


# Business Intelligence

## 10 Predictive Modeling II

Prof. Dr. Bastian Amberg  
(summer term 2024)  
12.6.2024

# Schedule

	Wed., 10:00-12:00			Fr., 14:00-16:00 (Start at 14:30)		Self-study	
Basics	W1	17.4.	(Meta-)Introduction	19.4.		Python-Basics	Chap. 1
	W2	24.4.	Data Warehouse – Overview & OLAP	26.4.	[Blockveranstaltung SE Prof. Gersch]		Chap. 2
	W3	1.5.		3.5.			Chap. 3
	W4	8.5.	Data Warehouse Modeling I & II	10.5.	Data Mining Introduction		
Main Part	W5	15.5.	CRISP-DM, Project understanding	17.5.	Python-Basics-Online Exercise	Python-Analytics	Chap. 1
	W6	22.5.	Data Understanding, Data Visualization I	24.5.	No lectures, but bonus tasks 1.) Co-Create your exam 2.) Earn bonus points for the exam		Chap. 2
	W7	29.5.	Data Visualization II	31.5.			
	W8	5.6.	Data Preparation	7.6.	Predictive Modeling I (10:00 -12:00)	BI-Project	Start
	W9	12.6.	Predictive Modeling II	14.6.	Python-Analytics-Online Exercise		
	W10	19.6.	Guest Lecture Dr. Ionescu	21.6.	Fitting a Model		
	W11	26.6.	How to avoid overfitting	28.6.	What is a good Model?		
Deepening	W12	3.7.	Project status update Evidence and Probabilities	5.7.	Similarity (and Clusters) From Machine to Deep Learning I		
	W13	10.7.		12.7.	From Machine to Deep Learning II		
	W14	17.7.	Project presentation	19.7.	Project presentation		End
Ref.					Klausur 1. Termin, 31.7. '24 Klausur 2. Termin, 2.10. '24	Projektbericht	



# Exercise – Information gain

$$-\sum p_i \log_2(p_i)$$

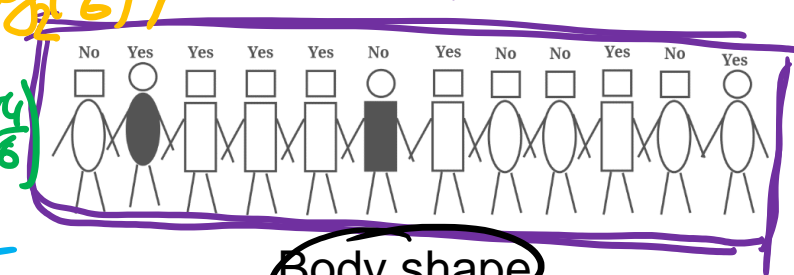
0,98

Freie Universität Berlin

10 Min.

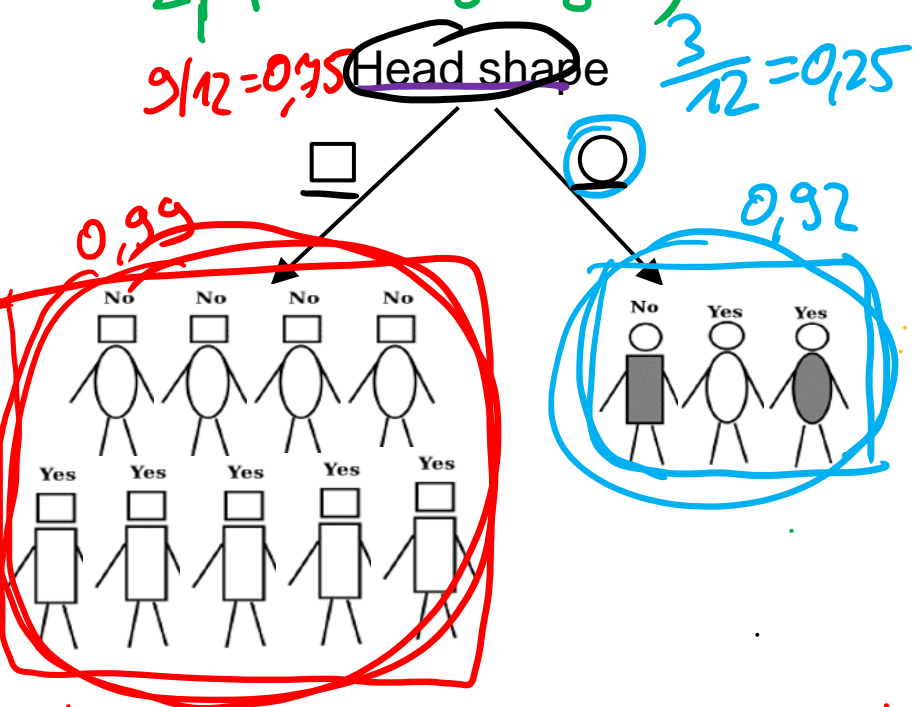
$$-\frac{4}{10} \log_2\left(\frac{4}{10}\right) - \frac{6}{10} \log_2\left(\frac{6}{10}\right)$$

Which attribute to choose?



Yes: 7  
No: 5

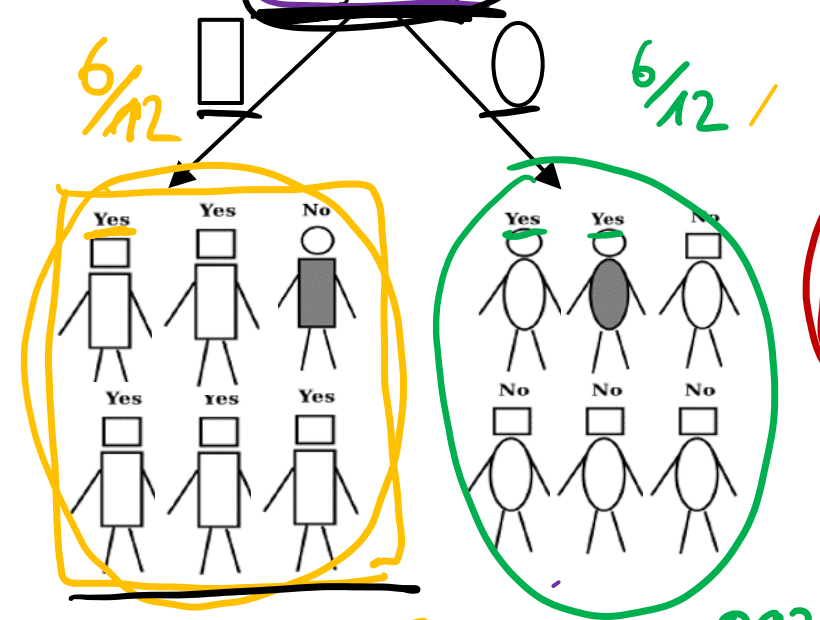
entropy(parent) ≈ 0,98



entropy(□) ≈ 0,99    entropy(○) ≈ 0,92  
p(□) ≈ 0,75    p(○) ≈ 0,25

IG ≈ 0,007

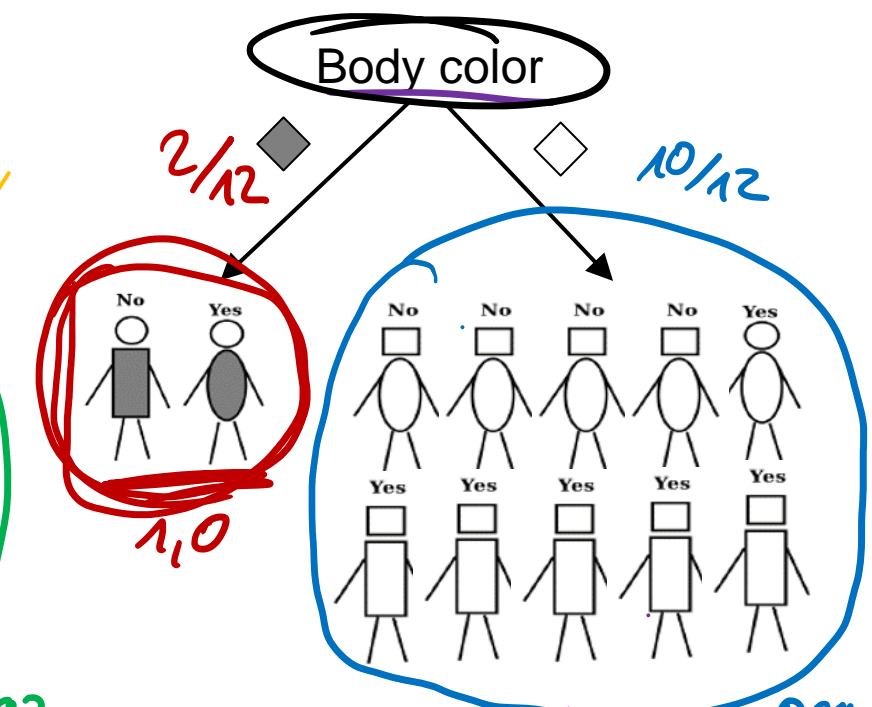
0,98 - 0,75 · 0,99 - 0,25 · 0,92



entropy(□) ≈ 0,65    entropy(○) ≈ 0,92  
p(□) ≈ 0,50    p(○) ≈ 0,50

IG ≈ 0,196

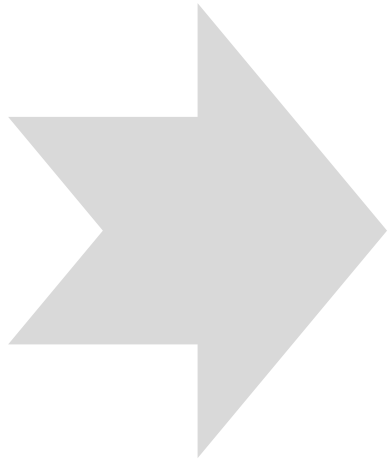
0,98 - 0,5 · 0,65 - 0,5 · 0,92



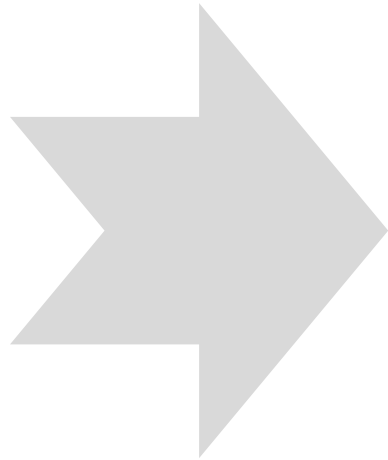
entropy(◆) ≈ 1,00    entropy(◇) ≈ 0,97  
p(◆) ≈ 0,17    p(◇) ≈ 0,83

IG ≈ 0,004

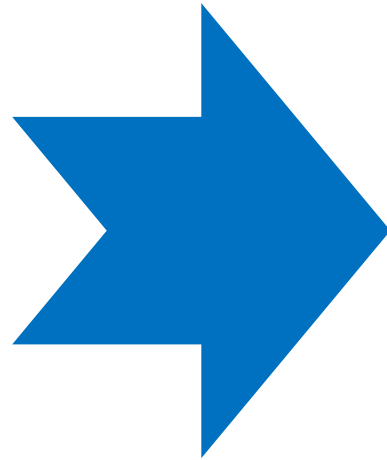
0,98 - 0,17 · 1,0 - 0,83 · 0,97



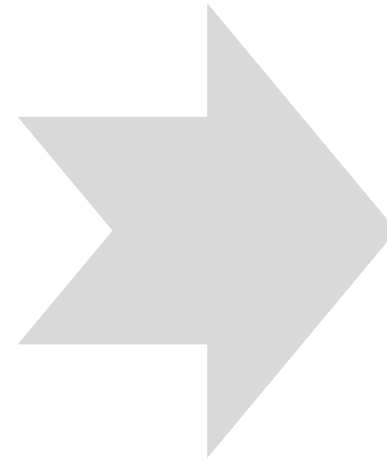
(1) Models and induction



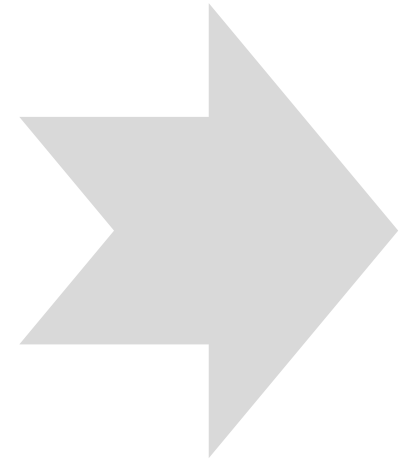
(2) Attribute Selection



(3a) Decision Trees  
Algorithmic View



(3b) Probability Estimation



(3c) Decision Tree Examples

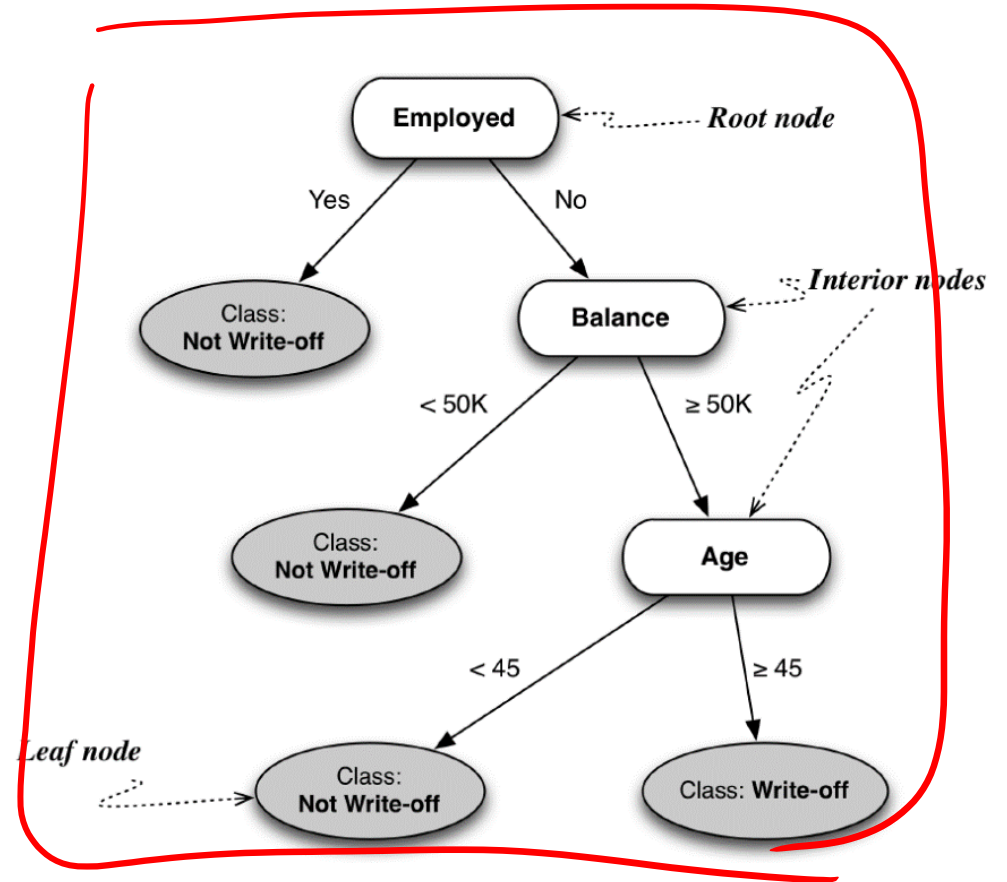


# Decision Trees

If we select multiple attributes each giving some information gain, it's not clear how to put them together → **decision trees**

The tree creates a segmentation of the data  
Each *node* in the tree contains a test of an attribute  
Each *path* eventually terminates at a *leaf*  
Each leaf corresponds to a *segment*,  
and the attributes and values along  
the path give the characteristics  
Each leaf contains a value for the  
target variable

Decision trees are often used as **predictive models**



→ **Prediction of target attribute, e.g.**  
Michi, 40,000, 24, yes, Write-off?  
Micki, 115,000, 40, no, Write-off?

# How to build a decision tree

if bodyShape = "Rectangular"  
 ^ bodyColor = "Black"  $\Rightarrow$  NO

Manually build the tree deductively, based on **expert knowledge**

- very time-consuming
- trees are sometimes corrupt (redundancy, contradictions, non-completeness, inefficient)

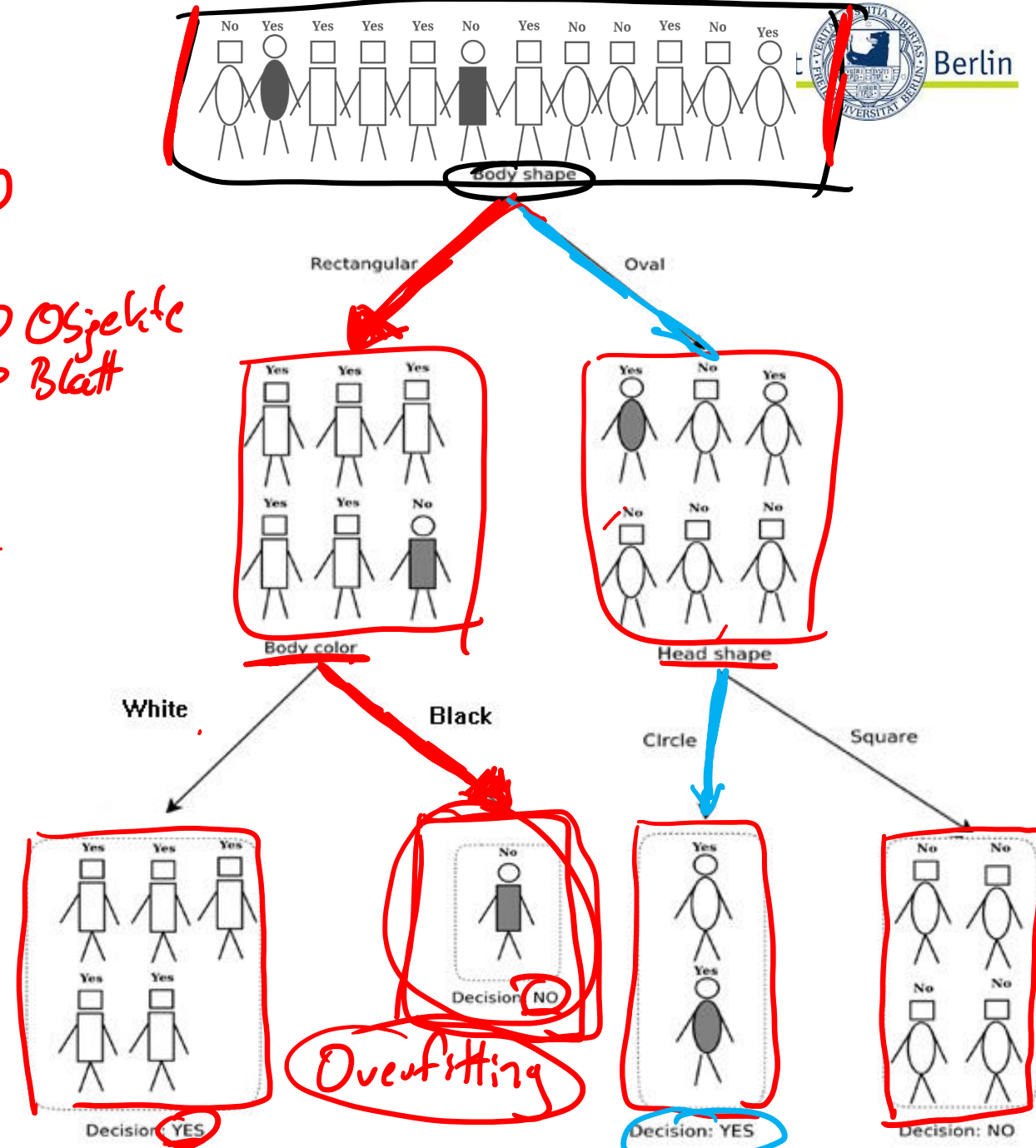
## Build the tree **automatically** by induction

- recursively partition the instances based on their attributes (divide-and-conquer)
- easy to understand
- relatively efficient

Recursively apply **attribute selection** to find the best attribute to partition the data set

The goal at each step is to select an attribute to partition the current group into subgroups that are as pure as possible w.r.t. the target variable

Ref.



# ID3 – an algorithm for tree induction

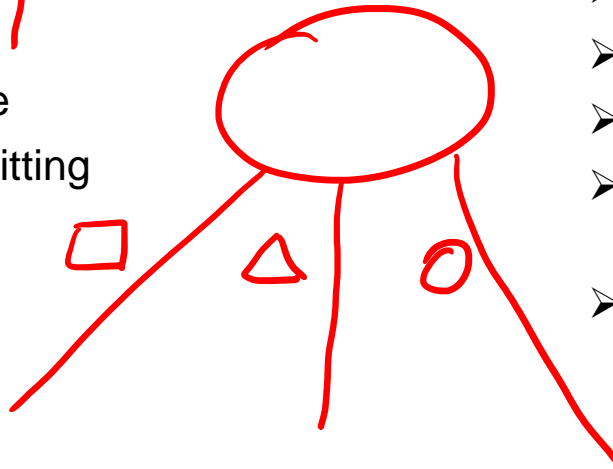
$\square, 0$   $\boxed{\square, 0, \triangle}$   $\square, 0, \triangle$   
 $[0 \dots 1,10] [1,10 \dots 1,40]$   
*Körpergröße?*  
 $[0 - 2,5m]$

**Iterative Dichotomiser (ID3)** is the most used machine learning algorithm in scientific literature and in commercial systems

ID3 encompasses a **top-down** decision tree building process

Basic step: split sets into disjoint subsets, where all the *individuals within a subset show the same value for a selected attribute*

- Invented by Ross Quinlan in the early 1980s
- Uses information gain as a measure for the quality of partitioning
- Requires categorical variables
- Has no features of handling noise
- Has no features of avoiding overfitting



- One attribute for testing is selected at the current node
- Testing separates the set into partitions
- For each partition, a subtree is built
- Subtree-building is terminated if all the samples in one partition belong to the same class
- Labels the tree leaves with the corresponding class



Ref.

Other algorithms, e.g. CART, CHAID, C4.5

We will use CART in the online-exercise.

(see <https://scikit-learn.org/stable/modules/tree.html#tree-algorithms>)



# ID3: algorithmic structure

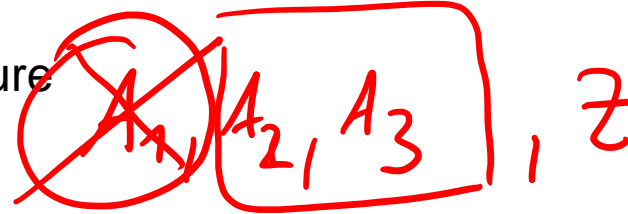
ID3



CART

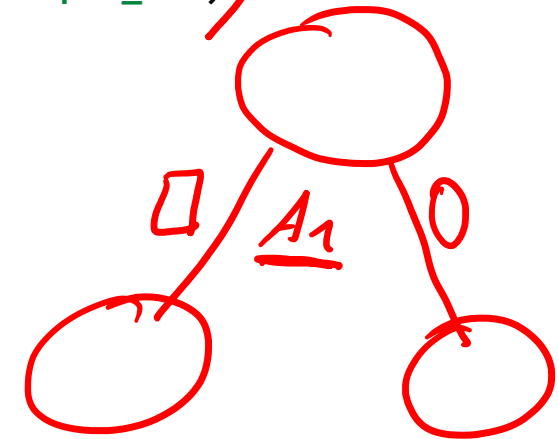
An **intelligent order of the tests** is the key feature of the ID3 algorithm (→ information gain)

*Recursive algorithmic structure of ID3:*



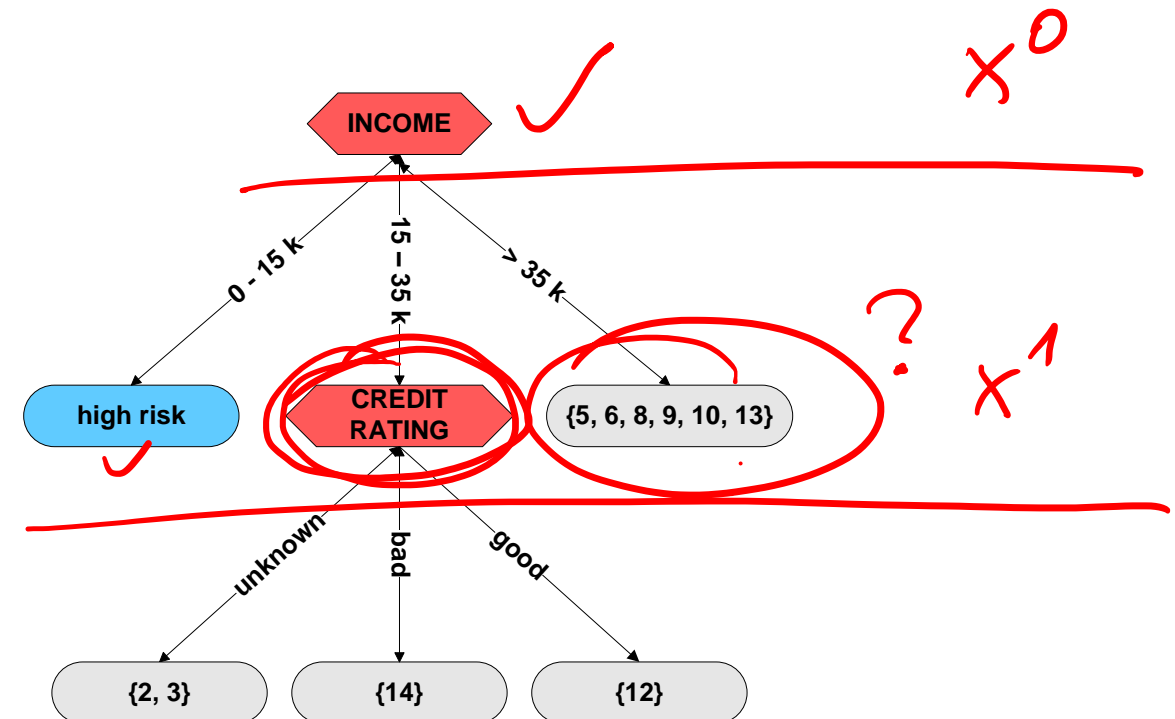
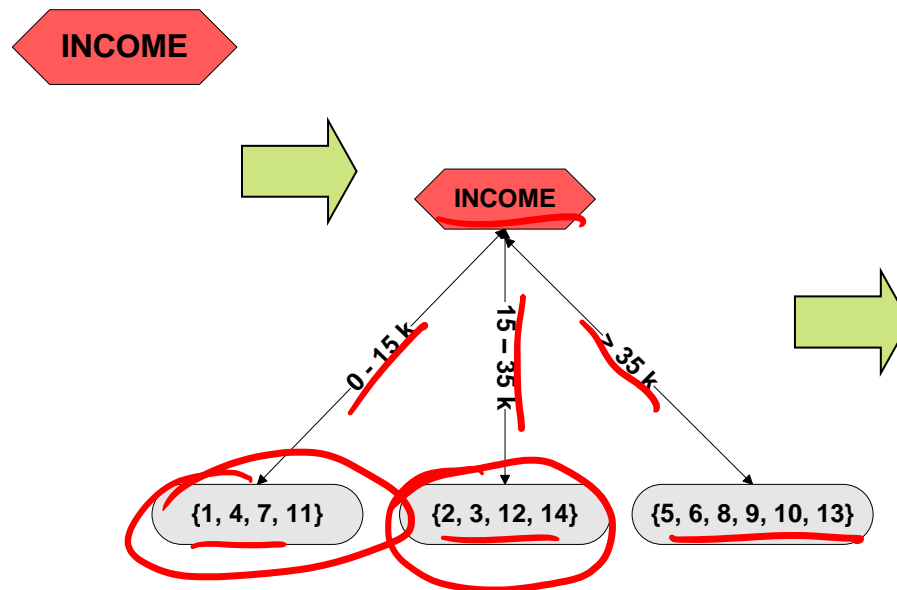
```
FUNCTION induce_tree(sample_set, attributes)
BEGIN
  IF (all the individuals from sample_set belong to the same class)
    RETURN (leave bearing the corresponding class label)
  ELSEIF (attributes is empty)
    RETURN (leave bearing a label of all the classes present in sample_set)
  ELSE
    1. select an attribute X_j SCORE FUNCTION
       make X_j the root of the current tree
       delete X_j from attributes
       FOR EACH (value V of X_j)
         construct a branch, labeled V
         V_partition = sample individuals for which X_j == V
         induce_tree(V_partition, attributes)
  END
```

Entropy bzw IG



# ID3: example steps

$X^0 = \{\text{INCOME}, \text{CREDIT RATING}, \text{SECURITIES}, \text{LEVEL OF DEBT}\}$   
→  $X^1 = \{\text{CREDIT RATING}, \text{SECURITIES}, \text{LEVEL OF DEBT}\}$  ←  
 $X^2 = \{\text{SECURITIES}, \text{LEVEL OF DEBT}\}$

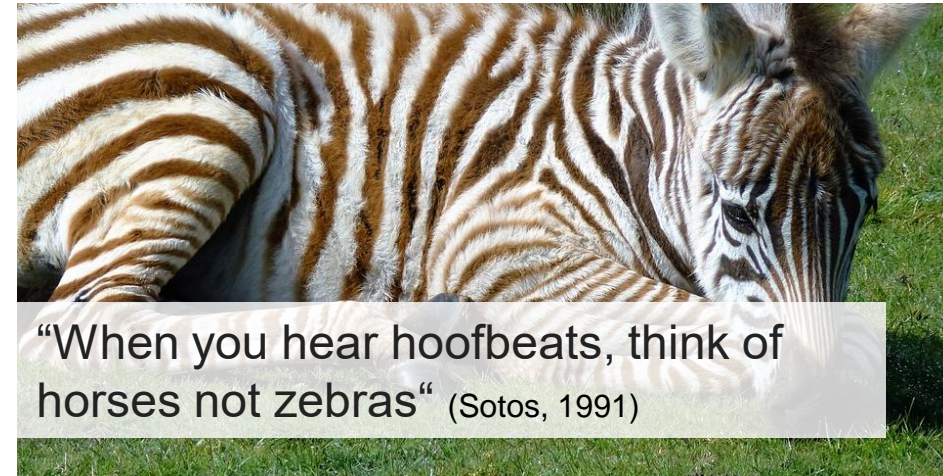


ID3 tries to build a tree, that

- classifies all the training samples correctly and
- provides a high probability for implying only a small number of tests when classifying an individual

Criterion for attribute selection:

- prefer attributes which contribute a **bigger amount of information** to the classification of an individual
- Information content is measured according to entropy

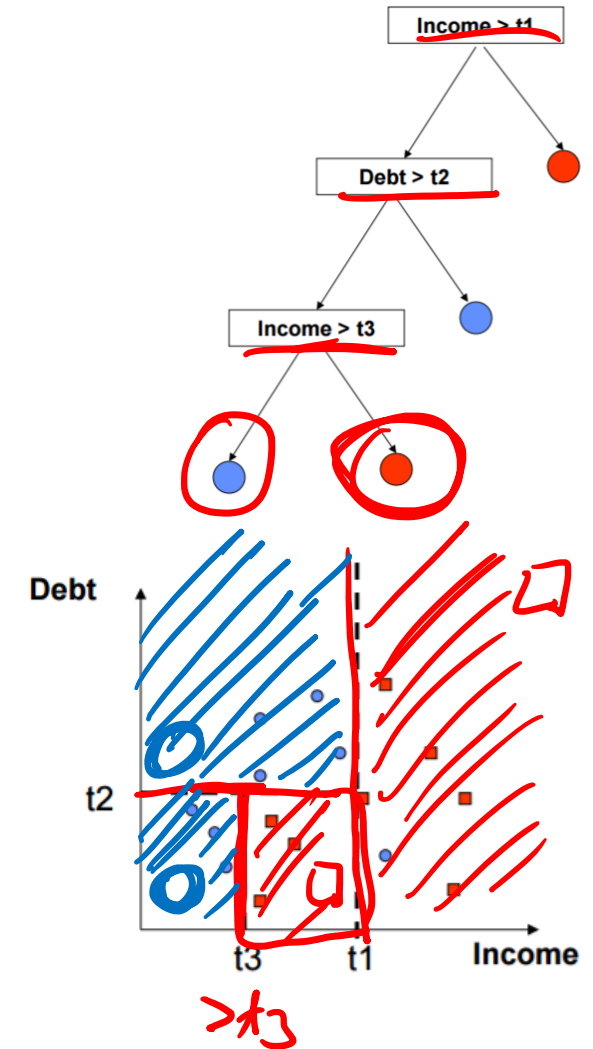
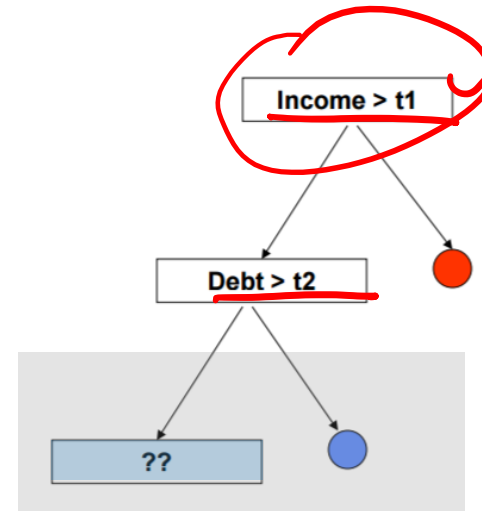
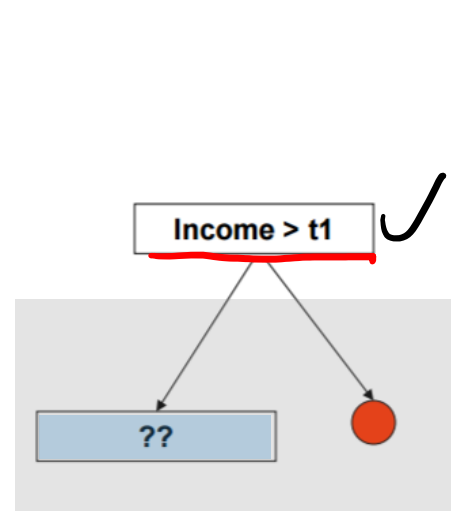


## **Occam's razor:**

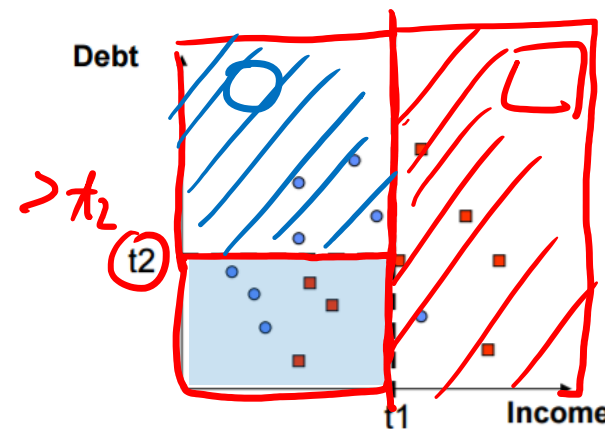
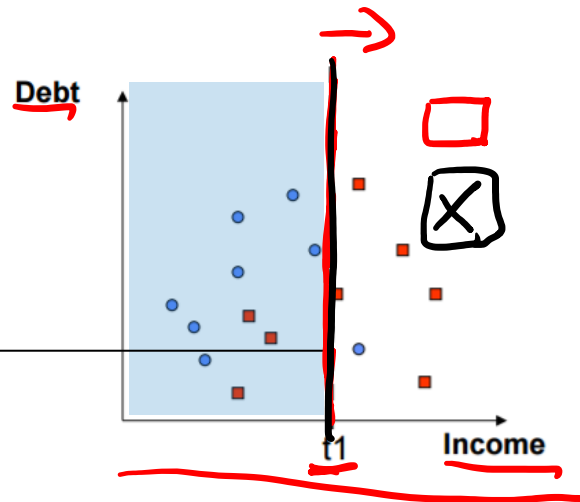
the simplest tree produces the smallest classification error rates, when applying the tree to new individuals

# Nonlinearity and Decision Surface

Example – Using two attributes to visualize segmentations



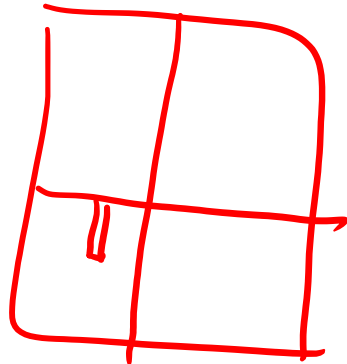
Tree boundaries  
are linear and axis-  
parallel



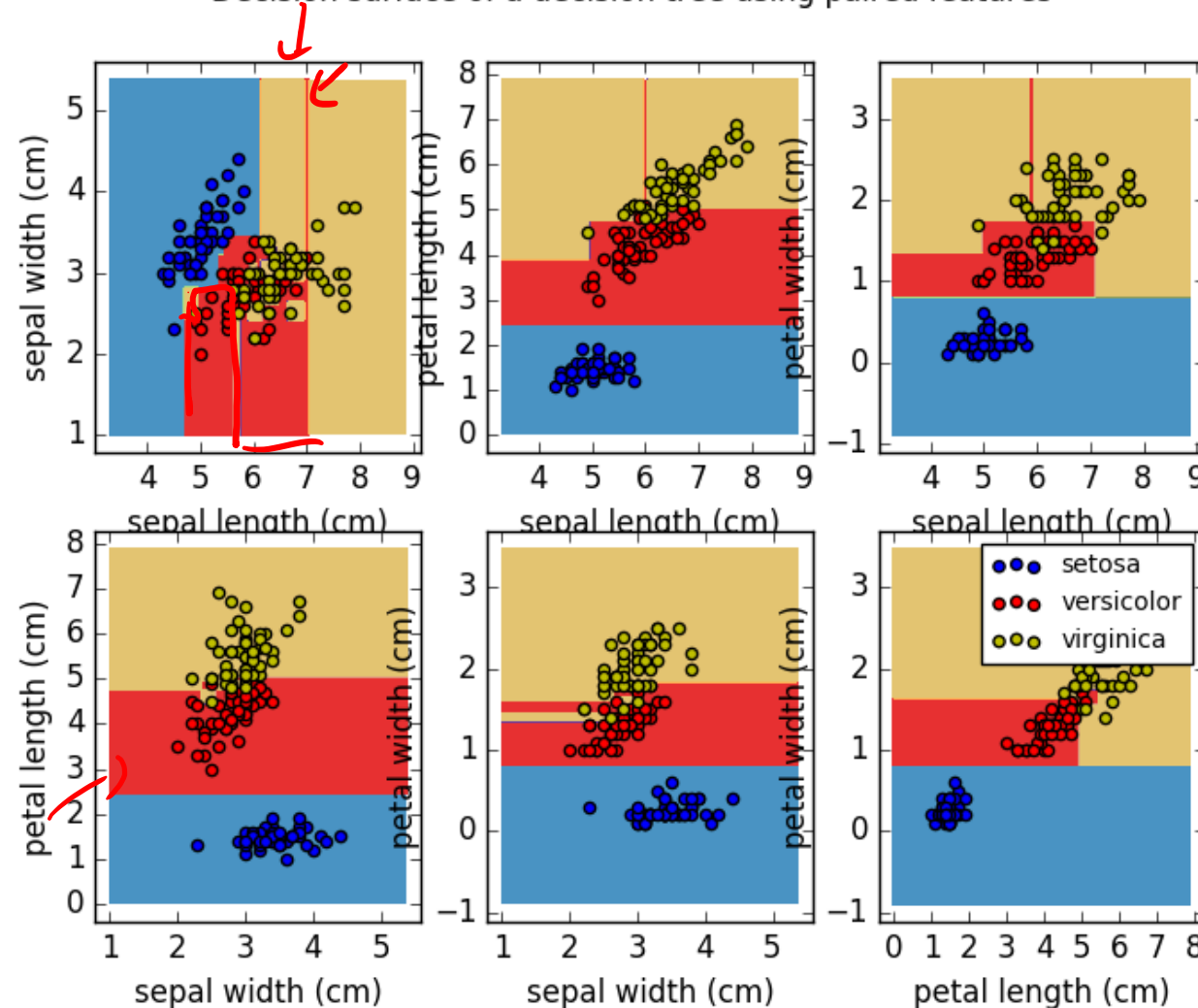
# Nonlinearity and Decision Surface

Example - Iris data

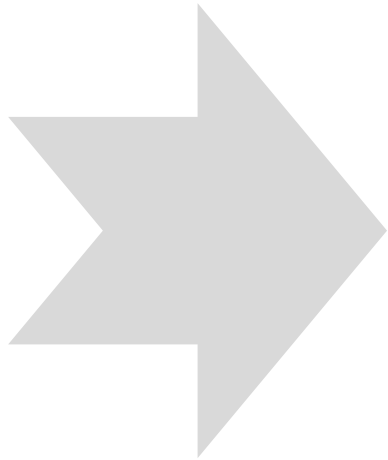
Decision Trees model  
non-linear relationships  
between attributes



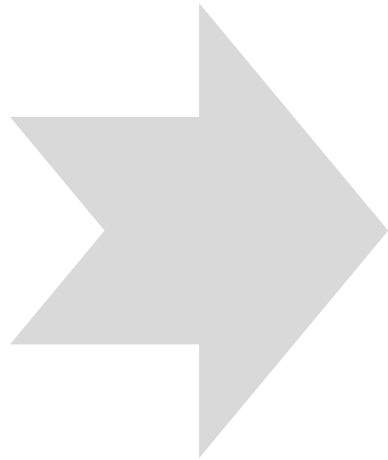
Decision surface of a decision tree using paired features



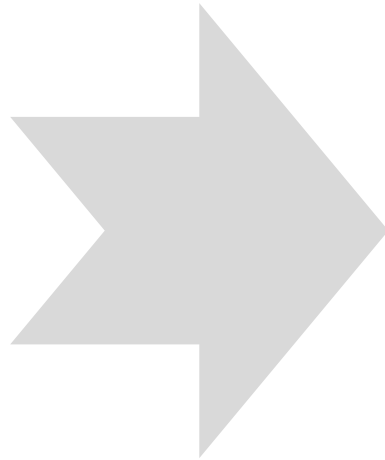




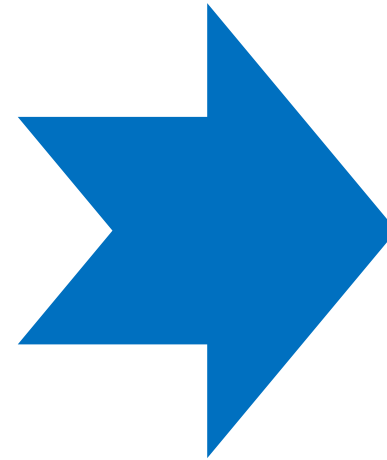
(1) Models and induction



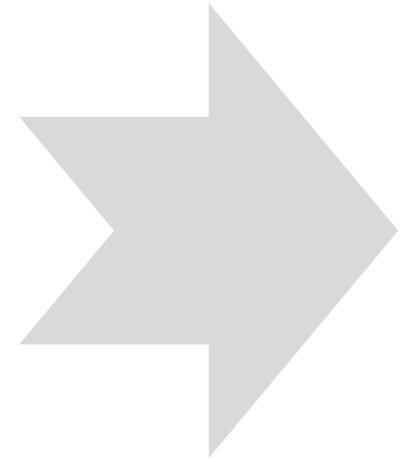
(2) Attribute selection



(3a) Decision Trees  
Algorithmic View



(3b) Probability Estimation



(3c) Decision Tree Examples

# Probability estimation (1/3)

We often need a **more informative prediction** than just a classification

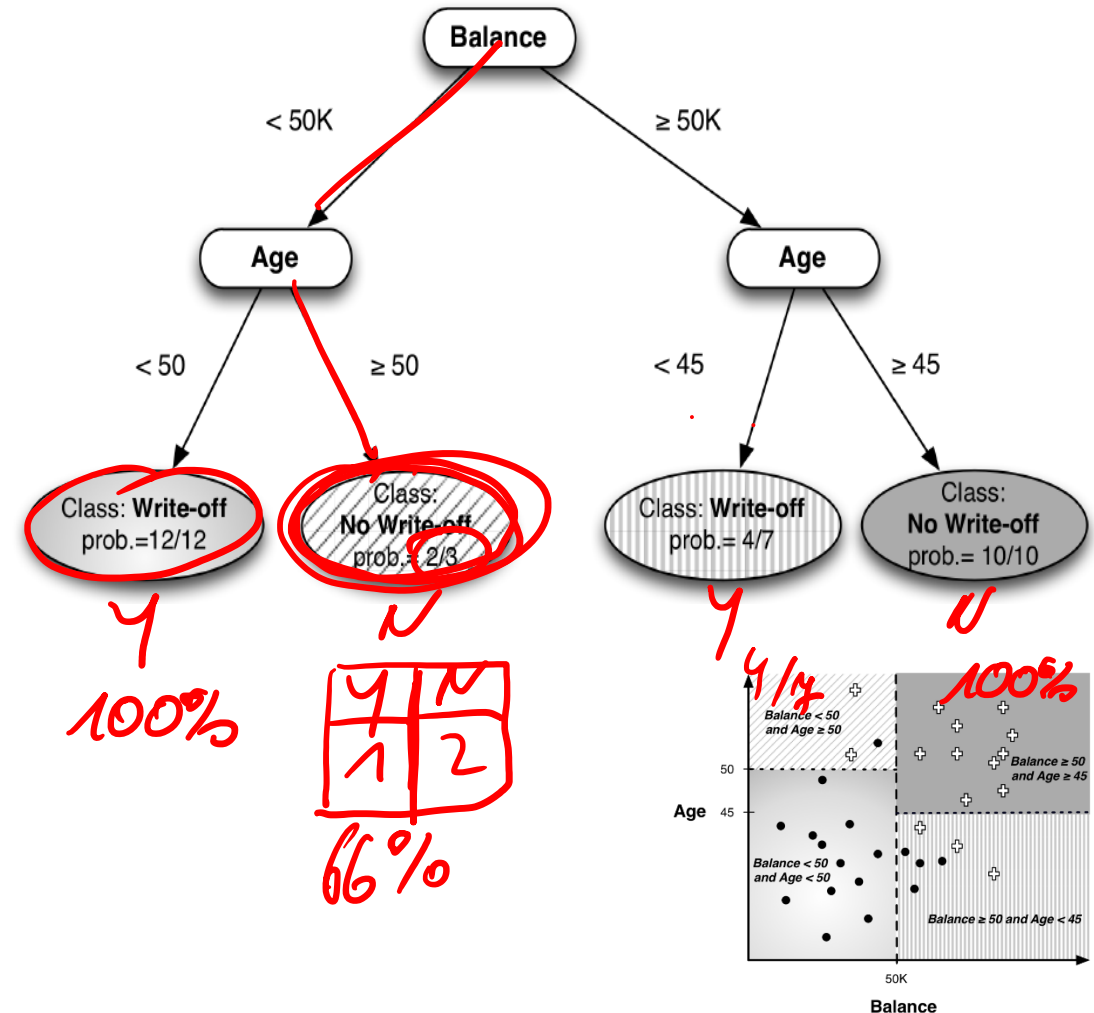
- E.g. allocate your budget to the instances with the highest expected loss
- More sophisticated decision-making process

Classification may oversimplify the problem

- E.g. if all segments have a probability of  $< 0.5$  for write-off, every leaf will be labeled „not write-off“
- We would like each segment (leaf) to be assigned an **estimate of the probability of membership** in the different classes

## Probability estimation tree

1# > min 3 Elemente



# Probability estimation (2/3)

Tree induction can easily produce probability estimation trees instead of simple classification trees

- **Instance counts** at each leaf provide class probability estimates
- **Frequency-based estimate** of class membership: if a leaf contains  $n$  positive and  $m$  negative instances, the probability of any new instance being positive may be estimated as  $n/(n + m)$ .

Approach may be too optimistic for segments with a very small number of instances ( $\rightarrow$  overfitting)

- Smoothed version of frequency-based estimate by **Laplace correction**, which moderates the influence of leaves with only a few instances:  $p(c) = \frac{n+1}{n+m+2}$  with  $n$  as number of instances that belong to class  $c$  and  $m$  instances not belonging to class  $c$

$\oplus$   
2  
100%

$\oplus$   
200  
sollte wichtiger sein

Example 1

$\oplus n \quad \ominus m$   
 $n=2 \quad \frac{n}{n+m}$

$n = 2$   
 $m = 0$   
 $p(c) = \frac{2}{2} = 1 \quad 100\%$   
 $p_{Laplace}(c) = \frac{2+1}{2+0+2} = 0,75 \quad 75\%$

Example 2

$n=20 \quad \frac{n}{n+m}$   
 $n = 20$   
 $m = 0$   
 $p(c) = \frac{20}{20+0} = 1 \quad 100\%$   
 $p_{Laplace}(c) = \frac{20+1}{20+0+2} = 0,95 \quad 95\%$

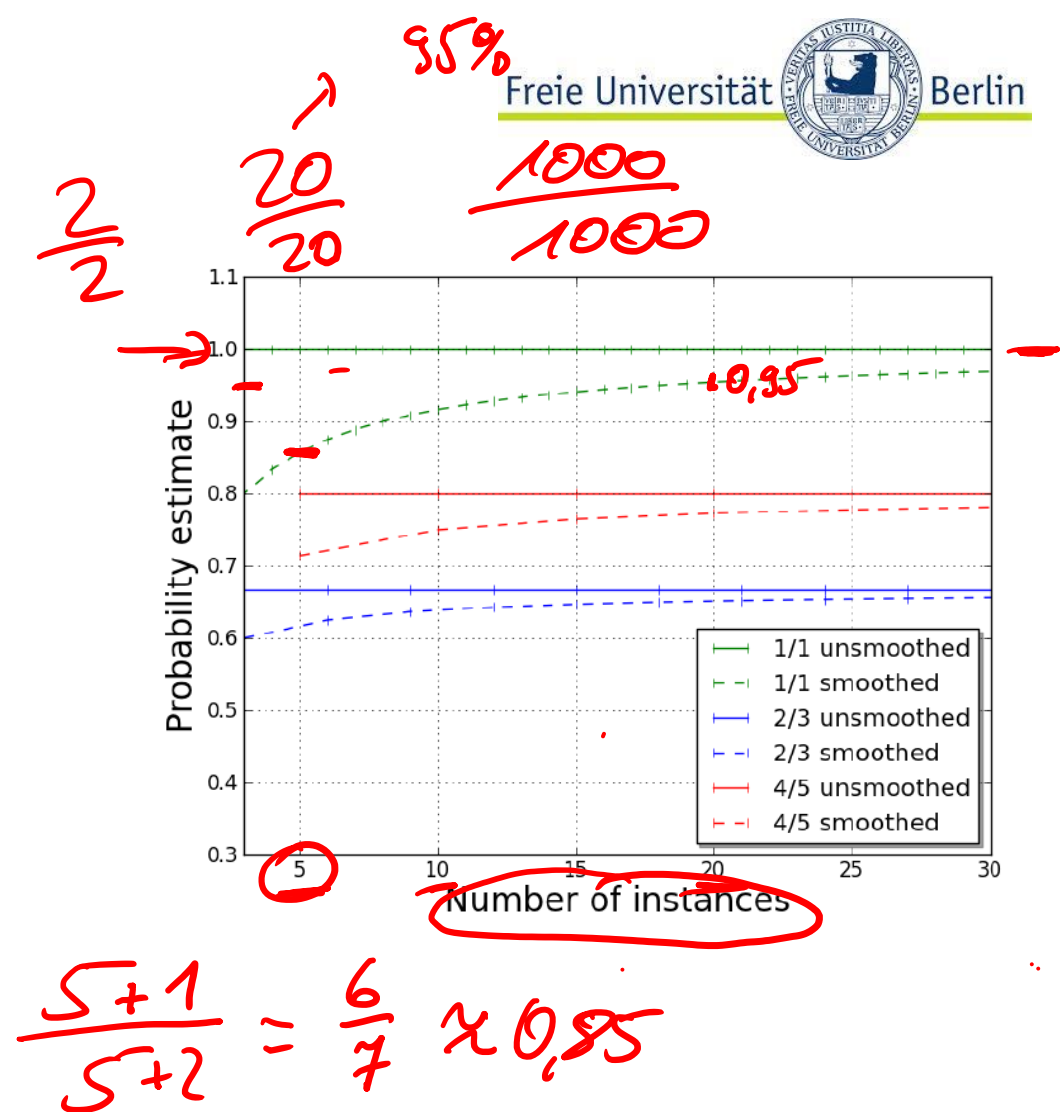
# Probability estimation (3/3)

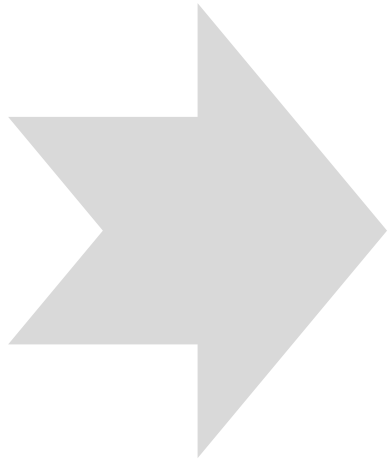
Effect of Laplace correction on several class ratios as the number of instances increases (2/3, 4/5, 1/1)

Example:

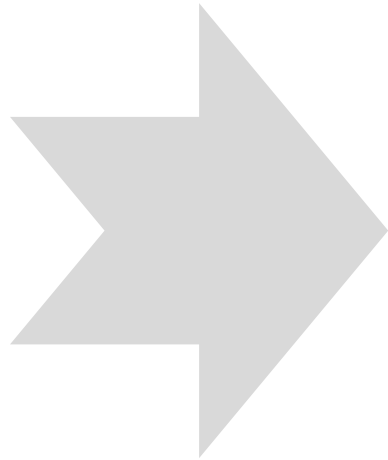
A leaf of the classification tree that has 2 pos. instances and no negative instances would produce the same f-b estimate ( $p = 1$ ) as a leaf node with 20 pos. and no negatives.

The Laplace correction smooths the estimate of the first leaf down to  $p = 0.75$  to reflect this uncertainty, but it has much less effect on the leaf with 20 instances ( $p \approx 0.95$ )

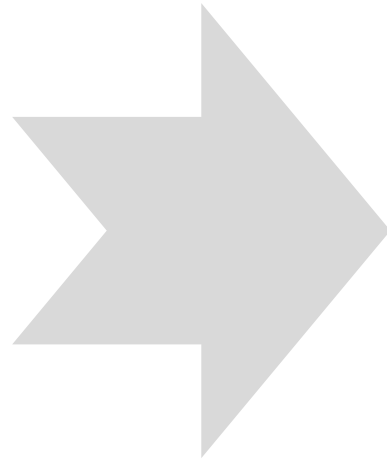




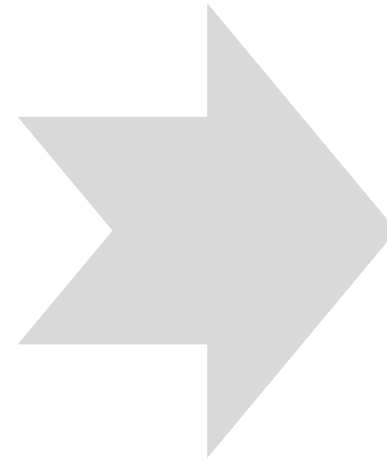
(1) Models and induction



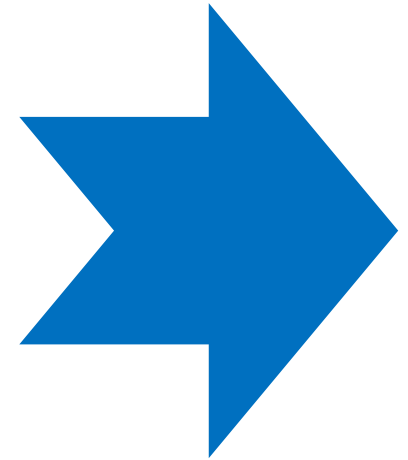
(2) Attribute selection



(3a) Decision Trees  
Algorithmic View



(3b) Probability Estimation



(3c) Decision Tree Examples



# Example - The Churn Problem

Solve the churn problem by tree induction

Historical data set of 20,000 customers

Each customer either had stayed with the company or left

Customers are described by the following variables:



Variable	Explanation
COLLEGE	Is the customer college educated?
INCOME	Annual <u>income</u>
OVERAGE	Average overcharges per month
LEFTOVER	Average number of leftover <u>minutes per month</u>
HOUSE	Estimated value of dwelling ( <u>from census tract</u> )
HANDSET_PRICE	Cost of phone
LONG_CALLS_PER_MONTH	Average number of long calls (15 mins or over) per month
AVERAGE_CALL_DURATION	Average duration of a call
REPORTED_SATISFACTION	Reported level of satisfaction
REPORTED_USAGE_LEVEL	Self-reported usage level
LEAVE ( <i>Target variable</i> )	<i>Did the customer stay or leave (churn)?</i>

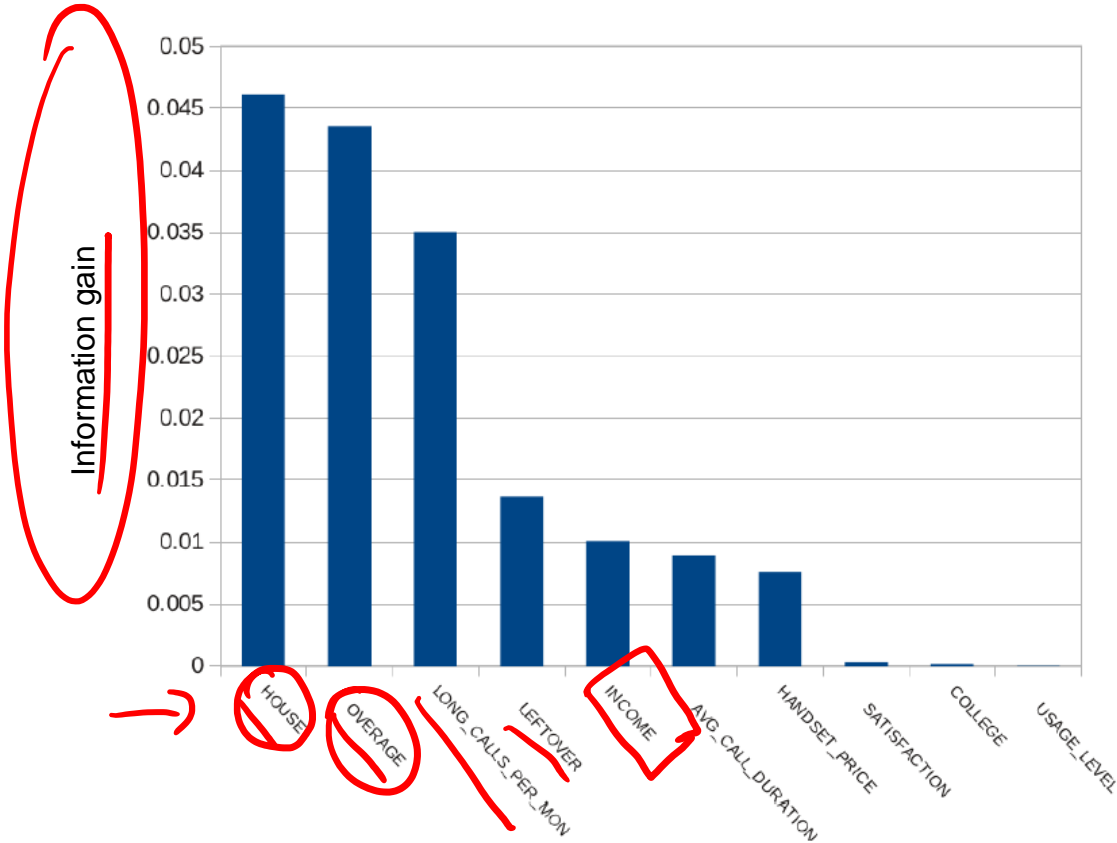
We want to use this data to **predict which new customers are going to churn.**

# Example - The Churn Problem

How good are each of these variables individually?

Measure the information gain of each variable

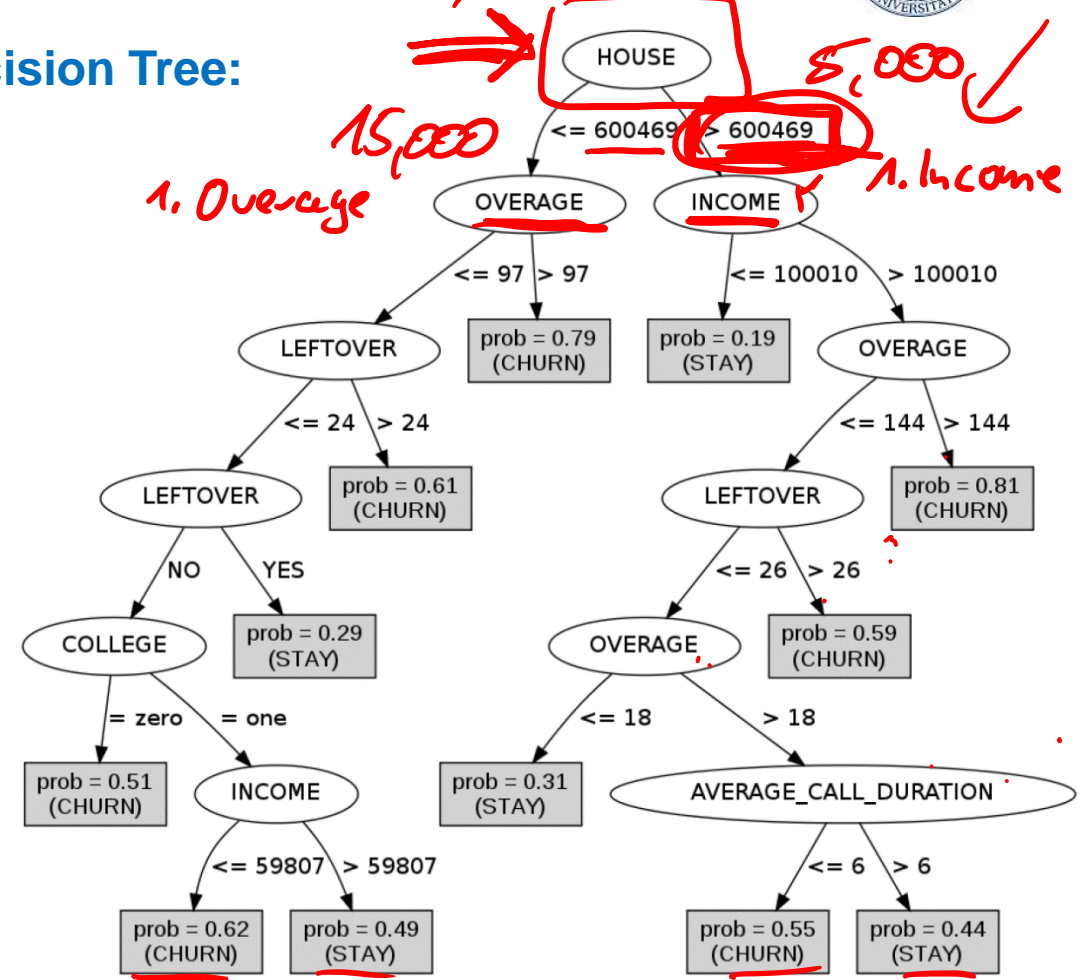
Compute information gain for each variable independently



The highest information gain feature (HOUSE) is at the root of the tree.

20,000

Decision Tree:



1. Why is the order of features chosen for the tree different from the ranking?
2. When to stop building the tree?
3. How do we know that this is a good model?

# Example - Decision Trees with Python

See [Python-Analytics](#) online exercise on Friday, 14.6.

If possible, prepare your system for the exercise:

**Achtung:** Die Pakete pandas und sklearn sind in der Regel schon vorinstalliert. Die Pakete graphviz und pydotplus müssen hingegen zunächst **außerhalb von Jupyter Notebooks** installiert werden, bevor man sie importieren kann. Öffne hierzu die Kommandozeile (Windows) oder das Terminal (Mac) und führe folgendes nacheinander aus:

```
1. conda install python-graphviz
2. conda install -c conda-forge pydotplus
```

Installationen direkt aus Jupyter Notebooks heraus oder mit pip führen in der Regel zu Fehlern!

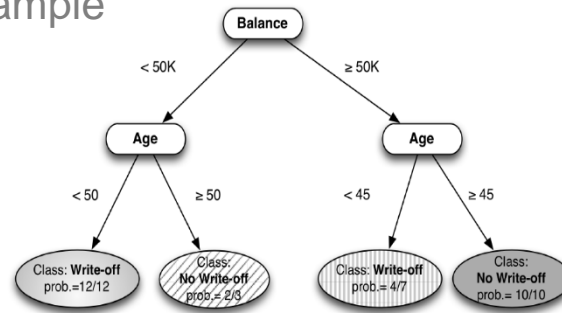
These components only need to be used for a small part of the exercise.

## Classification via Tree Induction

So far:

- ✓ we produced both the **structure of the model** (the particular tree model) and the **numeric parameters** of the model from the data

For example



Questions answered:

- ✓ How do we decide to classify data?
- ✓ Why do we not build „complete“ trees?
- ✓ If we have incomplete trees, we want to assess probabilities. What do we take into consideration?

Ref.

## Classification via Mathematical Functions

Now:

- We **specify the structure of the model**, but leave certain numeric parameters unspecified
- Data Mining calculates the best parameter values given a particular set of training data
- The form of the model and the attributes is specified
- The goal of DM is to tune the parameters so that the model fits the data as good as possible (**parameter learning**)

Simplifying assumptions:

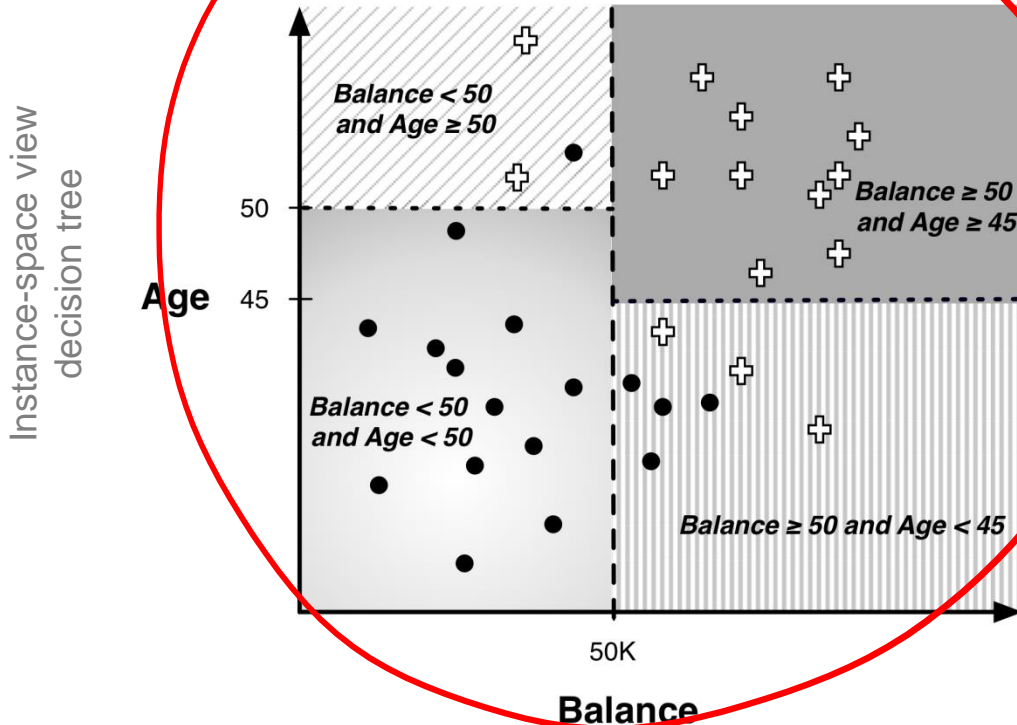
- For classification and class probability estimation, we will consider **only binary classes**.
- We assume that **all attributes are numeric**. (→ see data preparation)
- We **ignore the need to normalize numeric** measurements to a common scale (→ see data preparation)

# Linear classifiers

## Instance-space view:

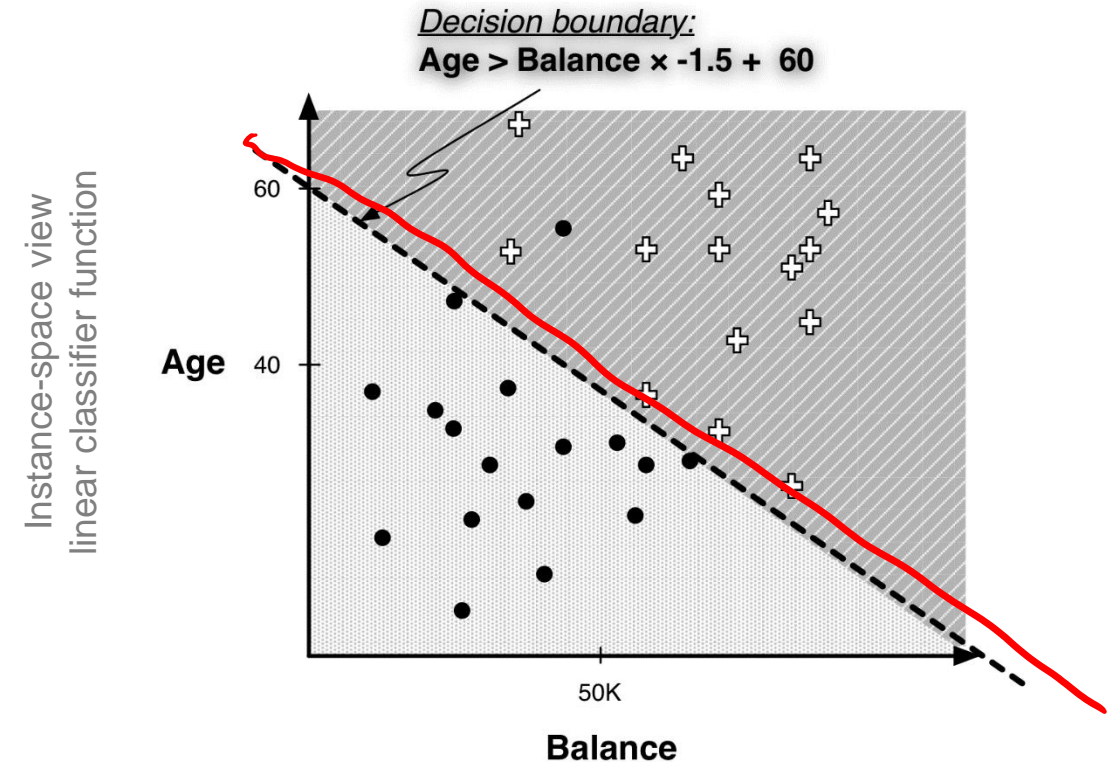
shows the space broken up into regions by decision boundaries

- Examples in each space should have similar values for the target variable
- Homogeneous regions help predicting the target variable of a new, unseen instance



We can separate the instance almost perfectly (by class) if we are allowed to introduce a boundary that is still a straight line, but is not perpendicular to the axes

- Linear classifier





## Fragen?

- ✓ Predictive modeling
  - ✓ Decision trees – Introduction
  - ✓ Decision trees – algorithmic view
  - ✓ Probability estimation tree
  - ✓ Decision trees – examples

# Recommended reading

Provost, F., Fawcett, T.	Data Science for Business Chapter 3
Berthold et al.	Guide to Intelligent Data Analysis Chapter 8.1
Hand, D.	Principles of Data Mining Chapter 10
Quinlan, J.R.	Induction of Decision Trees (in: Machine Learning, 1(1), p. 81-106, 1986)

- J. Bertin (1983) *Semiology of graphics: diagrams, networks, maps*. University of Wisconsin Press. Originally in French: *Semiologie Graphique*, 1967
- Cairo, A. (2012). *The Functional Art: An introduction to information graphics and visualization*. New Riders.
- Mertens, P., & Meier, M. (2009). *Integrierte Informationsverarbeitung*. Wiesbaden: Gabler.
- Woolman, M. (2002). *Digital information graphics*. Watson-Guptill Publications, Inc..