**Universität Stuttgart**
KI – Institute for Artificial Intelligence

Analytic Computing

# Machine Learning
# 2 Classification

Prof. Dr. Steffen Staab
Nadeen Fatallah
Daniel Frank
Akram Sadat Hosseini
Rodrigo Lopez
Osama Mohamed
Yi Wang
Tim Schneider

https://www.ki.uni-stuttgart.de/

Now all in different places!

- partially based on slides by
  - T. Gottron & M. Strohmaier, U. Koblenz-Landau
  - Florian Lemmerich et al, U. of Würzburg

http://west.uni-koblenz.de/en/studium/lehrveranstaltungen/ws1516/machine-learning-and-data-mining

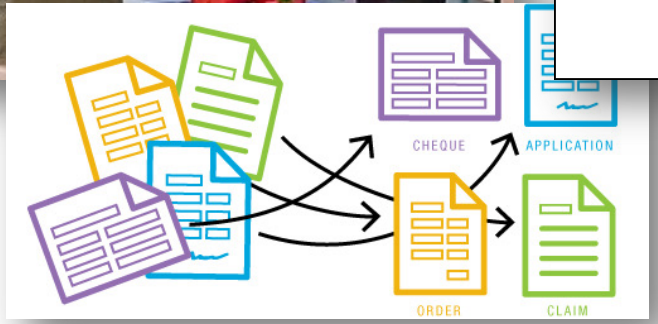
## Today‘s learning objectives (Monday, April 27, 2020)

Completing this slide deck you should know:

- What is *Classification*?
- What does it mean to *learn a classifier*?
- Machine Learning is optimization
  - In particular:
    A classifier is learned using optimization
- How does the algorithm *1-nearest-neighbor* work*?*
- *kNN*

# *Classification*

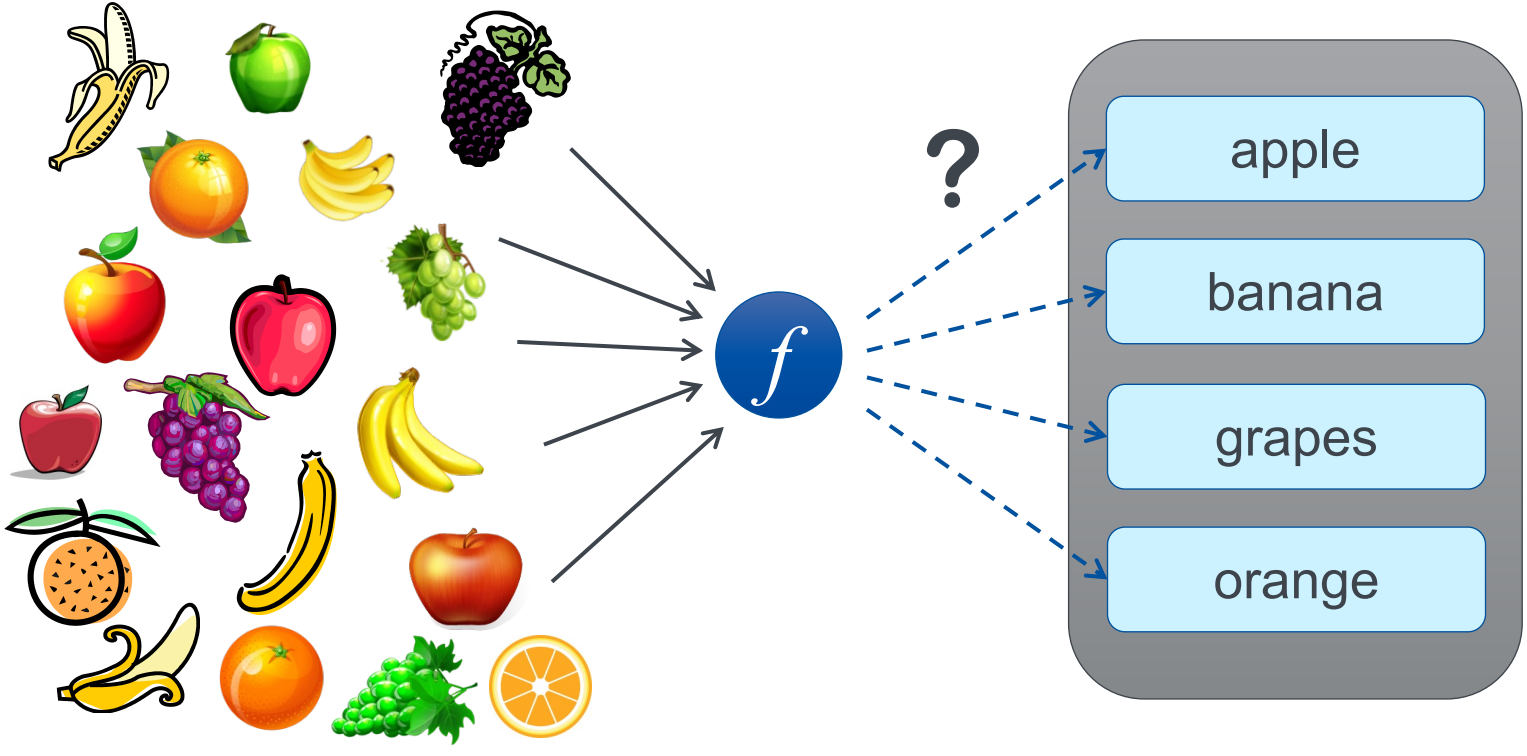
# Example use cases for classification




| | Moody's | S&P | Fitch | Meaning |
|---|---|---|---|---|
| Investment Grade | Aaa | AAA | AAA | Prime |
| | Aa1 | AA+ | AA+ | High Grade |
| | Aa2 | AA | AA | |
| | Aa3 | AA- | AA- | |
| | A1 | A+ | A+ | Upper Medium Grade |
| | A2 | A | A | |
| | A3 | A- | A- | |
| | Baa1 | BBB+ | BBB+ | Lower Medium Grade |
| | Baa2 | BBB | BBB | |
| | Baa3 | BBB- | BBB- | |
| Junk | Ba1 | BB+ | BB+ | Non Investment Grade Speculative |
| | Ba2 | BB | BB | |
| | Ba3 | BB- | BB- | |
| | B1 | B+ | B+ | Highly Speculative |
| | B2 | B | B | |
| | B3 | B- | B- | |
| | Caa1 | CCC+ | CCC+ | Substantial Risks |
| | Caa2 | CCC | CCC | Extremely Speculative |
| | Caa3 | CCC- | CCC- | In Default with Little Prospect of Recovery |
| | Ca | CC | CC+ | |
| | | C | CC | In Default |
| | | | CC- | |
| | D | D | DDD | |

# Classification

Classification is a method that assigns labels to object representations



# Classifier $f$

Each **object** $x_i \in X$ is described using
a list of $m$ attributes from a set $\mathcal{A} = \{A_1, \dots, A_m\}$

$$\forall i, k\colon\ x_i = \left(x_{i,1}, x_{i,2}, \dots, x_{i,m}\right)^T,$$

$$x_{i,k} \in A_k$$

(green, round, smooth)

(orange, round, rough)

**Category labels**

$$\mathrm{Y} = \{l_1, \dots, l_k\}$$

{apple, banana, grapes, orange}

**The classifier $f$ is** a function

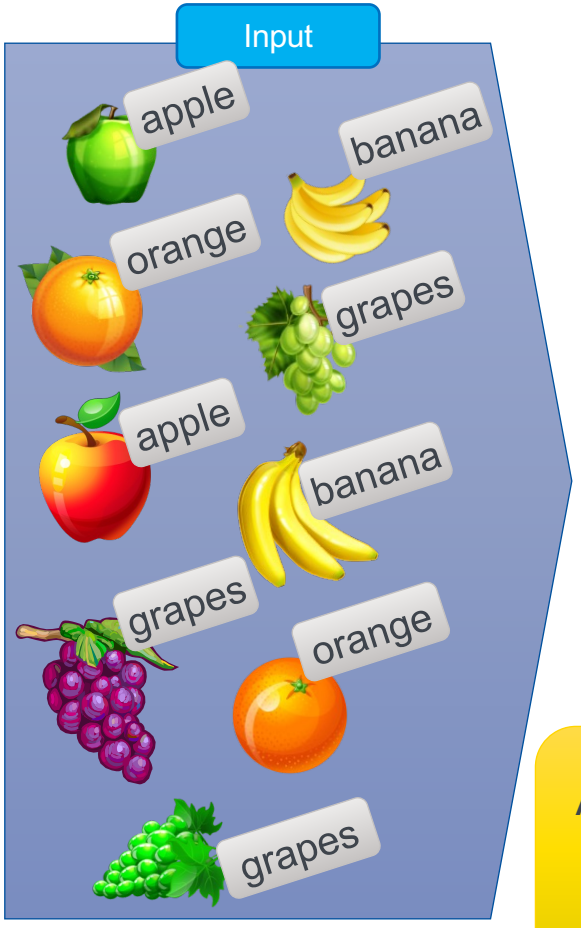
that maps descriptions of objects

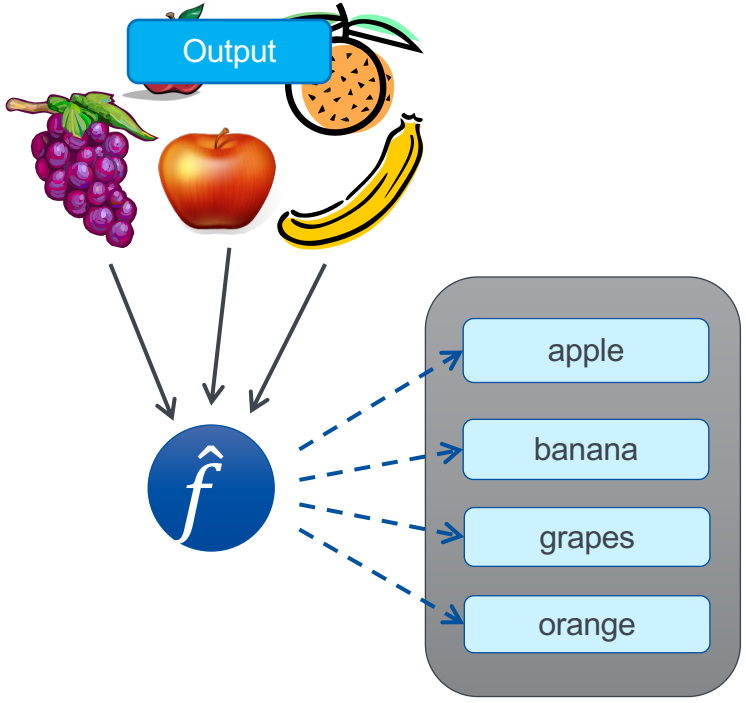
onto category labels

$$f\colon X \to Y$$

Many ways to model $f$

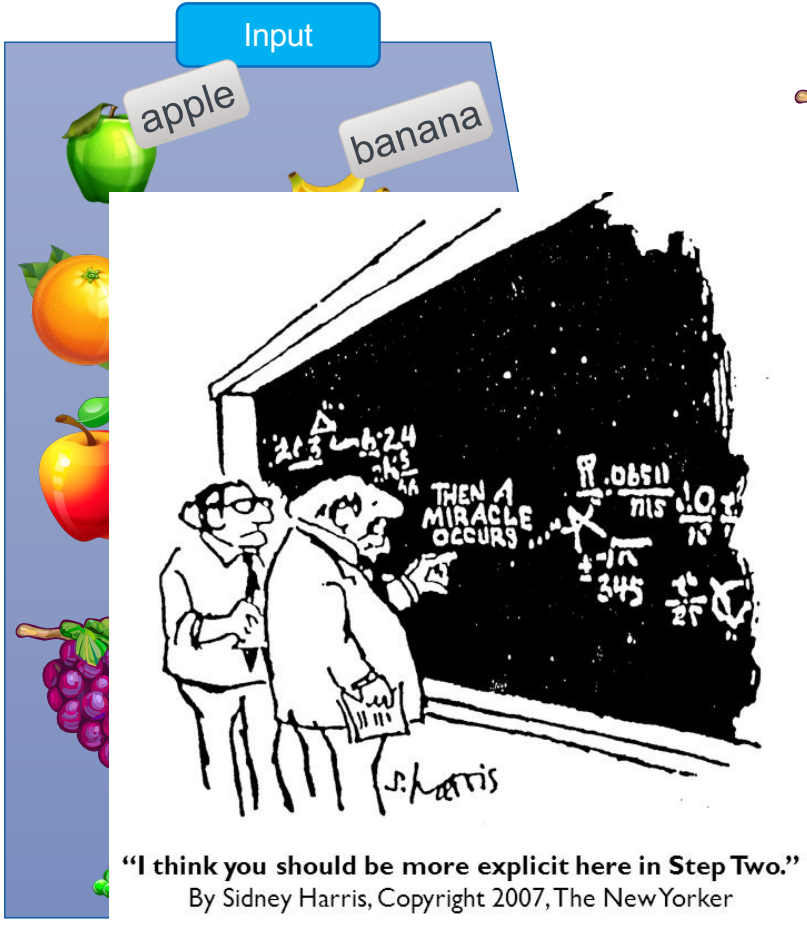
# Learning a classifier with pre-classified training data (labeled training data)

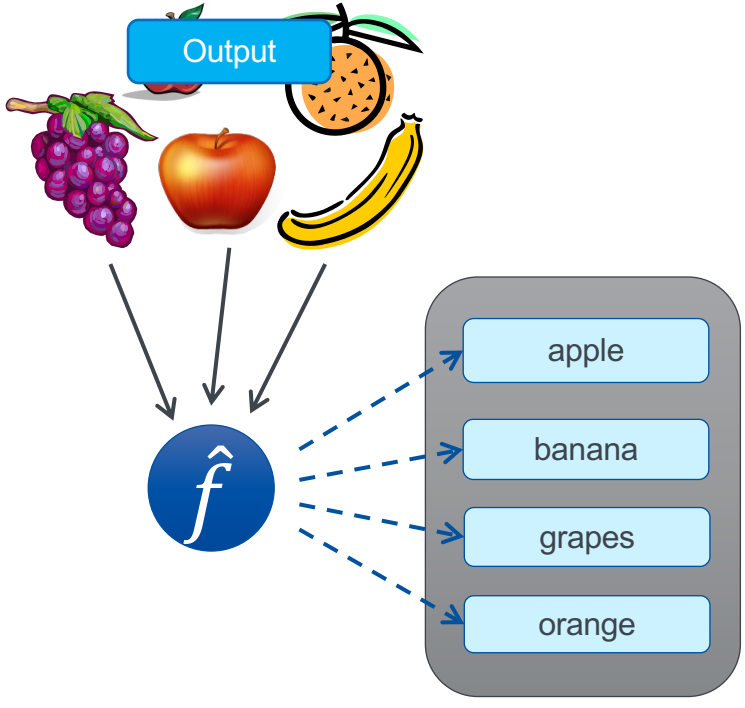


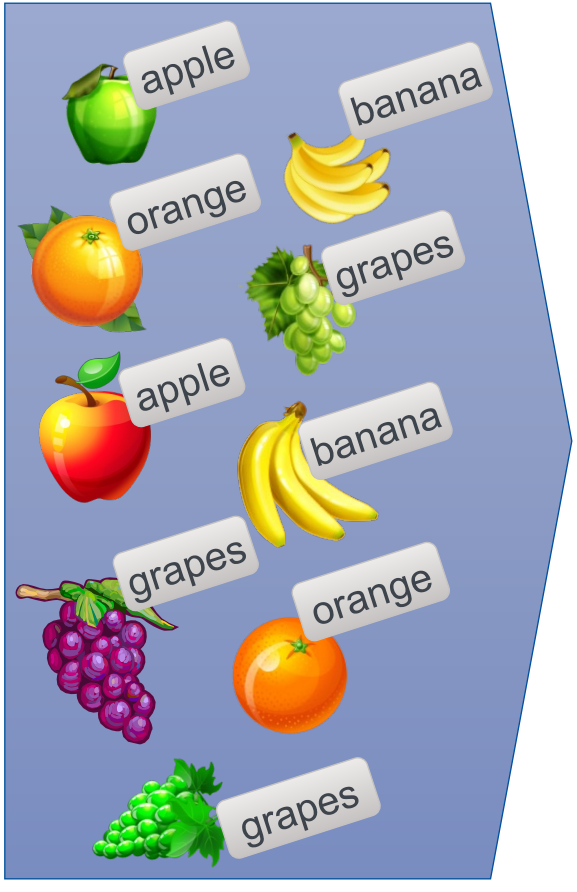

A frequently (but not always) used convention is to use $\hat{}$ in order to mark the learned function $\hat{f}$ (vs. “true“ function $f$), that predicts value $\hat{y}$ (where $y$ is the ”true” value/label)

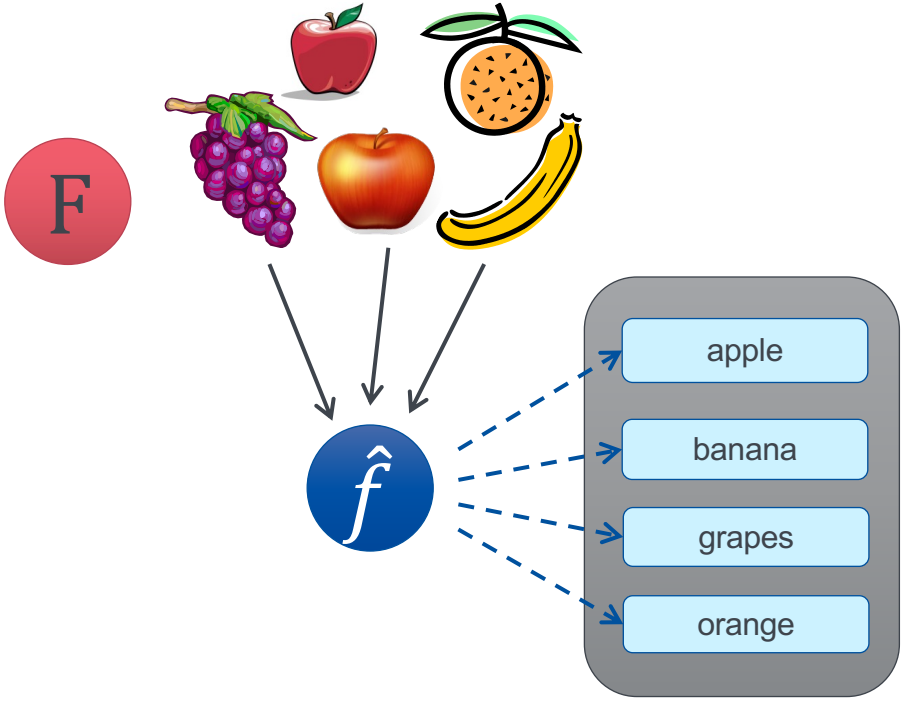
# Learning a classifier with pre-classified training data (labeled training data)



# Learning a classifier with pre-classified training data (labeled training data)

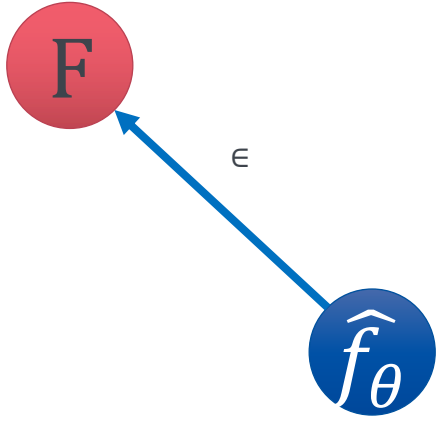


# Families of functions $\Gamma = \{f| \dots\}$

Which families of functions do you know?

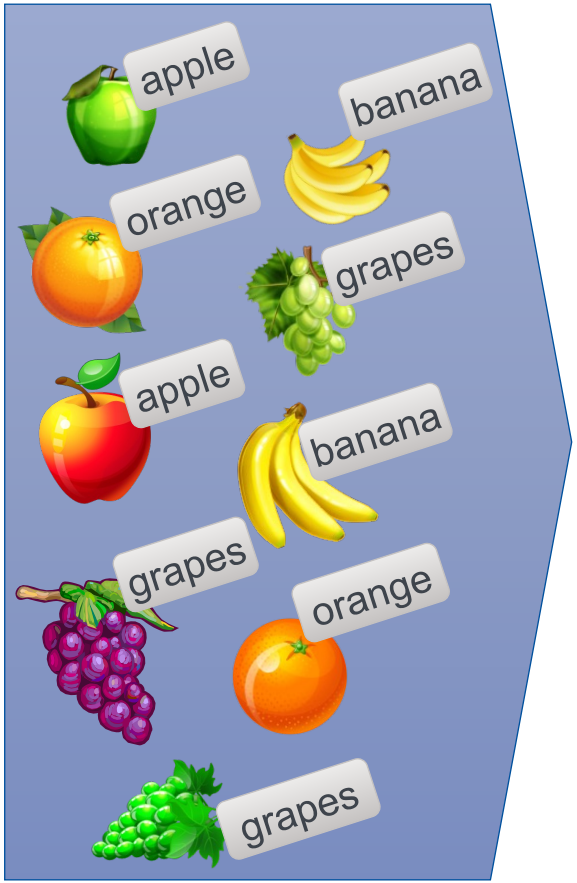
- family of constant functions
  $$\{f|f(x) = c\}$$
- family of linear functions
  $$\{f|f(x) = ax + c\}$$
- exponential functions
  $$\{f|f(x) = a^x\}$$
- ... $\hat{f}$

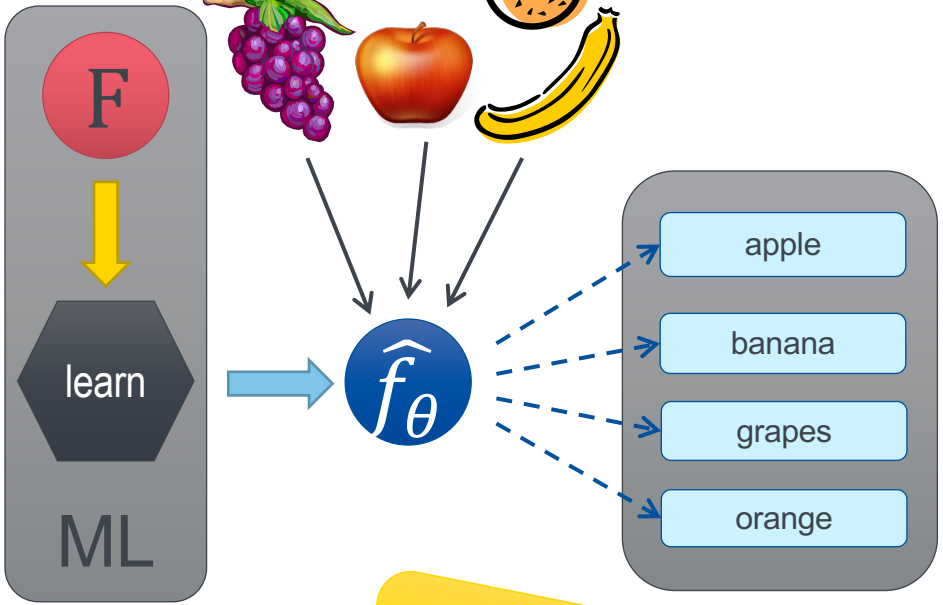



what characterizes a family of functions?

- form
- parameters $(a, b, c, \dots \theta_1, \dots \theta_t)$

# Learning a classifier with pre-classified training data (labelled training data)




Core questions of ML:
Which form does $\hat{f}$ have?
How to determine the parameters?

# Defining the task of learning a classifier

Given dataset $\mathcal{D}$ with $n$ objects $\mathcal{D} = \{(x_1, y_1), \ldots, (x_n, y_n)\} \sim X \times Y$

each described using $m$ attributes

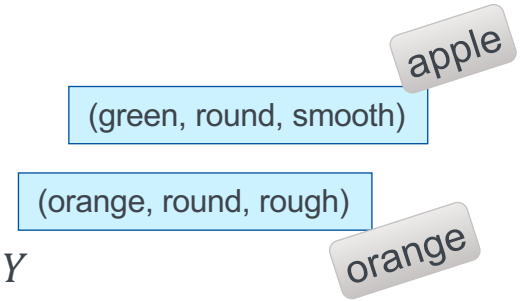
and an observed category label $y_i$

$$\forall i, k:\ x_i = (x_{i,1}, x_{i,2}, \ldots, x_{i,m}),$$

$$x_{i,k} \in A_k,$$

$$y_i \in L$$

where $X \times Y$ is the joint distribution of random variables $X$ and $Y$




Given a family of functions $\mathrm{F}$,

find a function $\widehat{f_\theta} \in \mathrm{F}$,

such that $\widehat{f_\theta}: A_1 \times \cdots \times A_m \to Y$

and $\widehat{f_\theta}$ minimizes the *empirical risk* ("loss")

on observed data (X,Y)

$$\widehat{f_\theta} = \underset{\tilde{f} \in \mathrm{F}}{\operatorname{argmin}} \sum_{i=1\ldots n} \ell(\tilde{f}(x_i), y_i)$$

$\widehat{f_\theta}(x_i)$ should reproduce observation $y_i$

# Empirical risk minimization and overfitting

Minimizing loss:

$$\min_{\theta} \mathbb{E}_{x,y\sim P(X,Y)}[\ell(\widehat{f_\theta}(x), y)]$$

Theory of machine learning: Law of large numbers:

$$\min_{\theta} \frac{1}{N}\sum_{i=1}^{N} \ell(\widehat{f_\theta}(x_n), y_n) \quad \overset{\text{“}LLN\text{“}}{\rightarrow} \quad \min_{\theta} \mathbb{E}[\ell(\widehat{f_\theta}(x), y)]$$

# Viewing machine learning as empirical risk minimization

Designing models:

$$\min_{\theta} \frac{1}{n} \sum_{i=1}^{n} \ell(\hat{f}_{\theta}(x_i), y_i)$$

Choosing/augmenting data:

$$\min_{\theta} \frac{1}{n} \sum_{i=1}^{n} \ell(\hat{f}_{\theta}(x_i), y_i)$$

Designing loss function:

$$\min_{\theta} \frac{1}{n} \sum_{i=1}^{n} \ell(\hat{f}_{\theta}(x_i), y_i)$$

Designing optimization methods:

$$\min_{\theta} \frac{1}{n} \sum_{i=1}^{n} \ell(\hat{f}_{\theta}(x_i), y_i)$$

# A Few Useful Things to Know about Machine Learning

- learning = data + representation + evaluation + optimization
  - data: quality of data
  - representation: which form does $\hat{f}_\theta$ have?
  - optimization: how to determine parameters $\theta$?
  - evaluation: empirical risk/$\mathrm{loss}$

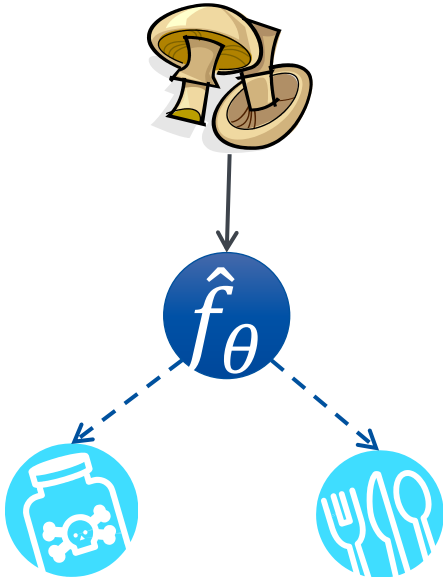
*1-nearest-neighbor*

Example

# Mushroom dataset (subset)


Example

# Mushroom dataset (subset)




- 100 objects, 8 attributes (features)*
- Category labels $L = \{e, p\}$ :
  - edible=e
  - poisonous=p
- prediction task:
  - classify unseen mushrooms

| cap-shape | cap-surface | cap-color | bruises | gill-spacing | gill-size | gill-color | habitat |
|---|---|---|---|---|---|---|---|
| bell=b<br>convex=x<br>flat=f<br>sunken=s | fibrous=f<br>scaly=y<br>smooth=s | brown=n<br>gray=g<br>white=w<br>yellow=y | bruises=t<br>no=f | close=c<br>crowded=w | broad=b<br>narrow=n | black=k<br>brown=n<br>gray=g<br>pink=p<br>white=w | grasses=g<br>meadows=m<br>paths=p<br>urban=u<br>woods=d |

* Original dataset contains more features and also more values for the features shown here

http://archive.ics.uci.edu/ml/datasets/Mushroom

# The first 4 entries of the mushroom data set

*p*,*x*,s,n,t,p,f,c,n,*k*,e,e,s,s,w,w,p,w,o,p,k,s,u

*e*,*b*,s,w,t,l,f,c,b,*n*,e,c,s,s,w,w,p,w,o,p,n,n,m

*p*,*x*,y,w,t,p,f,c,n,*n*,e,e,s,s,w,w,p,w,o,p,k,s,u

*e*,*x*,s,y,t,a,f,c,b,*k*,e,c,s,s,w,w,p,w,o,p,n,n,g

edible/
poisonous

bell shaped
yes (b), no (x)

black gills
yes (k), no (n)

# Data Preprocessing

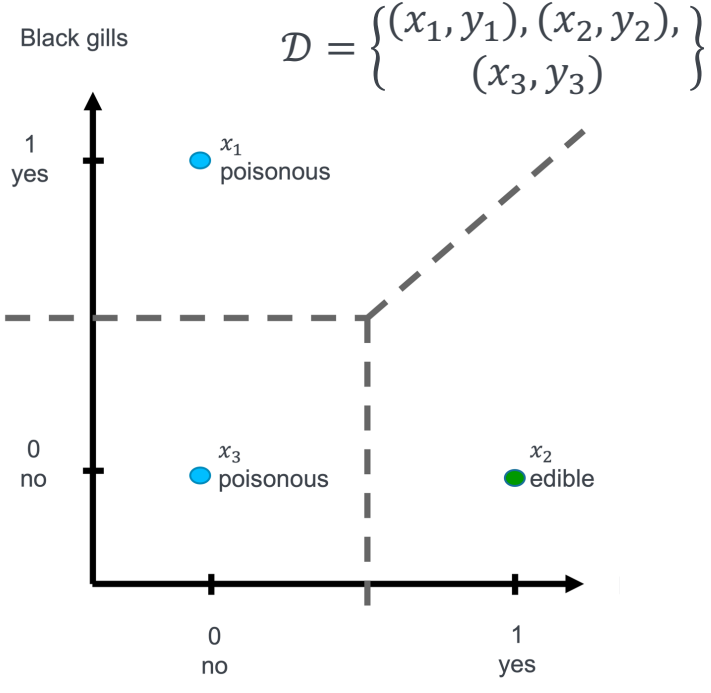
In this example: from CSV to suitable vector representation

p,x,s,n,t,p,f,c,n,k,e,e,s,s,w,w,p,w,o,p,k,s,u

e,b,s,w,t,l,f,c,b,n,e,c,s,s,w,w,p,w,o,p,n,n,m

p,x,y,w,t,p,f,c,n,n,e,e,s,s,w,w,p,w,o,p,k,s,u

e,x,s,y,t,a,f,c,b,k,e,c,s,s,w,w,p,w,o,p,n,n,g

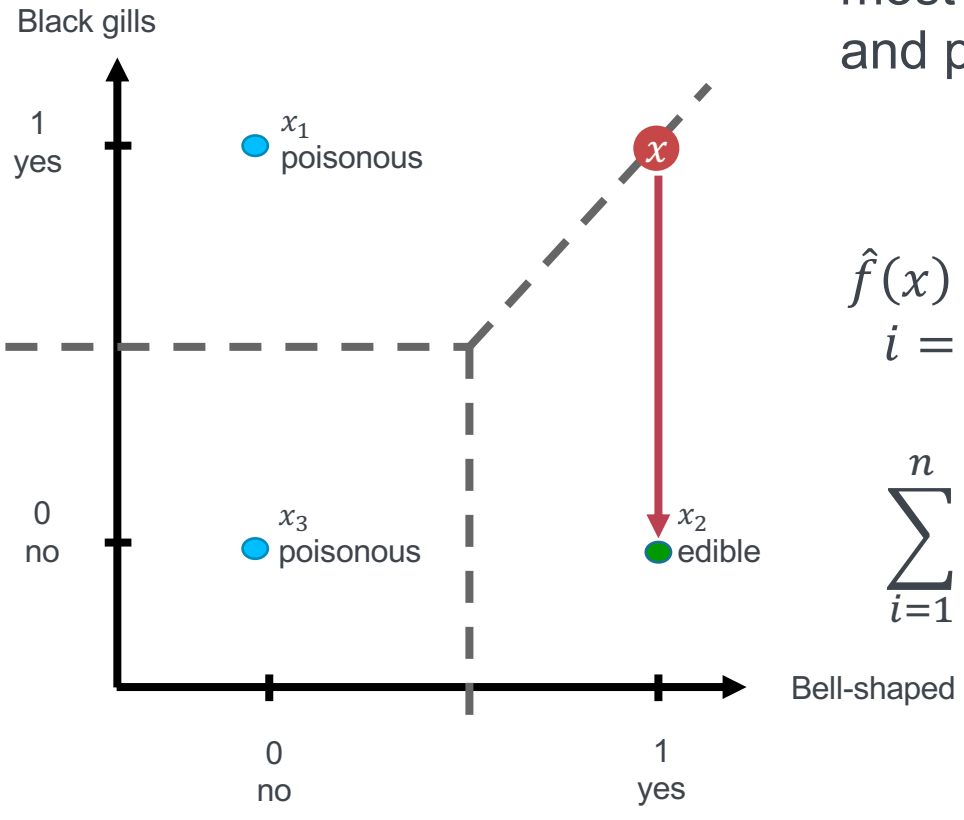


# 1-nearest-neighbor (1-NN): Lazy learning (represent in vector space)

ambiguous: many functions fulfill this description

$$\mathcal{D} = \{(x_{1,1}, x_{1,2}, y_1), (x_{2,1}, x_{2,2}, y_2), (x_{3,1}, x_{3,2}, y_3)\}$$
$$= \{(x_1, y_1), (x_2, y_2), (x_3, y_3)\}$$

$\mathrm{F} = \{\tilde{f} \,|\tilde{f}\,(x)$ consider most similar neighbors and pick their labels$\}$




$$\hat{f}(x) = y_i, \text{ where } i = \underset{j|x_j \in X}{\operatorname{argmin}} (x - x_j)^2$$

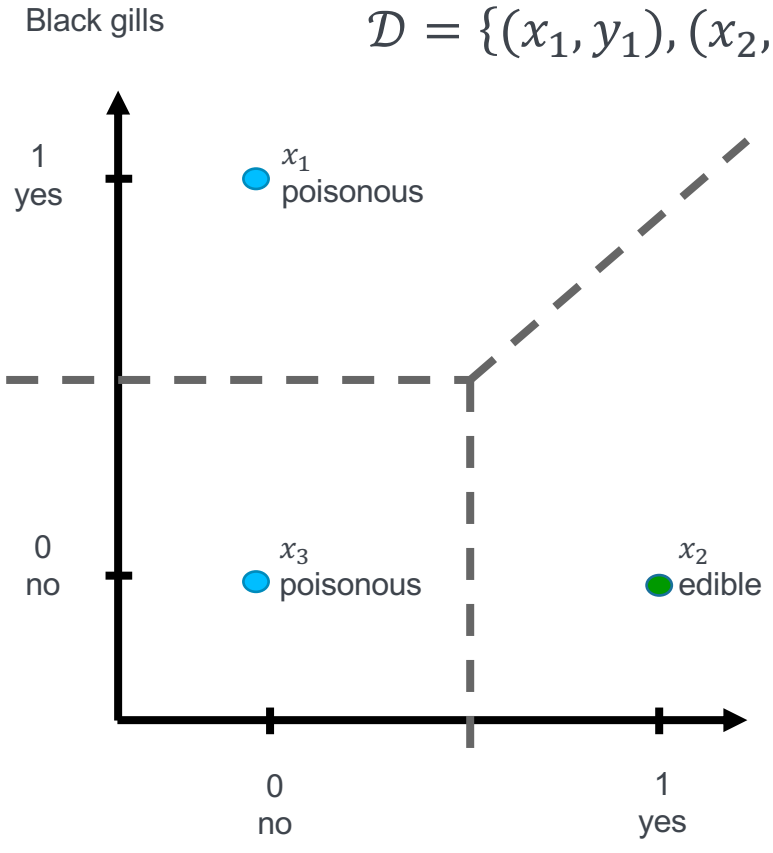
$$\sum_{i=1}^{n} \ell(\hat{f}_\theta(x_i), y_i) = ?$$

## Empirical risk (cost function / loss function)

Example loss function (not a good one for classification!)

- Quadratic Euclidean distance between
  - observed („true“) category labels: $y_i$
  - predicted category labels: $\hat{f}(x_i) = \widehat{y_i}$
- $\sum_{i=1}^{n} \ell(\hat{f}_\theta(x_i), y_i) = \sum_i (\hat{f}(x_i) - y_i)^2$
  where 0 indicates perfect agreement (minimal distance)

# 1-nearest neighbor: Lazy learning (represent in vector space)




$$\mathcal{D} = \{(x_1, y_1), (x_2, y_2), (x_3, y_3)\}$$

$$\hat{f}(x) = y_i, \text{ where } i = \underset{j|x_j \in X}{\operatorname{argmin}} (x - x_j)^2$$

$$\sum_{i=1}^{n} \ell(\hat{f}_\theta(x_i), y_i) = \mathbf{0}$$

## „Ingredients for 1-nearest-neighbor“

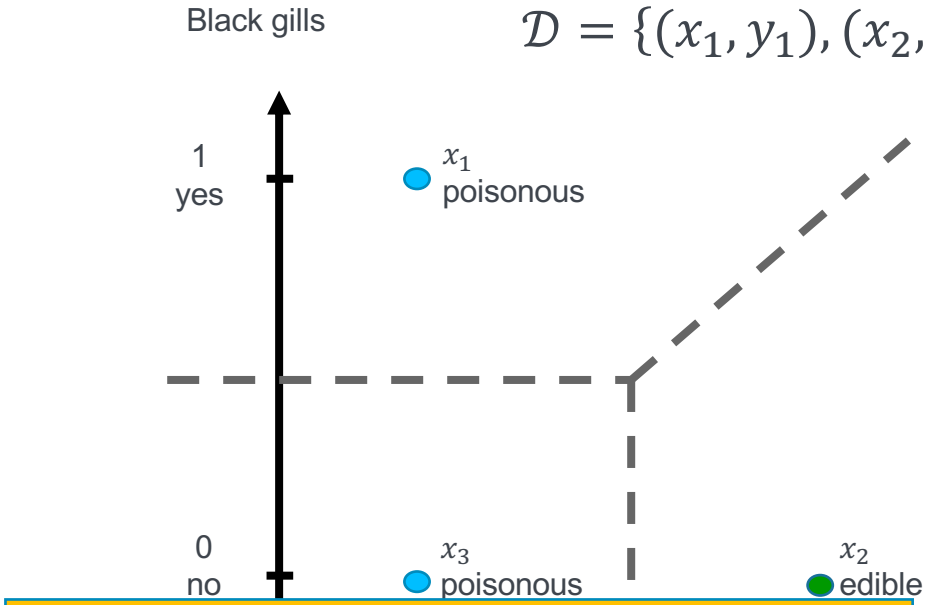
Effectiveness

- Representation of objects in vector space
- Definition of distance measure
- Definition of loss function

Efficiency

- Algorithmically efficient implementation
  - training: acquire $\hat{f}_\theta$
  - inference: apply $\hat{f}_\theta$

# 1-nearest neighbor: Lazy learning (represent in vector space)




$$\mathcal{D} = \{(x_1, y_1), (x_2, y_2), (x_3, y_3)\}$$

$$\hat{f}(x) = y_i, \text{ where } i = \underset{j|x_j \in X}{\operatorname{argmin}} (x - x_j)^2$$

$$\sum_{i=1}^{n} \ell(\hat{f}_\theta(x_i), y_i) = \mathbf{0}$$

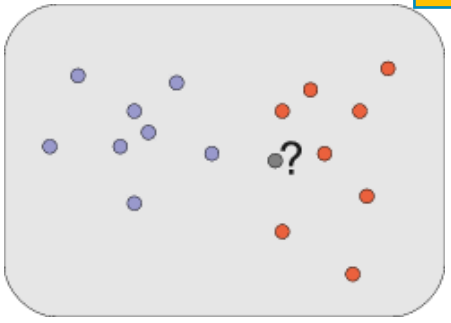
$$\sum_{i=1}^{n} \ell(\hat{f}_\theta(x_i), y_i) = \mathbf{0}$$

perfect?

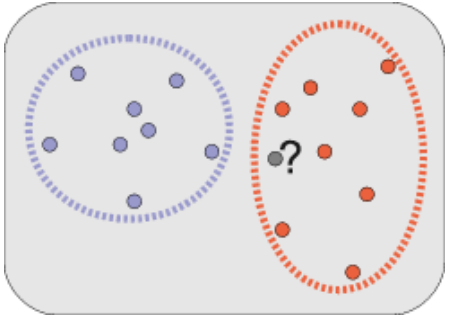
Bell-shaped

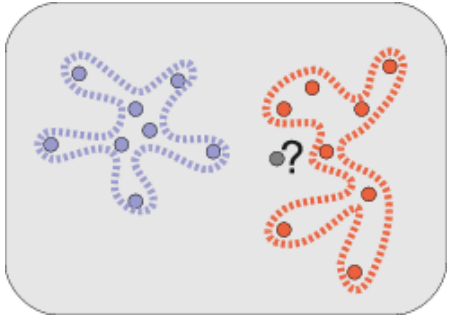
# Overfitting

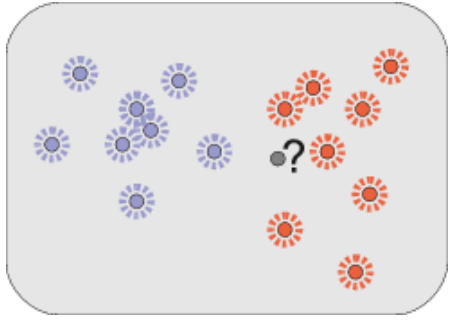
Many algorithms optimize $\sum_i \ell\left(\hat{f}(x_i), y_i\right)$, but only few yield good predictors




- Labelled training data provides only examples
- Generalization required
  - How to classify new/unseen data?

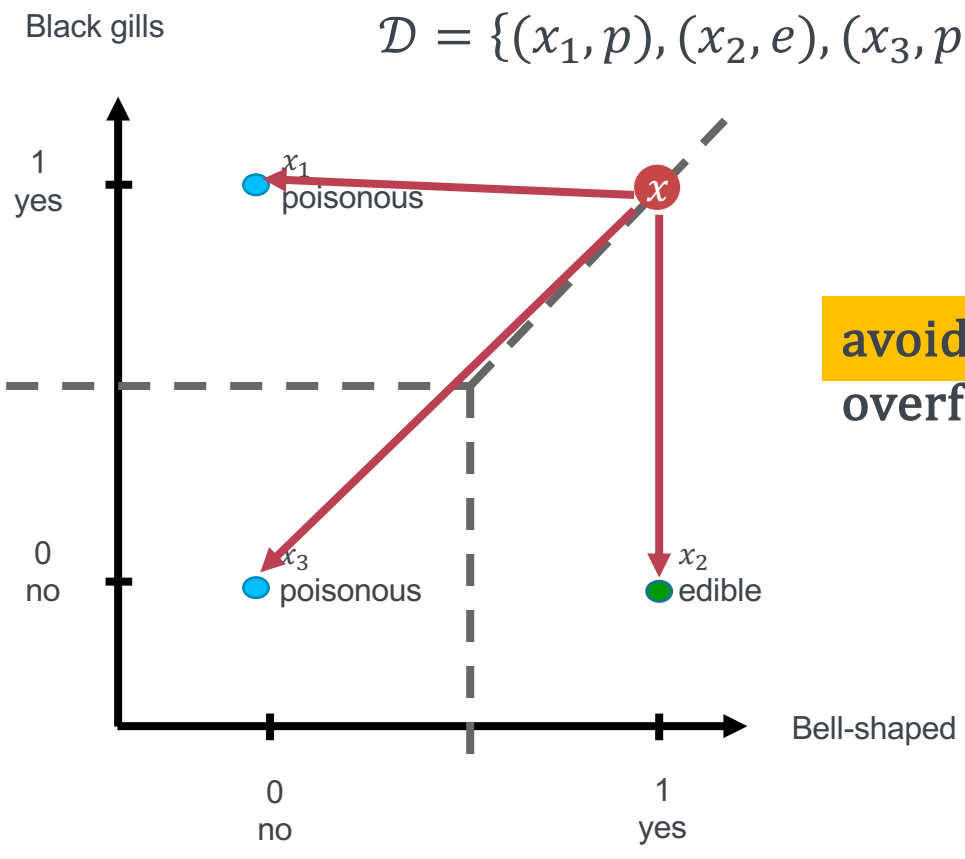


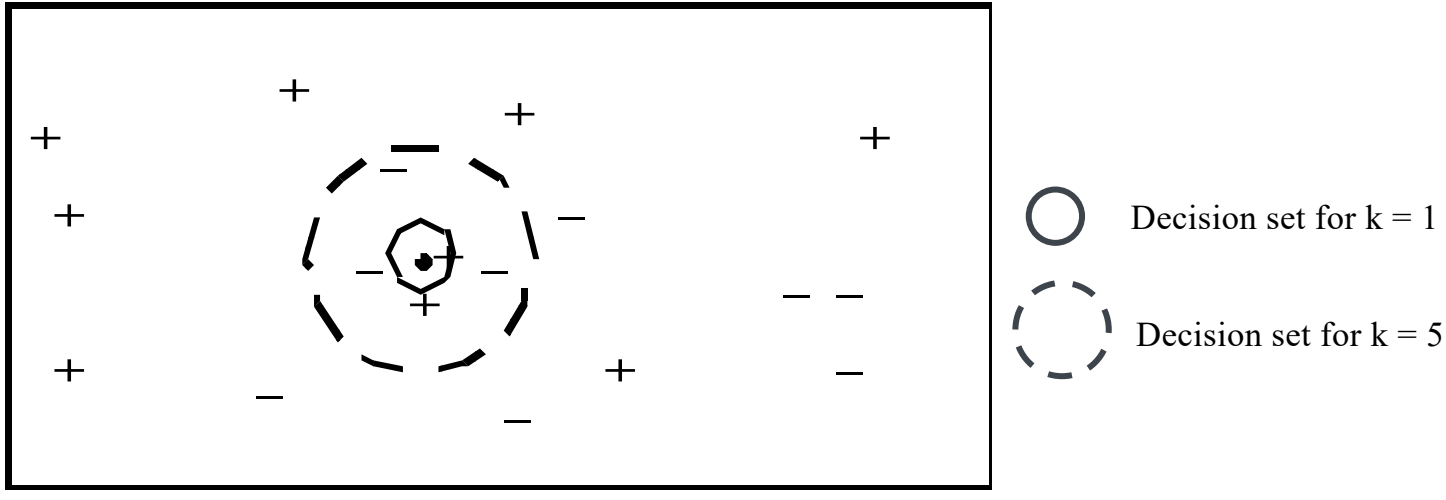

- Overfitting:
  - Not enough generalization
  - Bad performance on new data
  - „Learning by heart“ (extreme case: using an object ID)

# k-nearest neighbor (kNN): Lazy learning (represent in vector space)




avoids overfitting

# Example




Examples for classes „+“ and „-“

Decision:

*k* = 1: classified as „+“,
*k* = 5: classified as „–“

# Next week, Monday 29

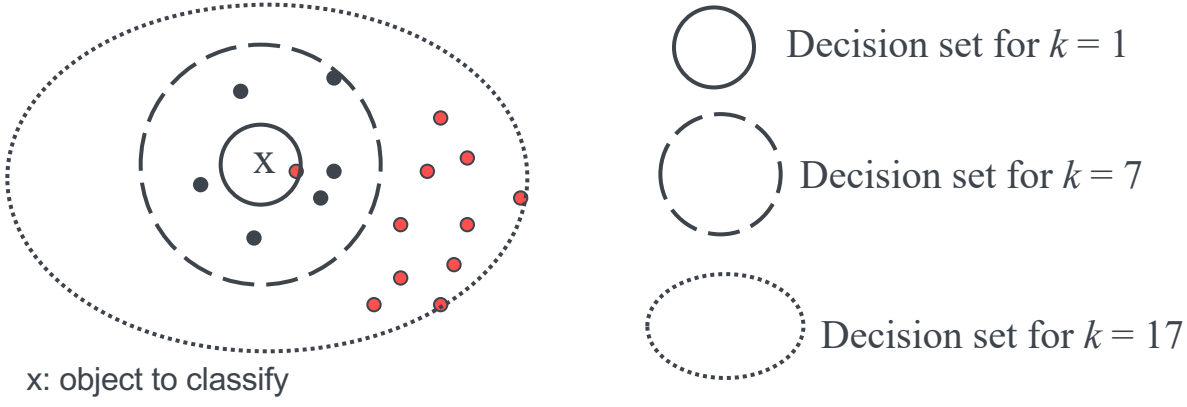
- No lecture
- Lecture video will be uploaded for you to watch

# Exam in Machine Learning

12. August 2024, 12.00-14.00 hrs

# The parameter *k*

- *k „too small“*:
  $\Rightarrow$ Too little generalization: high sensitivity regarding outliers
- *k „too large“:*
  $\Rightarrow$ too much generalization:
  many objects from different clusters (classes) included in the decision set.
- *„middle range“ k*: highest classification quality, often 1 << *k* < 10



## Selection of the parameter k

- Probing different parameters: k= 1, 2, 3, …
- For each k: determine classification error:
  - Either on the training set
  - Or via x-fold cross validation
- Use best k for classification

# Decision rules

- **Default rule**
  choose the majority class of the decision set

- **Weighted decision rules**

  Weigh the training examples in the decision set

  - w.r.t. the distance to the object to classify
  - w.r.t. the distribution of the class (which might be skewed!)
  - Example: class A: 95 %, class B 5 %
    Decision set = {A, A, A, A, B, B, B}
    default rule $\Rightarrow$ A, weighted rule $\Rightarrow$ B
  - w.r.t. a cost function
    (predicting A as B is less tolerable than vice versa)

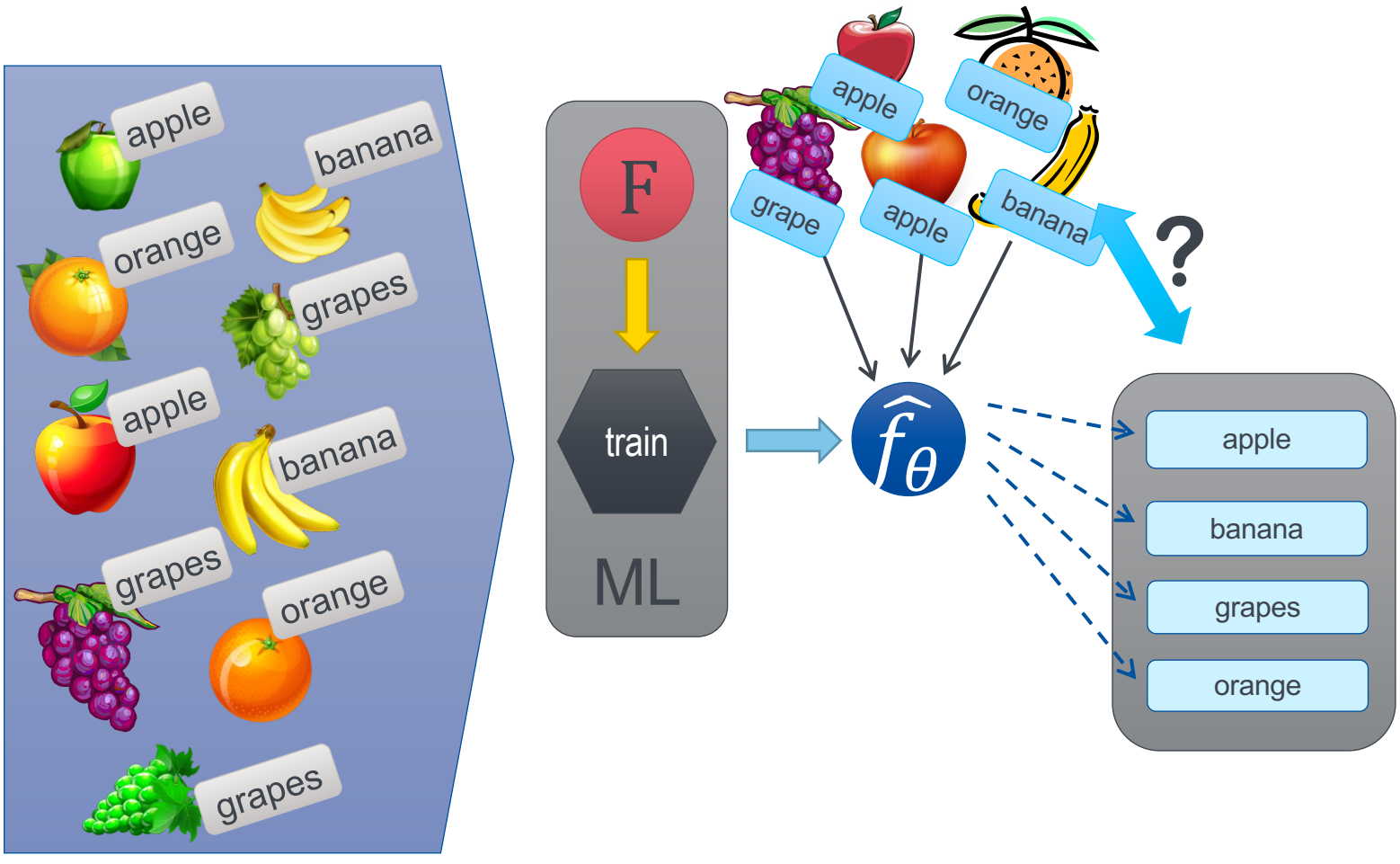
# Discussion: k-nearest neighbour

- Advantage:
  - Only requires training data as an input
  - High classification accuracy in many applications
  - Incremental: model does not need to be adapted
  - Variations for the prediction of numerical values exist

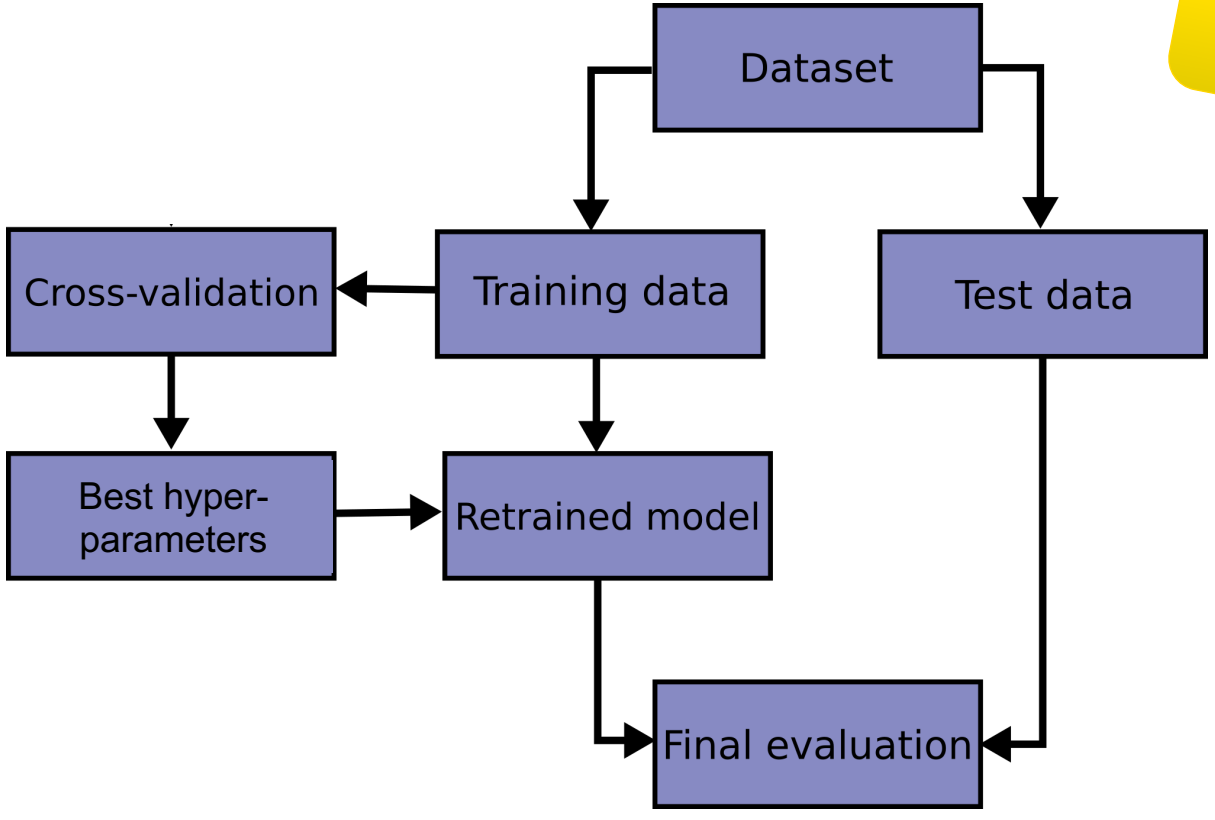
- Disadvantage:
  - Inefficient during classification (runtime)
  - Does not generate explicit knowledge about classes

# *Validation and Evaluation*

# Evaluation using Labeled Data



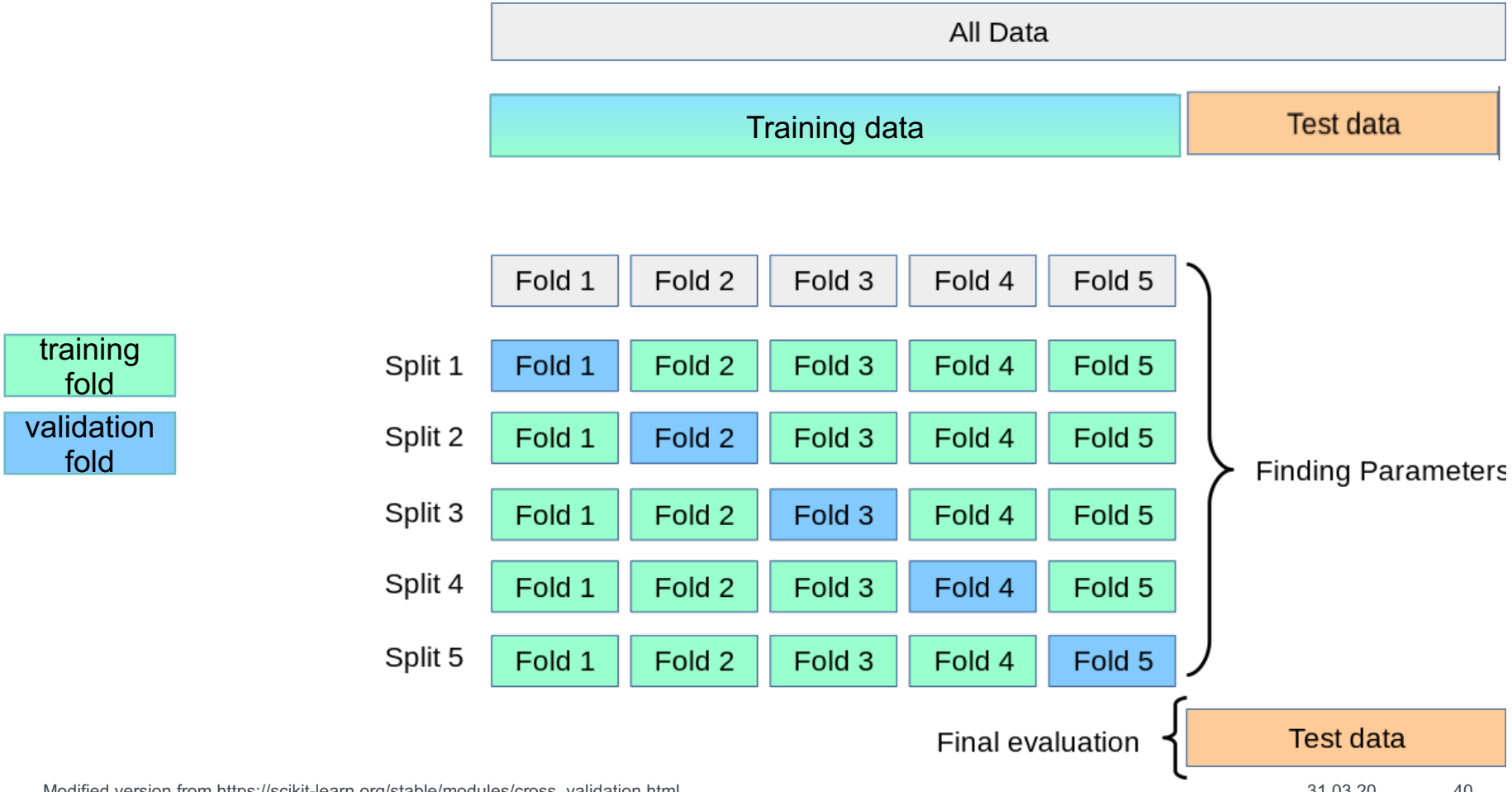
# Keep test data separate from your training efforts



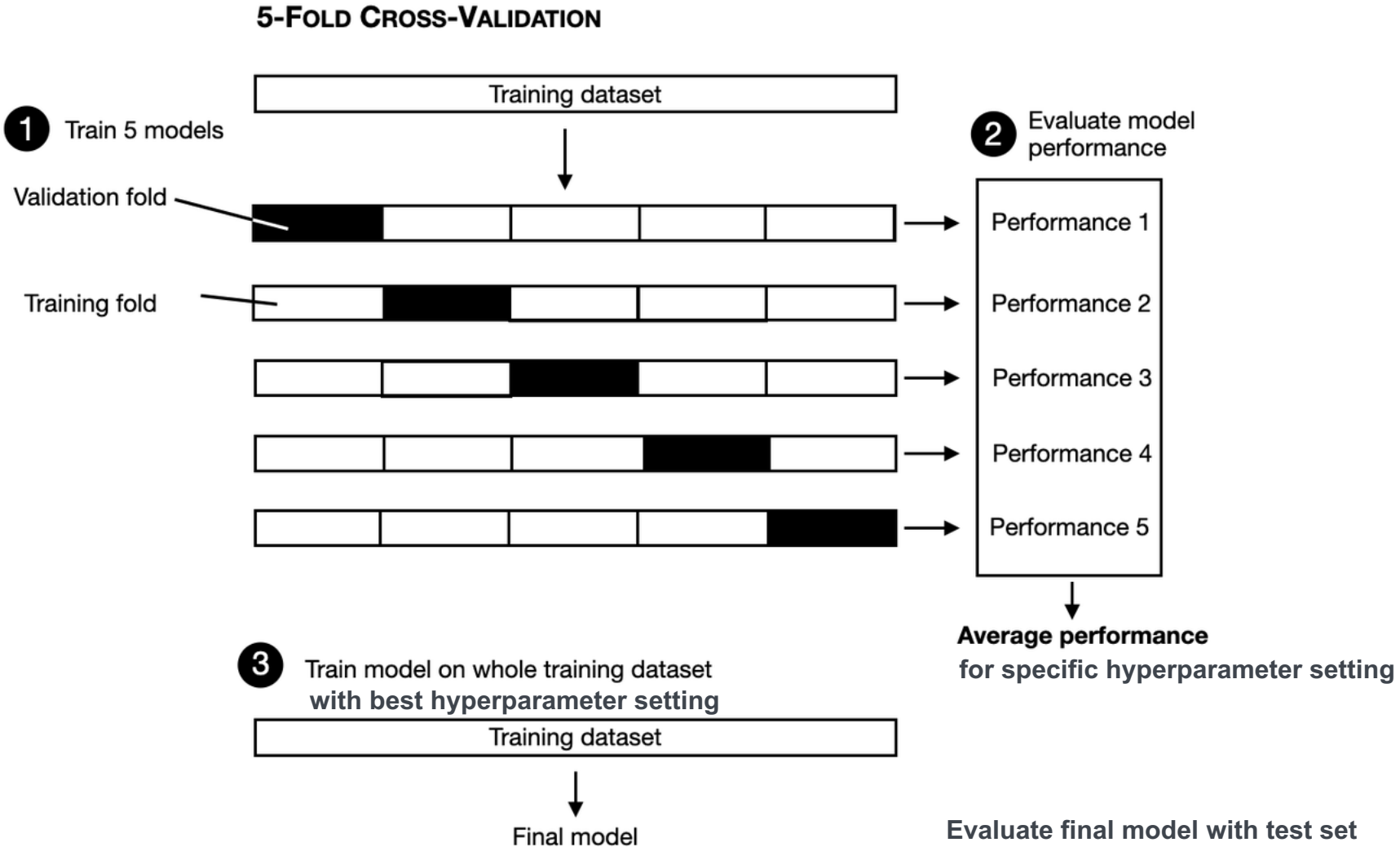

Otherwise no chance to detect overfitting

Modified version from https://scikit-learn.org/stable/modules/cross_validation.html

# Split Non-test Data into Training Data and Validation Data and k-fold cross-validation




Modified version from https://scikit-learn.org/stable/modules/cross_validation.html




Small k: shows sensitivity of hyperparameter choice; Large k: approximates performance

https://twitter.com/rasbt/status/1628427006386884614/photo/1

# Fighting overfitting when evaluating

- At each step: three non-overlapping sets
  - $\mathcal{D}_1 \cup \mathcal{D}_2 \cup \mathcal{D}_3 = \mathcal{D},\ \mathcal{D}_1 \cap \mathcal{D}_2 = \mathcal{D}_1 \cap \mathcal{D}_3 = \mathcal{D}_2 \cap \mathcal{D}_3 = \emptyset$
- Train to minimize $\sum_{(x,y)\in\mathcal{D}_1} \ell\left(\hat{f}(x), y\right)$
- Validate results with $\sum_{(x,y)\in\mathcal{D}_2} \ell\left(\hat{f}(x), y\right)$
  - determine best hyperparameters, e.g. k in kNN
- Evaluate results with $\sum_{(x,y)\in\mathcal{D}_3} \ell\left(\hat{f}(x), y\right)$
  - determine how well you do after modeling and optimization

Training data, validation data and evaluation data **must not overlap** in pairwise intersection

# How to evaluate?

- At each step: three non-overlapping sets
  - $\mathcal{D}_1 \cup \mathcal{D}_2 \cup \mathcal{D}_3 = \mathcal{D},\ \mathcal{D}_1 \cap \mathcal{D}_2 = \mathcal{D}_1 \cap \mathcal{D}_3 = \mathcal{D}_2 \cap \mathcal{D}_3 = \emptyset$
- Train to minimize $\sum_{(x,y)\in\mathcal{D}_1} \ell\left(\hat{f}(x), y\right)$
- Validate results with $\sum_{(x,y)\in\mathcal{D}_2} \ell\left(\hat{f}(x), y\right)$
  - determine best hyperparameters, e.g. k in kNN
- Evaluate results with $\sum_{(x,y)\in\mathcal{D}_3} \ell\left(\hat{f}(x), y\right)$
  - determine how well you do after modeling and optimization

**task**: guide parameter learning

Same loss function?

**task**: inform engineer or user about quality of system

# Confusion Matrix Representing Quality of System

- Extend confusion matrix to multiple categories

| | | Ground truth | | | | |
|---|---|---|---|---|---|---|
| | | $c_1$ | $c_2$ | $c_3$ | … | $c_J$ |
| $\gamma$ | $c_1$ | *correct* | error | error | | error |
| | $c_2$ | error | *correct* | error | | error |
| | $c_3$ | error | error | *correct* | | error |
| | … | | | | … | |
| | $c_J$ | error | error | error | | *correct* |

- Metrics for each category:
  - Recall:

$$r(c_i) = \frac{a_{ii}}{\sum_{k=1}^{J} a_{ki}}$$

Ratio of how many objects in $c_i$ have been correctly classified

  - Precision:

$$p(c_i) = \frac{a_{ii}}{\sum_{k=1}^{J} a_{ik}}$$

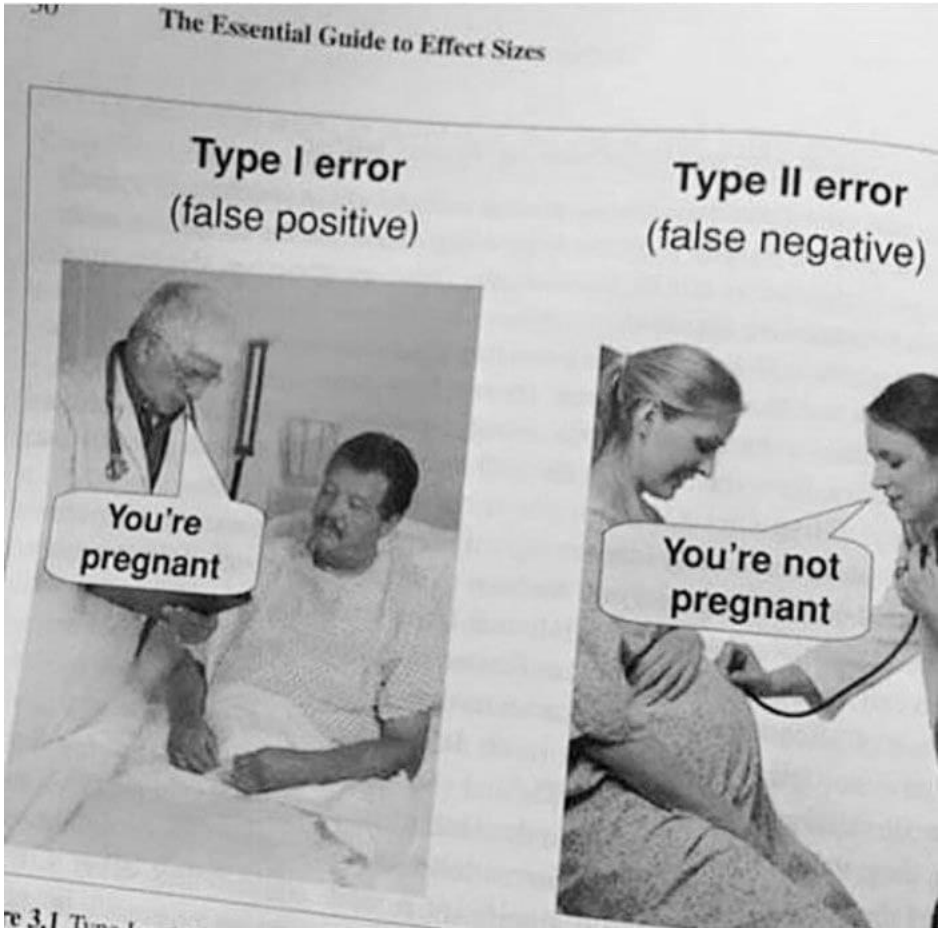
Ratio of how many objects have been correctly classified as $c_i$

  - $F_1$: harmonic mean of recall and precision:

$$F_1 = \frac{2 \cdot r \cdot p}{r + p}$$

A bit of both …

# Type I vs Type II error



# Confusion Matrix

- Extend confusion matrix to multiple categories

| | | Ground truth | | | | |
|---|---|---|---|---|---|---|
| | | $c_1$ | $c_2$ | $c_3$ | … | $c_J$ |
| $\gamma$ | $c_1$ | *correct* | error | error | | error |
| | $c_2$ | error | *correct* | error | | error |
| | $c_3$ | error | error | *correct* | | error |
| | … | | | | … | |
| | $c_J$ | error | error | error | | *correct* |

- Metrics:
  - Globally: Accuracy, 0-1-loss

$$acc = \frac{\sum_{i=1}^{J} a_{ii}}{\sum_{i=1}^{J} \sum_{j=1}^{J} a_{ij}}$$

Ratio of correct decisions

Example

# Evaluating a Mushroom Classifier

- Three categories: poisonous, edible, psychoactive
  - Confusion matrix

| | | Ground truth | | | |
|---|---|---|---|---|---|
| | | Poisonous | Edible | Psycho-active | ***Total*** |
| γ | Poisonous | *5* | 1 | 2 | ***8*** |
| | Edible | 2 | *10* | 4 | ***16*** |
| | Psycho-active | 0 | 0 | *6* | ***6*** |
| | ***Total*** | ***7*** | ***11*** | ***12*** | ***30*** |

- For „Poisonous“: Recall, Precision, $F_1$?

$$r = \frac{5}{7} = 0.71 \quad p = \frac{5}{8} = 0.63 \quad F_1 = \frac{2 \cdot r \cdot p}{r + p} = 0.67$$

- Accuracy, 0-1-loss? $$acc = \frac{21}{30} = 0.7$$

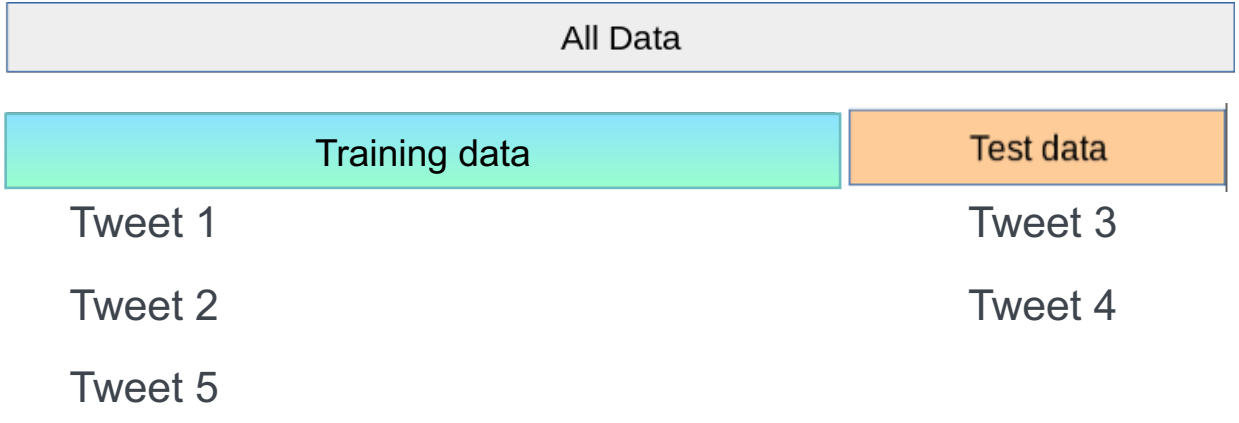
micro/macro/weighted

# Data Leakage

# Data leakage

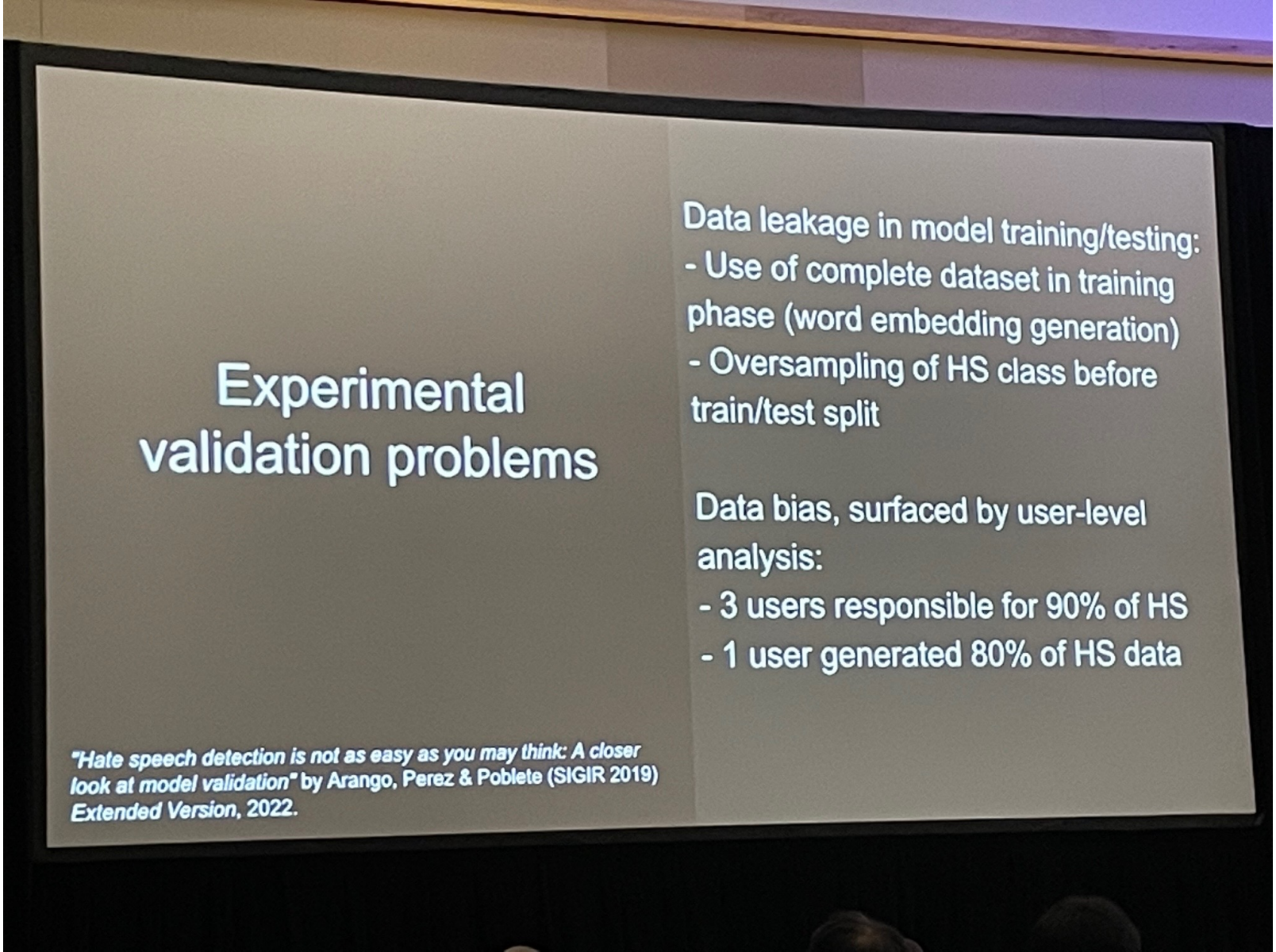
**Example: Classify social media posts into hate speech or not**

| | |
|---|---|
| Tweet 1 | Miller |
| Tweet 2 | Smith |
| Tweet 3 | Miller |
| Tweet 4 | Smith |
| Tweet 5 | McGuinness |




In this setup, one does not learn what hate speech is, but how Smith and Miller tweet


THE WEB CONFERENCE
Using diversity as a source of scientific innovation for the Web
Barbara Poblete
Department of Computer Science, University of Chile
Amazon Visiting Academic*
*content not associated to work done at Ama

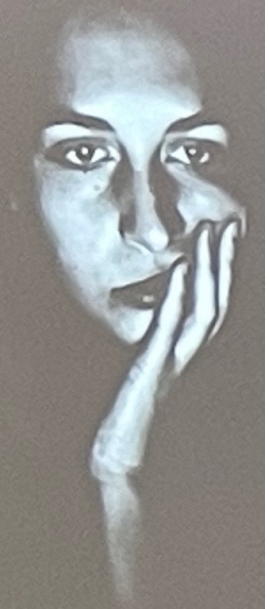

Experimental validation problems
Data leakage in model training/testing:
- Use of complete dataset in training phase (word embedding generation)
- Oversampling of HS class before train/test split
Data bias, surfaced by user-level analysis:
- 3 users responsible for 90% of HS
- 1 user generated 80% of HS data
"Hate speech detection is not as easy as you may think: A closer look at model validation" by Arango, Perez & Poblete (SIGIR 2019) Extended Version, 2022.

By fixing experimental issues, performance dropped to ~51% F1 (from ~93%)

Very close to our original Spanish baseline

Focus on the data is just as important as focus on performance metrics

As models become more obscure w/DL, experimental validation is key



*Hate speech detection is not as easy as you may think: A closer look at model validation"* by Arango, Perez & Poblete (2019)
*Extended Version*, 2022.

# Summary on Evaluation

- Evaluation is motivated by application
  - For example: you care about correct classification,
    you do not care whether you could have been almost correct
- A loss function is usually not a good evaluation function
  - As we will see later: A loss function needs to support the determination of parameters $\theta$

# Miscellaneous

# Variations of the Classification Task

- Category types:
  - Flat vs. hierarchical
  - Exclusive vs. multiple categories
- Function $\widehat{f_\theta}$
  - Hard vs. soft assignments
  - Manual provision vs. machine learning
- Purpose
  - Descriptive Modelling
    - Explain the data
  - Predictive Modelling
    - Classify new data

# Sources

- Optical recognition of fruits: http://de.ids-imaging.com/case-studies.html
- Document classification: http://www.ndm.net/opentext/capture-and-recognition/capture-center
- Email spam: http://www.gfi.com/blog/spam-emails-bringing-excitement-1978/
- Amanita muscaria: http://commons.wikimedia.org/wiki/File:Amanita_muscaria_3_vliegenzwammen_op_rij.jpg, CC-BY-SA 3.0-nl, Onderwijsgek
- Lactarius indigo: http://commons.wikimedia.org/wiki/File:Lactarius_indigo_48568.jpg, CC-BY-SA 3.0, Dan Molter

Universität Stuttgart
IPVS

# Thank you!

**Steffen Staab**

E-Mail Steffen.staab@ipvs.uni-stuttgart.de
Telefon +49 (0) 711 685-To be defined
www. ipvs.uni-stuttgart.de/departments/ac/

Universität Stuttgart
Analytic Computing, IPVS
Universitätsstraße 32, 50569 Stuttgart