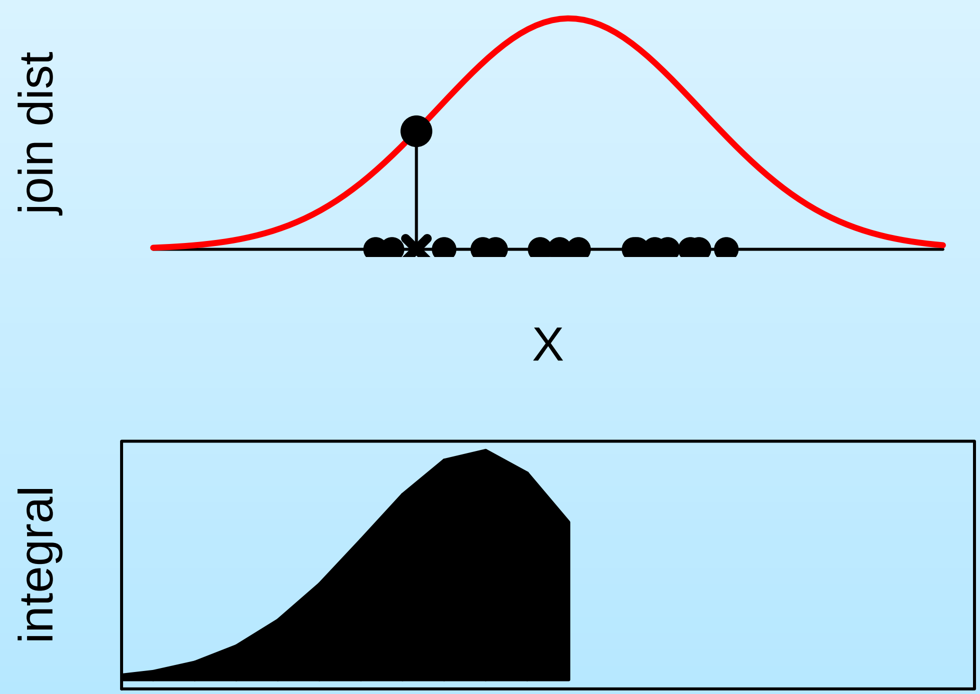
- **we can compute $p(\mathbf{x}|\mathcal{D})$ instead of $p(\mathbf{x}|\hat{\theta})$**
  - **integrate the join density $p(\mathbf{x},\theta|\mathcal{D}) = p(\mathbf{x}|\theta)p(\theta|\mathcal{D})$**

$$p(\mathbf{x}|\mathcal{D}) =$$
$$\int p(\mathbf{x}|\theta)p(\theta|\mathcal{D})d\theta$$



Bayesian estimation

- **we can compute $p(\mathbf{x}|\mathcal{D})$ instead of $p(\mathbf{x}|\hat{\theta})$**
  - **integrate the join density $p(\mathbf{x}, \theta|\mathcal{D}) = p(\mathbf{x}|\theta)p(\theta|\mathcal{D})$**



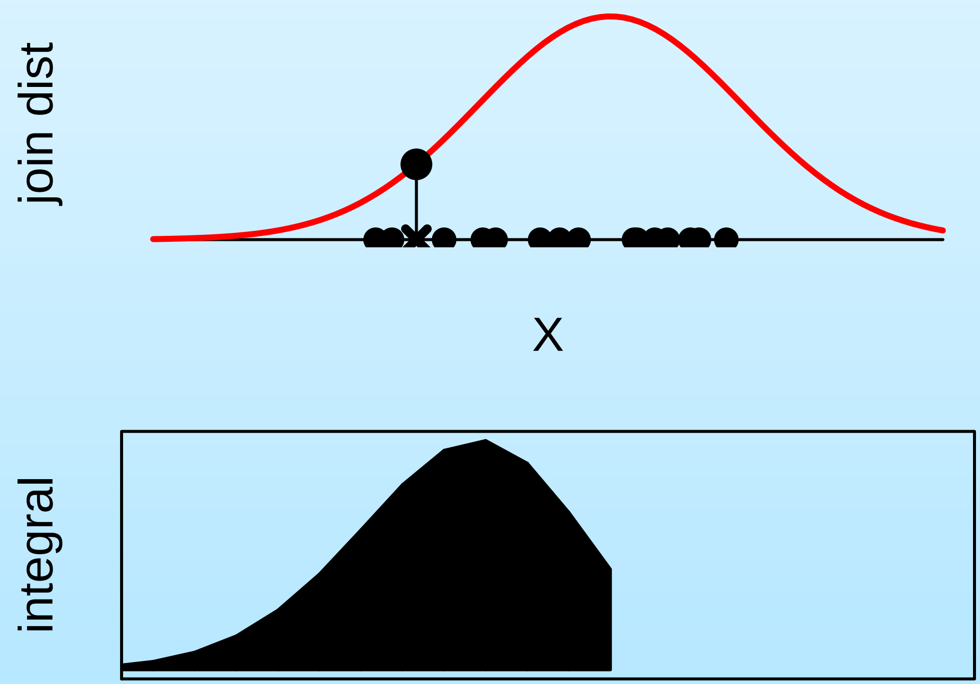
$$p(\mathbf{x}|\mathcal{D}) =$$

$$\int p(\mathbf{x}|\theta)p(\theta|\mathcal{D})d\theta$$

Bayesian estimation

- **we can compute** $p(\mathbf{x}|\mathcal{D})$ **instead of** $p(\mathbf{x}|\hat{\theta})$
  - **integrate the join density** $p(\mathbf{x},\theta|\mathcal{D}) = p(\mathbf{x}|\theta)p(\theta|\mathcal{D})$

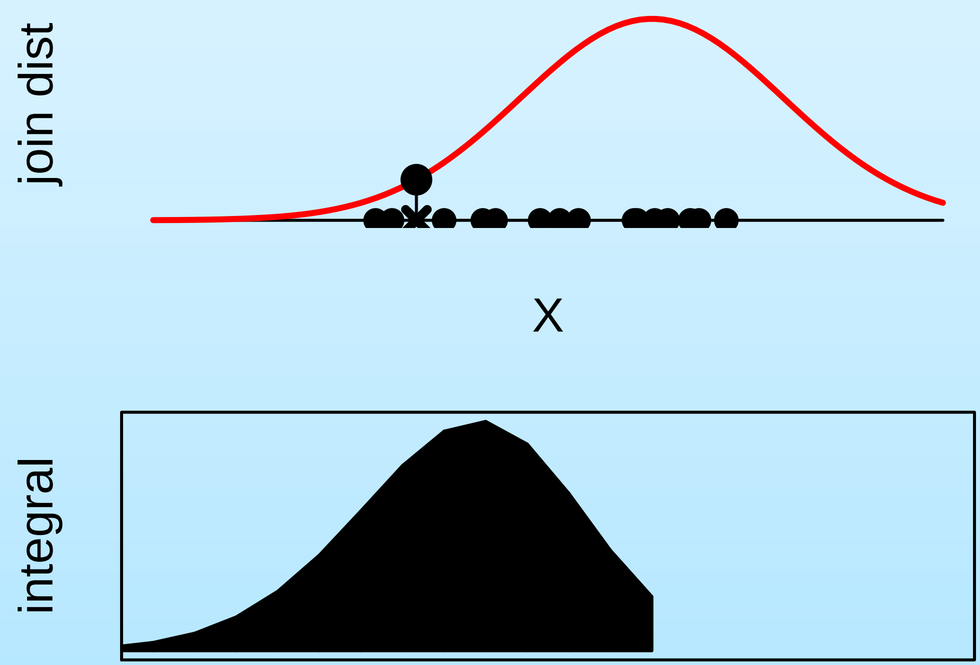
$$p(\mathbf{x}|\mathcal{D}) = \int p(\mathbf{x}|\theta)p(\theta|\mathcal{D})d\theta$$



Bayesian estimation

- **we can compute $p(\mathbf{x}|\mathcal{D})$ instead of $p(\mathbf{x}|\hat{\theta})$**
  - **integrate the join density $p(\mathbf{x}, \theta|\mathcal{D}) = p(\mathbf{x}|\theta)p(\theta|\mathcal{D})$**

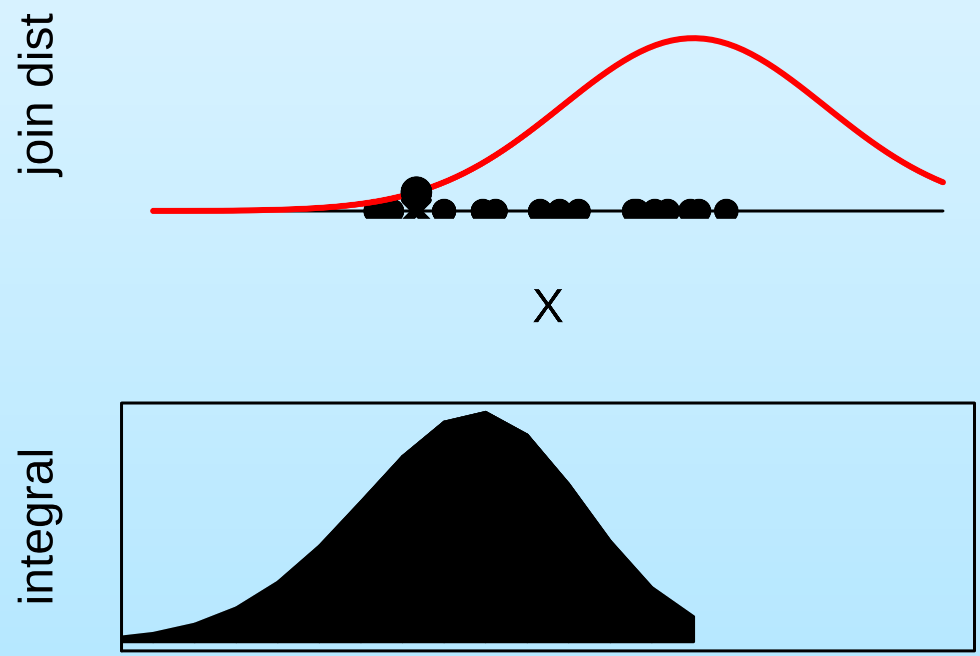
$$p(\mathbf{x}|\mathcal{D}) =$$
$$\int p(\mathbf{x}|\theta)p(\theta|\mathcal{D})d\theta$$



Bayesian estimation

- **we can compute $p(\mathbf{x}|\mathcal{D})$ instead of $p(\mathbf{x}|\hat{\theta})$**
  - **integrate the join density $p(\mathbf{x},\theta|\mathcal{D}) = p(\mathbf{x}|\theta)p(\theta|\mathcal{D})$**

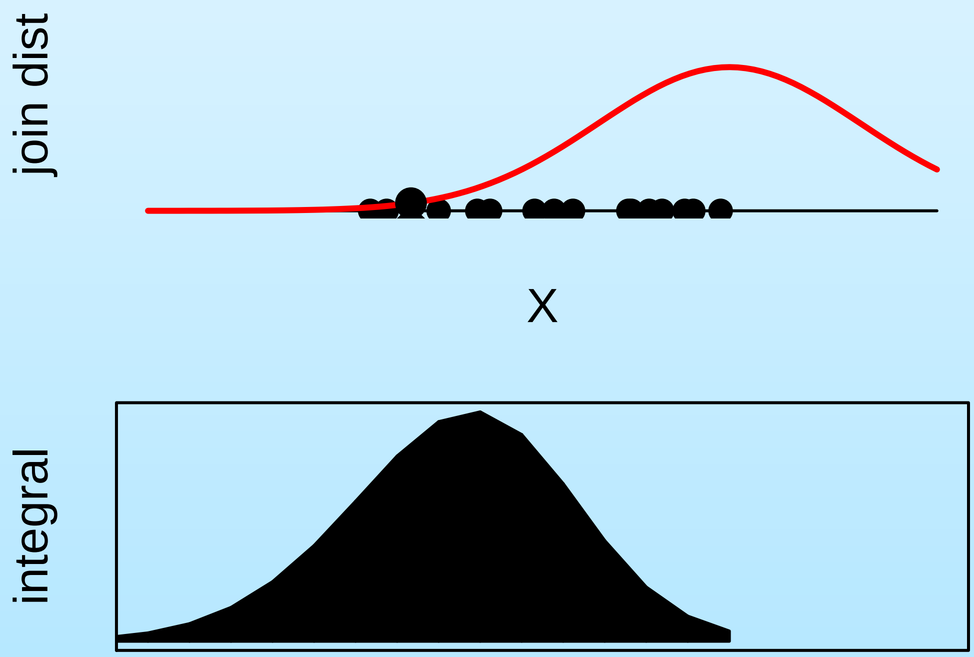
$$p(\mathbf{x}|\mathcal{D}) = \int p(\mathbf{x}|\theta)p(\theta|\mathcal{D})d\theta$$



Bayesian estimation

- **we can compute $p(\mathbf{x}|\mathcal{D})$ instead of $p(\mathbf{x}|\hat{\theta})$**
  - **integrate the join density $p(\mathbf{x}, \theta|\mathcal{D}) = p(\mathbf{x}|\theta)p(\theta|\mathcal{D})$**

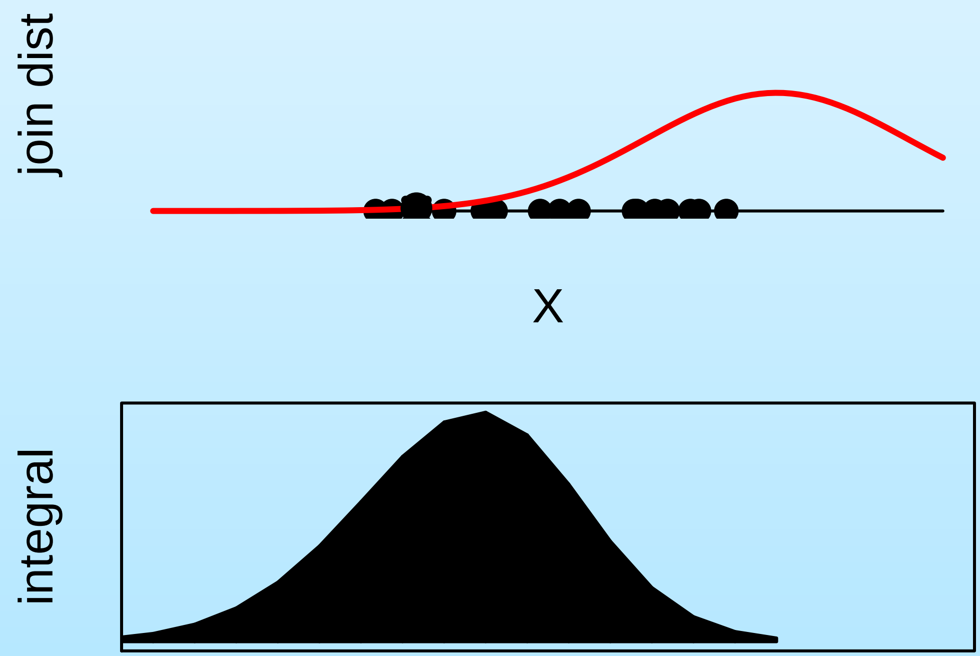
$$p(\mathbf{x}|\mathcal{D}) = \int p(\mathbf{x}|\theta)p(\theta|\mathcal{D})d\theta$$



Bayesian estimation

- **we can compute** $p(\mathbf{x}|\mathcal{D})$ **instead of** $p(\mathbf{x}|\hat{\theta})$
  - **integrate the join density** $p(\mathbf{x}, \theta|\mathcal{D}) = p(\mathbf{x}|\theta)p(\theta|\mathcal{D})$

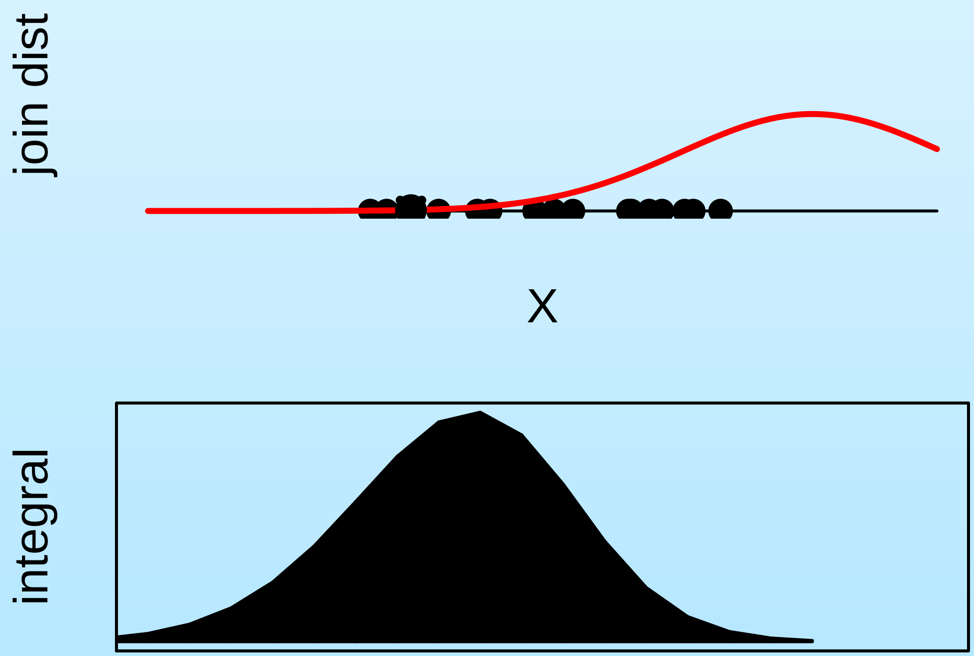
$$p(\mathbf{x}|\mathcal{D}) = \int p(\mathbf{x}|\theta)p(\theta|\mathcal{D})d\theta$$



Bayesian estimation

- **we can compute $p(\mathbf{x}|\mathcal{D})$ instead of $p(\mathbf{x}|\hat{\theta})$**
  - **integrate the join density $p(\mathbf{x},\theta|\mathcal{D}) = p(\mathbf{x}|\theta)p(\theta|\mathcal{D})$**

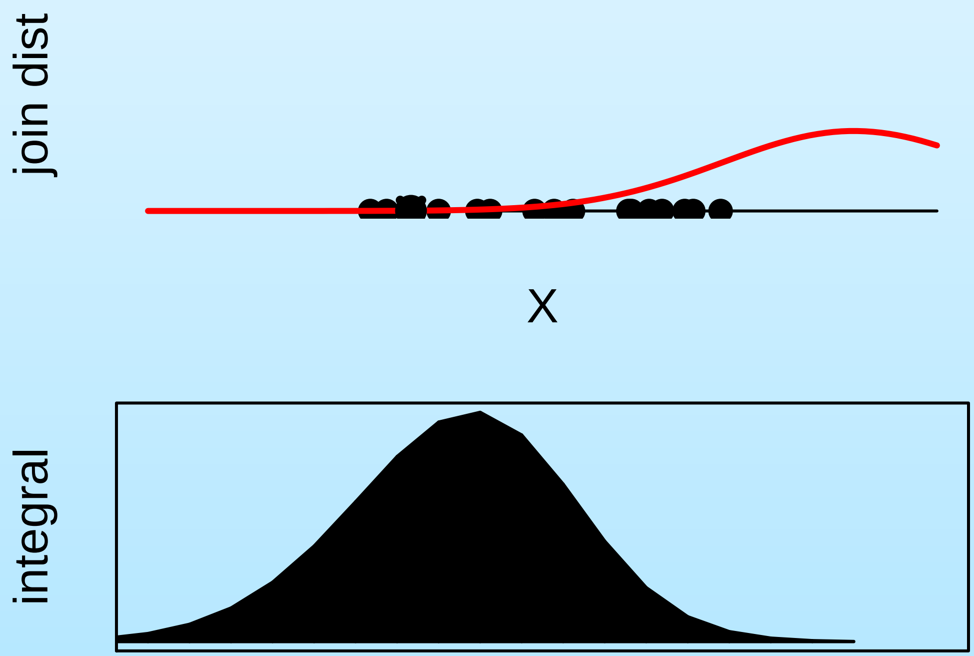
$$p(\mathbf{x}|\mathcal{D}) = \int p(\mathbf{x}|\theta)p(\theta|\mathcal{D})d\theta$$



Bayesian estimation

- **we can compute $p(\mathbf{x}|\mathcal{D})$ instead of $p(\mathbf{x}|\hat{\theta})$**
  - **integrate the join density $p(\mathbf{x}, \theta|\mathcal{D}) = p(\mathbf{x}|\theta)p(\theta|\mathcal{D})$**

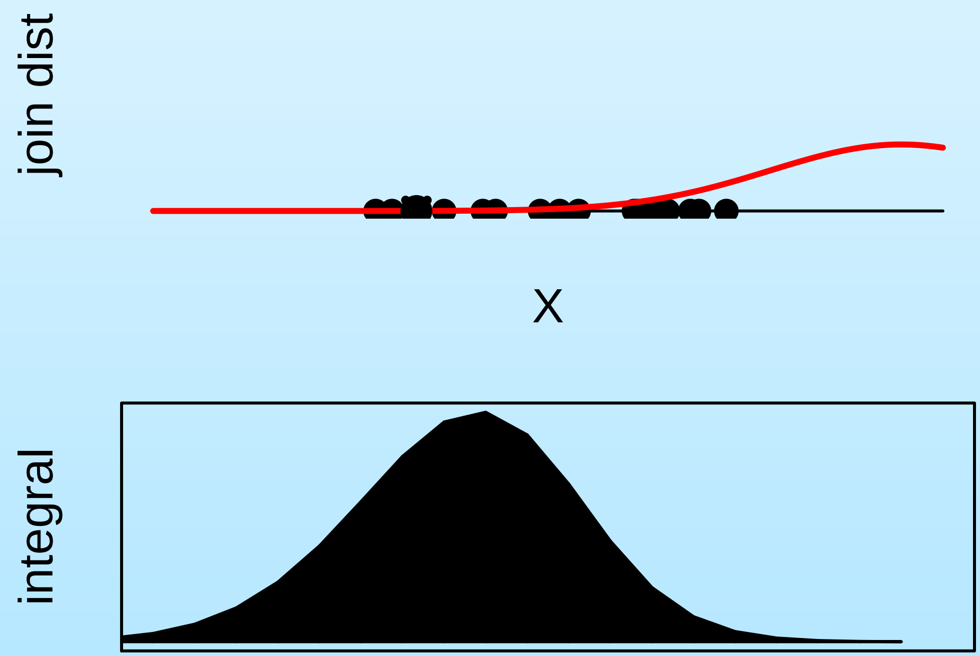
$$p(\mathbf{x}|\mathcal{D}) =$$
$$\int p(\mathbf{x}|\theta)p(\theta|\mathcal{D})d\theta$$



Bayesian estimation

- **we can compute $p(\mathbf{x}|\mathcal{D})$ instead of $p(\mathbf{x}|\hat{\theta})$**
  - **integrate the join density $p(\mathbf{x},\theta|\mathcal{D}) = p(\mathbf{x}|\theta)p(\theta|\mathcal{D})$**

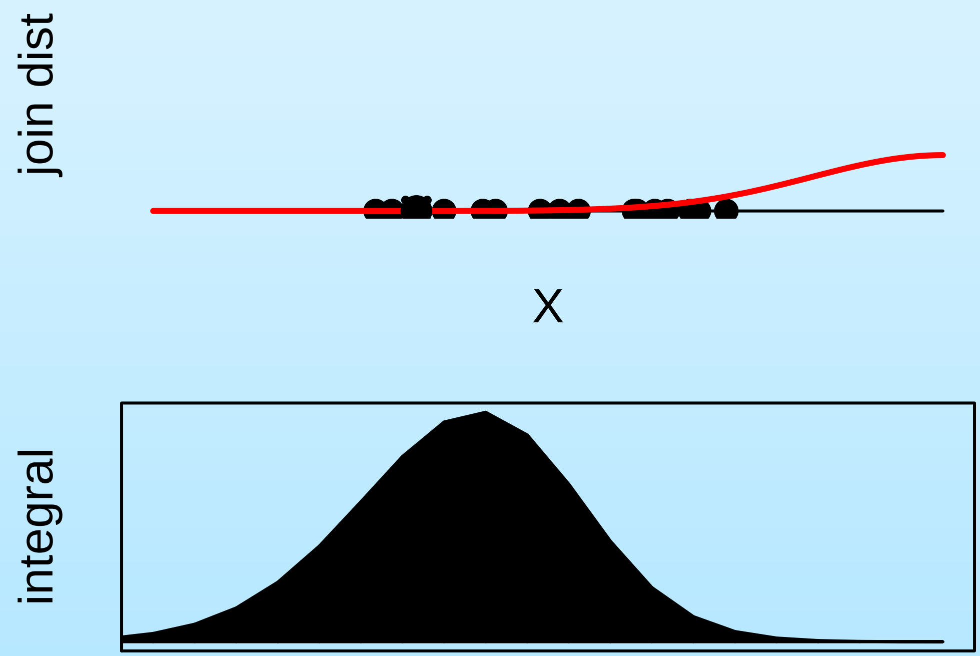
$$p(\mathbf{x}|\mathcal{D}) = \int p(\mathbf{x}|\theta)p(\theta|\mathcal{D})d\theta$$



Bayesian estimation

- **we can compute $p(\mathbf{x}|\mathcal{D})$ instead of $p(\mathbf{x}|\hat{\theta})$**
  - **integrate the join density $p(\mathbf{x}, \theta|\mathcal{D}) = p(\mathbf{x}|\theta)p(\theta|\mathcal{D})$**

$$p(\mathbf{x}|\mathcal{D}) = \int p(\mathbf{x}|\theta)p(\theta|\mathcal{D})d\theta$$



Bayesian estimation (cont.)

Pros:

- **better use of the data**
- **makes a priori assumptions explicit**
- **easily implemented recursively**
  - use posterior $p(\theta|\mathcal{D})$ as new prior

Bayesian estimation (cont.)

Pros:

- better use of the data
- makes a priori assumptions explicit
- easily implemented recursively
  - use posterior $p(\theta|\mathcal{D})$ as new prior

Cons:

- definition of noninformative priors can be tricky
- often requires numerical integration
- not widely accepted by traditional statistics (frequentism)

Outline

- What is learning?
- Parametric methods
- **Non-parametric methods**
- Stochastic methods
- Non-metric methods (skip)
- Universal principles
- Unsupervised learning
- Examples

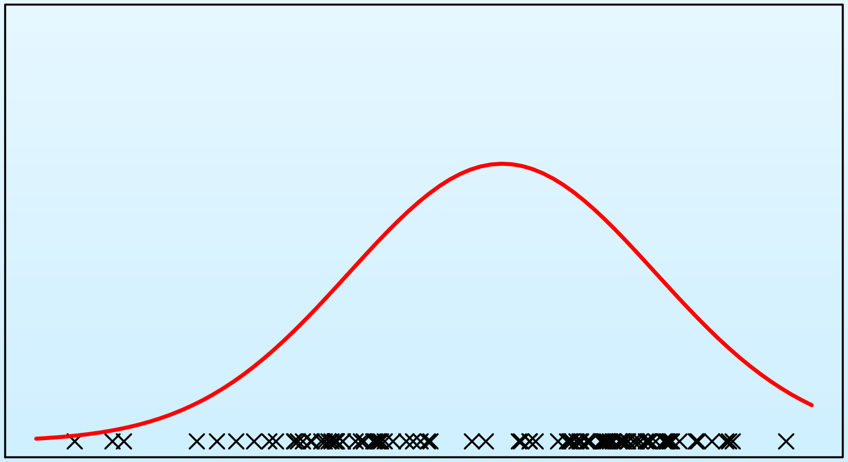
Probabilistic nonparametric methods

Parametric

non parametric

Probabilistic nonparametric methods

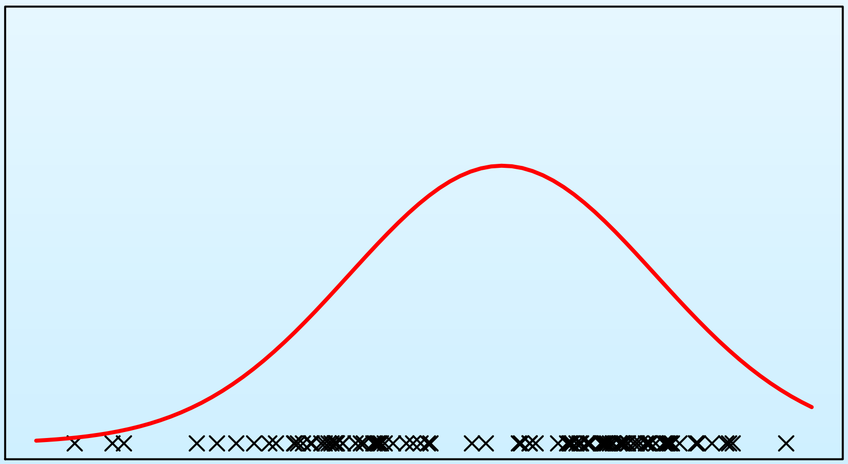
Parametric



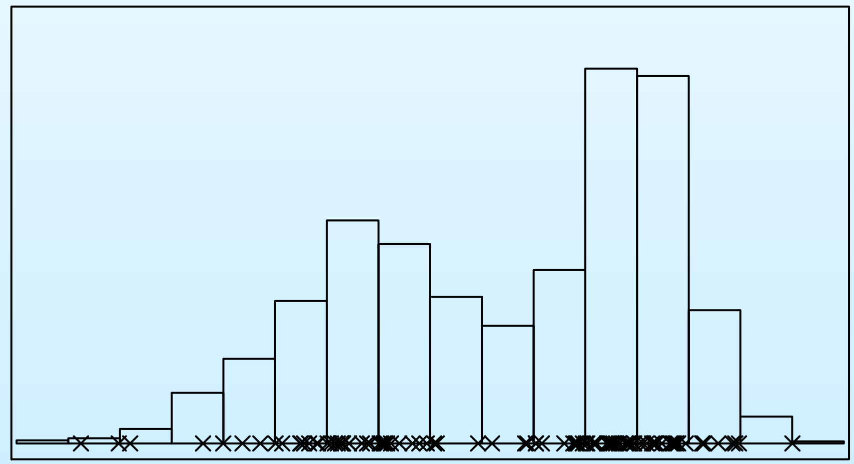
non parametric

Probabilistic nonparametric methods

Parametric

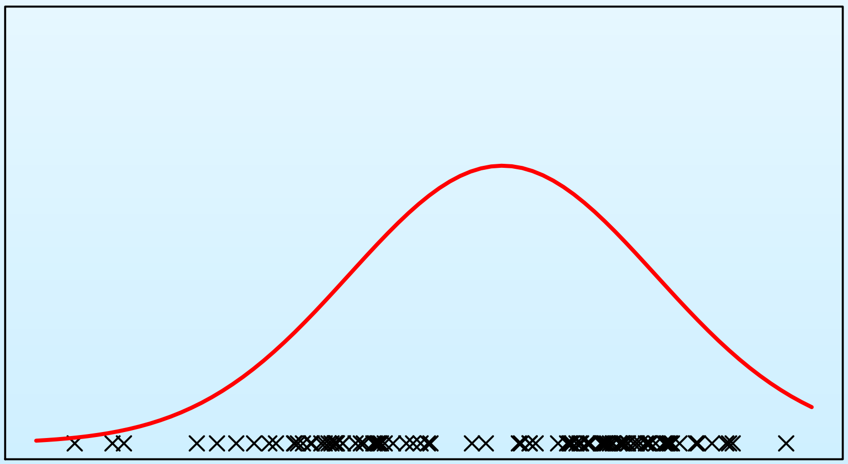


non parametric

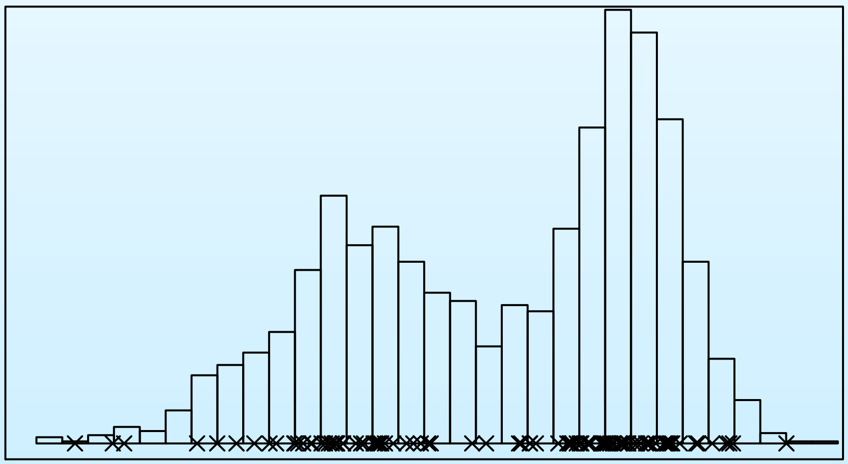


Probabilistic nonparametric methods

Parametric

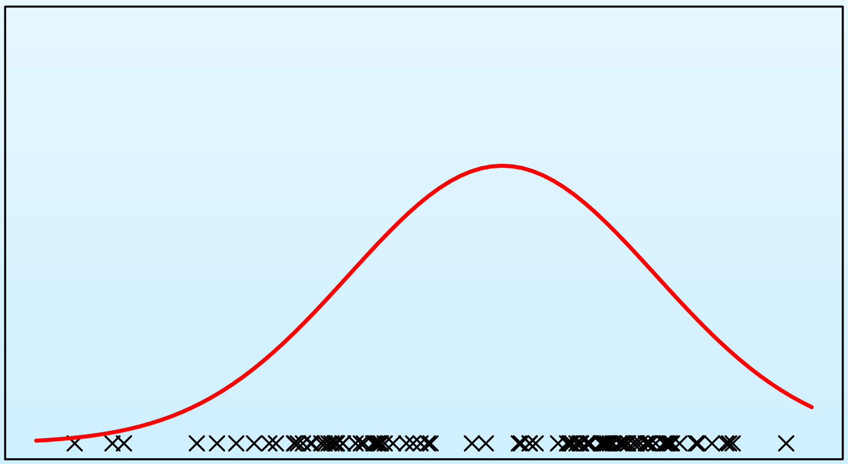


non parametric

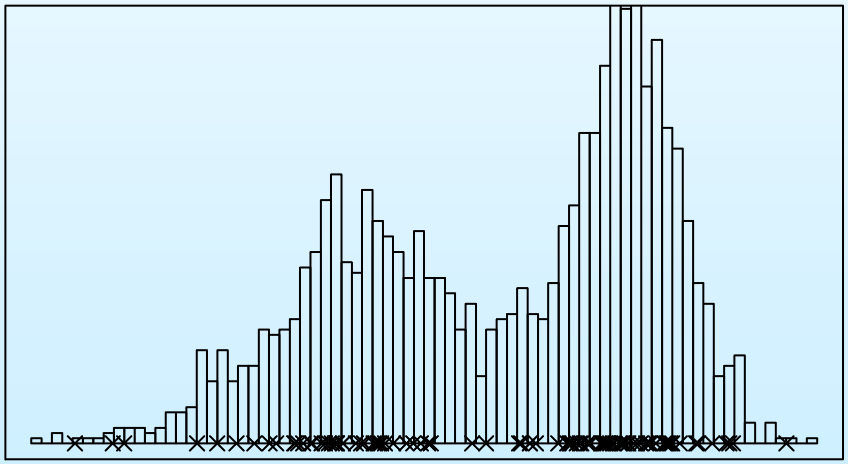


Probabilistic nonparametric methods

Parametric

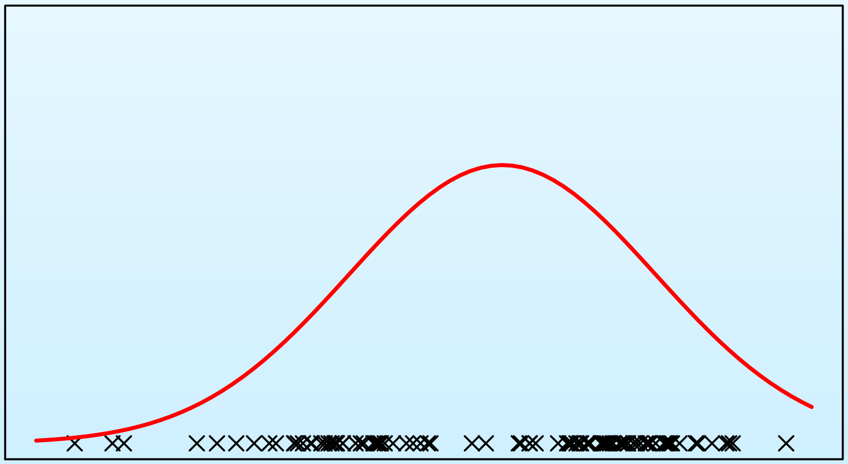


non parametric

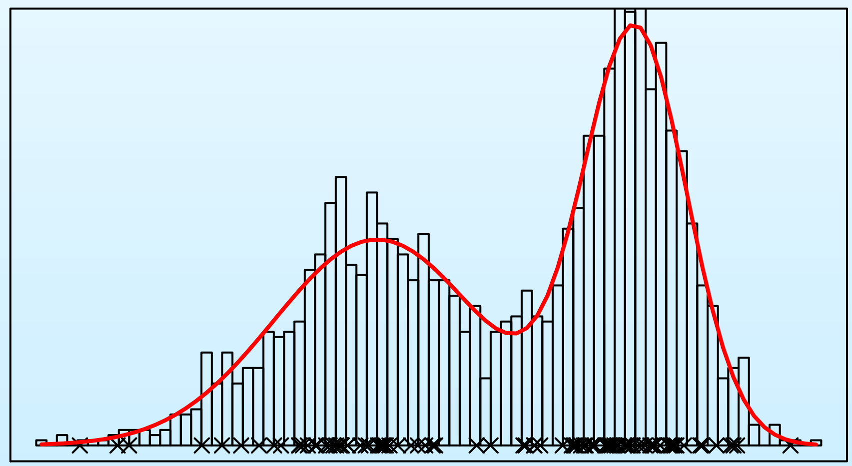


Probabilistic nonparametric methods

Parametric

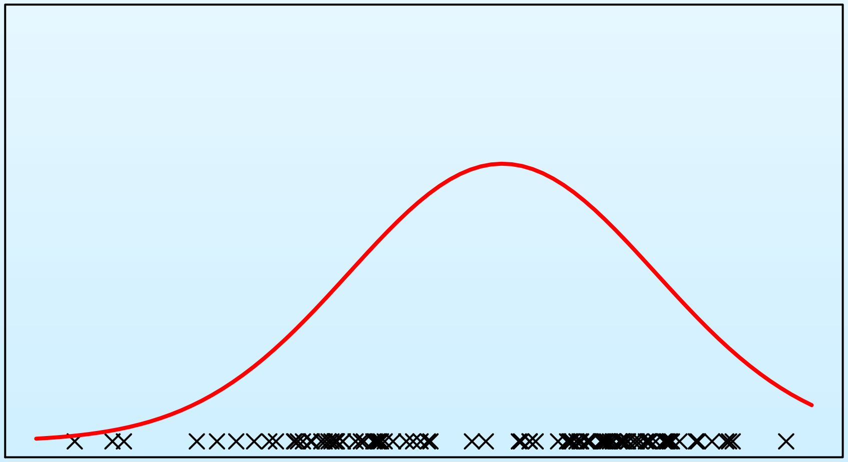


non parametric

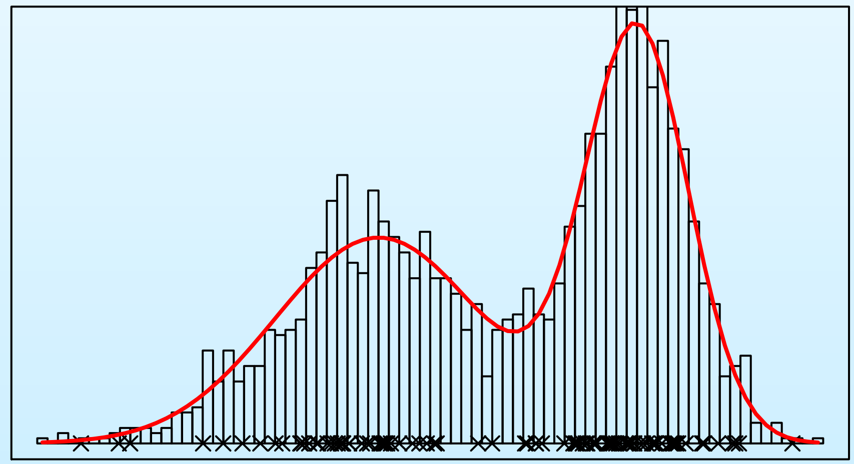


Probabilistic nonparametric methods

Parametric



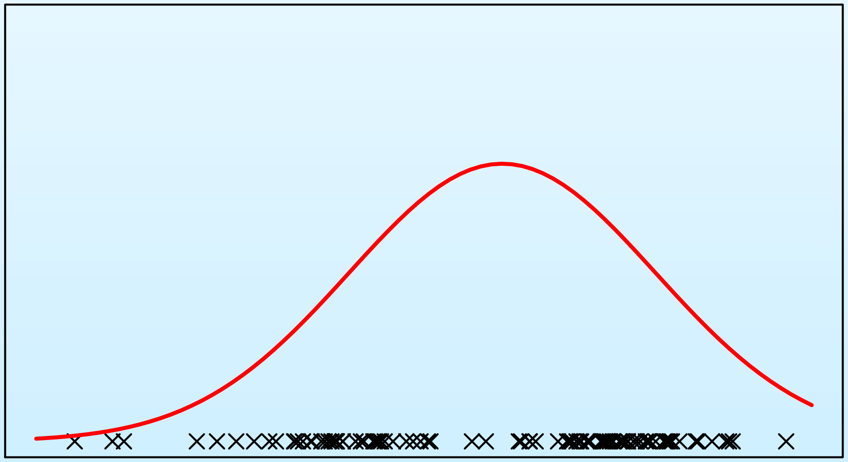
non parametric



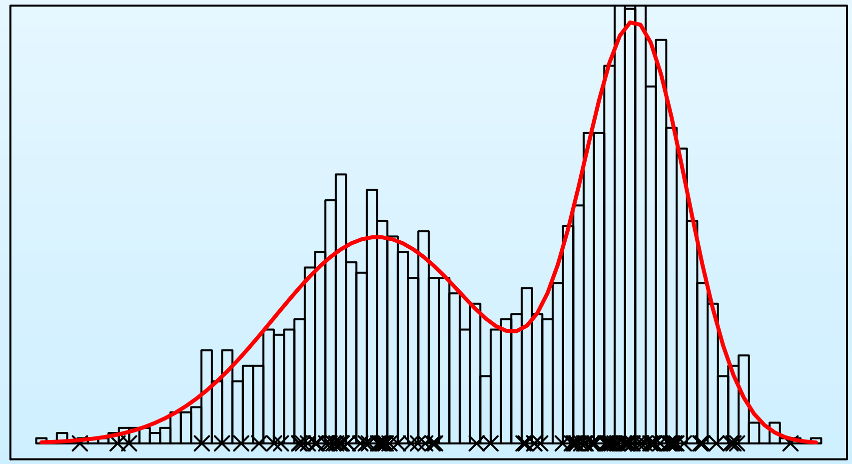
- **Parzen window**
  - **define cell volume as a function of total number of samples n**

Probabilistic nonparametric methods

Parametric



non parametric



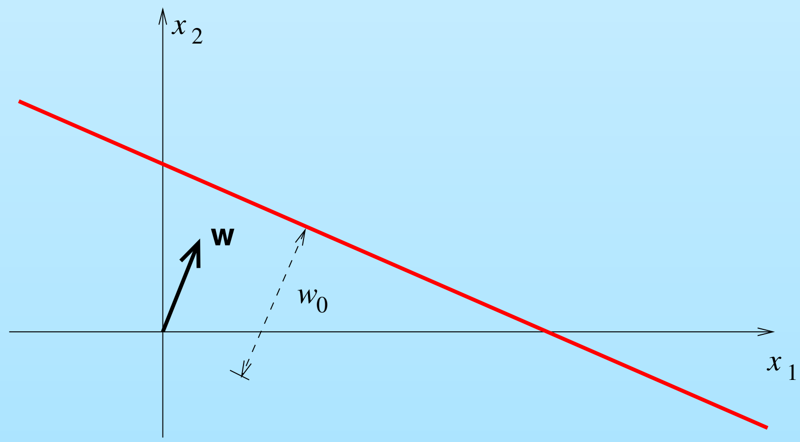
- **Parzen window**
  - **define cell volume as a function of total number of samples n**
- **k_n-nearest neighbour**
  - **define number of samples in a cell as a function of n**

Linear discriminant functions

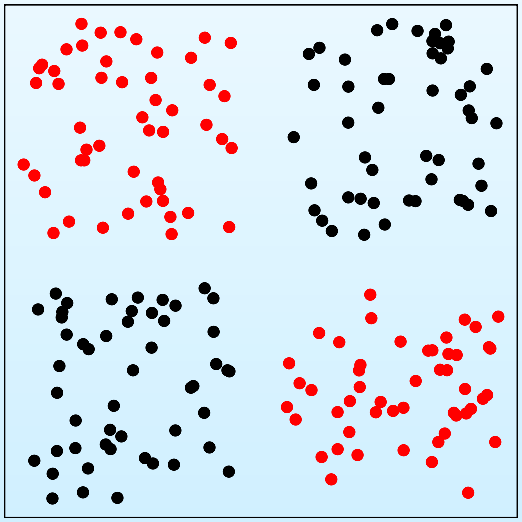
- use a linear combination of the components of $\mathbf{x}$ to rank a class

$$g_i(\mathbf{x}) = \mathbf{w}_i^t \mathbf{x} + w_{i0}$$

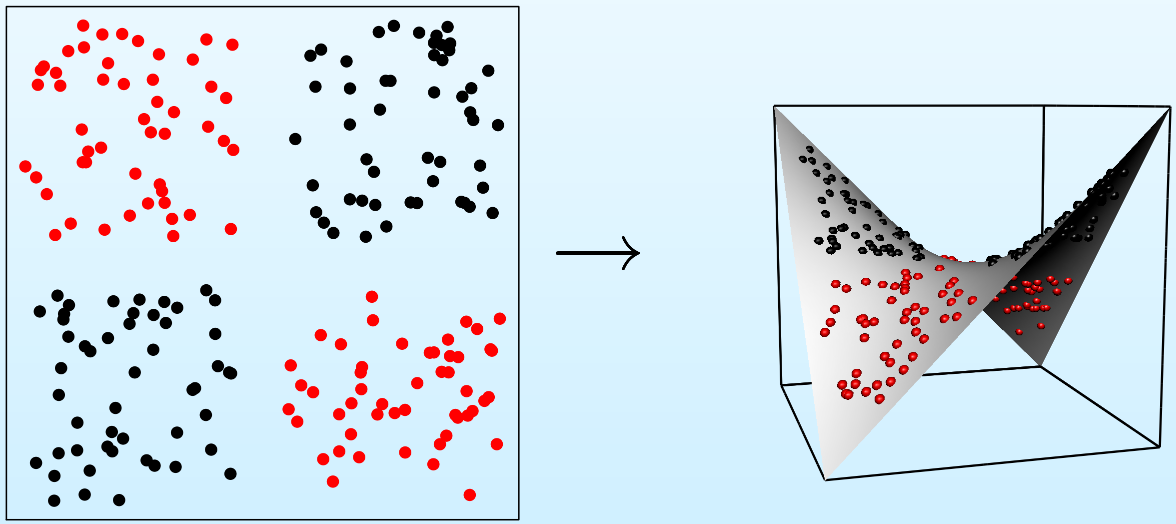
- compare the g_is to choose the best class
- for two categories $g_1(\mathbf{x}) = g_2(\mathbf{x})$ defines a hyperplane



Nonlinear extension



Nonlinear extension



- non-linearly map the features in a higher dimensional space $\mathbf{x} \rightarrow \mathbf{y}$

$$g(\mathbf{x}) = \mathbf{a}^t \mathbf{y}$$

Linear discriminant functions (cont.)

- **Gradient descent procedures**
  - define a criterion $J(\mathrm{a})$ that is maximised if a is a solution
  - update the current a with a fraction of the gradient of J

$$\mathbf{a} \leftarrow \mathbf{a} - \eta\ \boldsymbol{\Delta} J(\mathbf{a})$$

Linear discriminant functions (cont.)

■ Gradient descent procedures

- define a criterion $J(\mathrm{a})$ that is maximised if a is a solution
- update the current a with a fraction of the gradient of J

$$\mathbf{a} \leftarrow \mathbf{a} - \eta\ \mathbf{\Delta} J(\mathbf{a})$$

- Perceptron criterion: $J_p(\mathbf{a}) = \sum_{y \in \mathcal{Y}}(-\mathbf{a}^t\mathbf{y})$ where $\mathcal{Y}$ is the set of misclassified samples

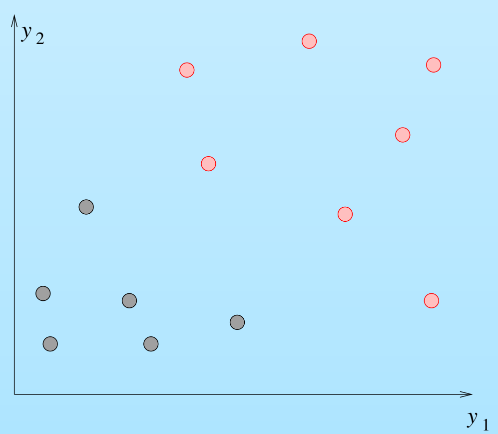
Linear discriminant functions (cont.)

Gradient descent procedures

- define a criterion $J(\mathbf{a})$ that is maximised if $\mathbf{a}$ is a solution
- update the current a with a fraction of the gradient of J

$$\mathbf{a} \leftarrow \mathbf{a} - \eta\ \boldsymbol{\Delta} J(\mathbf{a})$$

- Perceptron criterion: $J_p(\mathbf{a}) = \sum_{y \in \mathcal{Y}} (-\mathbf{a}^t \mathbf{y})$ where $\mathcal{Y}$ is the set of misclassified samples

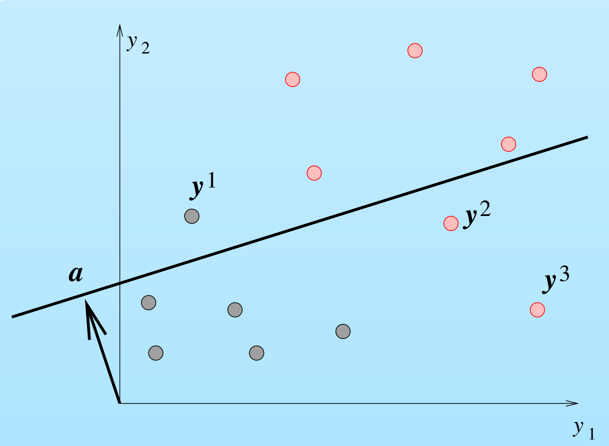


Linear discriminant functions (cont.)

- **Gradient descent procedures**
  - define a criterion $J(\mathbf{a})$ that is maximised if $\mathbf{a}$ is a solution
  - update the current a with a fraction of the gradient of J

$$\mathbf{a} \leftarrow \mathbf{a} - \eta\ \mathbf{\Delta} J(\mathbf{a})$$

  - Perceptron criterion: $J_p(\mathbf{a}) = \sum_{y \in \mathcal{Y}} (-\mathbf{a}^t \mathbf{y})$ where $\mathcal{Y}$ is the set of misclassified samples

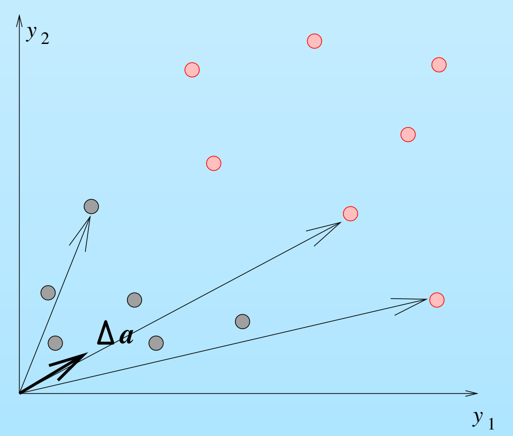


Linear discriminant functions (cont.)

- **Gradient descent procedures**
  - define a criterion $J(\mathbf{a})$ that is maximised if $\mathbf{a}$ is a solution
  - update the current a with a fraction of the gradient of J

$$\mathbf{a} \leftarrow \mathbf{a} - \eta\ \boldsymbol{\Delta} J(\mathbf{a})$$

  - Perceptron criterion: $J_p(\mathbf{a}) = \sum_{y \in \mathcal{Y}} (-\mathbf{a}^t \mathbf{y})$ where $\mathcal{Y}$ is the set of misclassified samples



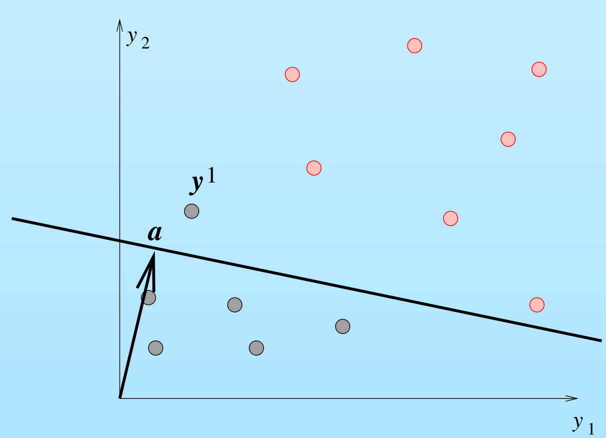
Linear discriminant functions (cont.)

Gradient descent procedures

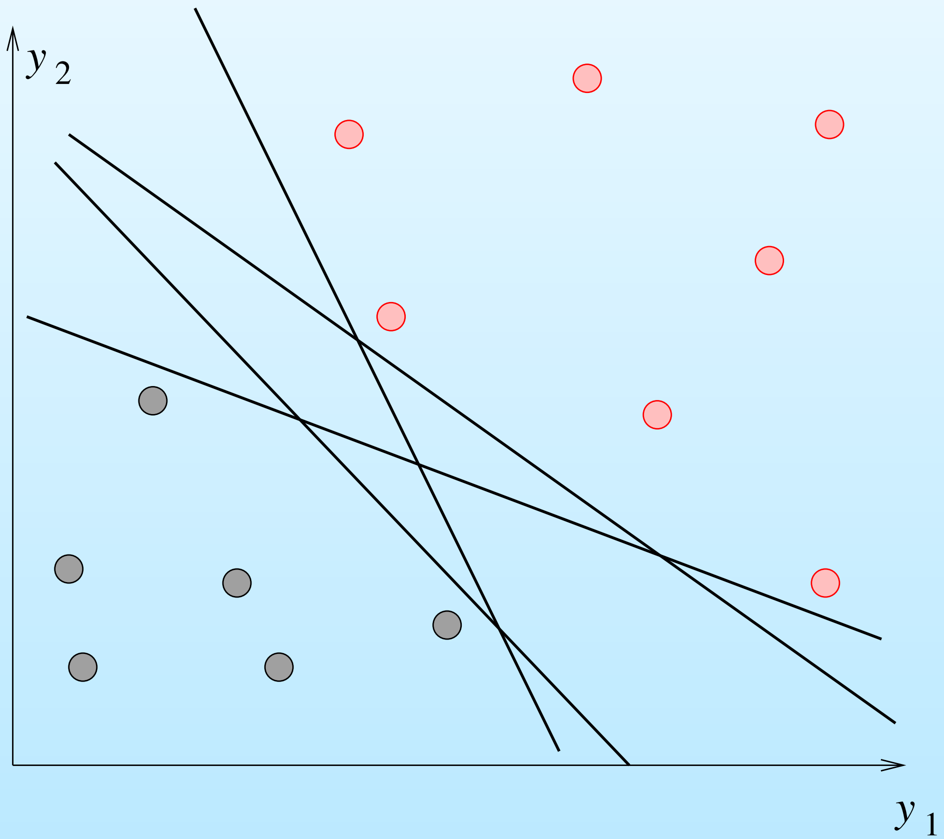
- define a criterion $J(\mathbf{a})$ that is maximised if $\mathbf{a}$ is a solution
- update the current a with a fraction of the gradient of J

$$\mathbf{a} \leftarrow \mathbf{a} - \eta\ \boldsymbol{\Delta} J(\mathbf{a})$$

- Perceptron criterion: $J_p(\mathbf{a}) = \sum_{y \in \mathcal{Y}} (-\mathbf{a}^t \mathbf{y})$ where $\mathcal{Y}$ is the set of misclassified samples

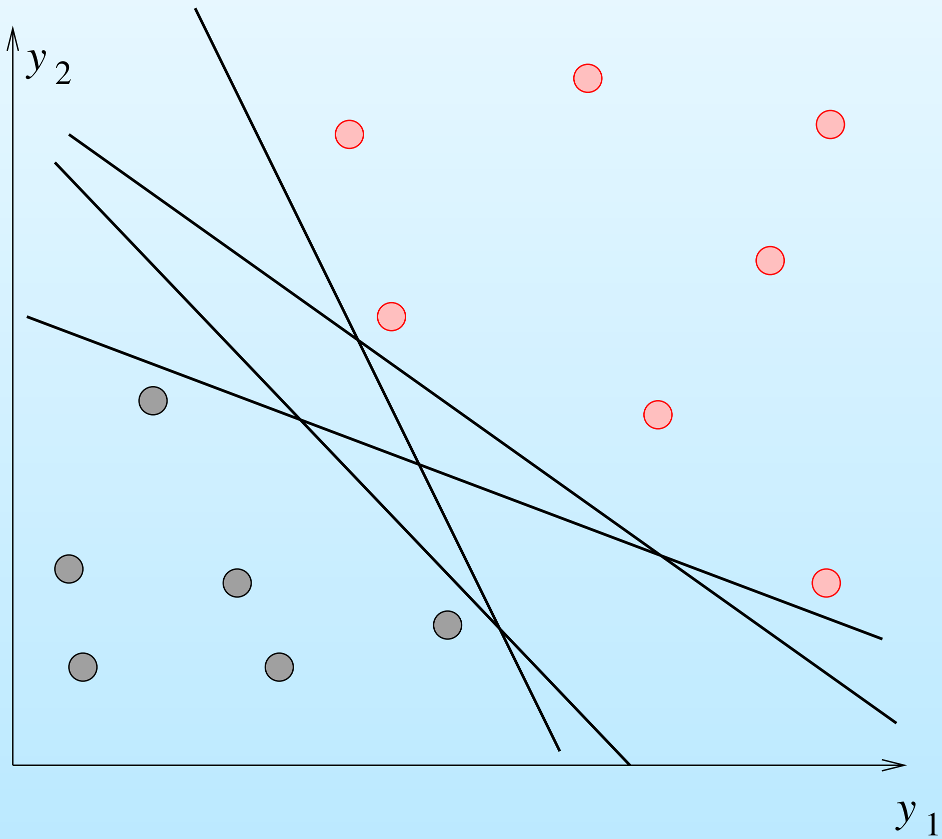


Perceptron

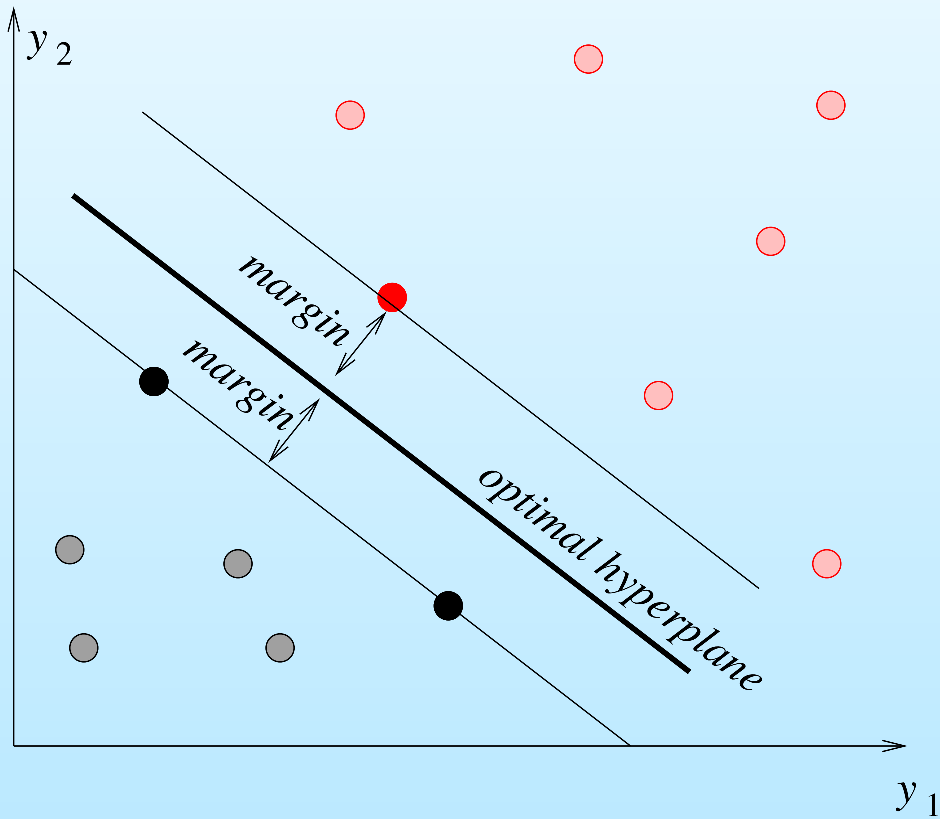


Support vector machines

Perceptron

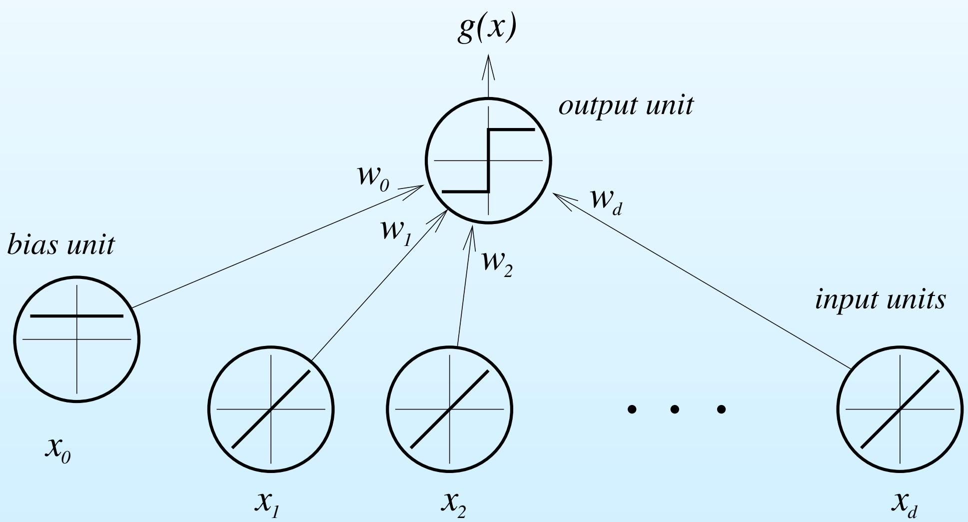


Support vector machine



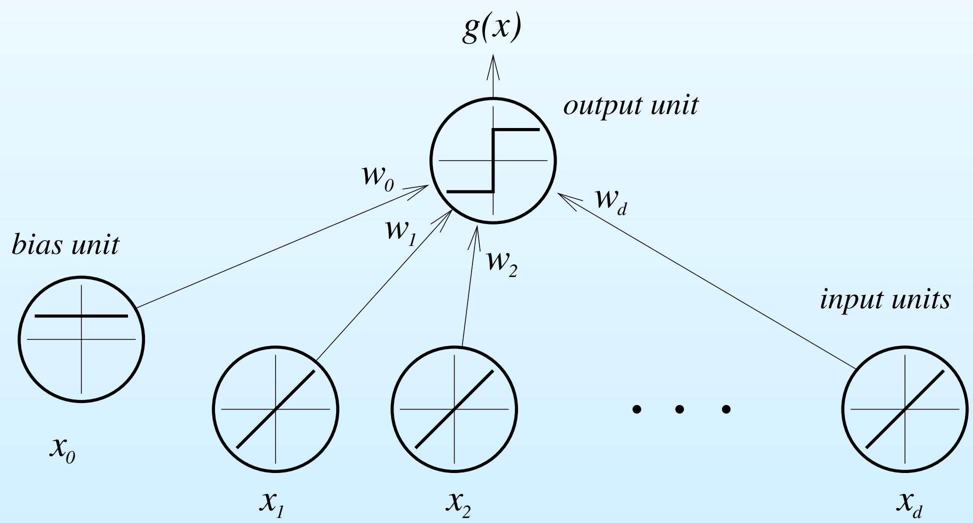
Multi layer neural networks

Linear discriminants

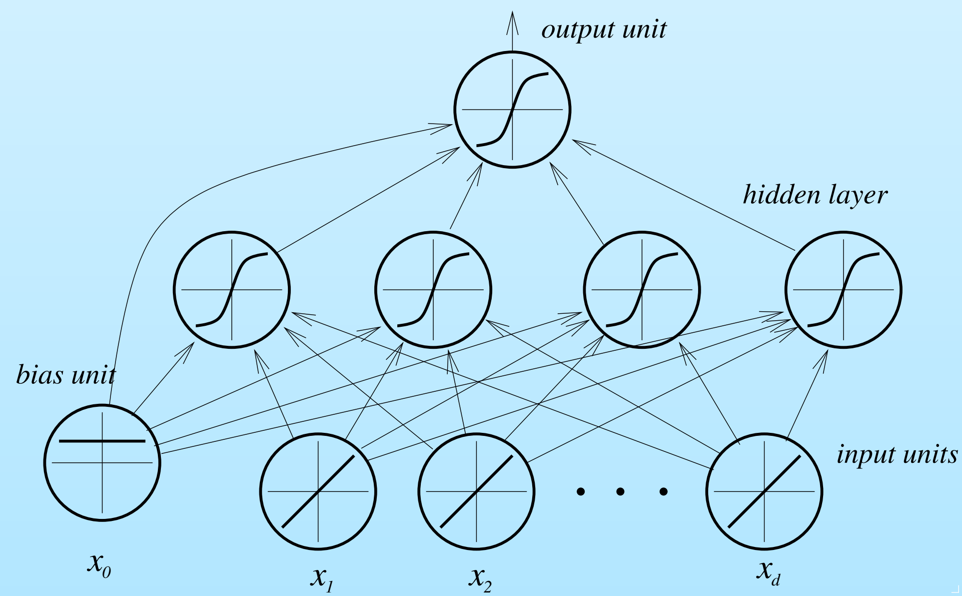


Multi layer neural networks

Linear discriminants

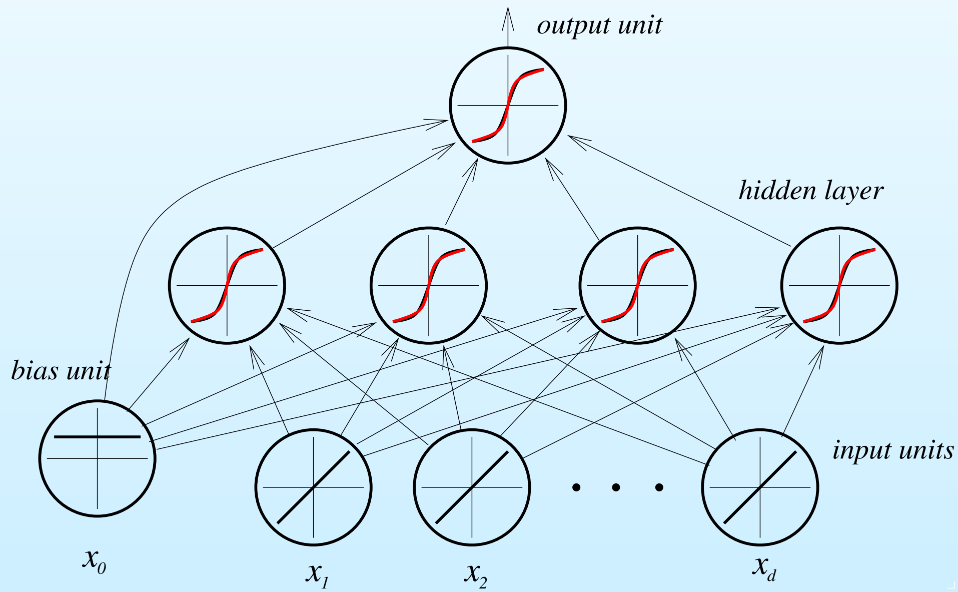


Multi layer neural networks



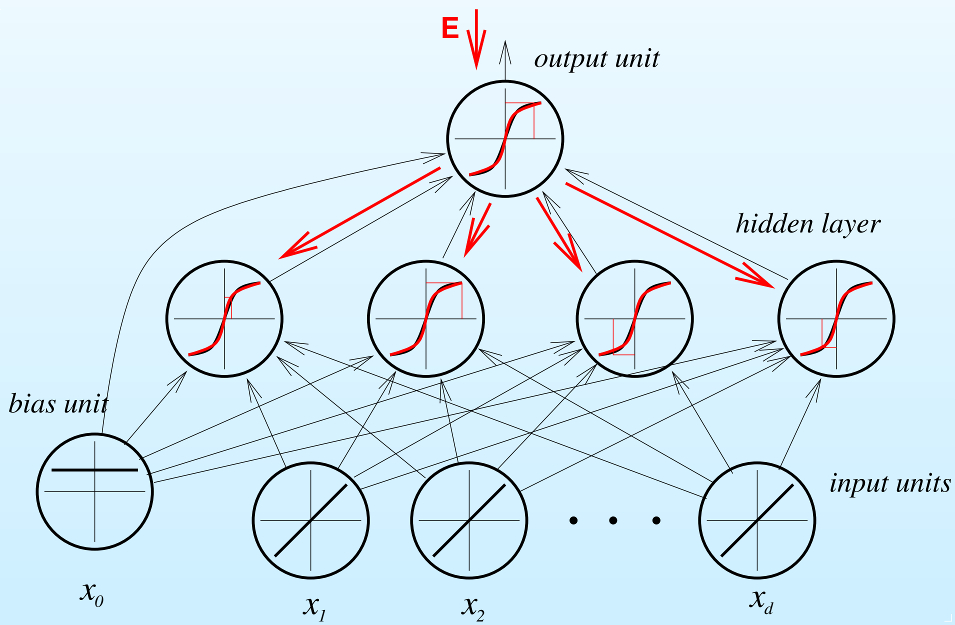
Multi layer neural networks (cont.)

Multi layer neural networks



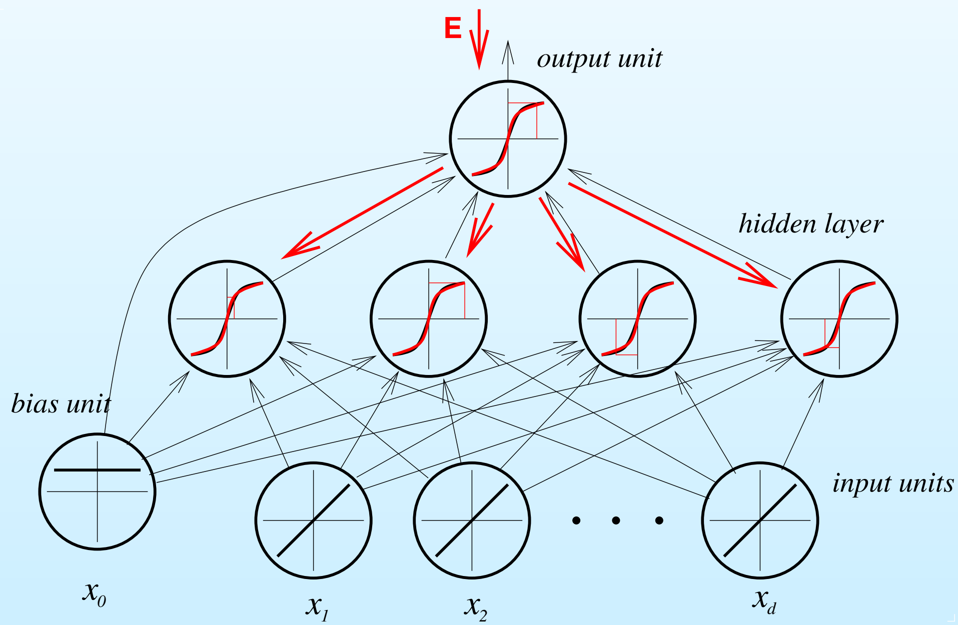
Multi layer neural networks (cont.)

Multi layer neural networks



Multi layer neural networks (cont.)

Multi layer
neural networks



- **Backpropagation algorithm**

Outline

- What is learning?
- Parametric methods
- Non-parametric methods
- **Stochastic methods**
- Non-metric methods (skip)
- Universal principles
- Unsupervised learning
- Examples

Stochastic methods

- **Gradient descent procedure find local minima**

Stochastic methods

- Gradient descent procedure find local minima
- solution: repeat training several times with different initialisations

Stochastic methods

- Gradient descent procedure find local minima
- solution: repeat training several times with different initialisations
- Simulated annealing
  - based on concepts from physics
  - well grounded theoretically
- Boltzmann learning

Stochastic methods

- Gradient descent procedure find local minima
- solution: repeat training several times with different initialisations
- Simulated annealing
  - based on concepts from physics
  - well grounded theoretically
- Boltzmann learning
- Evolutionary methods (Genetic algorithms)
  - based on concepts from biology
  - no theory behind: heuristic

Generations

Generation k: chromosomes	Generation k: after ranking	Generation k+1: survival + reproduction
01110110100100101001	10010100101010101010	10010100101010101010
10010100101010101010	01010111110100101100	01010111110100101100
01001010001010010101	10101001000101010010	10101001000101010010
10101001000101010010	01010010101010001010	01010010101010001010
01001001010101001000	00101011110100011111	10010100100100101100
00010111101010101010	01001010001010010101	01010111110101010010
01010111110100101100	00010111101010101010	10101001001010001010
00101011110100011111	01110110100100101001	01010010100101010010
01010010101010001010	01001001010101001000	01010010101010101010

Genetic algorithms

Generations

Generation k: chromosomes	Generation k: after ranking	Generation k+1: survival + reproduction
01110110100100101001	100101001010101010	100101001010101010
100101001010101010	01010111110100101100	01010111110100101100
01001010001010010101	10101001000101010010	10101001000101010010
10101001000101010010	01010010101010001010	01010010101010001010
01001001010101001000	00101011110100011111	10010100100100101100
00010111101010101010	01001010001010010101	01010111110101010010
01010111110100101100	00010111101010101010	10101001001010001010
00101011110100011111	01110110100100101001	01010010100101010010
01010010101010001010	01001001010101001000	01010010101010101010

Genetic operators

	replication (survival)	crossover	mutation
gen k	01110110100100101001	1001010010 | 1010101010 0101011111 | 0100101100	100101001010101010
gen k+1	01110110100100101001	10010100100100101100 01010111111010101010	10000100111010001110

Outline

- What is learning?
- Parametric methods
- Non-parametric methods
- Stochastic methods
- Non-metric methods (skip)
- **Universal principles**
- Unsupervised learning
- Examples

Universal principles

- **No free lunch theorem**
  - **if we make no** *prior assumptions* **on the nature of the problem, no** *learning method* **can be proved to be superior to any other, not even random guessing**

Universal principles

- **No free lunch theorem**
  - if we make no *prior assumptions* on the nature of the problem, no *learning method* can be proved to be superior to any other, not even random guessing
- **Ugly duckling theorem**
  - if we make no *prior assumptions* on the nature of the problem, no *feature representation* should be preferred to any other

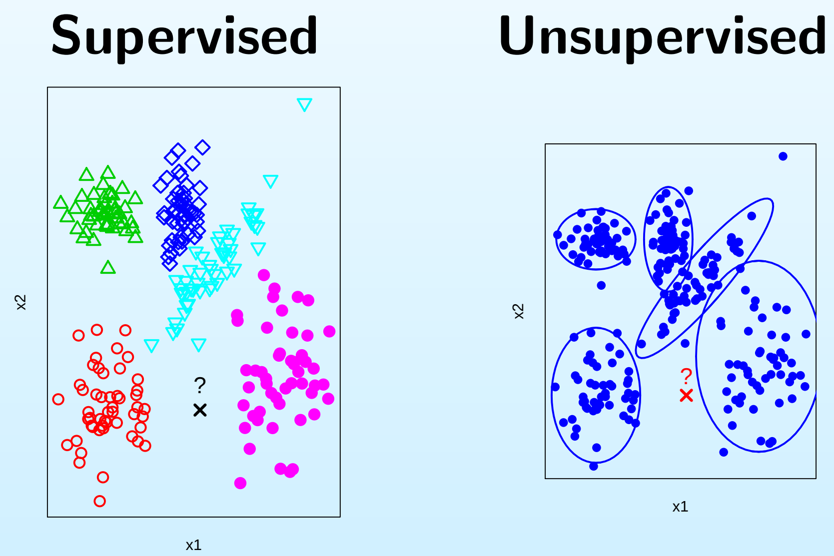
Universal principles

- **No free lunch theorem**
  - if we make no *prior assumptions* on the nature of the problem, no *learning method* can be proved to be superior to any other, not even random guessing
- **Ugly duckling theorem**
  - if we make no *prior assumptions* on the nature of the problem, no *feature representation* should be preferred to any other
- **Minimum description length principle**
  - prefer low complexity solutions. True only asymptotically, but valid in practice
- **Occam's razor**
  - avoid overfitting

Outline

- What is learning?
- Parametric methods
- Non-parametric methods
- Stochastic methods
- Non-metric methods (skip)
- Universal principles
- **Unsupervised learning**
- Examples

Unsupervised learning



- release assumption on class independence
- learn a mixture of distributions
- the parametric solution is formally similar, but different in practice

Model based clustering

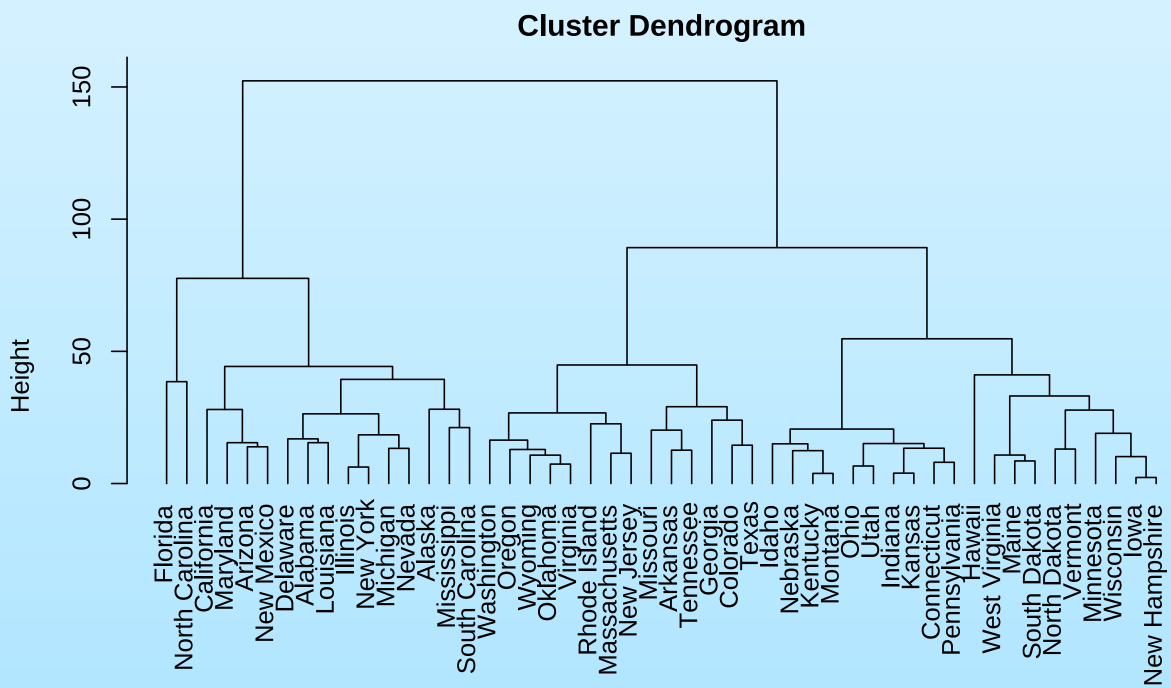
- A maximum likelihood solution is the Expectation Maximisation algorithm
- Problem with missing data (class membership $\forall \mathbf{x}_k \in \mathcal{D}$)
- Solution:
  - assume the missing data is known
  - compute and maximise likelihood
  - estimate the new best guess for the missing data
  - iterate
- guaranteed to find ML solution with *marginalised* missing data

Heuristic methods

- k-means clustering
  - use Euclidean distance as similarity measure
  - define k centroids
  - assign data points to the nearest centroid
  - recompute centroids
  - iterate
- Properties
  - is equivalent to Model Based Clustering with equal and spherical covariances

Heuristic methods (cont.)

- **hierarchical clustering**
  - start with one cluster per data point
  - iteratively merge most similar clusters
  - single linkage, complete linkage, average linkage, ...



Number of clusters

- what if the number of clusters is not known?
- Large number of heuristic methods
  - measure the within and across cluster spread

Number of clusters

- what if the number of clusters is not known?
- Large number of heuristic methods
  - measure the within and across cluster spread
- Bayes Information Criterion
  - model fit to the data: likelihood
  - model complexity in number of parameters (minimum description length principle)
  - number of data points available for parameter estimation

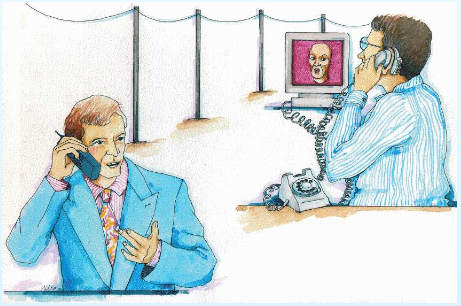
Outline

- What is learning?
- Parametric methods
- Non-parametric methods
- Stochastic methods
- Non-metric methods (skip)
- Universal principles
- Unsupervised learning
- **Examples**

Examples

- **Synface: map acoustic to visual information in speech**
- **Accent analysis with hierarchical agglomerative clustering**
- **Mille, model first language learning with Model Based Clustering**

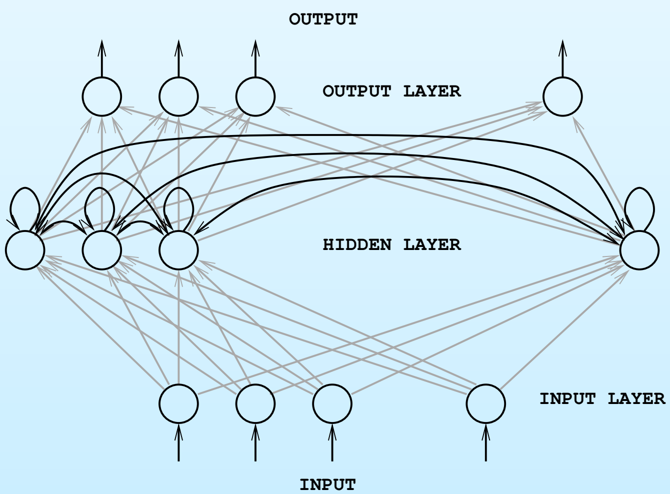
Synface



- idea: use a synthesized talking face derived from speech as a hearing aid for users of voice channels
- problem: extract (phonetic) information from the speech signal with very low latencies ($\sim 50ms$)
- it is a regression problem
- ...but, solved as a classification problem
  - map acoustic signal to visemes
  - use rules to generate the lip movements

Synface: methods

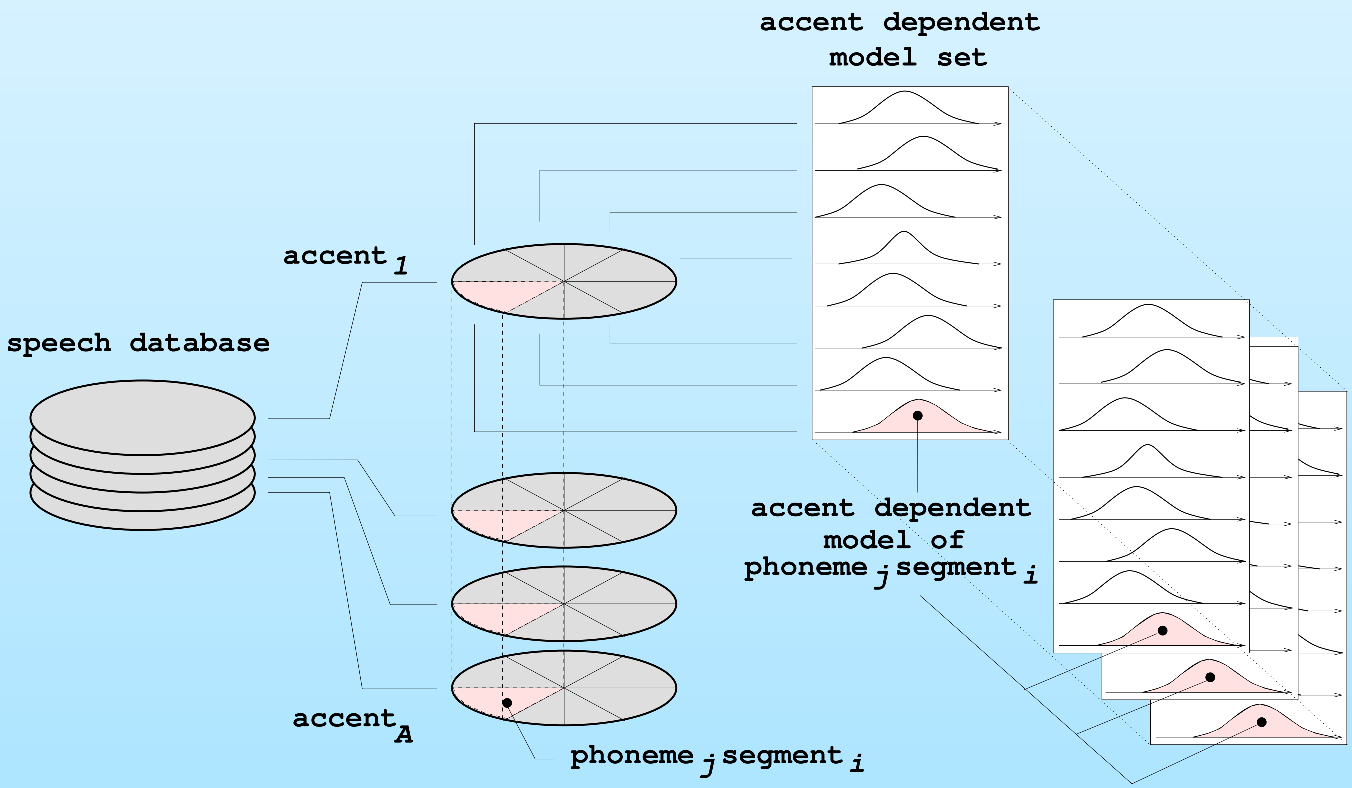
- **Recurrent neural network**



- **Hidden Markov models**

Accent clustering

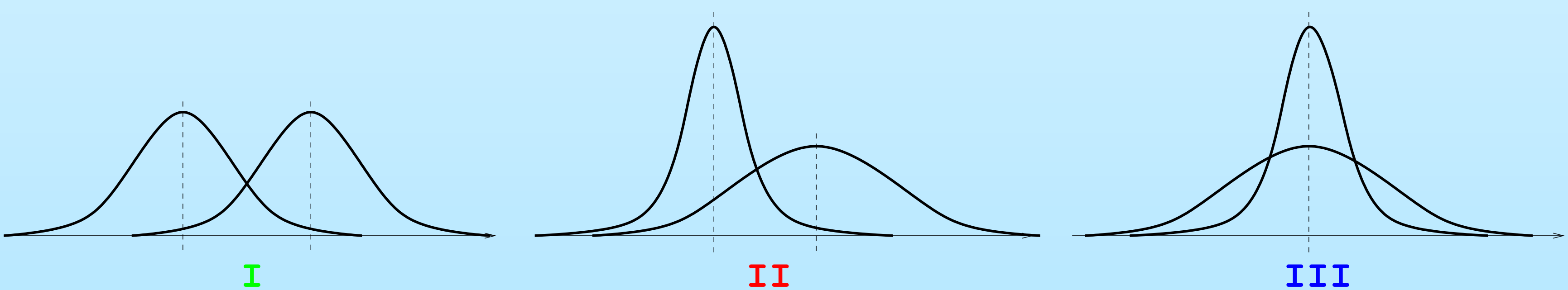
- aim: analysis of regional pronunciation variation on large data sets (∼5000 speakers)
- how? Automate part of the process with data mining techniques



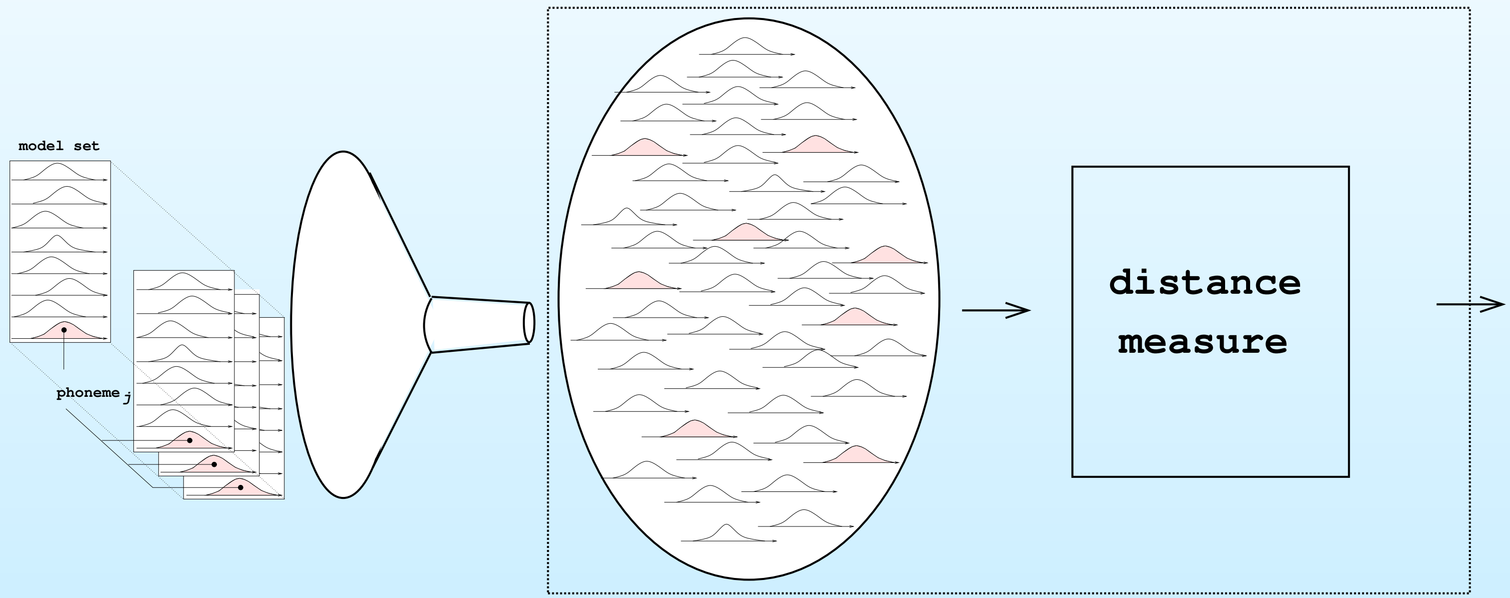
Accent clustering: distance measure

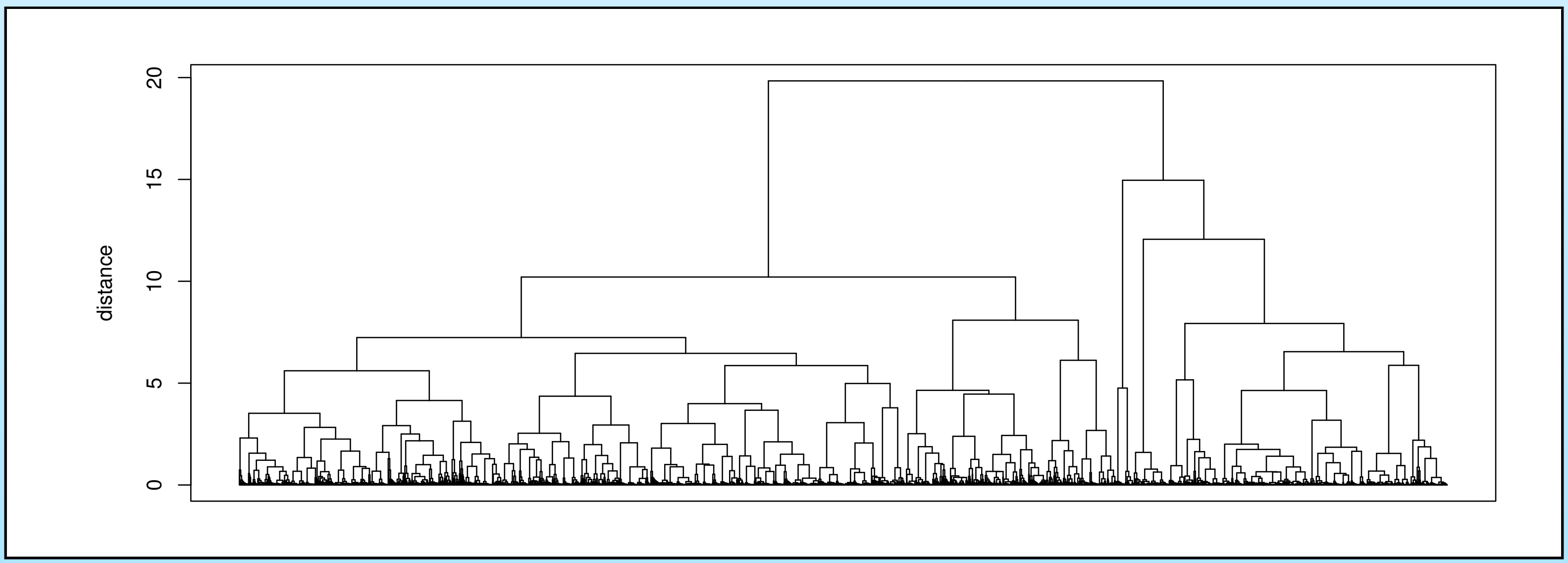
- **Analyse differences between groups by comparing distributions**
  - **metric based on Bhattacharyya distance**

$$D_{\mathsf{bhatt}}(\Theta_1,\Theta_2) = \overbrace{\underbrace{\frac{1}{8}(M_2-M_1)^T\left[\frac{\Sigma_1+\Sigma_2}{2}\right]^{-1}(M_2-M_1)}_{\mathsf{I}} + \underbrace{\frac{1}{2}\ln\frac{\left|\frac{\Sigma_1+\Sigma_2}{2}\right|}{\sqrt{|\Sigma_1||\Sigma_2|}}}_{\mathsf{III}}}^{\mathsf{II}}$$



Accent clustering (cont.)



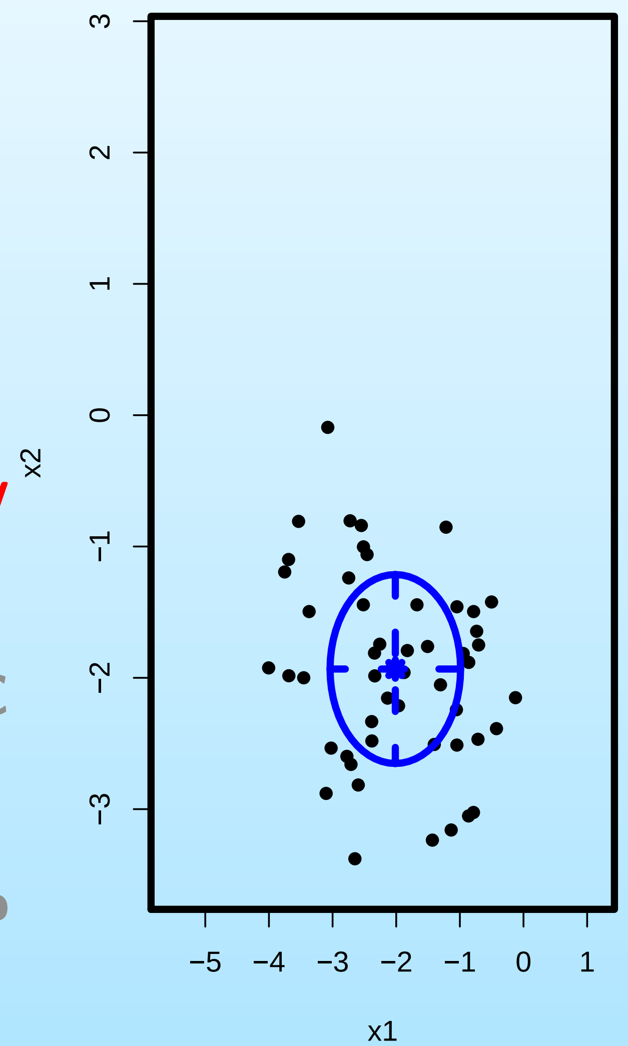


Mille

- Background: infants have no innate linguistic knowledge
- Aim (long term): mathematical modelling of the learning process
  - acoustic features classification
  - time integration into meaningful sequences
- Aim (so far): spectral features classification
  - unsupervised
  - incremental

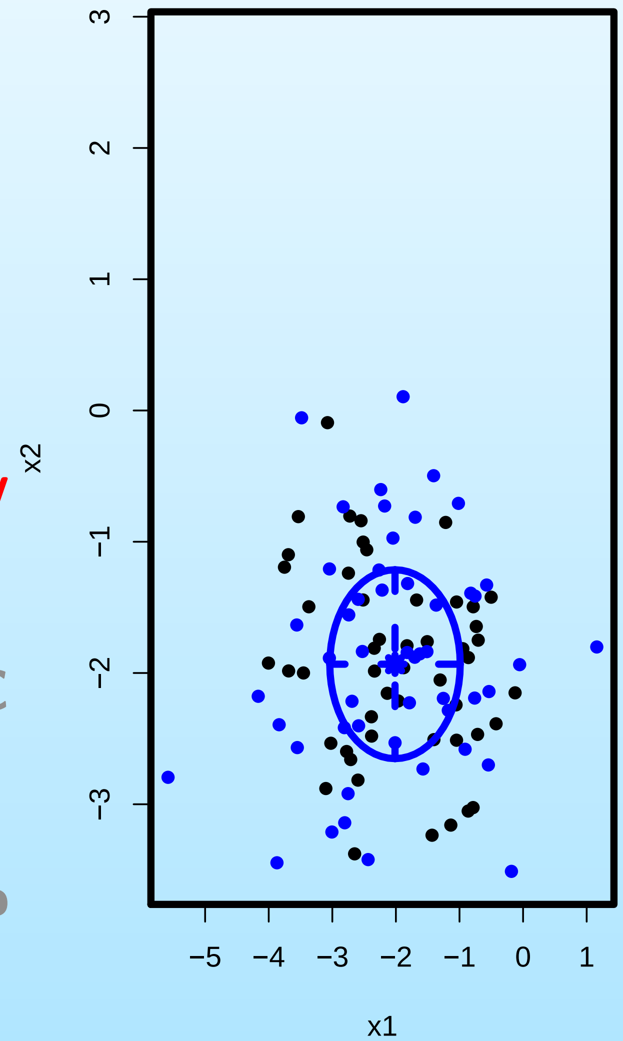
Mille: Algorithm

1. start with a MCLUST model
2. get new data
3. adjust old model to new data
4. divide new data into well and poorly modelled points
5. try a more complex model, if better BIC set as best and go back to 4
6. set the current best model and go back to 2



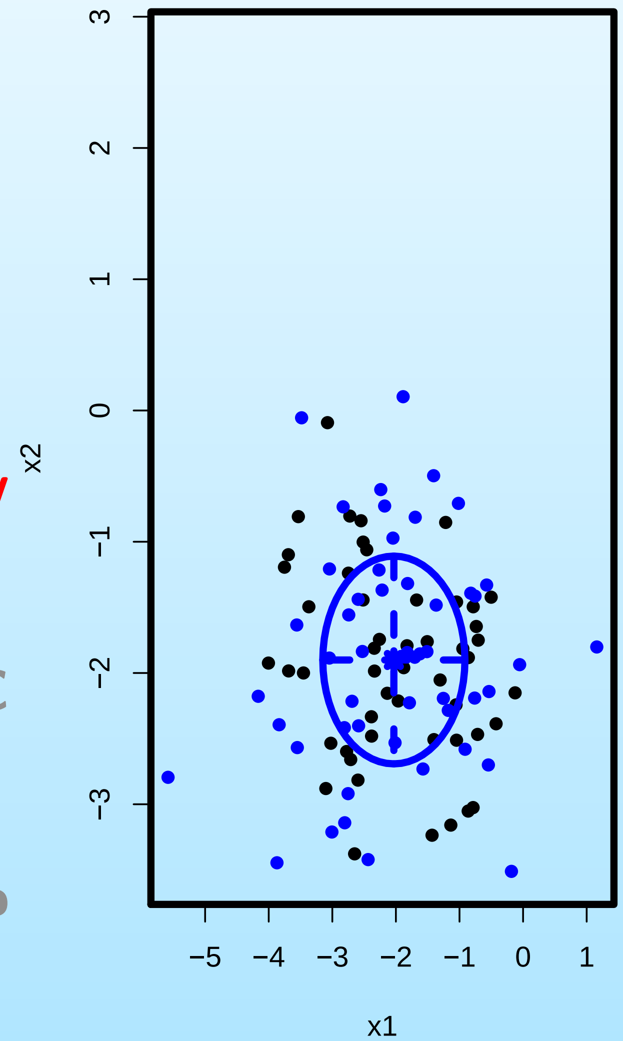
Mille: Algorithm

1. start with a MCLUST model
2. **get new data**
3. adjust old model to new data
4. divide new data into well and poorly modelled points
5. try a more complex model, if better BIC set as best and go back to 4
6. set the current best model and go back to 2



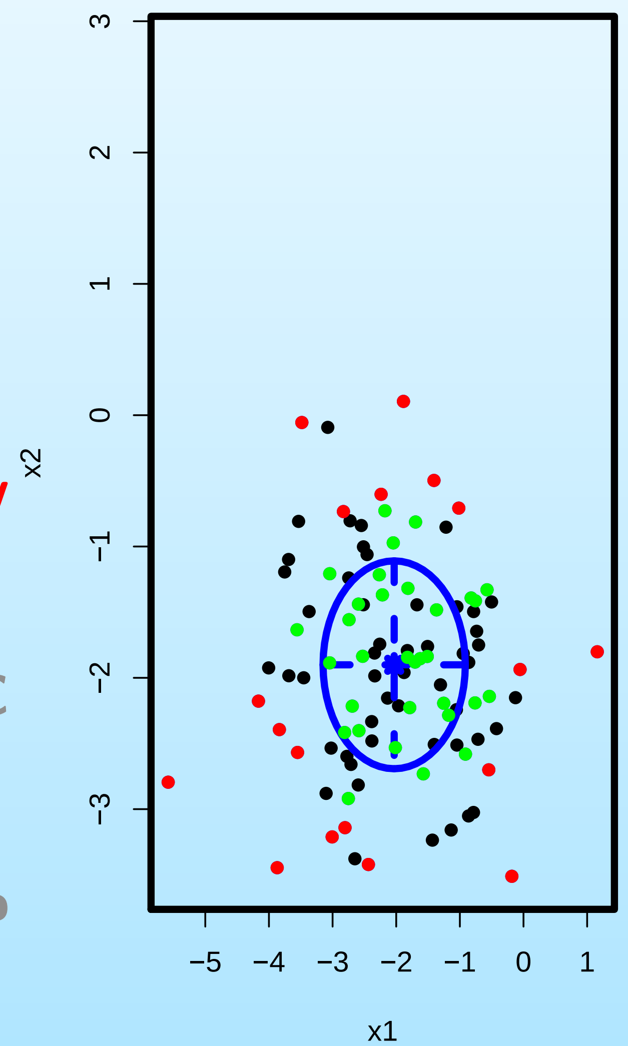
Mille: Algorithm

1. start with a MCLUST model
2. get new data
3. **adjust old model to new data**
4. divide new data into well and poorly modelled points
5. try a more complex model, if better BIC set as best and go back to 4
6. set the current best model and go back to 2



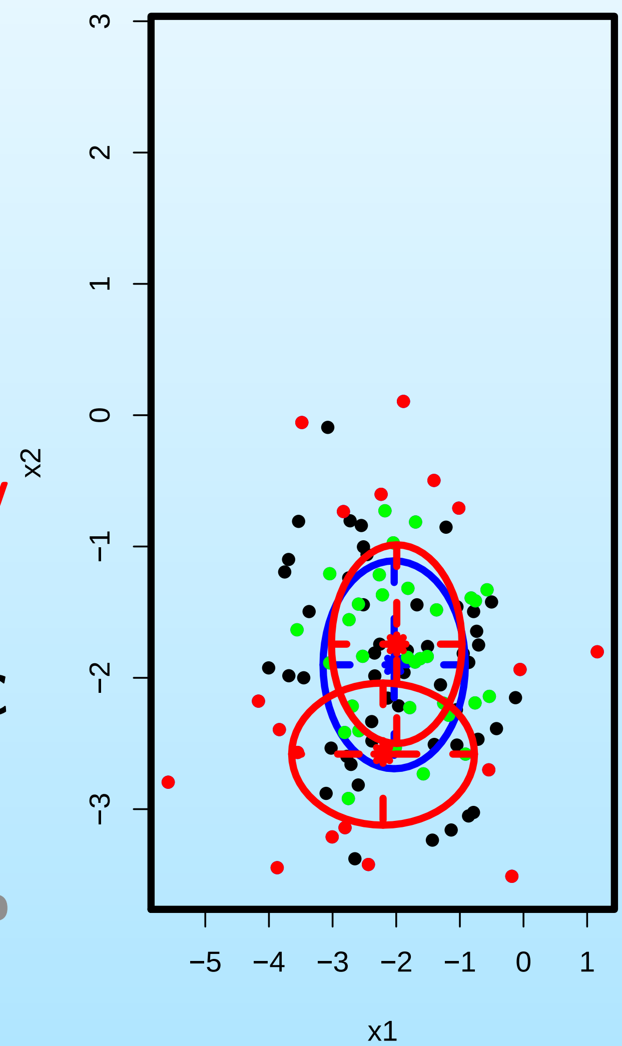
Mille: Algorithm

1. start with a MCLUST model
2. get new data
3. adjust old model to new data
4. **divide new data into well and poorly modelled points**
5. try a more complex model, if better BIC set as best and go back to 4
6. set the current best model and go back to 2



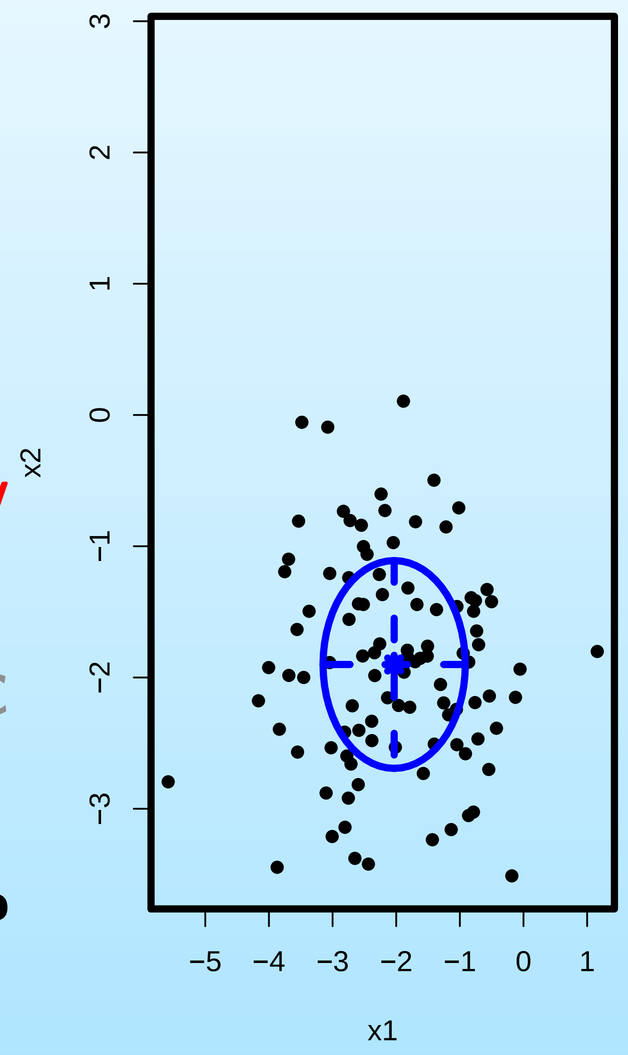
Mille: Algorithm

1. start with a MCLUST model
2. get new data
3. adjust old model to new data
4. divide new data into well and poorly modelled points
5. **try a more complex model, if better BIC set as best and go back to 4**
6. set the current best model and go back to 2



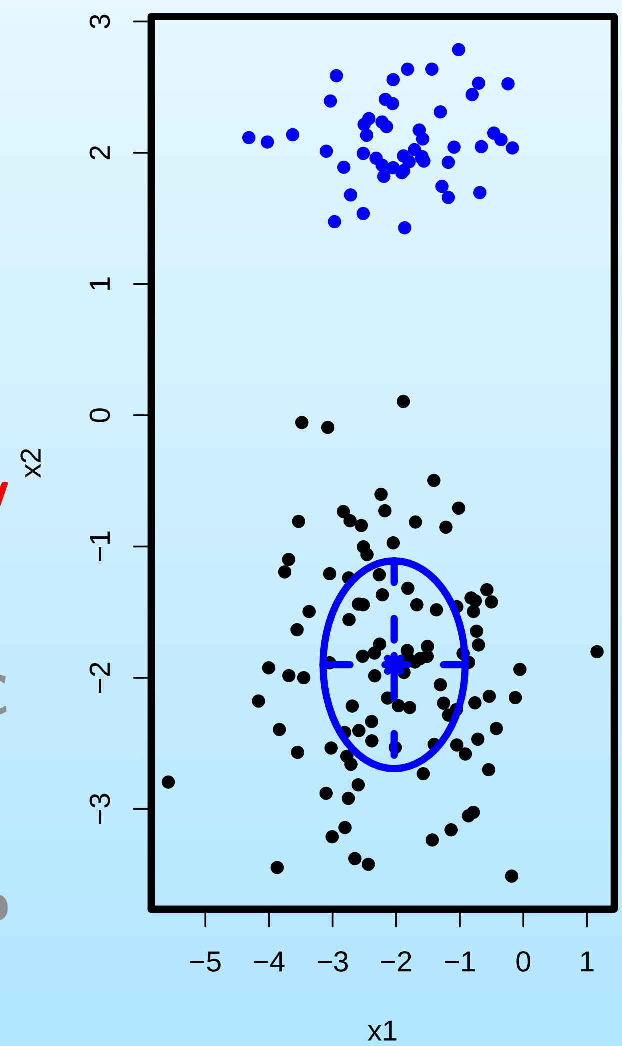
Mille: Algorithm

1. start with a MCLUST model
2. get new data
3. adjust old model to new data
4. divide new data into well and poorly modelled points
5. try a more complex model, if better BIC set as best and go back to 4
6. **set the current best model and go back to 2**



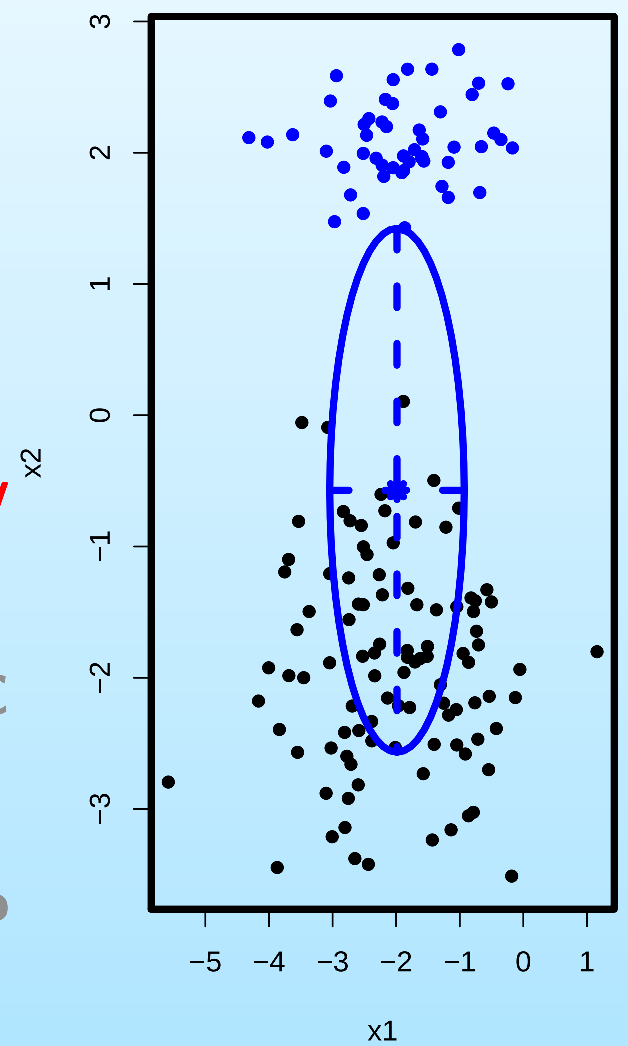
Mille: Algorithm

1. start with a MCLUST model
2. **get new data**
3. adjust old model to new data
4. divide new data into well and poorly modelled points
5. try a more complex model, if better BIC set as best and go back to 4
6. set the current best model and go back to 2



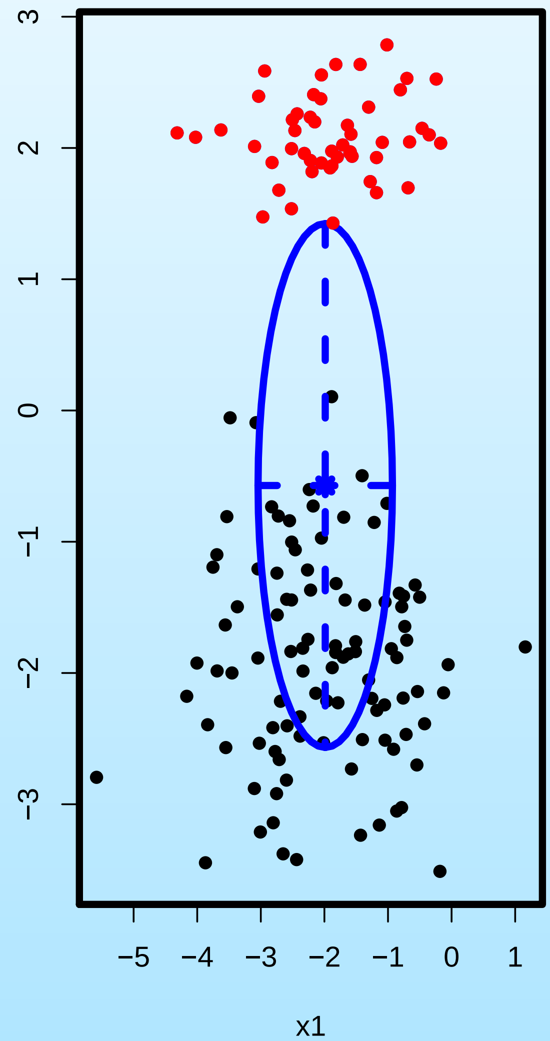
Mille: Algorithm

1. start with a MCLUST model
2. get new data
3. **adjust old model to new data**
4. divide new data into well and poorly modelled points
5. try a more complex model, if better BIC set as best and go back to 4
6. set the current best model and go back to 2



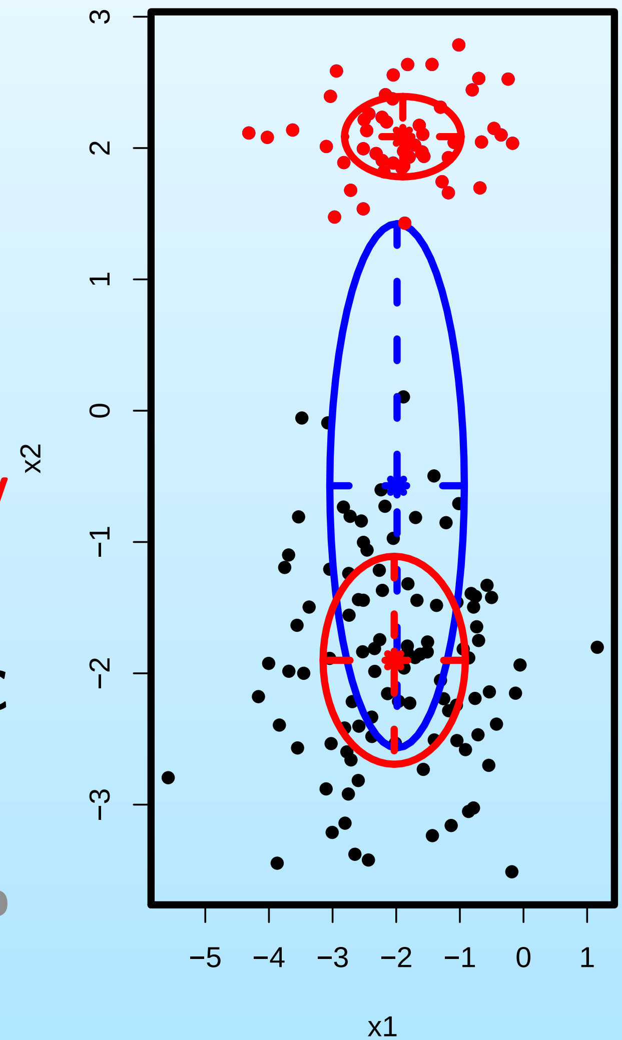
Mille: Algorithm

1. start with a MCLUST model
2. get new data
3. adjust old model to new data
4. **divide new data into well and poorly modelled points**
5. try a more complex model, if better BIC set as best and go back to 4
6. set the current best model and go back to 2



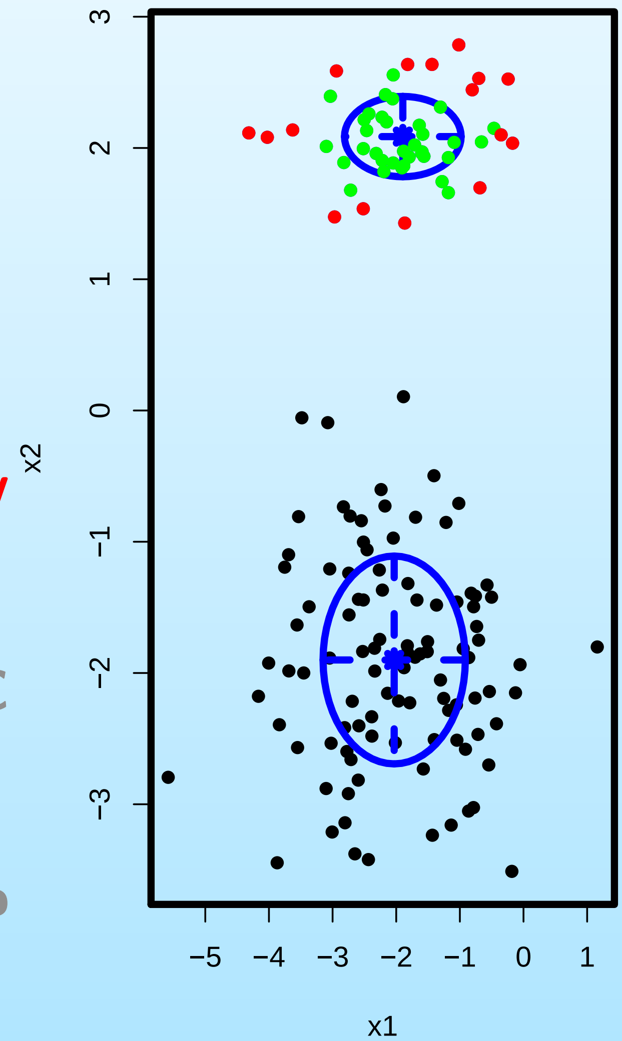
Mille: Algorithm

1. start with a MCLUST model
2. get new data
3. adjust old model to new data
4. divide new data into well and poorly modelled points
5. **try a more complex model, if better BIC set as best and go back to 4**
6. set the current best model and go back to 2



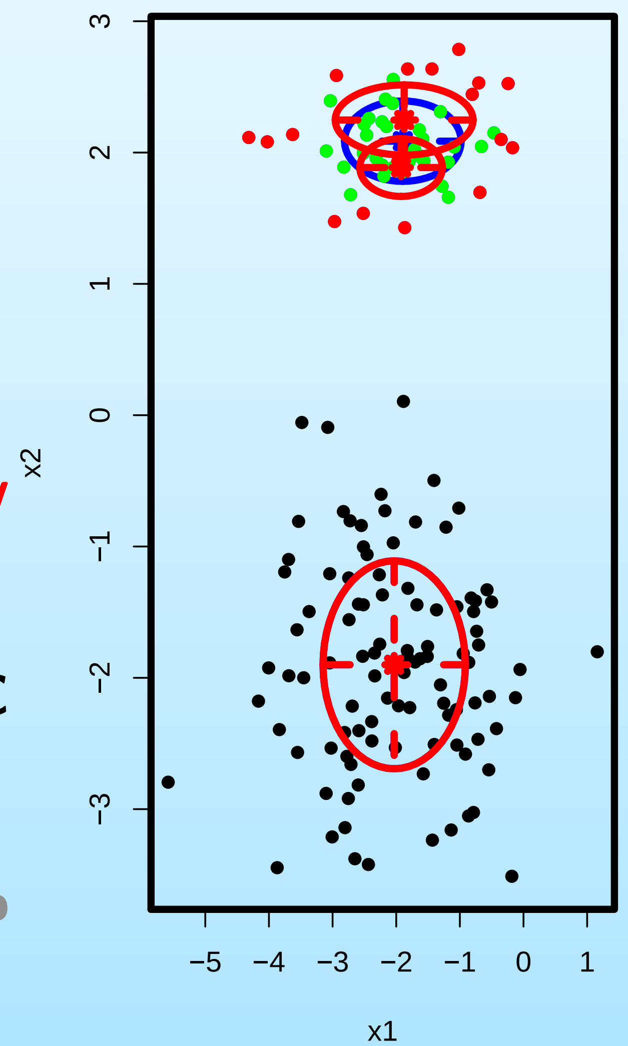
Mille: Algorithm

1. start with a MCLUST model
2. get new data
3. adjust old model to new data
4. **divide new data into well and poorly modelled points**
5. try a more complex model, if better BIC set as best and go back to 4
6. set the current best model and go back to 2



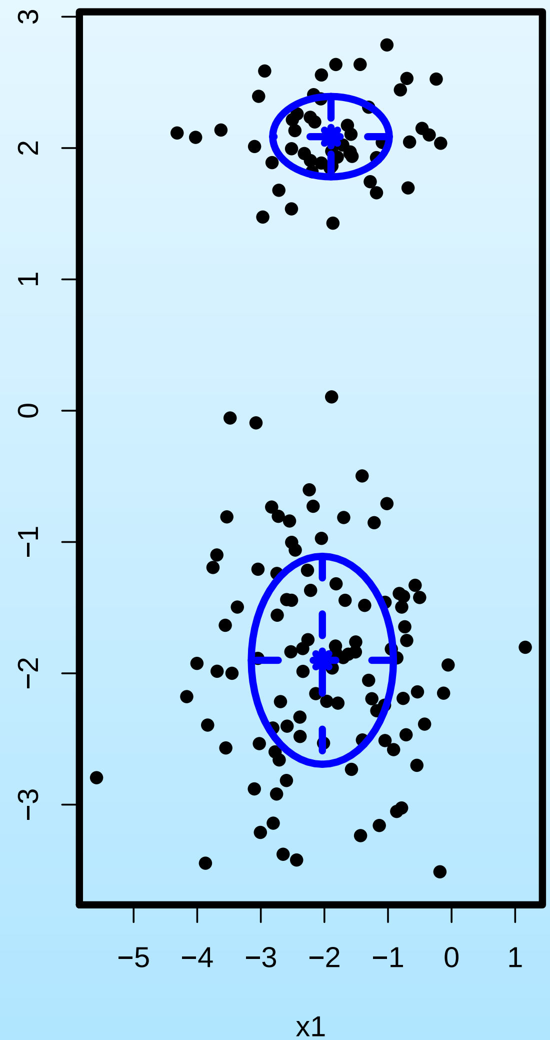
Mille: Algorithm

1. start with a MCLUST model
2. get new data
3. adjust old model to new data
4. divide new data into well and poorly modelled points
5. **try a more complex model, if better BIC set as best and go back to 4**
6. set the current best model and go back to 2



Mille: Algorithm

1. start with a MCLUST model
2. get new data
3. adjust old model to new data
4. divide new data into well and poorly modelled points
5. try a more complex model, if better BIC set as best and go back to 4
6. **set the current best model and go back to 2**



End

Thank you!