Elements of Machine Learning & Data Science

Winter semester 2023/24

Lecture 3 – Bayes Decision Theory

17.10.2023

Prof. Bastian Leibe

Announcement: Small-Group Exercises

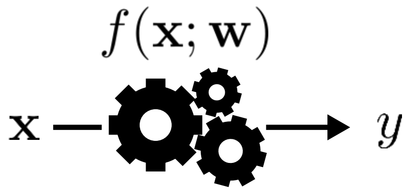
Monday	Tuesday	Wednesday	Thursday	Friday
14:30-18:00h	3x 14:30-16:00h	2x 14:30-16:00h		
3x 16:30-18:00h	3x 16:30-18:00h	2x 16:30-18:00h		
18:30-20:00h	2x 18:30-20:00h			

- **Bi-weekly small-group exercises**
  - We’re currently setting up a poll to collect your preferences for the exercise slots
  - Please enter your choices until Wed, 18.10. evening!
  - Based on the poll results, we will assign you to exercise slots

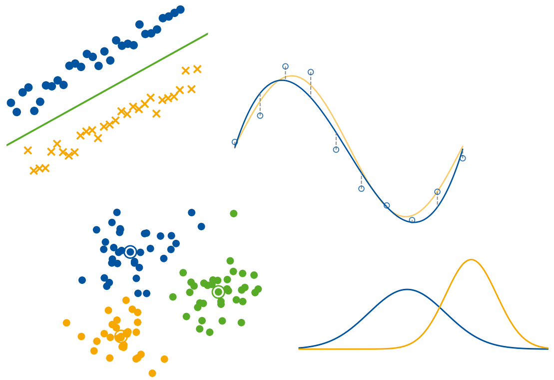
  - *Please sign up for your time slot preferences by Wed evening…*

Machine Learning Topics

1. **Introduction to ML**
2. Probability Density Estimation
3. Linear Discriminants
4. Linear Regression
5. Logistic Regression
6. Support Vector Machines
7. AdaBoost
8. Neural Network Basics



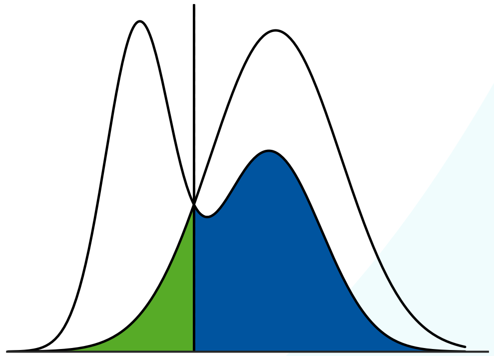
Machine Learning Concepts



Forms of Machine Learning

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

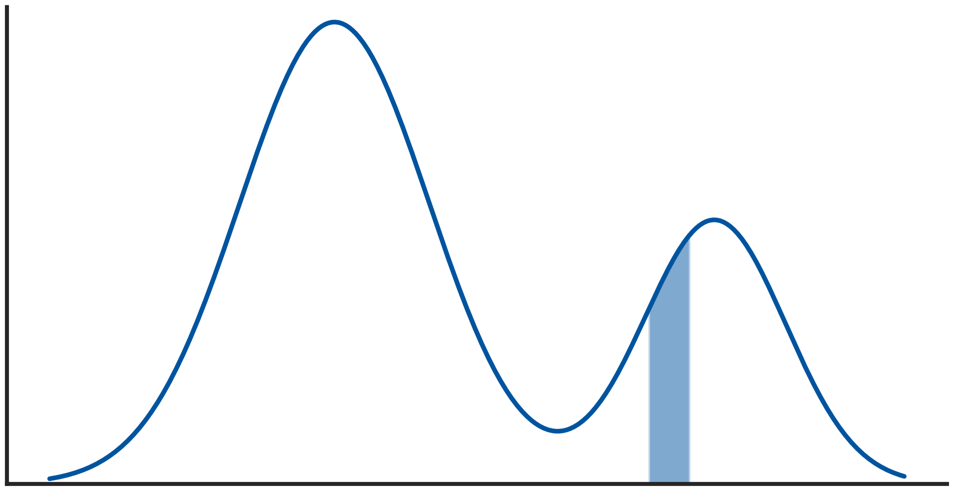
Bayes Decision Theory



Bayes Optimal Classification

Introduction

1. Motivation
2. Forms of learning
3. **Terms, Concepts, and Notation**
4. Bayes Decision Theory



Rules of Probability - Summary

- Sum rule:

$$p(A) = \sum_B p(A, B)$$

- Product rule:

$$p(A, B) = p(B|A)p(A)$$

- Combine into Bayes' Theorem:

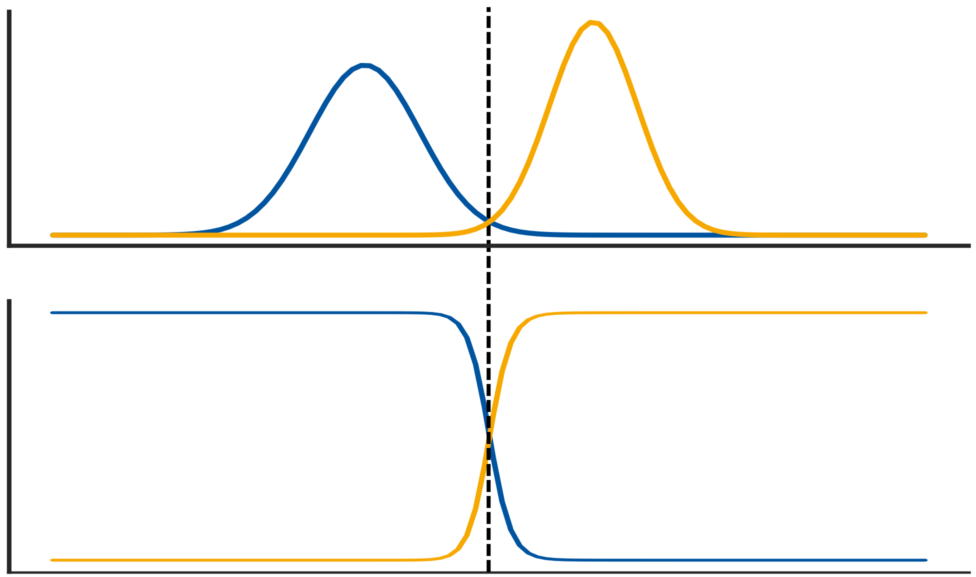
$$p(A|B) = \frac{p(B|A)p(A)}{p(B)}$$

$$= \frac{p(B|A)p(A)}{\sum_A p(B|A)p(A)}$$

This is the most important equation in this course!

Introduction

1. Motivation
2. Forms of learning
3. Terms, Concepts, and Notation
4. **Bayes Decision Theory**



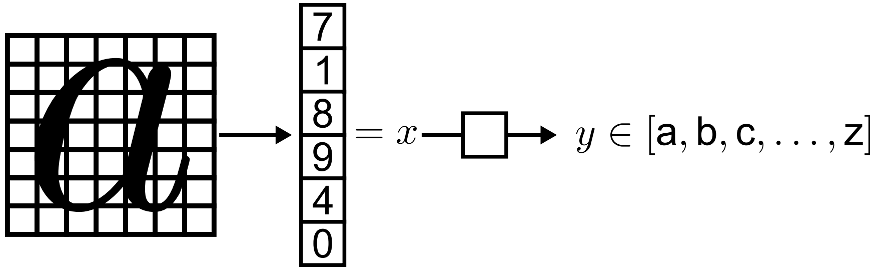
Bayes Decision Theory

- Goal: predict an output class $\mathcal{C}$ from measurements $\mathbf{x}$, by minimizing the probability of misclassification.
- *How can we make such decisions optimally?*
- Bayes Decision Theory gives us the tools for this
  - Based on Bayes' Theorem:

$$p(Y|X) = \frac{p(X|Y)p(Y)}{p(X)}$$

  - In the following, we will introduce its basic concepts…

Example: handwritten character recognition



$\mathbf{x}$: e.g., pixel values

Core Concept: Priors

- What can we tell about the outcome of an experiment *before* making any measurements?
- The a-priori probability $p(\mathcal{C})$ captures the probability distribution over the different class outcomes
  - Based on previously observed data
  - i.e., independent of the actual measurement
- The prior probabilities over all possible class outcomes sum to one.

Example: in English text, the letter "e" makes up ~13% of all letters:

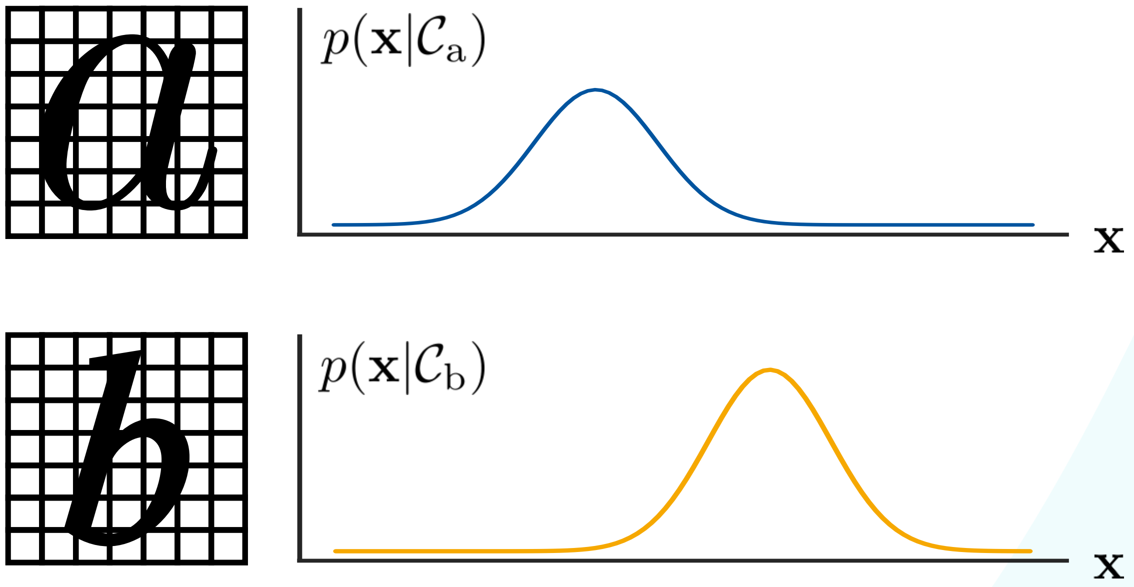
$$p(\mathcal{C}_{\mathrm{e}}) = 0.13$$

And there are 26 letters in the English alphabet:

$$\sum_{\alpha \in \{\mathrm{a},\ldots,\mathrm{z}\}} p(\mathcal{C}_\alpha) = 1$$

Core Concept: Likelihood

- How *likely* is it that we *observe* a certain measurement $\mathbf{x}$ *given* an example of class $\mathcal{C}$?
- This is expressed by the likelihood $p(\mathbf{x}|\mathcal{C})$
  - It is called a *class-conditional distribution*, since it specifies the distribution of $\mathbf{x}$ conditioned on the class $\mathcal{C}$.
  - We can estimate the likelihood from the distribution of measurements $\mathbf{x}$ observed on the given training data.



- Here, $\mathbf{x}$ measures certain properties of the input data.
  - E.g., the fraction of black pixels
  - We simply treat it as a vector $\mathbf{x} \in \mathbb{R}^D$.

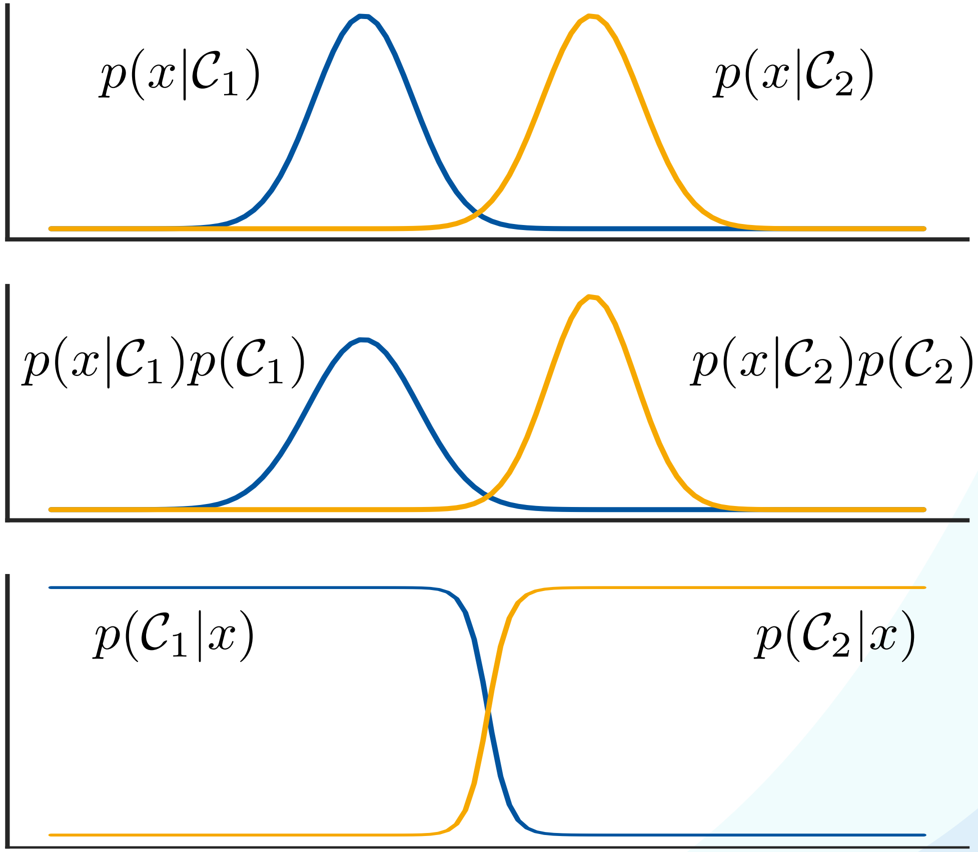
Core Concept: Posterior

- What is the probability for class $\mathcal{C}_k$ if we made a measurement $\mathbf{x}$?
- This a-posteriori probability $p(\mathcal{C}_k|\mathbf{x})$ can be computed via Bayes' Theorem after we observed $\mathbf{x}$:

$$p(\mathcal{C}_k|\mathbf{x}) = \frac{p(\mathbf{x}|\mathcal{C}_k)p(\mathcal{C}_k)}{p(\mathbf{x})} = \frac{p(\mathbf{x}|\mathcal{C}_k)p(\mathcal{C}_k)}{\sum_j p(\mathbf{x}|\mathcal{C}_j)p(\mathcal{C}_j)}$$

- *This is usually what we're interested in!*
- Interpretation

$$posterior = \frac{likelihood \cdot prior}{normalization\ factor}$$



Making Optimal Decisions

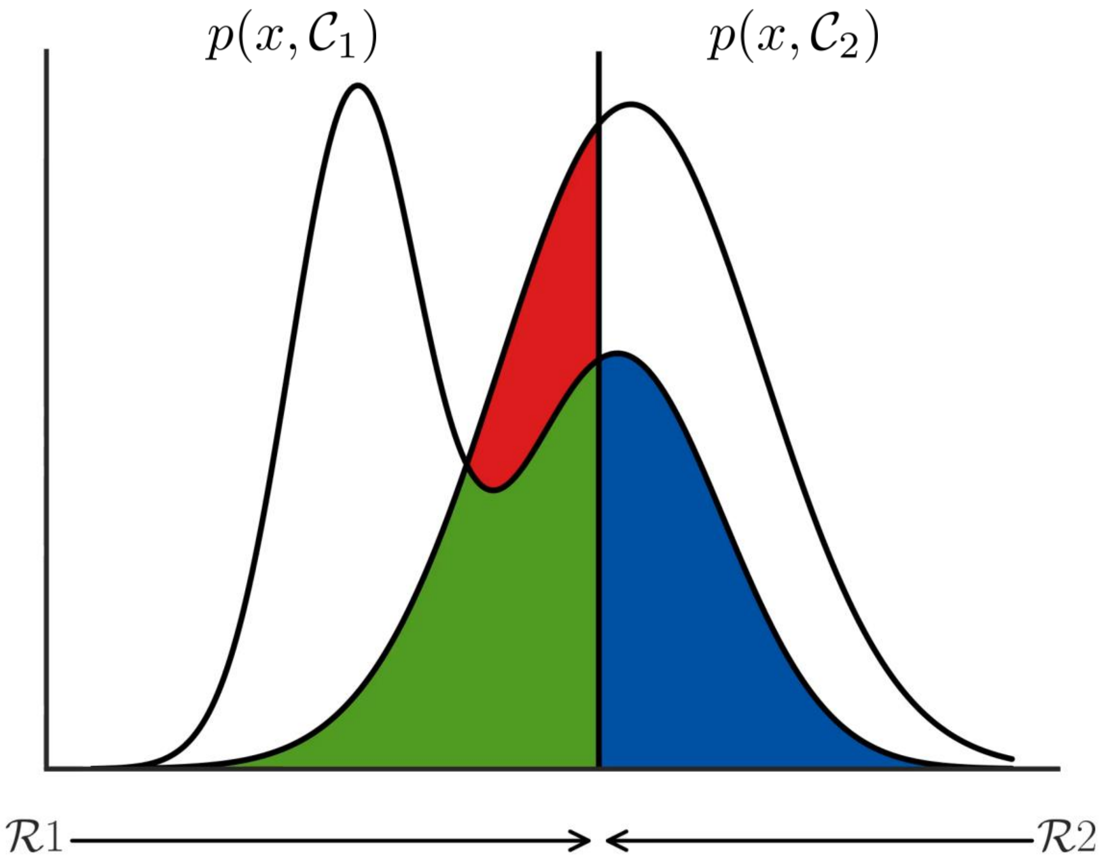
- Goal: minimize the probability of misclassification.

$$p(\text{mistake}) = p(x \in \mathcal{R}_1, \mathcal{C}_2) + p(x \in \mathcal{R}_2, \mathcal{C}_1)$$
$$= \int_{\mathcal{R}_1} p(x, \mathcal{C}_2)\,\mathrm{d}x + \int_{\mathcal{R}_2} p(x, \mathcal{C}_1)\,\mathrm{d}x$$
$$= \int_{\mathcal{R}_1} p(\mathcal{C}_2|x)p(x)\,\mathrm{d}x + \int_{\mathcal{R}_2} p(\mathcal{C}_1|x)p(x)\,\mathrm{d}x$$

- Note:

+ = constant

We can only reduce



$\mathcal{R}_1$ and $\mathcal{R}_2$ are the decision regions after setting a decision threshold.

Making Optimal Decisions

- Goal: minimize the probability of misclassification.

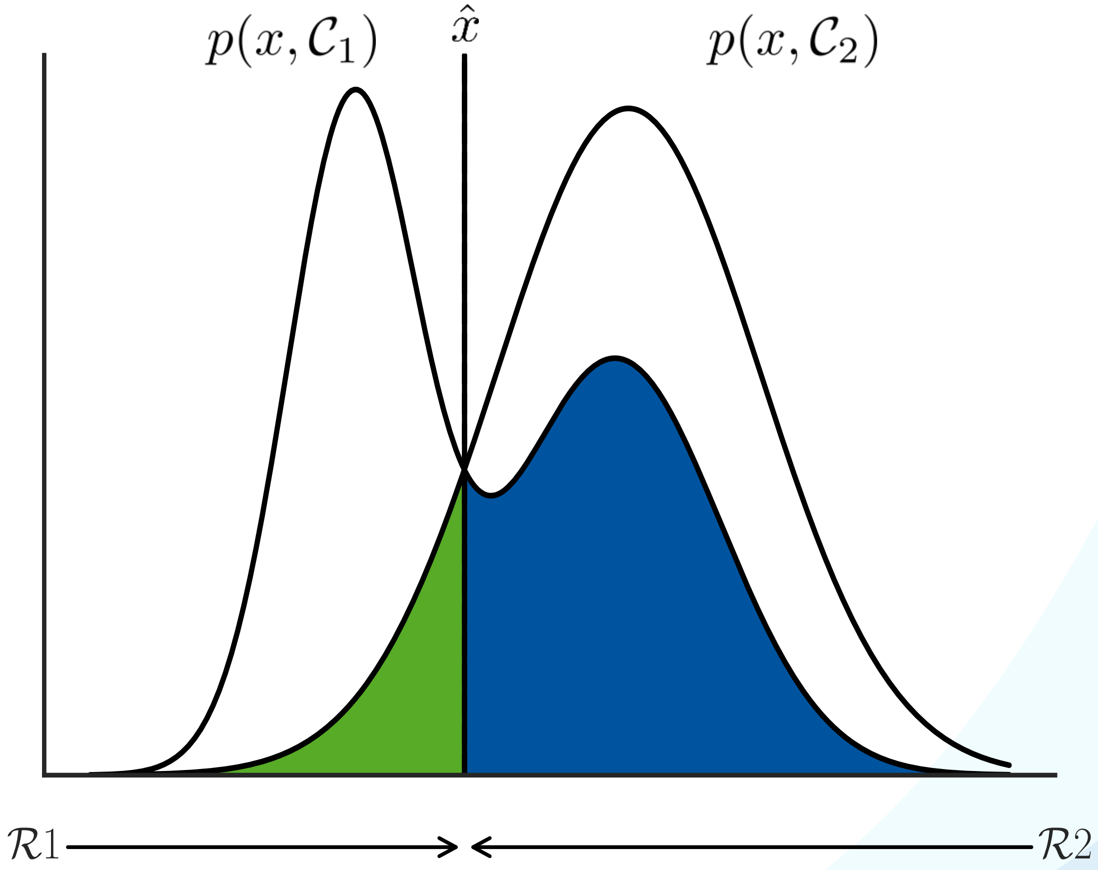
$$p(\text{mistake}) = p(x \in \mathcal{R}_1, \mathcal{C}_2) + p(x \in \mathcal{R}_2, \mathcal{C}_1)$$
$$= \int_{\mathcal{R}_1} p(x, \mathcal{C}_2)\,\mathrm{d}x + \int_{\mathcal{R}_2} p(x, \mathcal{C}_1)\,\mathrm{d}x$$
$$= \int_{\mathcal{R}_1} p(\mathcal{C}_2|x)p(x)\,\mathrm{d}x + \int_{\mathcal{R}_2} p(\mathcal{C}_1|x)p(x)\,\mathrm{d}x$$

- Note:

+ = constant

We can only reduce

- *Minimal error at the intersection* $\hat{x}$



$\mathcal{R}_1$ and $\mathcal{R}_2$ are the decision regions after setting a decision threshold.

Making Optimal Decisions

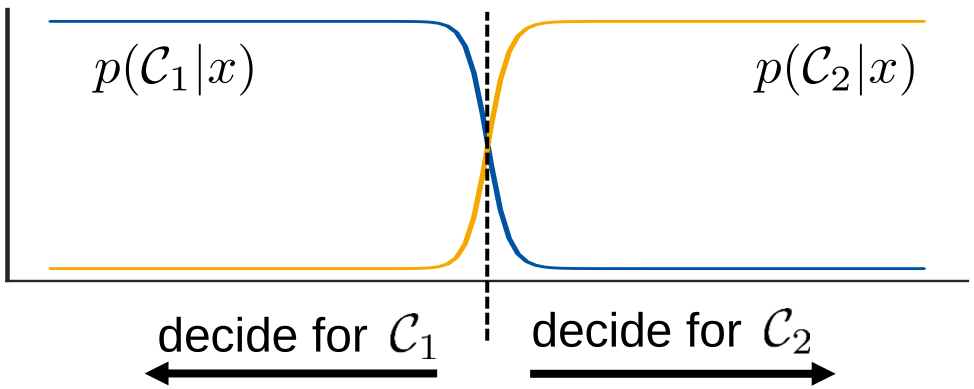
- Our goal is to minimize the probability of a misclassification.
- The optimal decision rule is: decide for $\mathcal{C}_1$ iff

$$p(\mathcal{C}_1|\mathbf{x}) > p(\mathcal{C}_2|\mathbf{x})$$

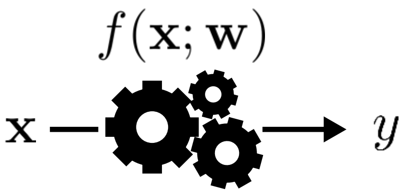
- Or for multiple classes: decide for $\mathcal{C}_k$ iff

$$p(\mathcal{C}_k|\mathbf{x}) > p(\mathcal{C}_j|\mathbf{x}) \ \forall j \neq k$$

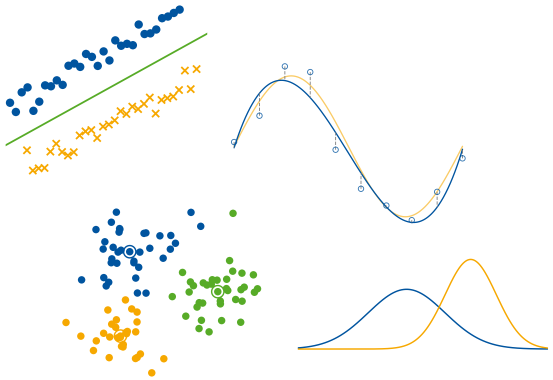
- *Once we can estimate posterior probabilities, we can use this rule to build classifiers.*



Summary: Introduction to ML



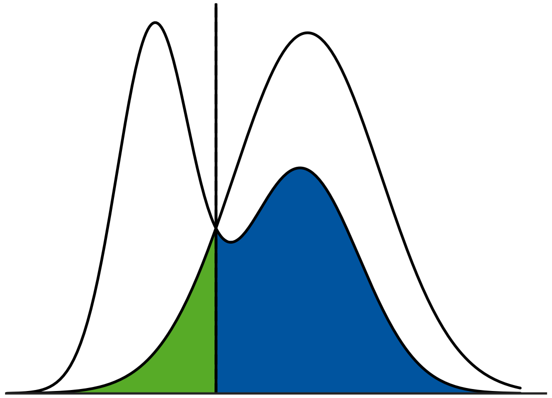
Machine Learning



Forms of Machine Learning

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

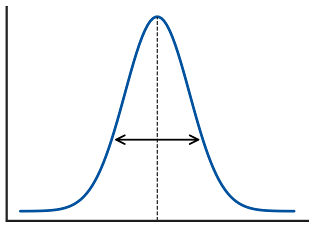
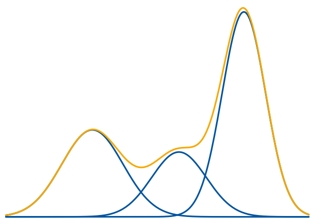
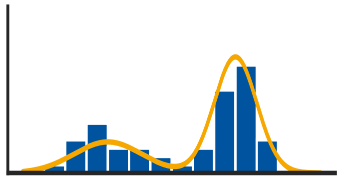
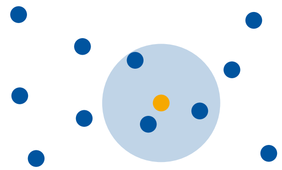
Bayes Theorem



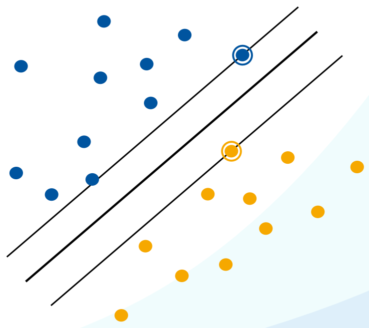
Bayes Optimal Classification

Next Lectures…

- Ways how to estimate the probability densities $p(\mathbf{x}|\mathcal{C}_k)$
  - Parametric methods
    - Gaussian distribution
    - Mixtures of Gaussians
  - Non-parametric methods
    - Histograms
    - k-Nearest Neighbor
    - Kernel Density Estimation

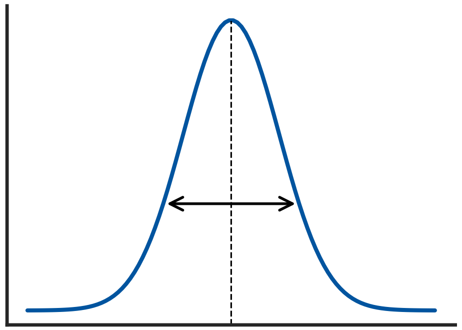





- Ways to directly model the posteriors $p(\mathcal{C}_k|\mathbf{x})$
  - Linear discriminants
  - Logistic regression, SVMs, Neural Networks, …

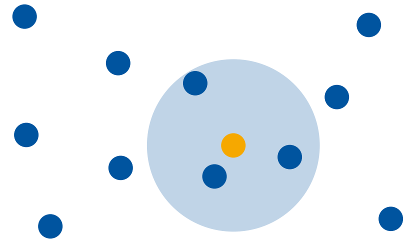


Machine Learning Topics

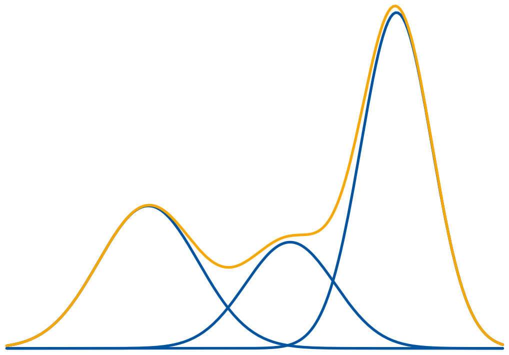
1. Introduction to ML
2. **Probability Density Estimation**
3. Linear Discriminants
4. Linear Regression
5. Logistic Regression
6. Support Vector Machines
7. AdaBoost
8. Neural Network Basics



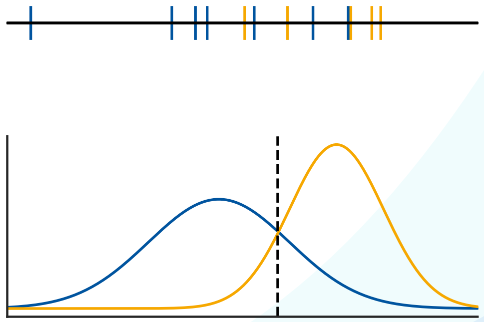
Parametric Methods & ML-Algorithm



Nonparametric Methods



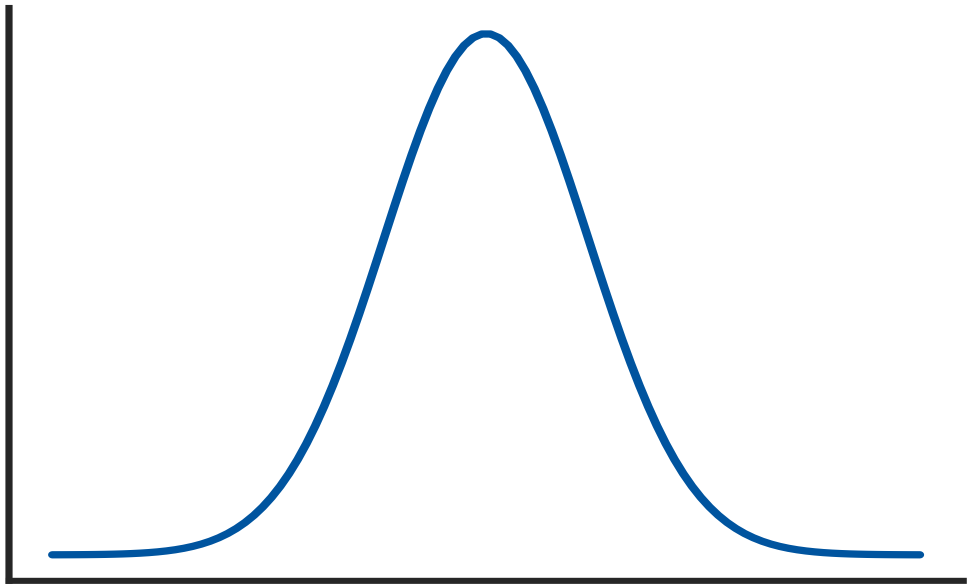
Mixtures of Gaussians & EM-Algorithm



Bayes Classifiers

Probability Density Estimation

1. **Probability Distributions**
2. Parametric Methods
3. Nonparametric Methods
4. Mixture Models
5. Bayes Classifier
6. K-NN Classifier



Probability Distributions

- Up to now: Bayes optimal classification based on $p(\mathbf{x}|\mathcal{C}_k)$ and $p(\mathcal{C}_k)$.

- *How can we estimate (= learn) those probability densities?*
  - Supervised training case: data and class labels are known.
  - Estimate the probability density for each class $\mathcal{C}_k$ separately.

$$p(\mathbf{x}|\mathcal{C}_k) \quad \text{given} \quad \mathbf{x} \in \mathcal{C}_k$$

  - (For simplicity of notation, we will drop the class label $\mathcal{C}_k$ in the following $\Rightarrow p(\mathbf{x})$).

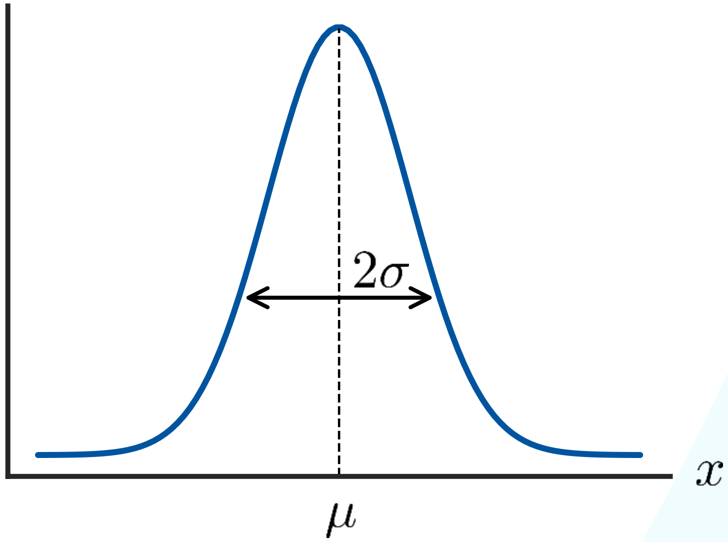
- *First, we look at the Gaussian distribution in more detail…*

The Gaussian (or Normal) Distribution

- One-dimensional (univariate) case:

$$\mathcal{N}(x|\mu,\sigma^2) = \frac{1}{\sqrt{2\pi}\sigma}\exp\left(-\frac{(x-\mu)^2}{2\sigma^2}\right)$$

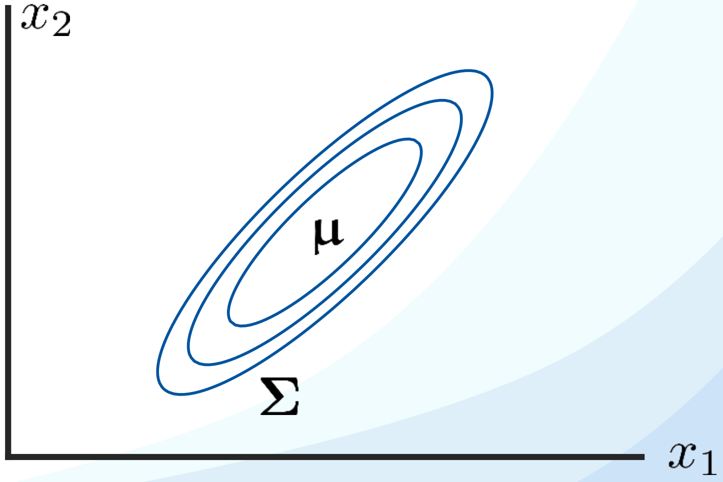
Mean Variance



- Multi-dimensional (multivariate) case:

$$\mathcal{N}(\mathbf{x}|\boldsymbol{\mu},\boldsymbol{\Sigma}) = \frac{1}{(2\pi)^{D/2}|\boldsymbol{\Sigma}|^{1/2}}\exp\left(-\frac{1}{2}(\mathbf{x}-\boldsymbol{\mu})^\mathsf{T}\boldsymbol{\Sigma}^{-1}(\mathbf{x}-\boldsymbol{\mu})\right)$$

Mean vector Covariance matrix

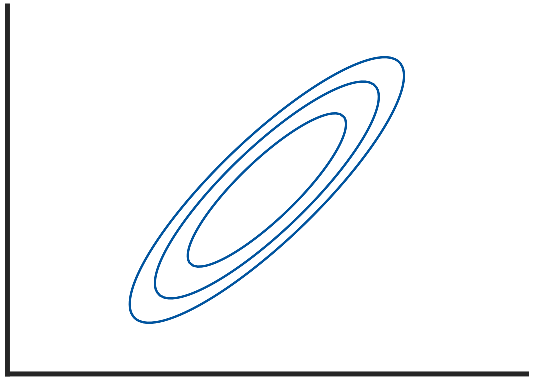


Gaussian Distribution: Shape

Full covariance matrix:

$$\boldsymbol{\Sigma} = [\sigma_{ij}]$$

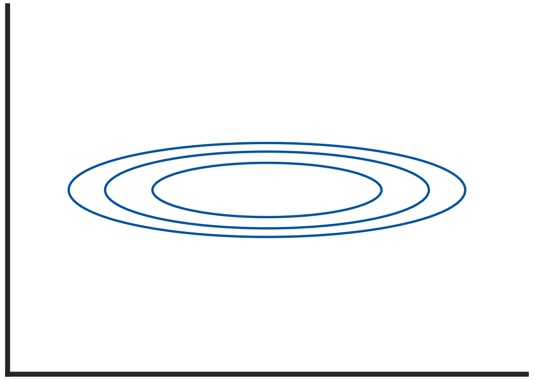
General ellipsoid shape



Diagonal covariance matrix:

$$\boldsymbol{\Sigma} = \mathrm{diag}\{\sigma_i\}$$

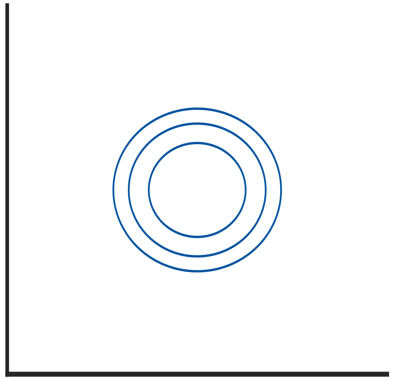
Axis-aligned ellipsoid



Uniform variance:

$$\boldsymbol{\Sigma} = \sigma^2\mathbf{I}$$

Hypersphere

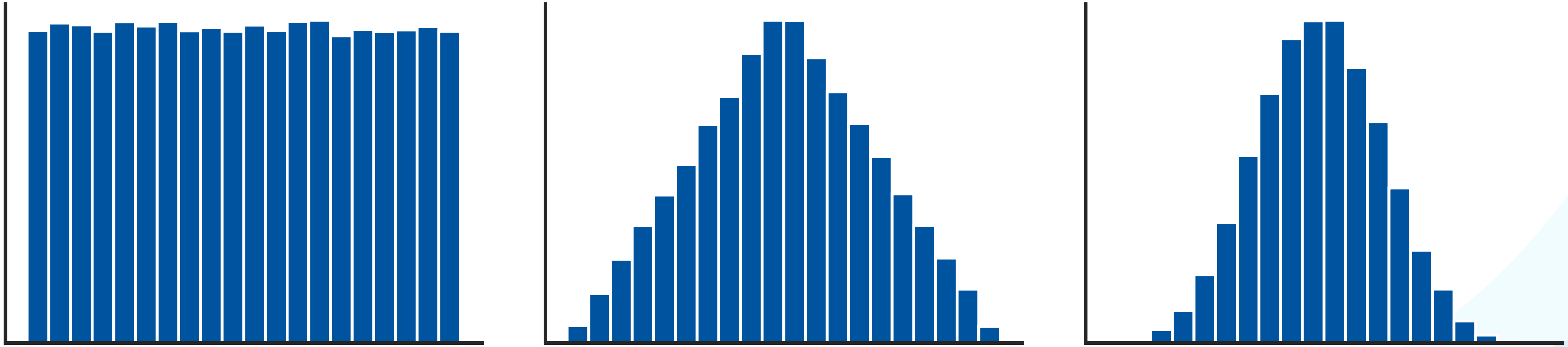


Gaussian Distribution: Motivation

- Central Limit Theorem
  - The distribution of a sum of N *i.i.d.* random variables becomes increasingly Gaussian as N grows.
  - In practice, the convergence to a Gaussian can be very rapid.
  - This makes the Gaussian interesting for many applications.

i.i.d. = independent and identically distributed

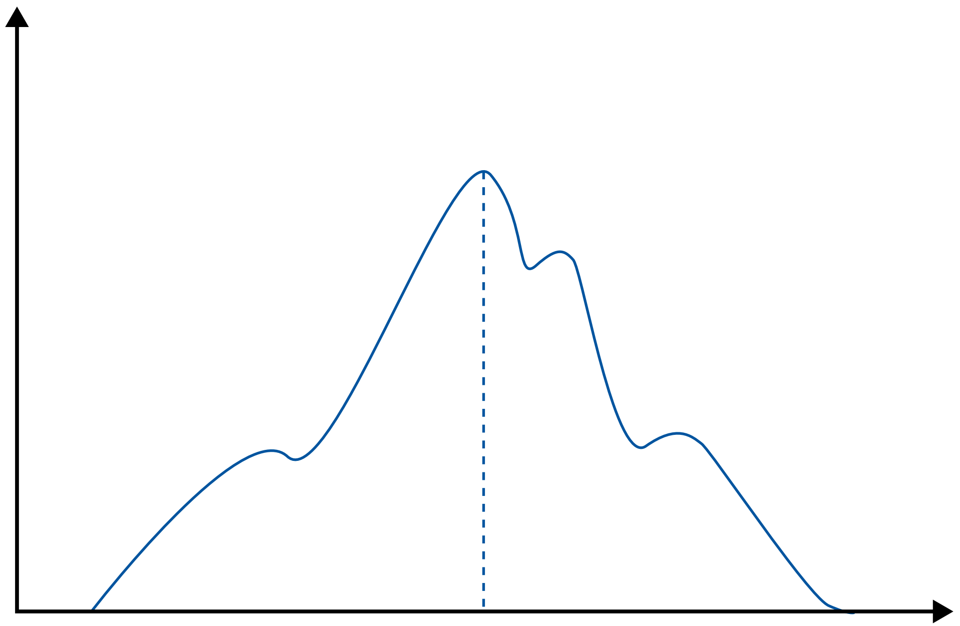
- Example: Sum over N uniform [0,1] random variables.



$N = 1$ $\quad$ $N = 2$ $\quad$ $N = 5$

Probability Density Estimation

1. Probability Distributions
2. **Parametric Methods**
3. Nonparametric Methods
4. Mixture Models
5. Bayes Classifier
6. K-NN Classifier

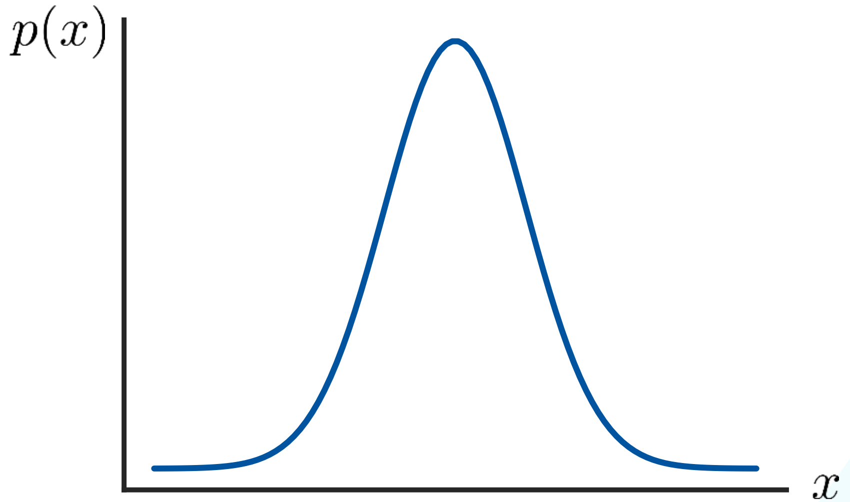


Parametric Methods

- In parametric methods, we assume that we know the parametric form of the underlying data distribution.
  - I.e., the equation of the pdf with parameters θ.

Example: $p(x) = \mathcal{N}(x|\mu, \sigma)$

$$\theta = (\mu, \sigma)$$



Parametric Methods

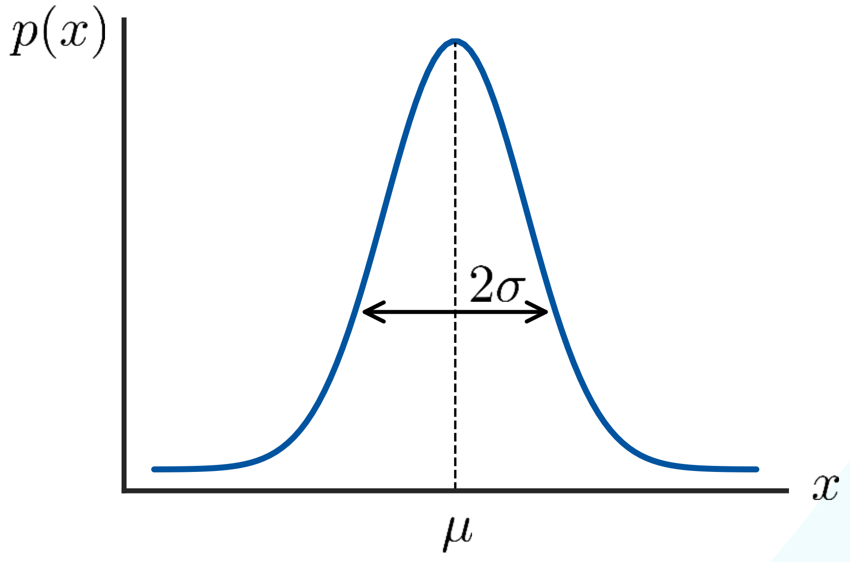
- In parametric methods, we assume that we know the parametric form of the underlying data distribution.
  - I.e., the equation of the pdf with parameters θ.
- Goal: Estimate θ from training data $\mathcal{X} = \{\mathbf{x}_1, \ldots, \mathbf{x}_N\}$.
- Likelihood of θ:

$$L(\theta) = p(\mathcal{X}|\theta)$$

Probability that the data $\mathcal{X}$ was indeed generated by a distribution with parameters θ.

Example: $p(x) = \mathcal{N}(x|\mu, \sigma)$

$\theta = (\mu, \sigma)$



Maximum Likelihood Approach

- Idea: Find optimal parameters by maximizing $L(\theta)$.
- Computation of the likelihood:
  - Single data point (e.g., for Gaussian):

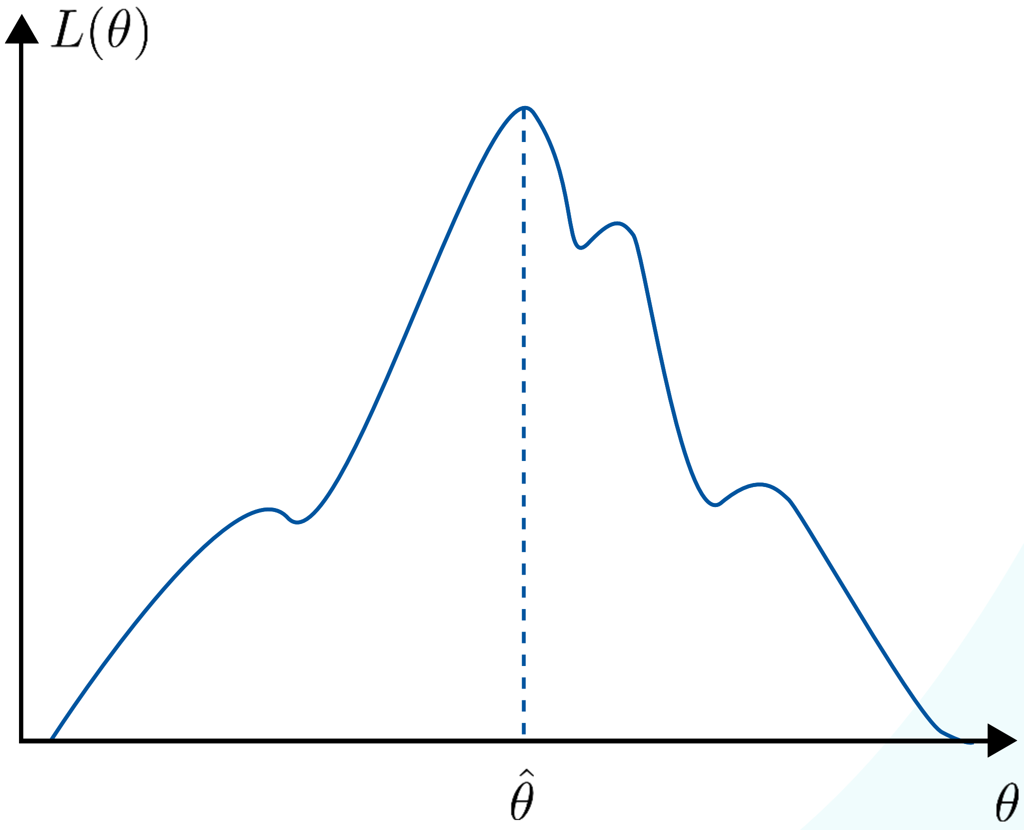
$$p(x_n|\theta) = \frac{1}{\sqrt{2\pi}\sigma} \exp\left(-\frac{(x_n - \mu)^2}{2\sigma^2}\right)$$

  - Assumption: all data points are independent

$$L(\theta) = p(\mathcal{X}|\theta) = \prod_{n=1}^{N} p(x_n|\theta)$$

  - Negative Log-Likelihood ("Energy"):

$$E(\theta) = -\ln L(\theta) = -\sum_{n=1}^{N} \ln p(x_n|\theta)$$



Maximizing the likelihood $\Leftrightarrow$ minimizing the negative log-likelihood.

- Minimizing the negative log-likelihood:
  - Take the derivative and set it to zero.

$$\frac{\partial}{\partial \theta} E(\theta) = -\frac{\partial}{\partial \theta} \sum_{n=1}^{N} \ln p(x_n|\theta) = -\sum_{n=1}^{N} \frac{\frac{\partial}{\partial \theta} p(x_n|\theta)}{p(x_n|\theta)} \overset{!}{=} 0$$

  - Log-likelihood for Normal distribution (1D case):

$$\boxed{\begin{aligned} p(x_n|\mu,\sigma) &= \frac{1}{\sqrt{2\pi}\sigma} \exp\left(-\frac{(x_n-\mu)^2}{2\sigma^2}\right) \\ \frac{\partial}{\partial \mu} p(x_n|\mu,\sigma) &= -\frac{2(x_n-\mu)}{2\sigma^2} p(x_n|\mu,\sigma) \end{aligned}}$$

$$\begin{aligned} \frac{\partial}{\partial \mu} E(\mu,\sigma) &= -\sum_{n=1}^{N} \frac{\frac{\partial}{\partial \mu} p(x_n|\mu,\sigma)}{p(x_n|\mu,\sigma)} \\ &= -\sum_{n=1}^{N} -\frac{\cancel{2}(x_n-\mu)}{\cancel{2}\sigma^2} \frac{\cancel{p(x_n|\mu,\sigma)}}{\cancel{p(x_n|\mu,\sigma)}} \\ &= \frac{1}{\sigma^2} \sum_{n=1}^{N} (x_n-\mu) = \cancel{\frac{1}{\sigma^2}} \left(\sum_{n=1}^{N} x_n - N\mu \right) \overset{!}{=} 0 \quad \Longleftrightarrow \quad \hat{\mu} = \frac{1}{N} \sum_{n=1}^{N} x_n \end{aligned}$$

- By minimizing the negative log-likelihood, we found:

$$\hat{\mu} = \frac{1}{N}\sum_{n=1}^{N} x_n$$

sample mean

Similarly, we can derive:

$$\hat{\sigma}^2 = \frac{1}{N}\sum_{n=1}^{N}(x_n - \hat{\mu})^2$$

sample variance

- $\hat{\theta} = (\hat{\mu}, \hat{\sigma})$ is the Maximum Likelihood estimate for the parameters of a Gaussian distribution.
  - This is a very important result.
  - Unfortunately, it is wrong…

- To be precise, the result is not wrong, but biased.
- Assume the samples $x_1, x_2, \ldots, x_N$ come from a true Gaussian distribution with mean μ and variance σ^2
  - It can be shown that the expected estimates are then

$$\mathbb{E}[\mu_{\mathrm{ML}}] = \mu$$

$$\mathbb{E}[\sigma^2_{\mathrm{ML}}] = \left(\frac{N-1}{N}\right)\sigma^2$$

$\Rightarrow$ *The ML estimate will underestimate the true variance!*

- We can correct for this bias:

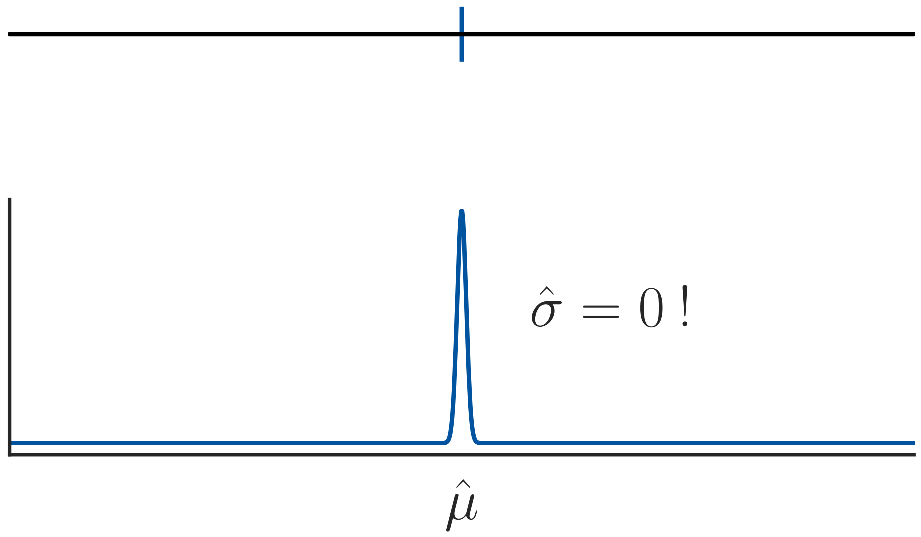
$$\hat{\sigma}^2 = \frac{N}{N-1}\sigma^2_{\mathrm{ML}} = \frac{1}{N-1}\sum_{n=1}^{N}(x_n - \hat{\mu})^2$$

- Maximum Likelihood has several significant limitations.
  - It systematically underestimates the variance of the distribution!
  - E.g., consider the estimate for a single sample:

$$N = 1,\ \mathcal{X} = \{x_1\}$$

$$\hat{\mu} = \frac{1}{N}\sum_{n=1}^{N} x_n = x_1$$

$$\hat{\sigma}^2 = \frac{1}{N}\sum_{n=1}^{N} (x_n - \hat{\mu})^2 = 0$$

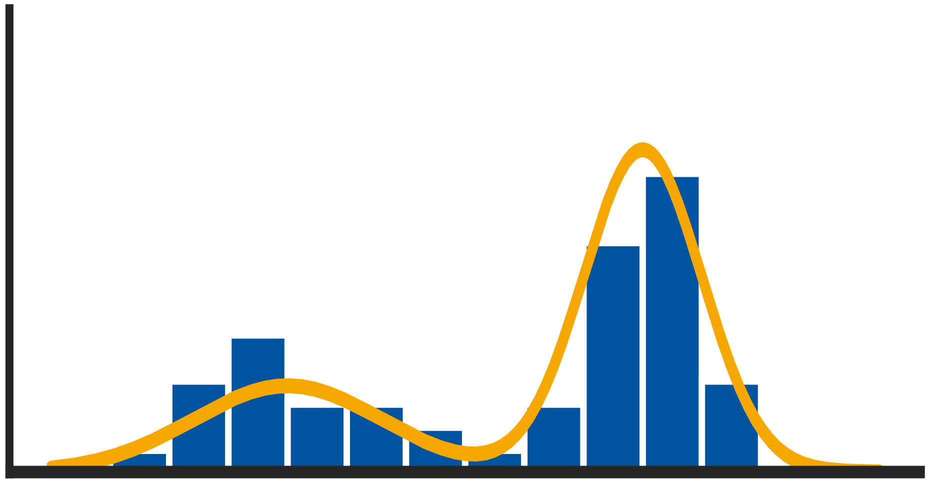


- We say ML overfits to the observed data.

- *We will still often use Maximum Likelihood, but it is important to know about this effect.*

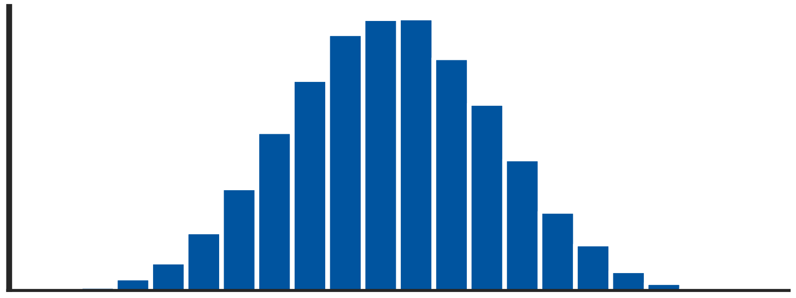
Probability Density Estimation

1. Probability Distributions
2. Parametric Methods
3. **Nonparametric Methods**
    a) **Histograms**
    b) Kernel Methods & k-Nearest Neighbors
4. Mixture Models
5. Bayes Classifier
6. K-NN Classifier

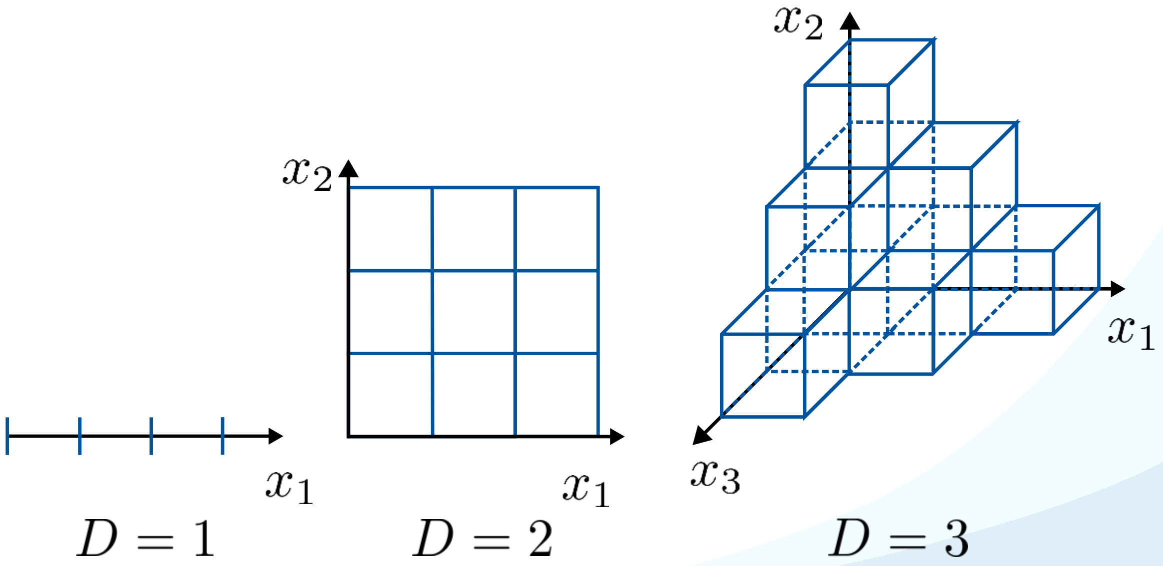


Histograms

- Partition the data space into N distinct bins with widths Δ_i and count the number of observations n_i in each bin.
- Then, $p_i = \dfrac{n_i}{N\Delta_i}$.
- Often the same width is used for all bins.
- This can be done, in principle, for any dimensionality D.

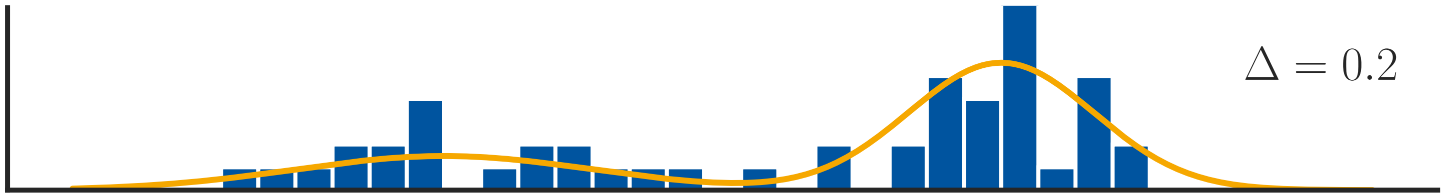


…but the required number of bins grows exponentially with D!



The bin width Δ acts as a smoothing factor.

Not smooth enough



About ok

Too smooth

Advantages

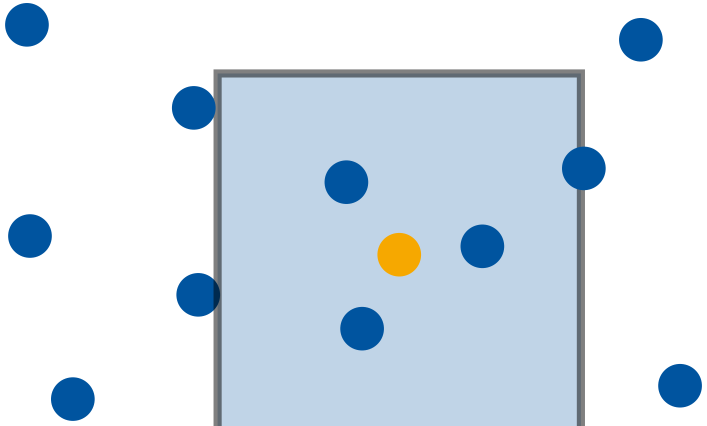
- Very general method. In the limit $(N \to \infty)$, every probability density can be represented.
- No need to store the data points once histogram is computed.

Limitations

- Rather brute-force.
- Discontinuities at bin edges.
- Choosing right bin size is hard.
- Unsuitable for high-dimensional feature spaces.

Probability Density Estimation

1. Probability Distributions
2. Parametric Methods
3. **Nonparametric Methods**
   a) Histograms
   b) **Kernel Methods & k-Nearest Neighbors**
4. Mixture Models
5. Bayes Classifier
6. K-NN Classifier



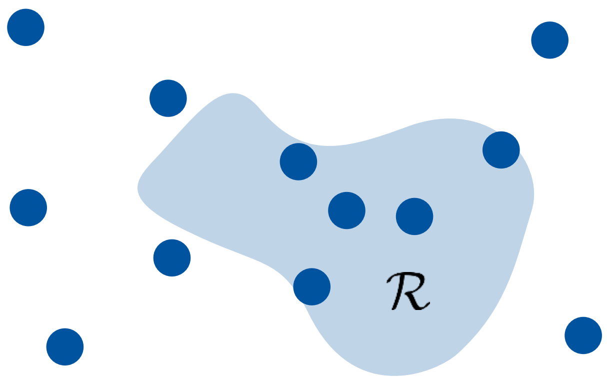
Kernel Methods and k-Nearest Neighbors

- Data point $\mathbf{x}$ comes from pdf $p(\mathbf{x})$.
  - Probability that $\mathbf{x}$ falls into small region $\mathcal{R}$:

$$P = \int_{\mathcal{R}} p(y)dy \approx p(\mathbf{x})V$$

- Estimate $p(\mathbf{x})$ from samples
  - Let K be the number of samples that fall into $\mathcal{R}$.
  - If the number of samples N is sufficiently large, we can estimate P as:

$$P = \frac{K}{N} \quad \Rightarrow \quad p(\mathbf{x}) \approx \frac{K}{NV}$$



$K = 5$
$N = 12$

For sufficiently small $\mathcal{R}$, $p(\mathbf{x})$ is roughly constant.

V : volume of $\mathcal{R}$.

$$p(\mathbf{x}) \approx \frac{K}{NV}$$

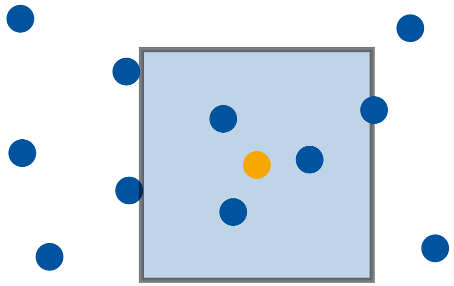
fixed V
determine K

fixed K
determine V

Kernel Methods

k-Nearest Neighbors

Example: Determine the number K of data points inside a fixed hypercube



Kernel Methods

- Hypercube of dimension D with edge length h:

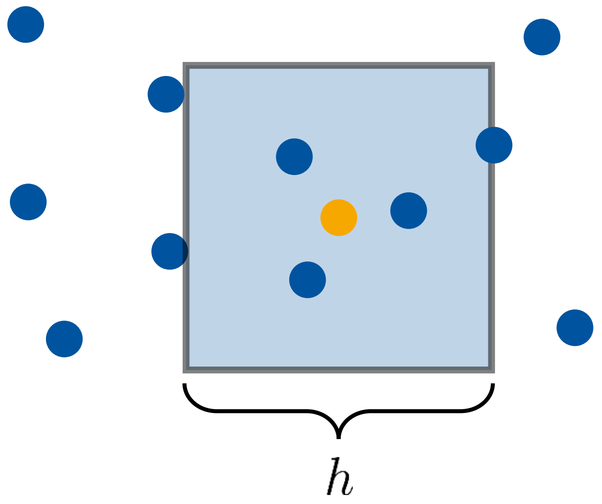
$$k(\mathbf{u}) = \begin{cases} 1, & \text{if } |u_i| \leq \frac{1}{2}h, \;\; i = 1, \ldots, D \\ 0, & \text{otherwise} \end{cases}$$

$$K = \sum_{n=1}^{N} k(\mathbf{x} - \mathbf{x}_n) \qquad V = \int k(\mathbf{u}) d\mathbf{u} = h^D$$

- Probability density estimate:

$$p(\mathbf{x}) \approx \frac{K}{NV} = \frac{1}{Nh^D} \sum_{n=1}^{N} k(\mathbf{x} - \mathbf{x}_n)$$

- This method is known as Parzen Window estimation.



- In general, we can use any kernel such that

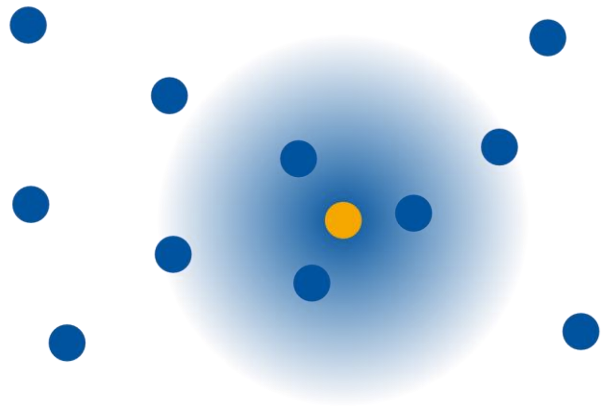
$$k(\mathbf{u}) \geq 0, \quad \int k(\mathbf{u}) d\mathbf{u} = 1$$

$$K = \sum_{n=1}^{N} k(\mathbf{x} - \mathbf{x}_n)$$

- Then, we get the probability density estimate

$$p(\mathbf{x}) \approx \frac{K}{NV} = \frac{1}{N} \sum_{n=1}^{N} k(\mathbf{x} - \mathbf{x}_n)$$

- This is known as Kernel Density Estimation.



E.g., a Gaussian kernel for smoother boundaries.

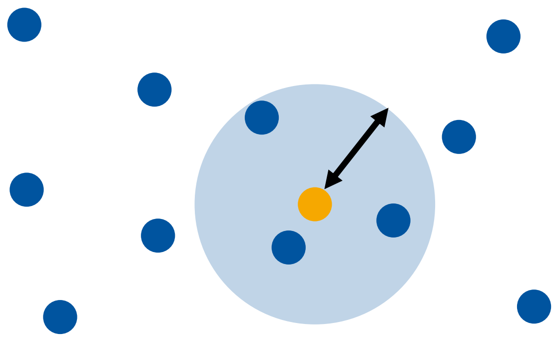
$$p(\mathbf{x}) \approx \frac{K}{NV}$$

fixed V
determine K

Kernel Methods

fixed K
determine V

k-Nearest Neighbors



Increase the volume V until the K next data points are found.

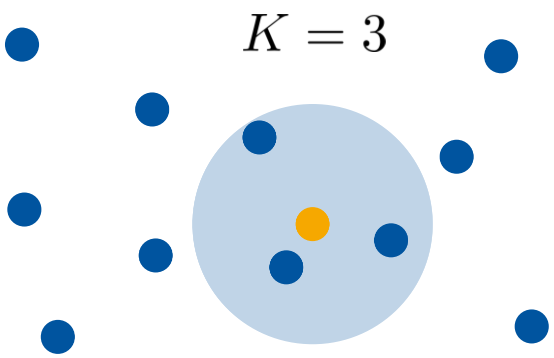
k-Nearest Neighbors

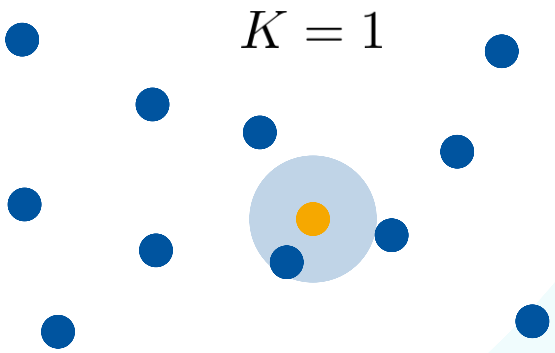
- Fix K, estimate V from the data.
- Consider a hypersphere centered on $\mathbf{x}$ and let it grow to a volume V^* that includes K of the given N data points.
- Then

$$p(\mathbf{x}) \approx \frac{K}{NV^*}$$

- Side note:
  - Strictly speaking, the model produced by k-NN is not a true density model, because the integral over all space diverges.
  - E.g. consider $K = 1$ and a sample exactly on a data point.

$$V^* = 0 \quad \Rightarrow \quad p(\mathbf{x}) \approx \frac{K}{N \cdot 0}$$





Advantages

- Very general. In the limit $(N \to \infty)$, every probability density can be represented.
- No computation during training phase
  - Just need to store training set

Limitations

- Requires storing and computing with the entire dataset.
  - Computational costs linear in the number of data points.
  - Can be improved through efficient storage structures (at the cost of some computation during training).
- Choosing the kernel size/K is a hyperparameter optimization problem.

Bias-Variance Tradeoff

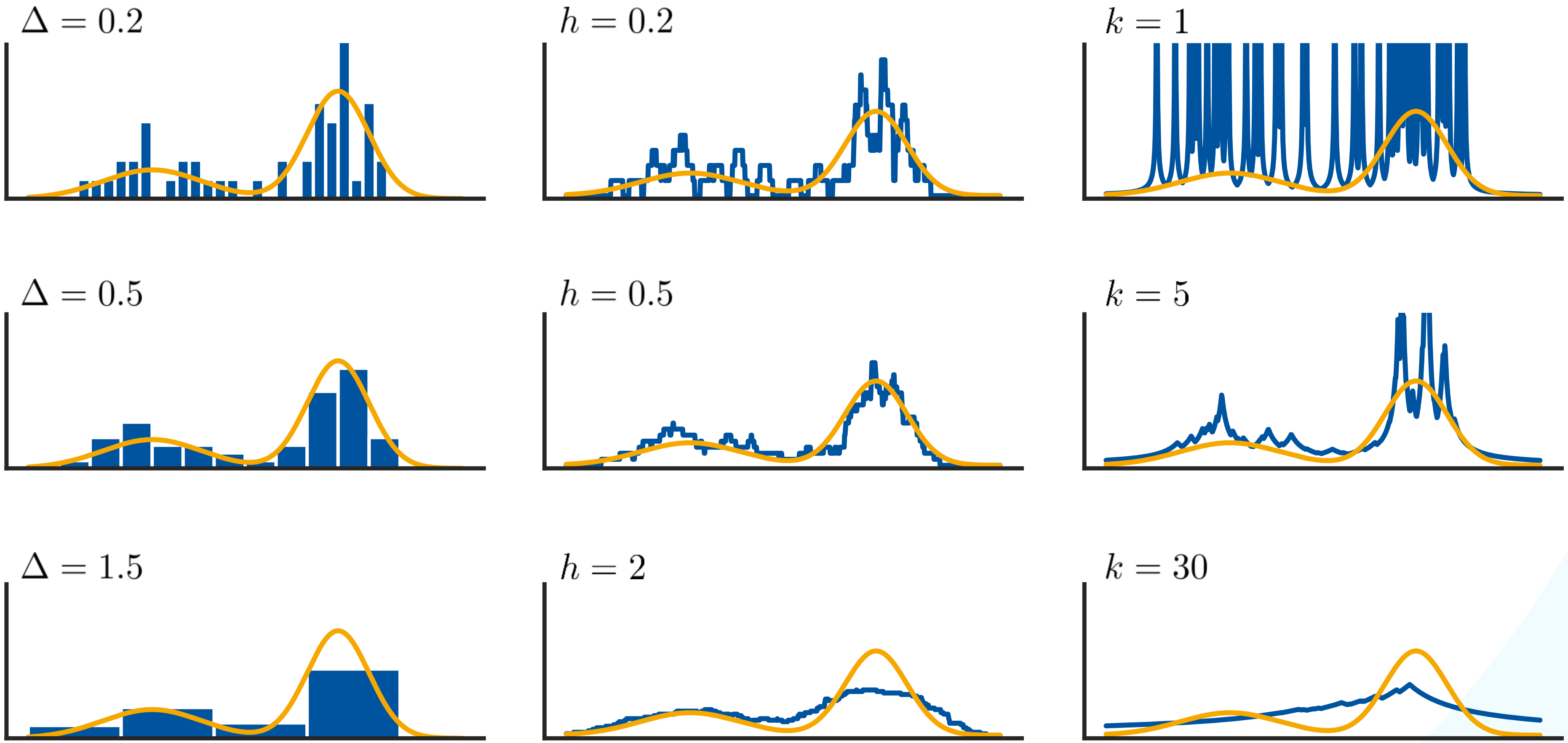
Not smooth enough

Too much variance

About ok

Too smooth

Too much bias



Histograms:
Bin width Δ

Parzen Window:
Kernel size h

k-NN:
\# of neighbors k

References and Further Reading

- More information in Bishop's book
  - Gaussian distribution and ML: Ch. 1.2.4 and 2.3.1-2.3.4.
  - Nonparametric methods: Ch. 2.5.
-

Christopher M. Bishop
Pattern Recognition and Machine Learning
Springer, 2006