Institut für Kartographie und Geoinformatik | Leibniz Universität Hannover

Classification: Introduction, Information Gain, Decision Trees

Claus.Brenner@ikg.uni-hannover.de

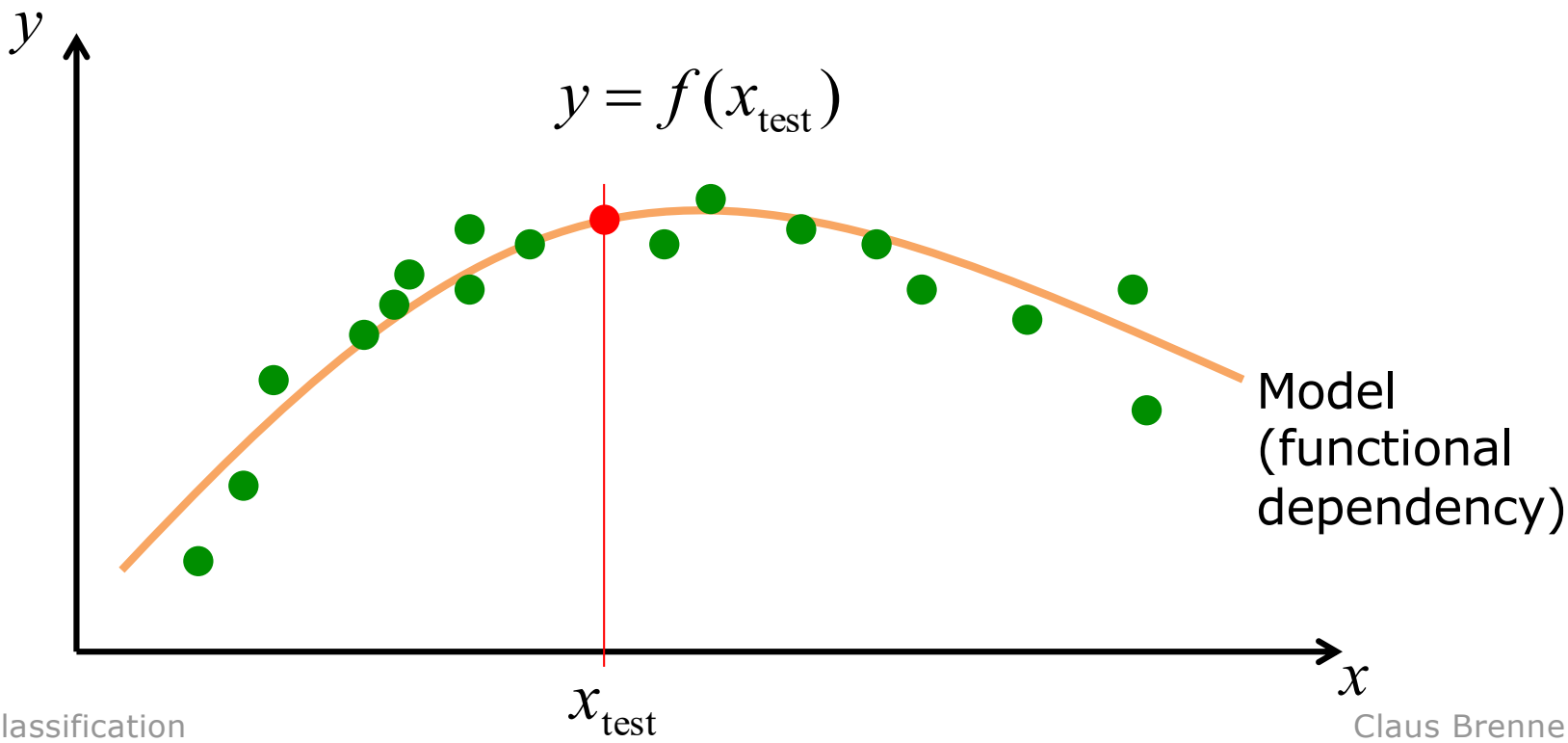
Leibniz Universität Hannover

Machine learning

- Task
  - Regression
  - Classification ←
  - Dimensionality reduction
- Two main cases
  - Supervised learning
  - Unsupervised learning

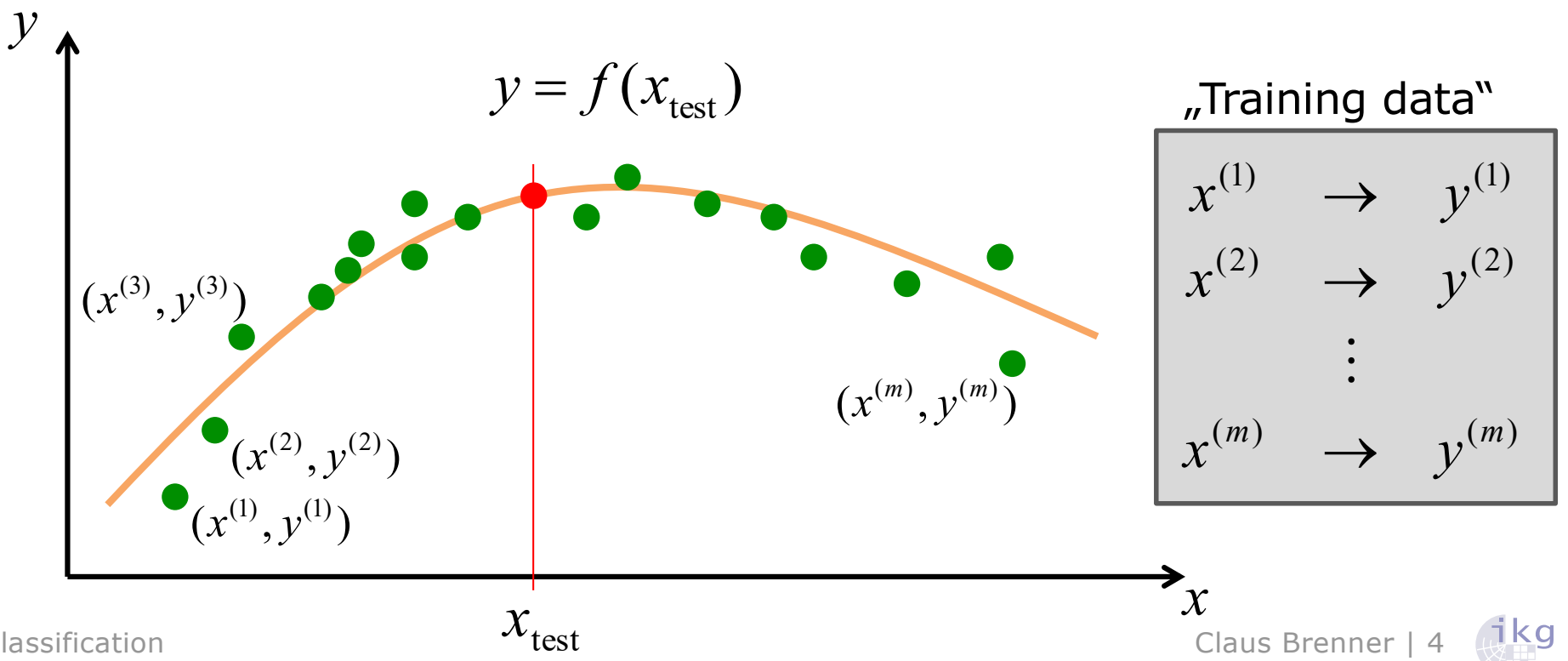
Regression

- A model (e.g. polynomial of n^{th} order: constant, line, parabola,…) shall be estimated
- After estimation, the obtained model can be used to predict values: $y = f(x_{\text{test}})$
- y is a **continuous** variable.



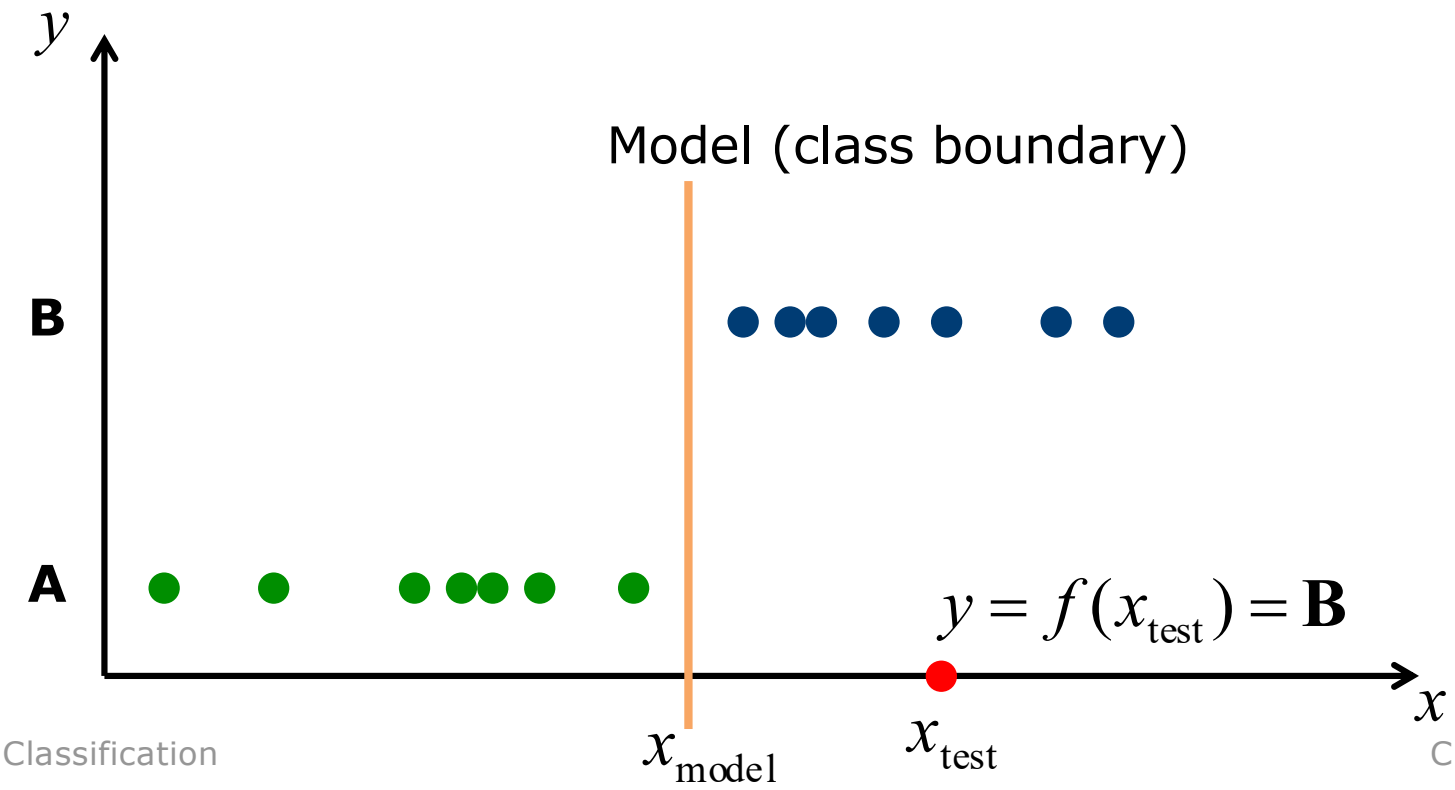
Regression

- The model describes the relationship between the *features* x and the *output* y
- The pairs $(x^{(i)}, y^{(i)})$, which are used to estimate the function can be considered as "training data".



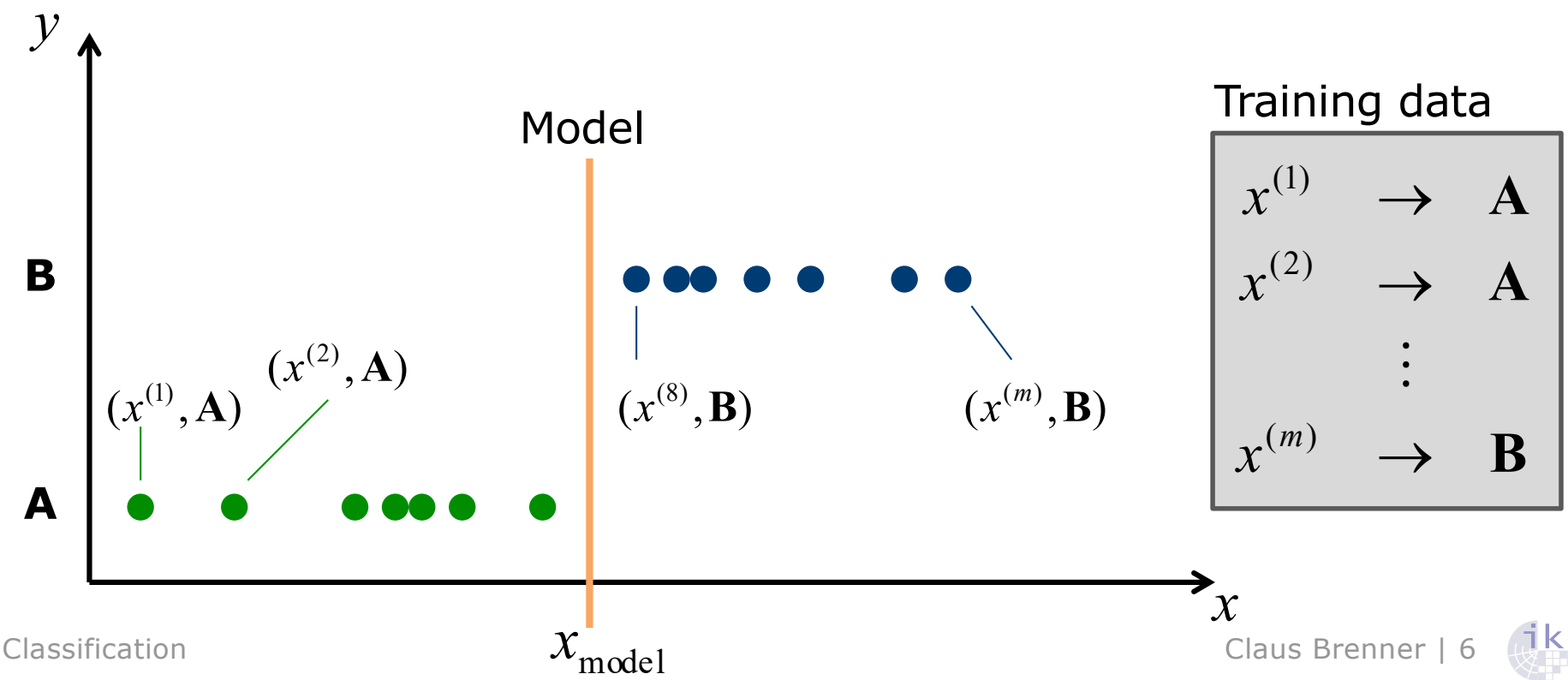
Classification

- A model shall be estimated which describes the membership in a class
- After estimation, the model can be used to predict class membership for a test value x_{test}
- y is a **discrete** variable: {0, 1}, {A, B}, {sun, rain}, …



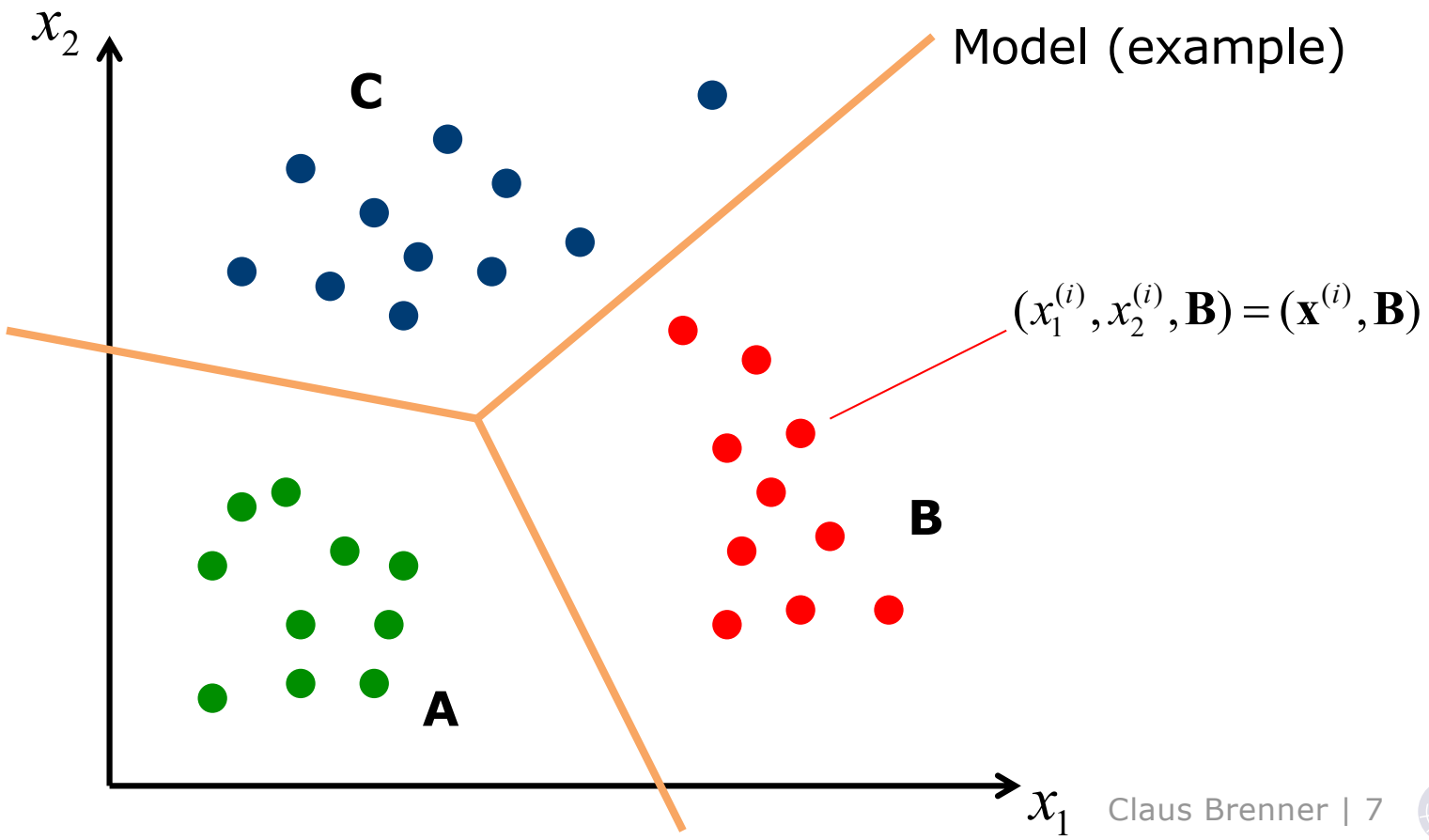
Classification

- The model describes the relationship between the *features* x and the *class membership* y
- The pairs $(x^{(i)}, y^{(i)})$ are the training data.



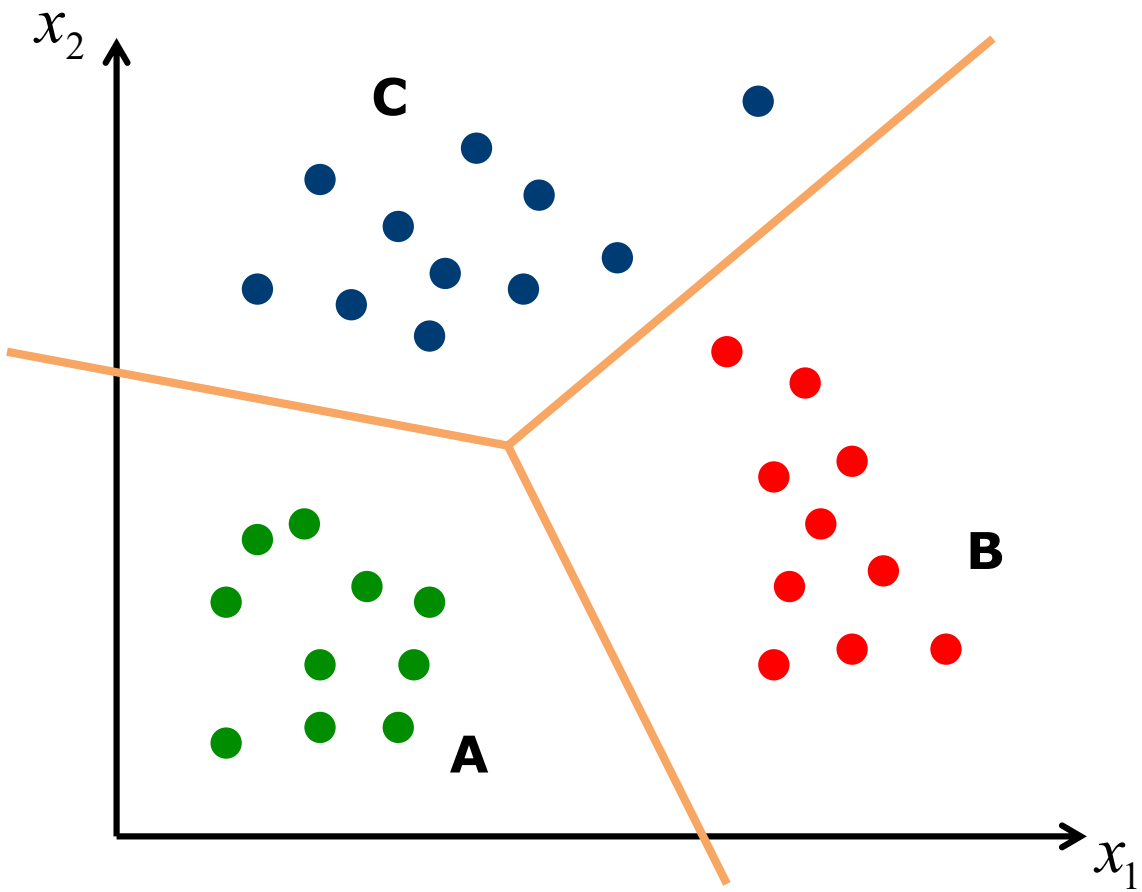
Classification

- x usually has a **high dimension**: *feature* (or *attribute*) *vector*
- There may be two or **more classes** (for y)
- Example: x two-dimensional, three classes



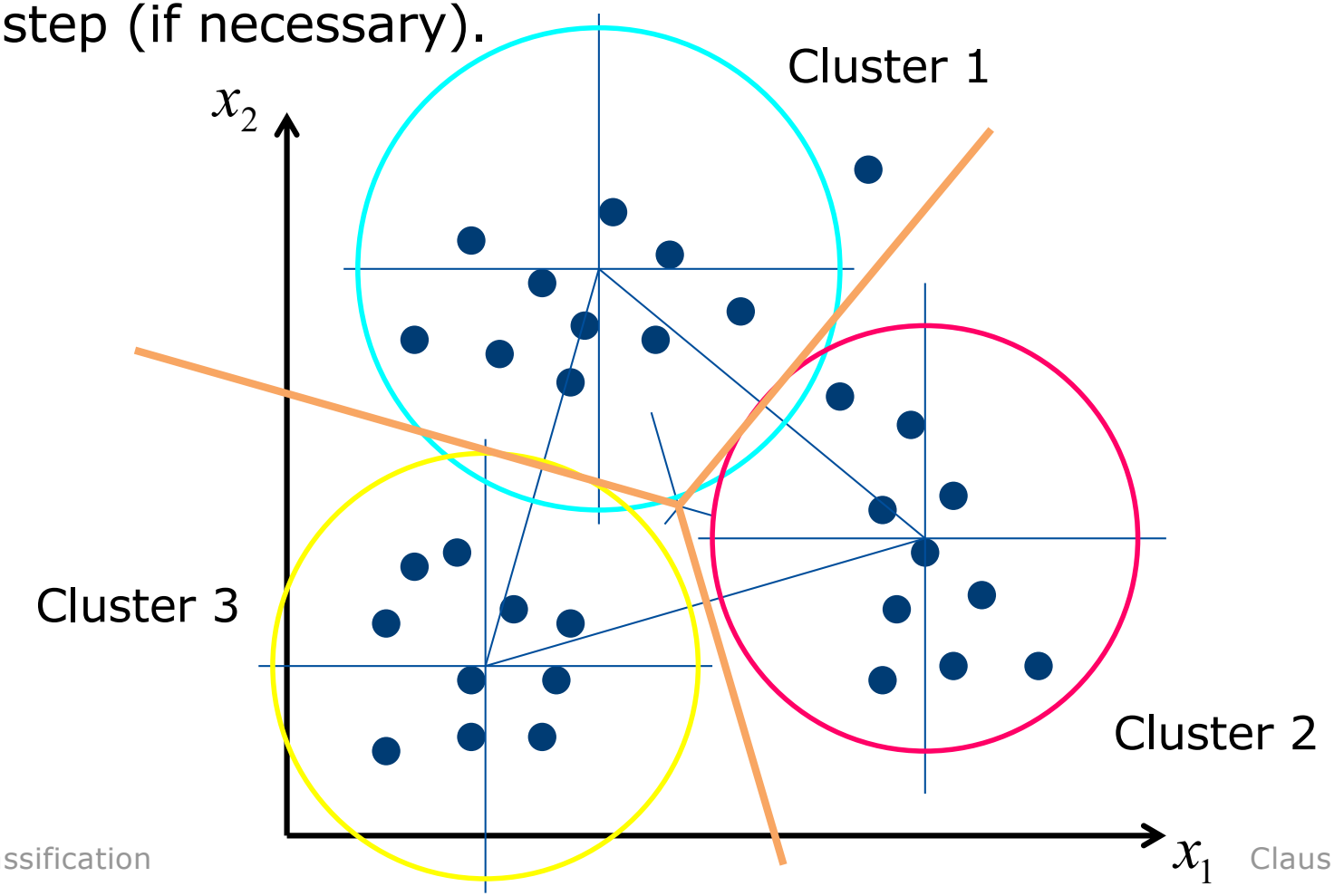
Classification: supervised vs. unsupervised

- In supervised training, the assignment of training examples to classes is known



Classification: supervised vs. unsupervised

- In unsupervised training, this assignment is not known
- Then, the goal is to identify clusters (accumulations)
- The "meaning" of those clusters has to be found in an additional step (if necessary).



Example of a classification task

- Under which conditions can a certain (not further specified) game be played?

No.	outlook Nominal	temperature Numeric	humidity Numeric	windy Nominal	**play** Nominal
1	sunny	85.0	85.0	FALSE	no
2	sunny	80.0	90.0	TRUE	no
3	overcast	83.0	86.0	FALSE	yes
4	rainy	70.0	96.0	FALSE	yes
5	rainy	68.0	80.0	FALSE	yes
6	rainy	65.0	70.0	TRUE	no
7	overcast	64.0	65.0	TRUE	yes
8	sunny	72.0	95.0	FALSE	no
9	sunny	69.0	70.0	FALSE	yes
10	rainy	75.0	80.0	FALSE	yes
11	sunny	75.0	70.0	TRUE	yes
12	overcast	72.0	90.0	TRUE	yes
13	overcast	81.0	75.0	FALSE	yes
14	rainy	71.0	91.0	TRUE	no

Instance, example

$$(\underbrace{x_1^{(14)}, \quad x_2^{(14)}, \quad x_3^{(14)}, x_4^{(14)}}, \quad y^{(14)})$$

Attributes (features)

(Source dataset: WEKA, http://www.cs.waikato.ac.nz/ml/index.html)

Example of a classification task

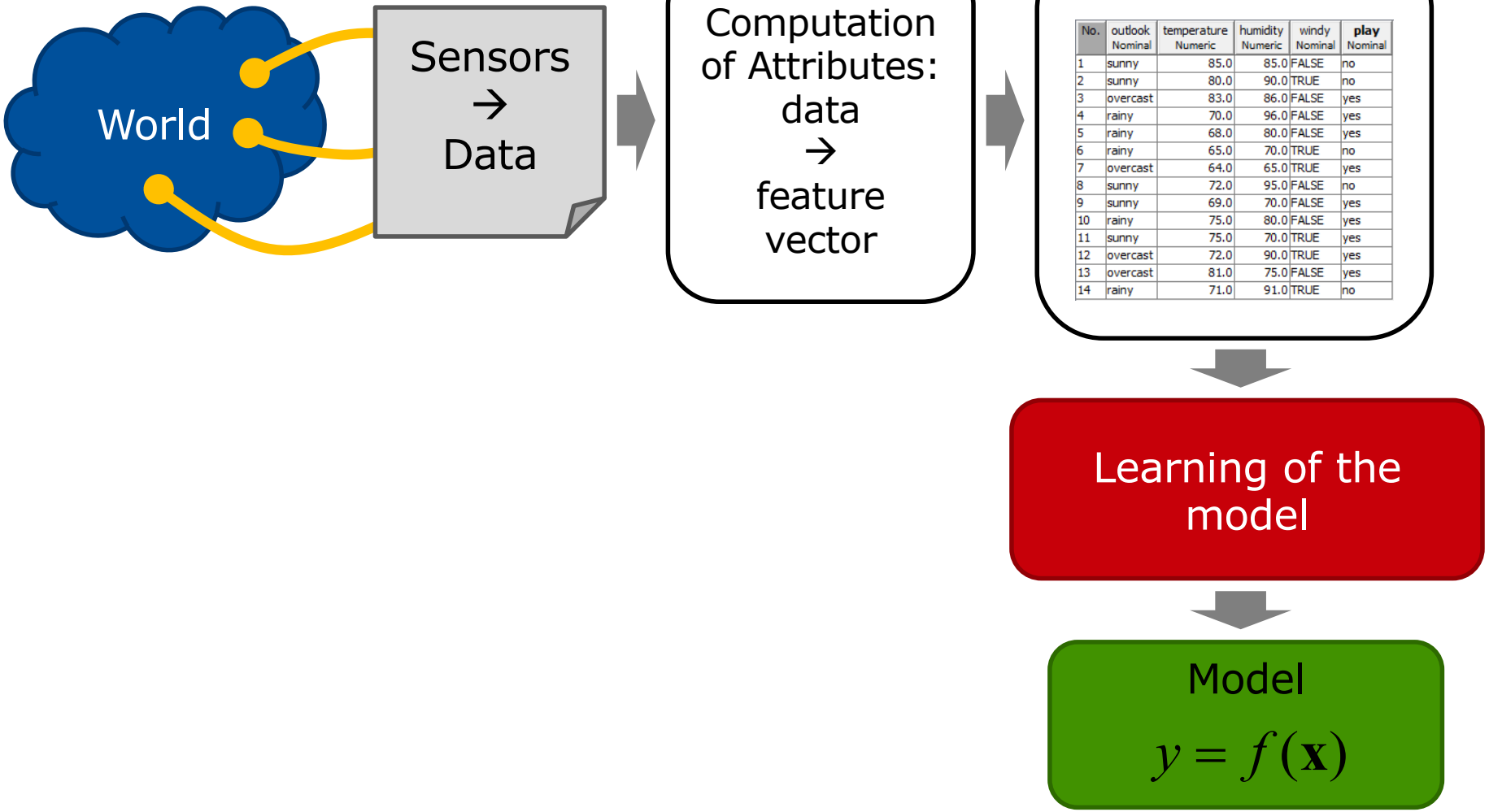
No.	outlook Nominal	temperature Numeric	humidity Numeric	windy Nominal	**play** Nominal
1	sunny	85.0	85.0	FALSE	no
2	sunny	80.0	90.0	TRUE	no
3	overcast	83.0	86.0	FALSE	yes
4	rainy	70.0	96.0	FALSE	yes
5	rainy	68.0	80.0	FALSE	yes
6	rainy	65.0	70.0	TRUE	no
7	overcast	64.0	65.0	TRUE	yes
8	sunny	72.0	95.0	FALSE	no
9	sunny	69.0	70.0	FALSE	yes
10	rainy	75.0	80.0	FALSE	yes
11	sunny	75.0	70.0	TRUE	yes
12	overcast	72.0	90.0	TRUE	yes
13	overcast	81.0	75.0	FALSE	yes
14	rainy	71.0	91.0	TRUE	no

(Source dataset: WEKA, http://www.cs.waikato.ac.nz/ml/index.html)

Nominal attribute (discrete values): {sunny, overcast, rainy}

Numeric attribute (continuous values): e.g. 0.0…100.0

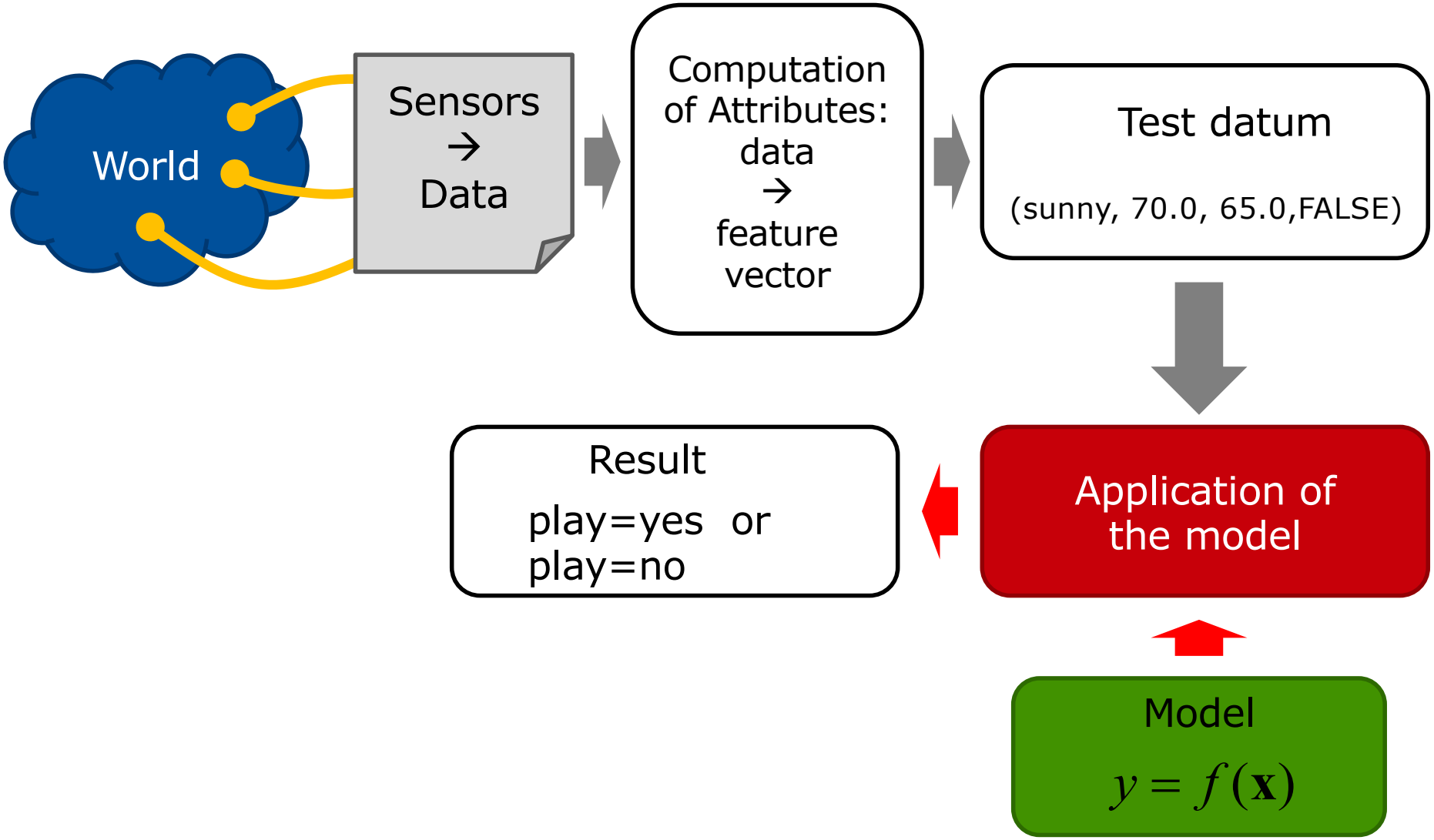
Typical approach in classification
Part 1: training phase



No.	outlook Nominal	temperature Numeric	humidity Numeric	windy Nominal	play Nominal
1	sunny	85.0	85.0	FALSE	no
2	sunny	80.0	90.0	TRUE	no
3	overcast	83.0	86.0	FALSE	yes
4	rainy	70.0	96.0	FALSE	yes
5	rainy	68.0	80.0	FALSE	yes
6	rainy	65.0	70.0	TRUE	no
7	overcast	64.0	65.0	TRUE	yes
8	sunny	72.0	95.0	FALSE	no
9	sunny	69.0	70.0	FALSE	yes
10	rainy	75.0	80.0	FALSE	yes
11	sunny	75.0	70.0	TRUE	yes
12	overcast	72.0	90.0	TRUE	yes
13	overcast	81.0	75.0	FALSE	yes
14	rainy	71.0	91.0	TRUE	no

(Source dataset: WEKA, http://www.cs.waikato.ac.nz/ml/index.html)

Typical approach in classification
Part 2: test phase (or, usage)



Underfitting vs. overfitting

- Hard: Select the "correct" complexity of the model
- In general: **model selection problem**

- Underfitting:
  - The model is too coarse to represent the training data (it is "too simple")
  - There are **large errors in training** and **large errors in testing**

- Overfitting:
  - The model is a too detailed representation of the training data (it is "too complex"), it does not generalize well
  - There are **small errors in training**, but **large errors in testing**

Underfitting vs. overfitting: regression

- Regression example: given the training data (blue dots)
- (They were obtained from the green sine curve by adding noise.)

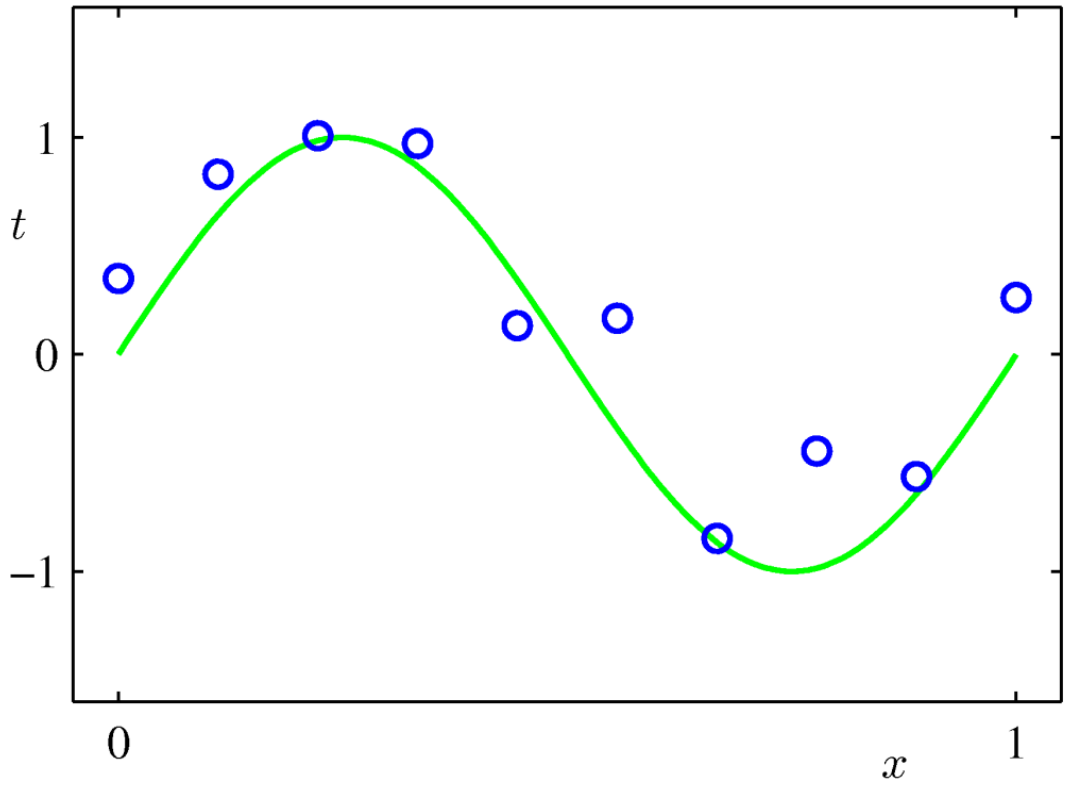
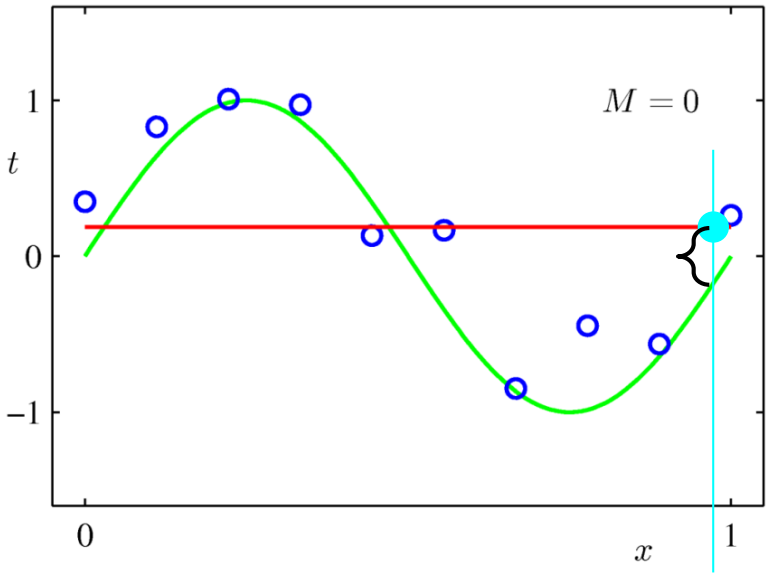
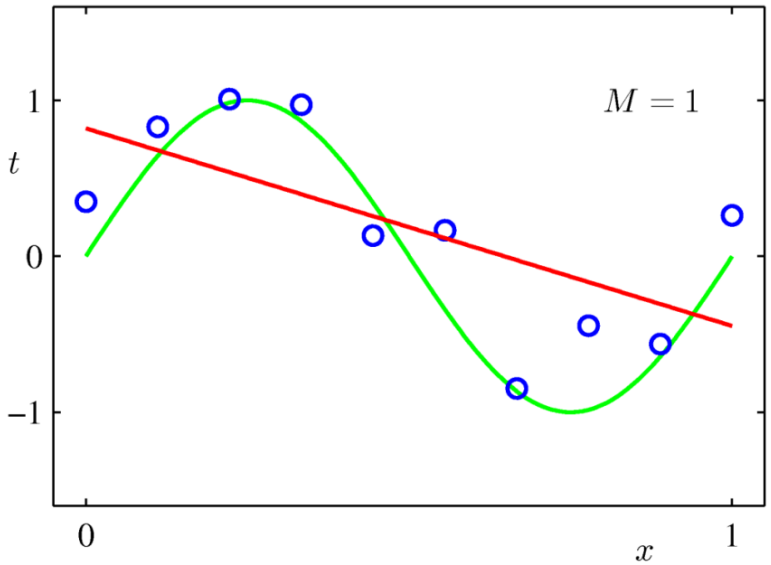


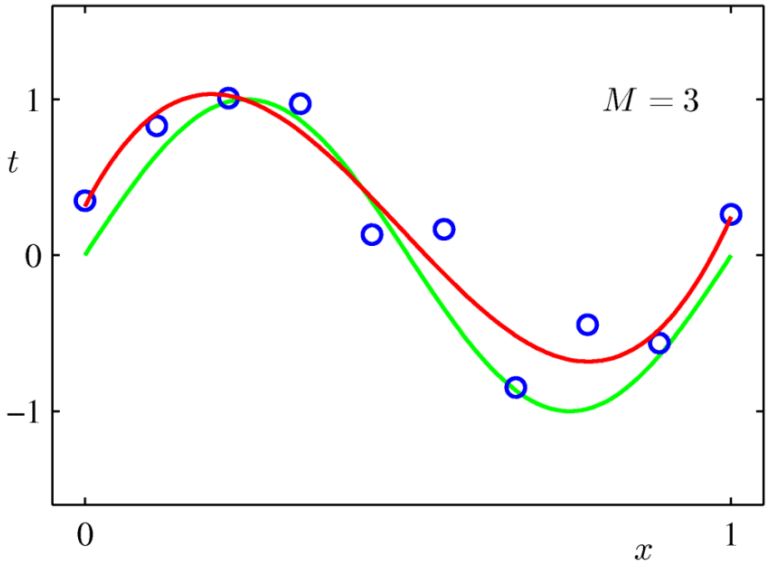
Image source: Christopher M. Bishop, Pattern Recognition and Machine Learning, Springer 2007

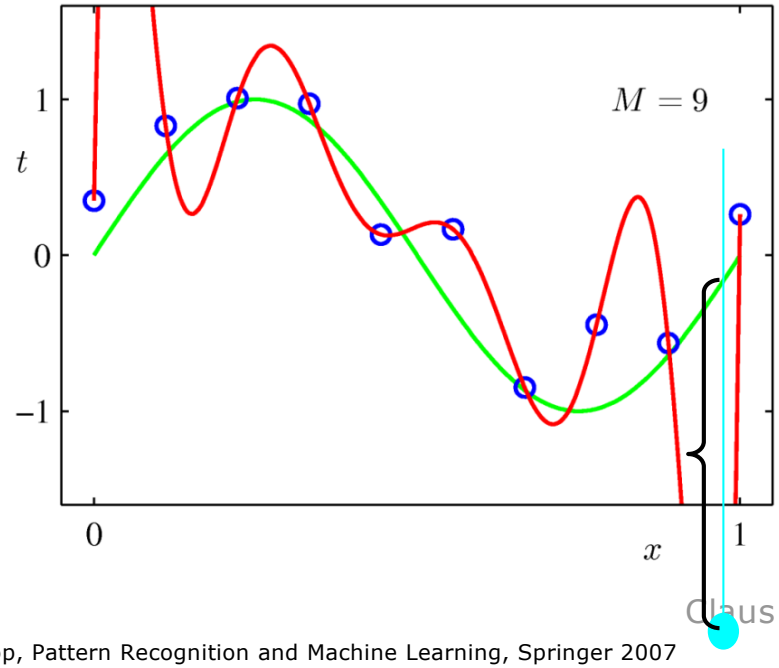
Underfitting vs. overfitting: regression





M = degree of polynomial: „model complexity“





Test datum

Image source: Christopher M. Bishop, Pattern Recognition and Machine Learning, Springer 2007

Underfitting vs. overfitting: regression

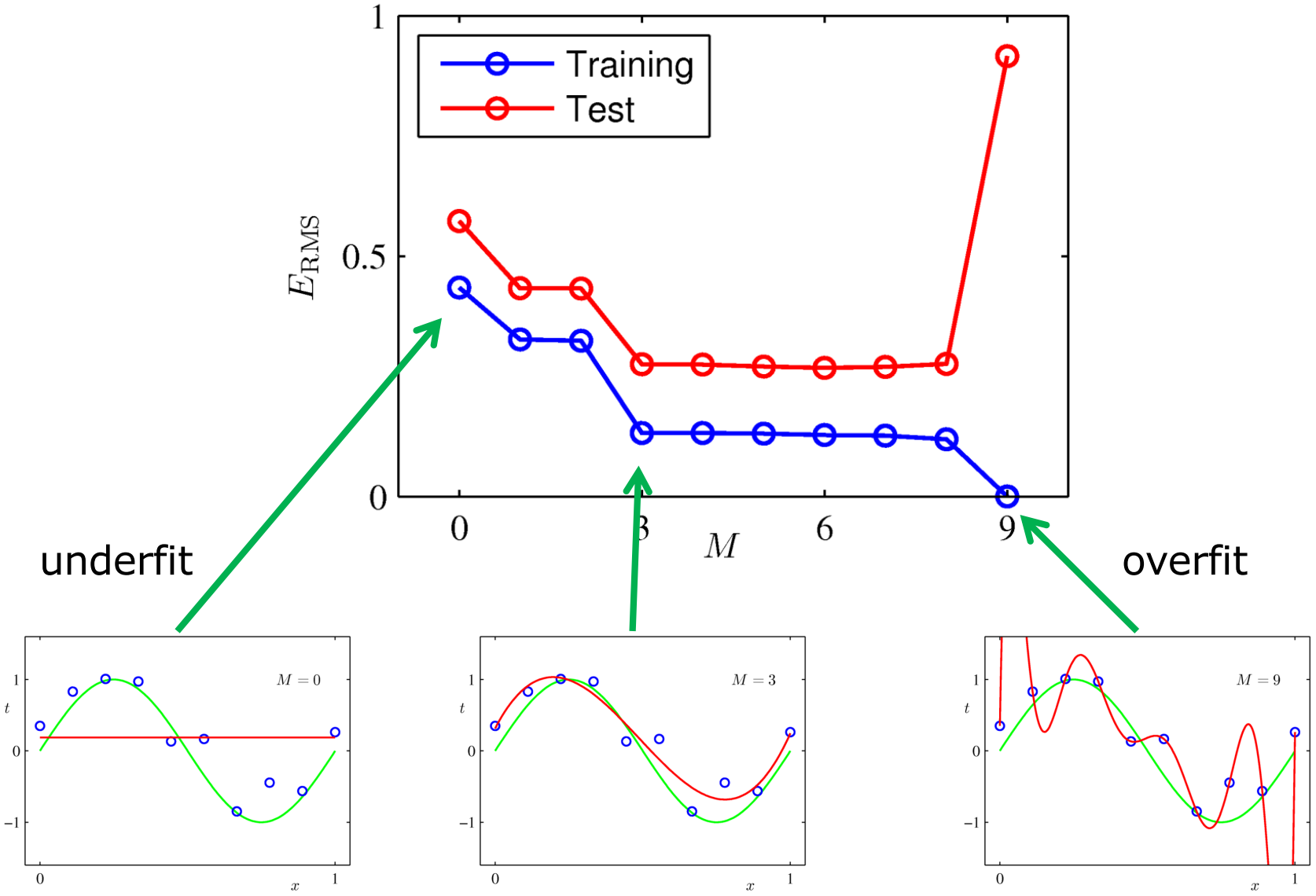
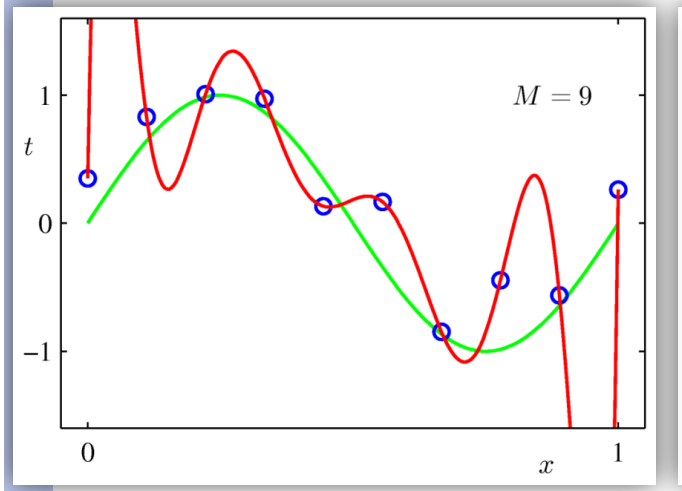


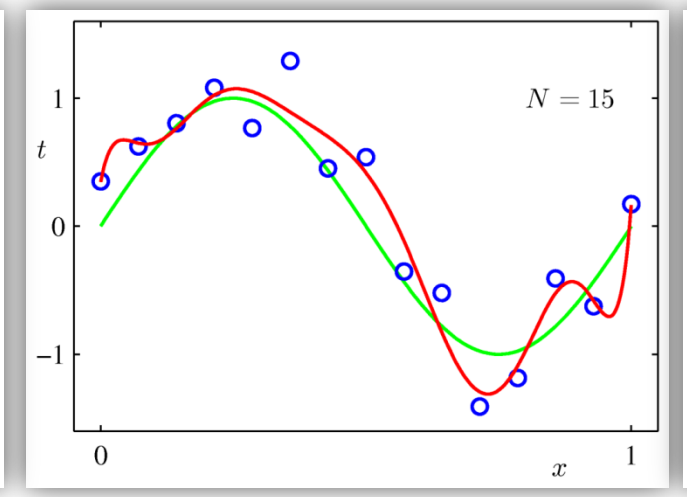
Image source: Christopher M. Bishop, Pattern Recognition and Machine Learning, Springer 2007

Underfitting vs. overfitting: regression

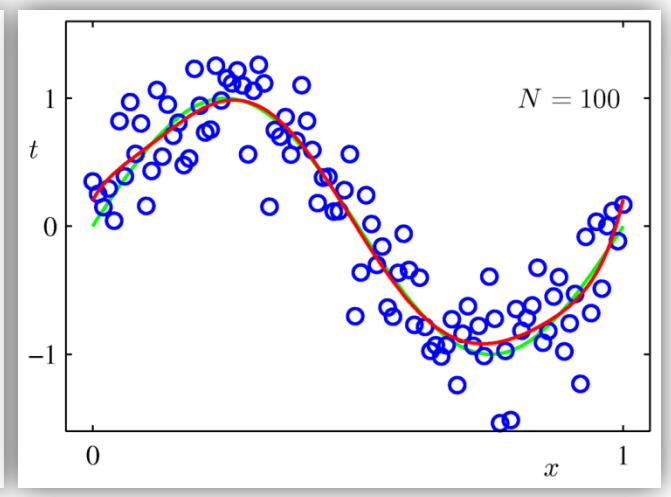
- Overfitting can be reduced by using more training data
- Example: polynomial degree M=9, training data N=10, 15, 100



N=10



N=15

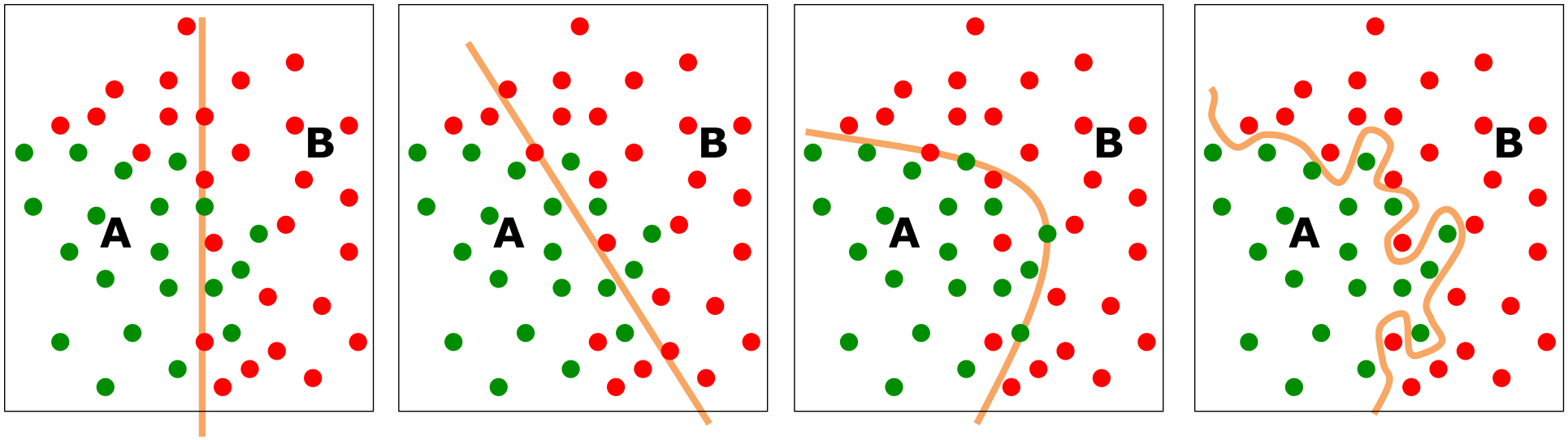


N=100

Image source: Christopher M. Bishop, Pattern Recognition and Machine Learning, Springer 2007

Underfitting vs. overfitting: classification

- The same problems arise in classification!

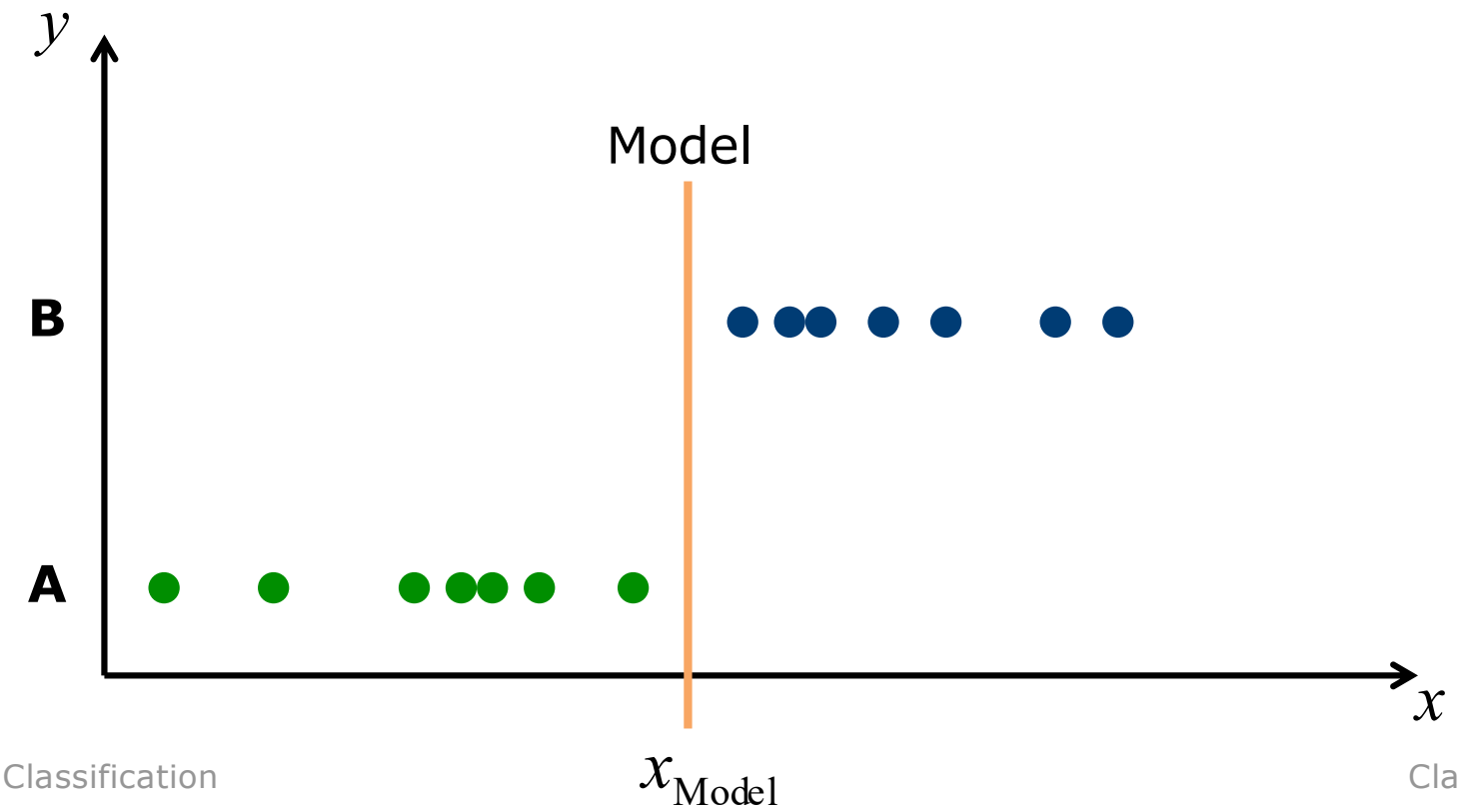


Underfit Underfit OK Overfit

Decision Trees

Multidimensional features

- Previously: only one attribute
- The values of the attribute x were distributed in such a way that a border between A and B could be determined which separates them



Multidimensional features

- Now: Instances have 4 attributes

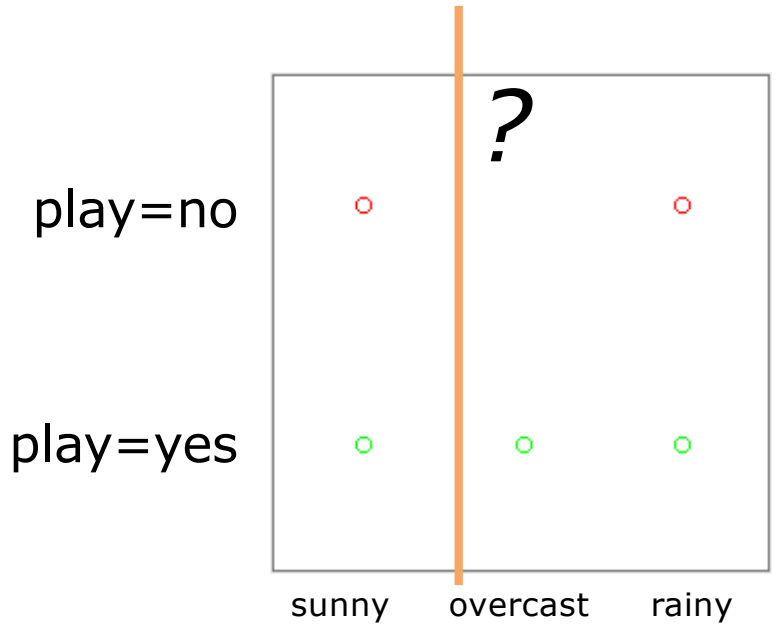
No.	outlook Nominal	temperature Numeric	humidity Numeric	windy Nominal	**play** Nominal
1	sunny	85.0	85.0	FALSE	no
2	sunny	80.0	90.0	TRUE	no
3	overcast	83.0	86.0	FALSE	yes
4	rainy	70.0	96.0	FALSE	yes
5	rainy	68.0	80.0	FALSE	yes
6	rainy	65.0	70.0	TRUE	no
7	overcast	64.0	65.0	TRUE	yes
8	sunny	72.0	95.0	FALSE	no
9	sunny	69.0	70.0	FALSE	yes
10	rainy	75.0	80.0	FALSE	yes
11	sunny	75.0	70.0	TRUE	yes
12	overcast	72.0	90.0	TRUE	yes
13	overcast	81.0	75.0	FALSE	yes
14	rainy	71.0	91.0	TRUE	no

Four attributes
→ can't be visualized easily

(Source dataset: WEKA, http://www.cs.waikato.ac.nz/ml/index.html)

Multidimensional features

- Plot: outlook vs. play
- All instances with “overcast” have “play=yes”
- But for “sunny” (and also, for “rainy”), there are instances with “play=yes” and “play=no”
- No separation possible

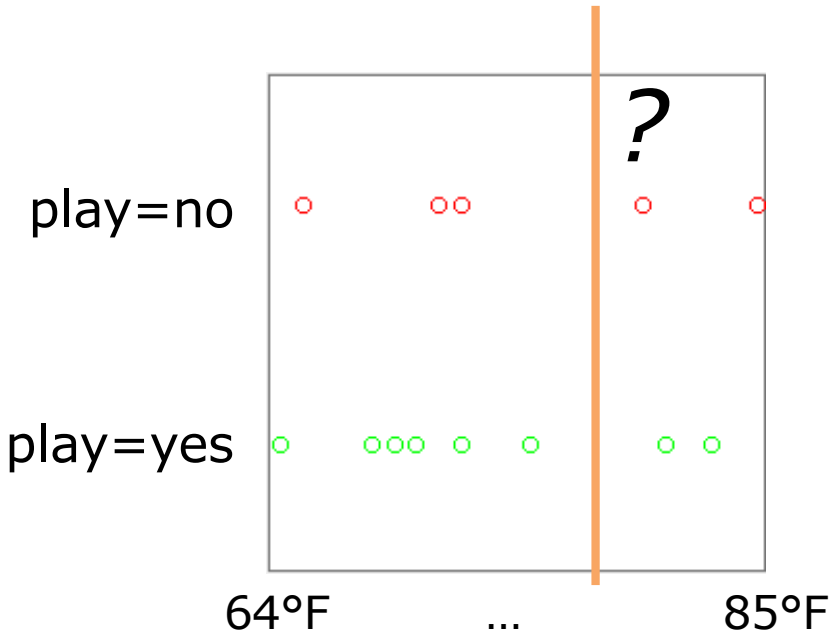


No.	outlook Nominal	temperature Numeric	humidity Numeric	windy Nominal	**play** Nominal
1	sunny	85.0	85.0	FALSE	no
2	sunny	80.0	90.0	TRUE	no
3	overcast	83.0	86.0	FALSE	yes
4	rainy	70.0	96.0	FALSE	yes
5	rainy	68.0	80.0	FALSE	yes
6	rainy	65.0	70.0	TRUE	no
7	overcast	64.0	65.0	TRUE	yes
8	sunny	72.0	95.0	FALSE	no
9	sunny	69.0	70.0	FALSE	yes
10	rainy	75.0	80.0	FALSE	yes
11	sunny	75.0	70.0	TRUE	yes
12	overcast	72.0	90.0	TRUE	yes
13	overcast	81.0	75.0	FALSE	yes
14	rainy	71.0	91.0	TRUE	no

(Source dataset: WEKA, http://www.cs.waikato.ac.nz/ml/index.html)

Multidimensional features

- The same holds for the (numeric) attribute “temperature”

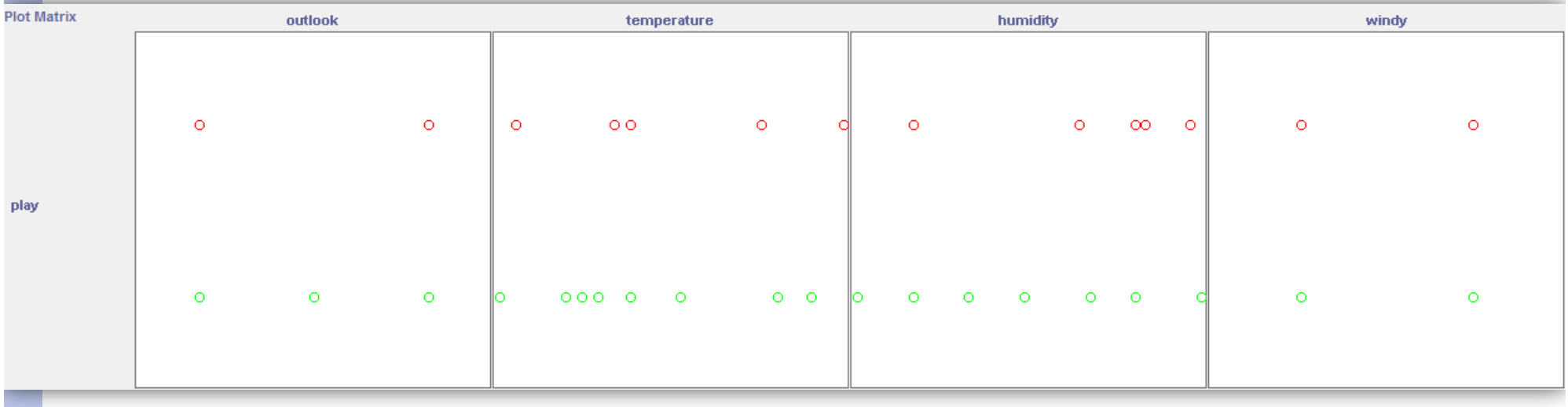


No.	outlook Nominal	temperature Numeric	humidity Numeric	windy Nominal	**play** Nominal
1	sunny	85.0	85.0	FALSE	no
2	sunny	80.0	90.0	TRUE	no
3	overcast	83.0	86.0	FALSE	yes
4	rainy	70.0	96.0	FALSE	yes
5	rainy	68.0	80.0	FALSE	yes
6	rainy	65.0	70.0	TRUE	no
7	overcast	64.0	65.0	TRUE	yes
8	sunny	72.0	95.0	FALSE	no
9	sunny	69.0	70.0	FALSE	yes
10	rainy	75.0	80.0	FALSE	yes
11	sunny	75.0	70.0	TRUE	yes
12	overcast	72.0	90.0	TRUE	yes
13	overcast	81.0	75.0	FALSE	yes
14	rainy	71.0	91.0	TRUE	no

(Source dataset: WEKA, http://www.cs.waikato.ac.nz/ml/index.html)

Multidimensional features

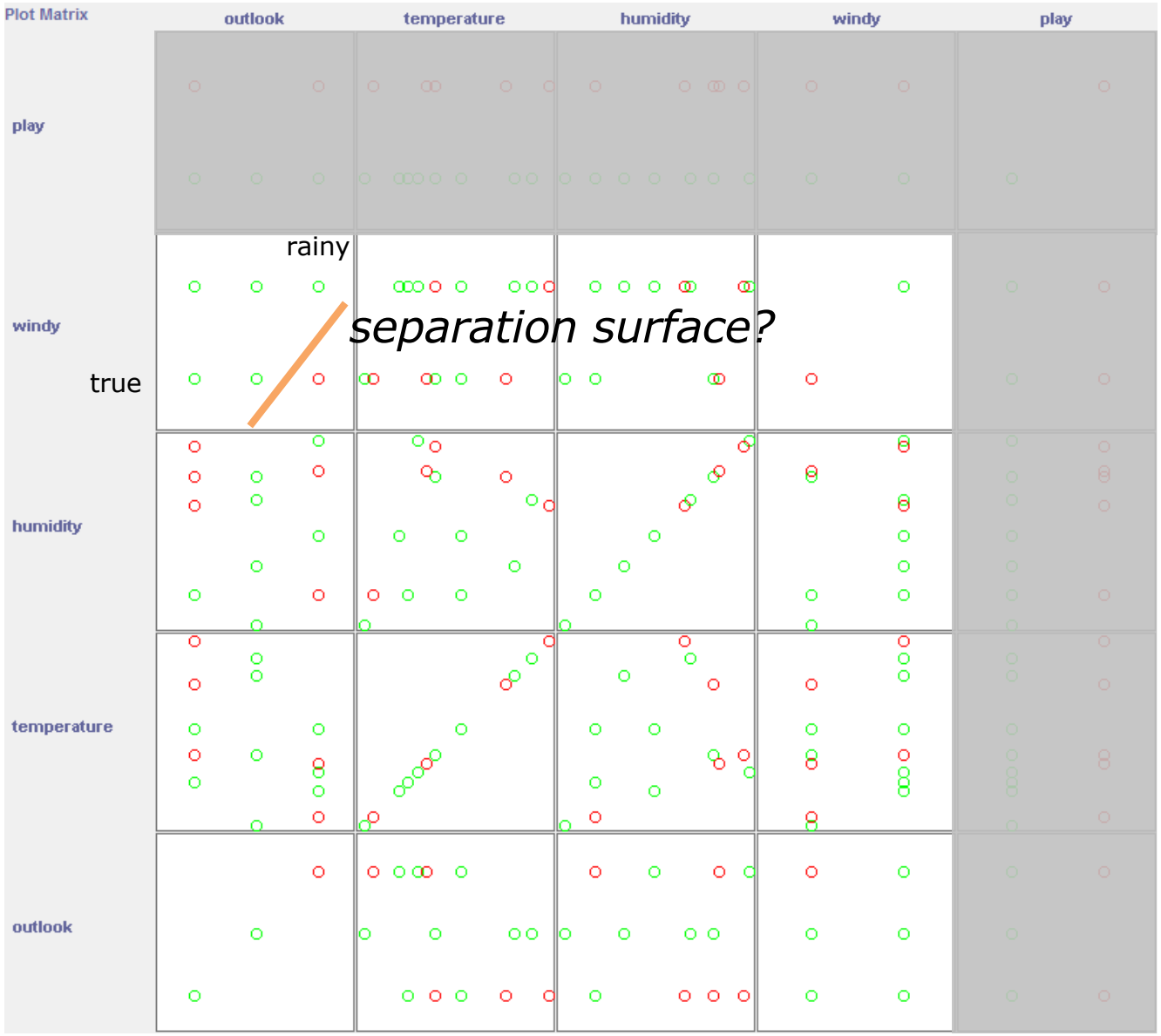
- …and with the other attributes
- In none of the projections, the training data can be separated using a simple threshold.



(Source dataset: WEKA, http://www.cs.waikato.ac.nz/ml/index.html)

Multidimensional features

- By visualizing 2D (or 3D) projections, one can try to identify "visually" if separation surfaces exist
- The task of training is to find such a separation surface automatically, in n dimensional space



(Source dataset: WEKA, http://www.cs.waikato.ac.nz/ml/index.html)

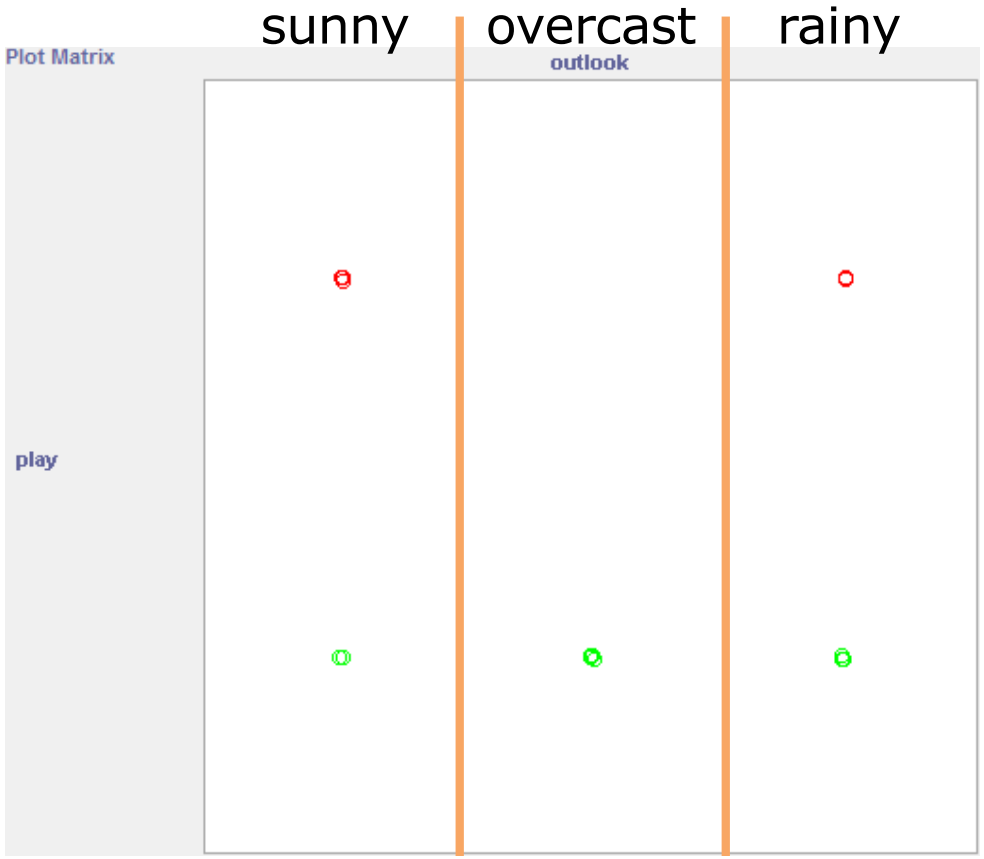
Multidimensional features

- Note: nominal attributes can be occluded by others
- WEKA Explorer: switch on small random perturbations in the visualization (called "jitter")

(Source dataset: WEKA, http://www.cs.waikato.ac.nz/ml/index.html)

Decision trees

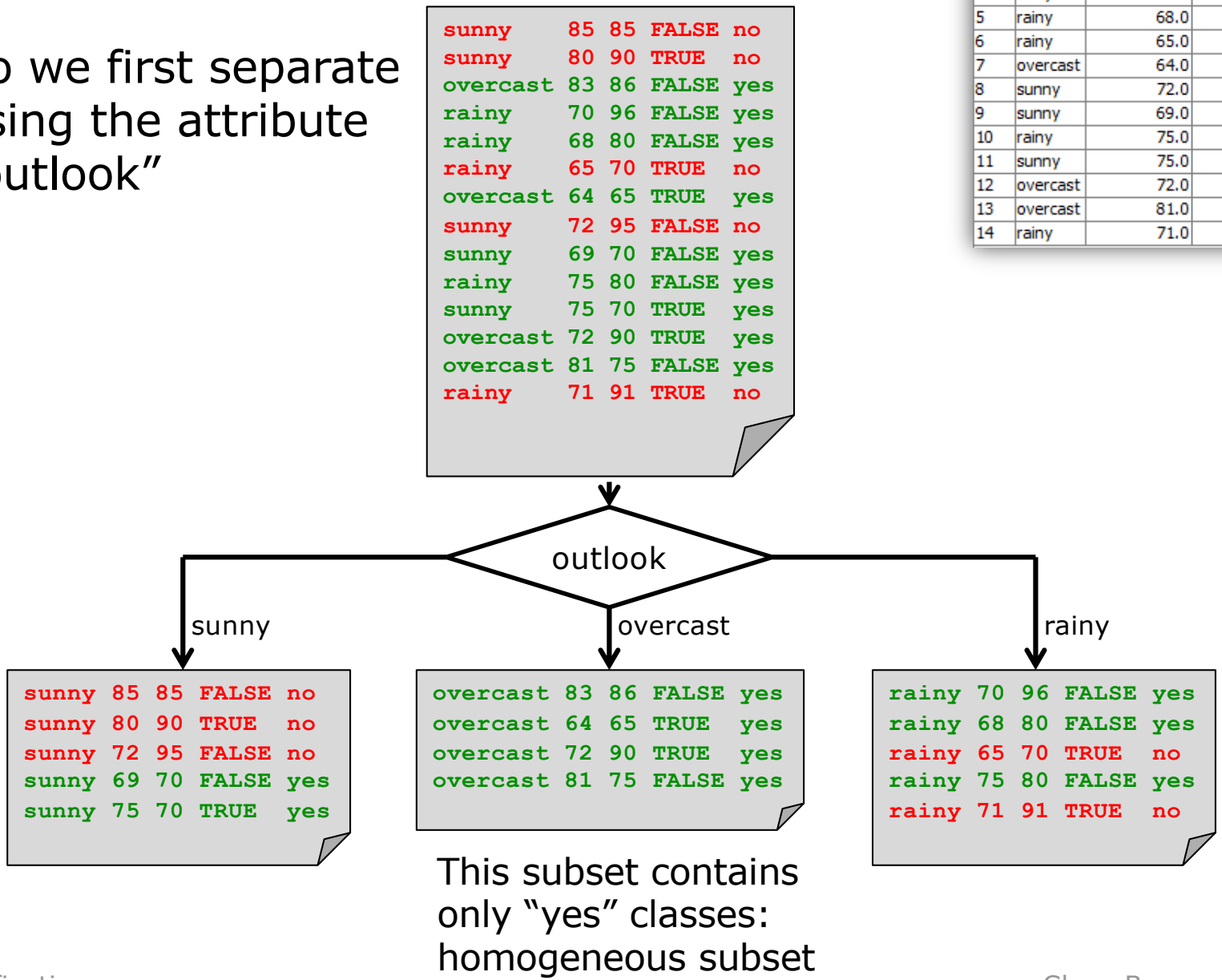
- We cannot separate based on a single attribute
- → First separate using one attribute, then further separate using another attribute, then using still another attribute…
- In our example: we have seen that "outlook" seems to make a "good" separation



Decision trees

- So we first separate using the attribute "outlook"

No.	outlook Nominal	temperature Numeric	humidity Numeric	windy Nominal	play Nominal
1	sunny	85.0	85.0	FALSE	no
2	sunny	80.0	90.0	TRUE	no
3	overcast	83.0	86.0	FALSE	yes
4	rainy	70.0	96.0	FALSE	yes
5	rainy	68.0	80.0	FALSE	yes
6	rainy	65.0	70.0	TRUE	no
7	overcast	64.0	65.0	TRUE	yes
8	sunny	72.0	95.0	FALSE	no
9	sunny	69.0	70.0	FALSE	yes
10	rainy	75.0	80.0	FALSE	yes
11	sunny	75.0	70.0	TRUE	yes
12	overcast	72.0	90.0	TRUE	yes
13	overcast	81.0	75.0	FALSE	yes
14	rainy	71.0	91.0	TRUE	no

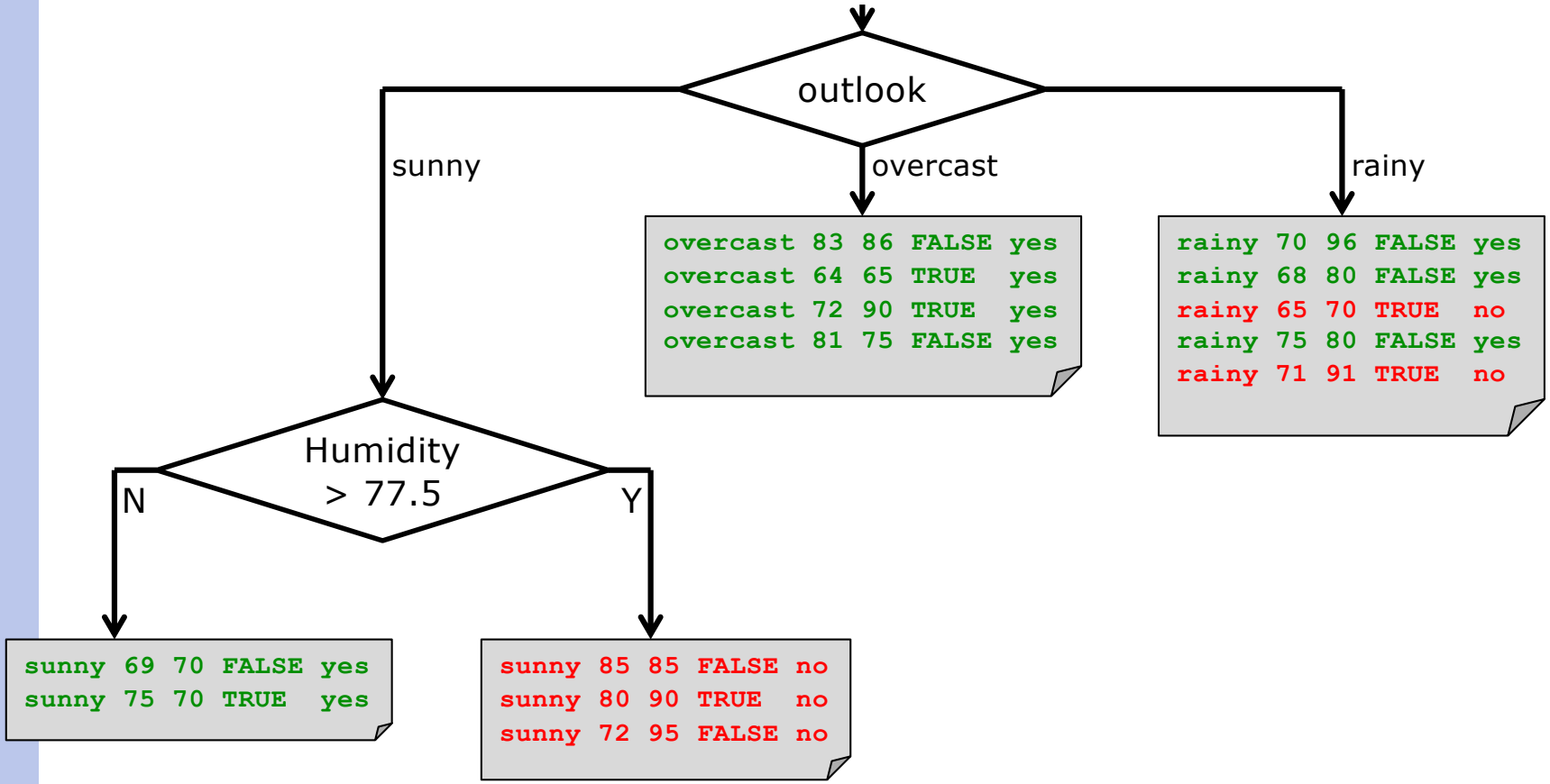


This subset contains only "yes" classes: homogeneous subset

(Source dataset: WEKA, http://www.cs.waikato.ac.nz/ml/index.html)

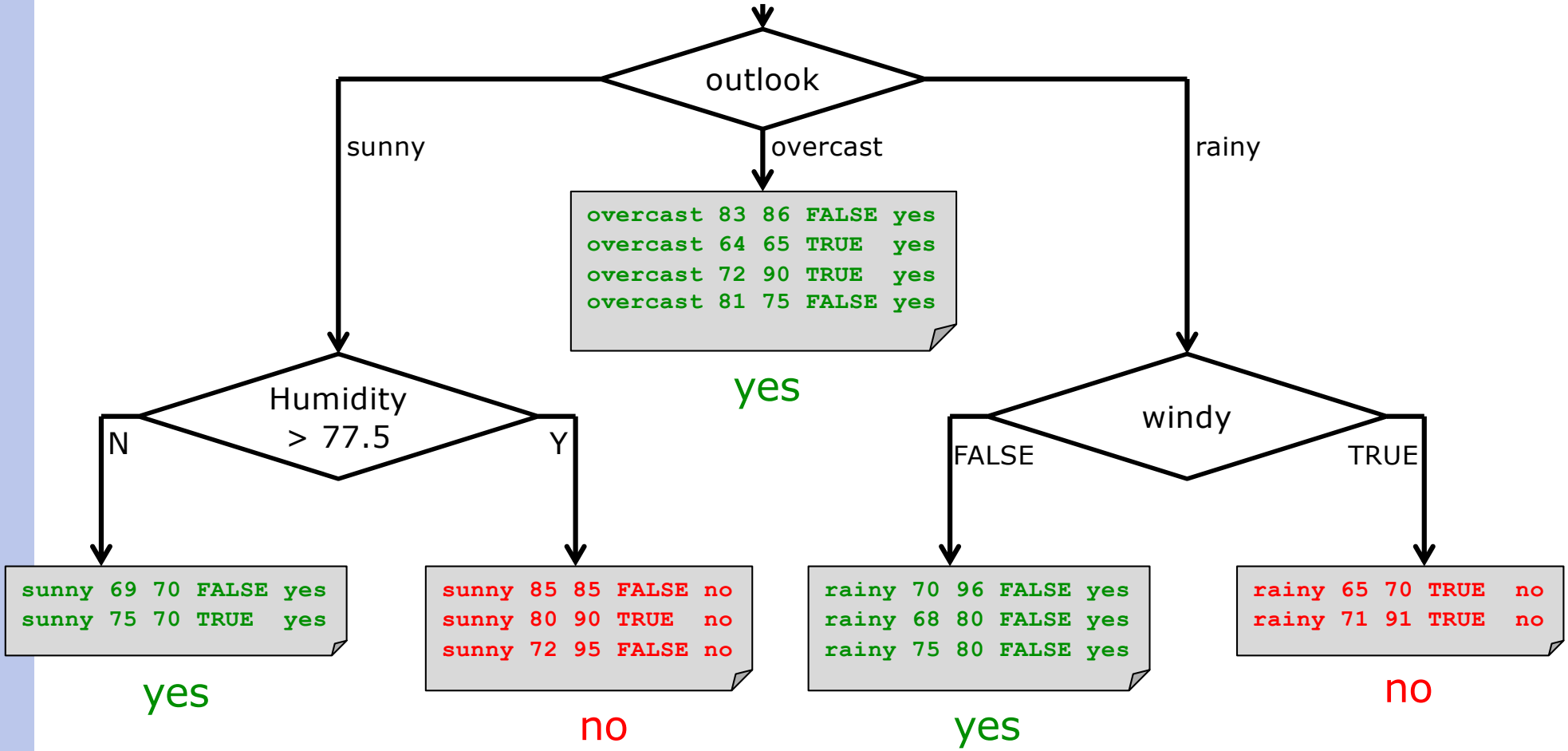
Decision trees

- In the left branch “sunny”, we further subdivide based on “humidity”



Decision trees

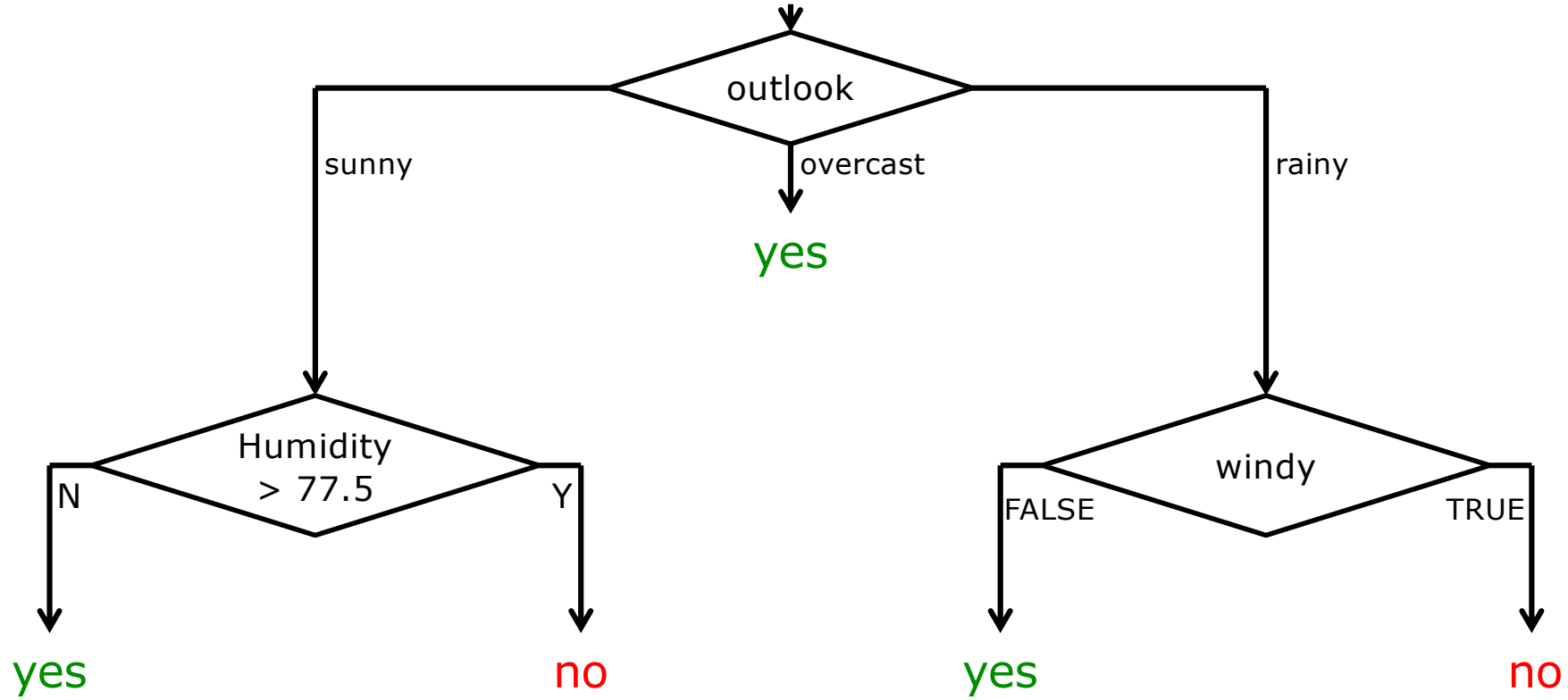
- In the right branch “rainy”, we subdivide based on “windy”



(Source dataset: WEKA, http://www.cs.waikato.ac.nz/ml/index.html)

Decision trees

- This is the decision tree which we finally obtain
- It is a representation of f(x)

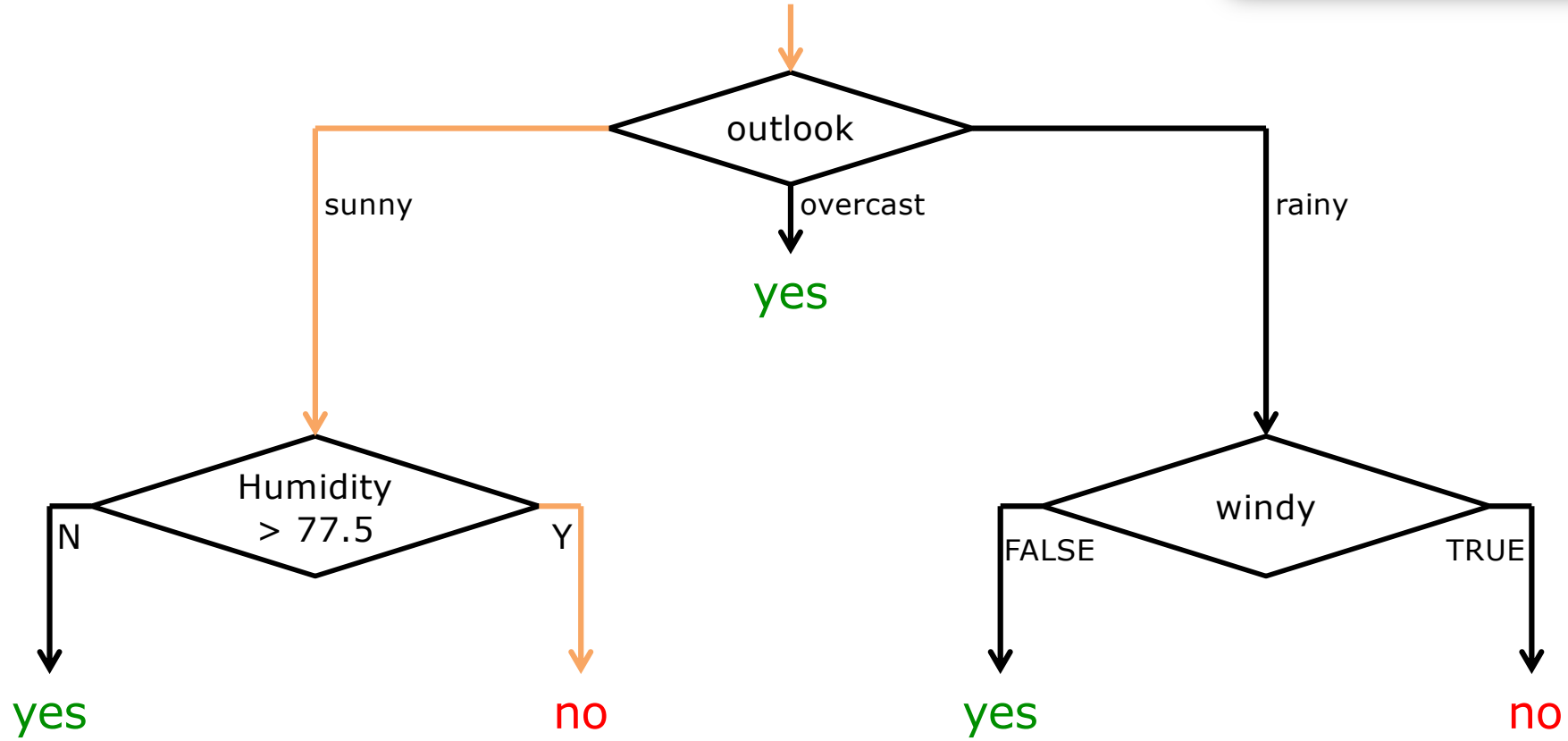


Decision trees

(outlook, temperature, humidity, windy)

- Example: f(sunny, 102, 91, TRUE) = no
- temperature =102 plays no role

No.	outlook Nominal	temperature Numeric	humidity Numeric	windy Nominal	play Nominal
1	sunny	85.0	85.0	FALSE	no
2	sunny	80.0	90.0	TRUE	no
3	overcast	83.0	86.0	FALSE	yes
4	rainy	70.0	96.0	FALSE	yes
5	rainy	68.0	80.0	FALSE	yes
6	rainy	65.0	70.0	TRUE	no
7	overcast	64.0	65.0	TRUE	yes
8	sunny	72.0	95.0	FALSE	no
9	sunny	69.0	70.0	FALSE	yes
10	rainy	75.0	80.0	FALSE	yes
11	sunny	75.0	70.0	TRUE	yes
12	overcast	72.0	90.0	TRUE	yes
13	overcast	81.0	75.0	FALSE	yes
14	rainy	71.0	91.0	TRUE	no



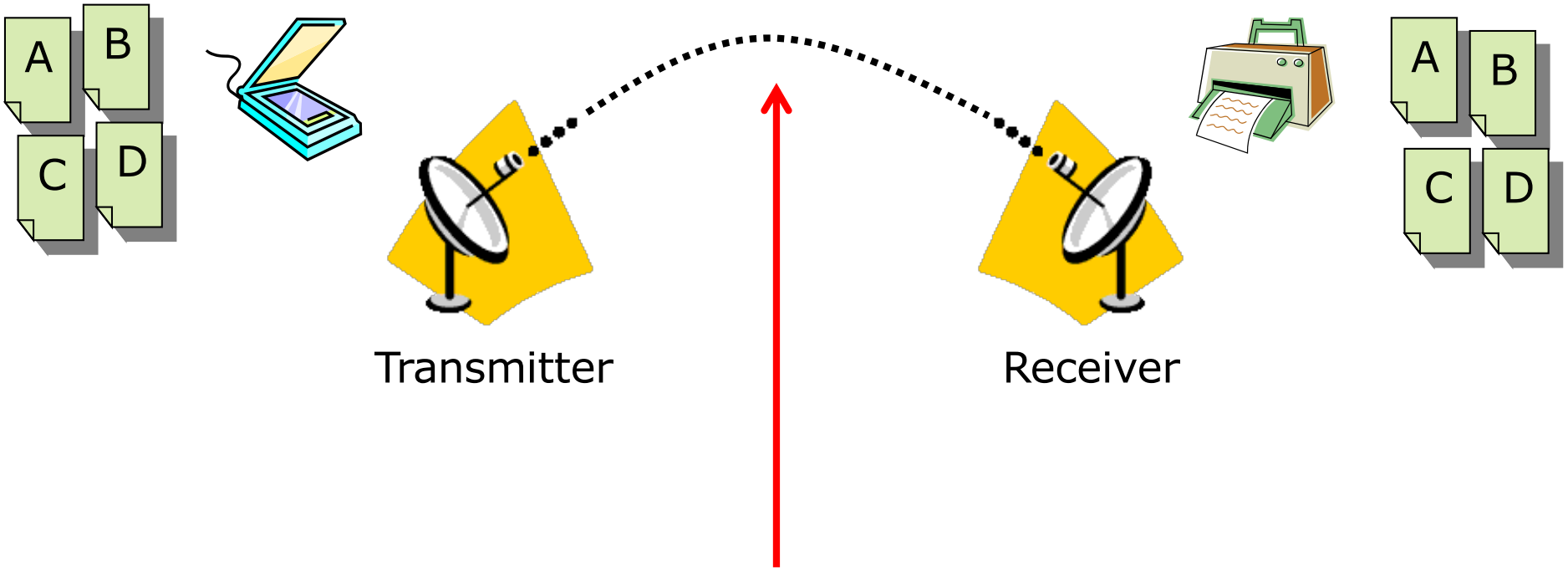
(Source dataset: WEKA, http://www.cs.waikato.ac.nz/ml/index.html)

Decision trees

- How shall we select the attributes to be tested?
  - In which order?
  - Which ones can be left out because they play no role for the classification?

Information and entropy

Number of bits required to encode a message

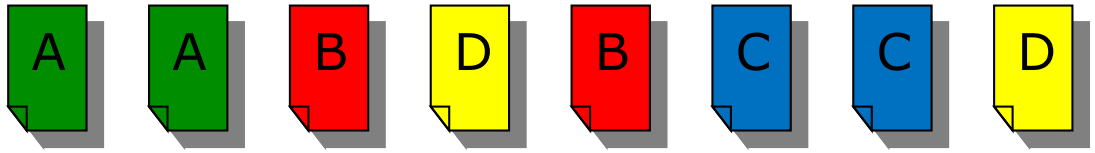


Symbol	Code
A	00
B	01
C	10
D	11

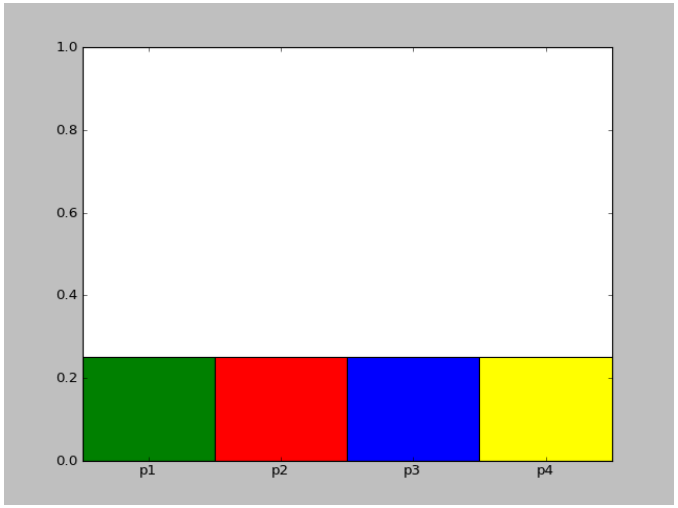
4 symbols →
2 bits are needed.

Number of bits required to encode a message

- If symbols are uniformly distributed:



00 00 01 11 01 10 10 11 Total bits: 16

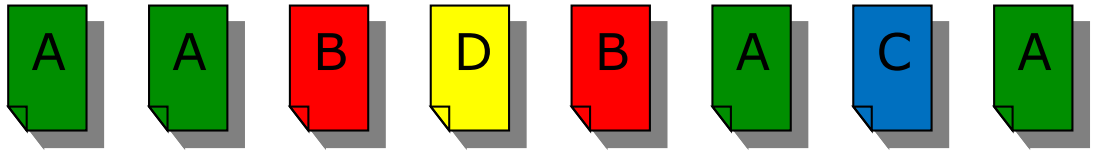


Expected number of bits:

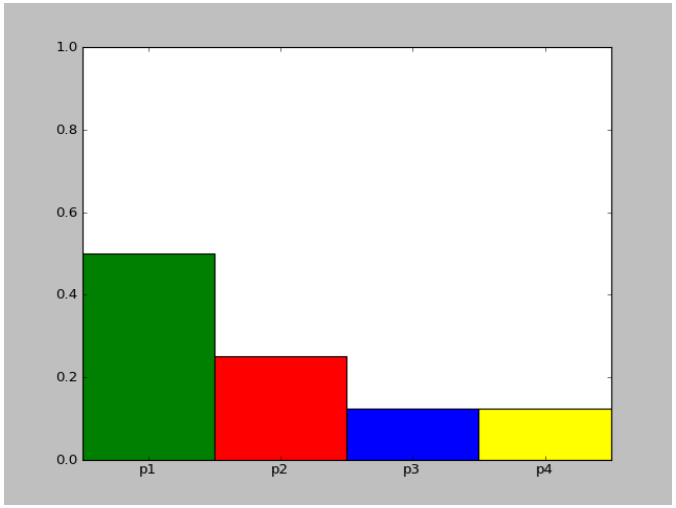
$$0.25 \cdot 2 + 0.25 \cdot 2 + 0.25 \cdot 2 + 0.25 \cdot 2$$
$$= (0.25 + 0.25 + 0.25 + 0.25) \cdot 2 = 2$$

Number of bits required to encode a message

- If the symbols are **not** uniformly distributed:



00 00 01 11 01 00 10 00 Total bits: 16



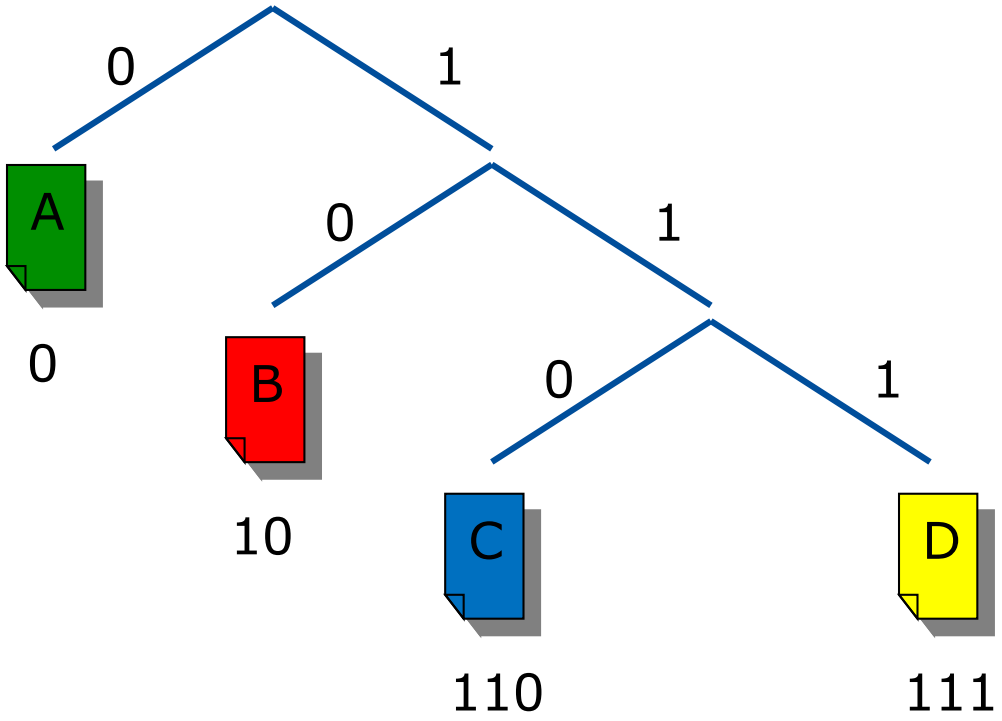
Expected number of bits:

$$0.5 \cdot 2 + 0.25 \cdot 2 + 0.125 \cdot 2 + 0.125 \cdot 2$$
$$= (0.5 + 0.25 + 0.125 + 0.125) \cdot 2 = 2$$

No change – that is no surprise!

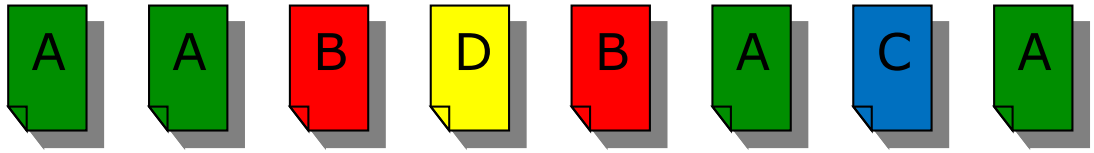
Number of bits required to encode a message

- Use fewer bits for symbols which are more frequent
  - (c.f. prefix codes used for phone dialing)
  - e.g. between one and three bits:

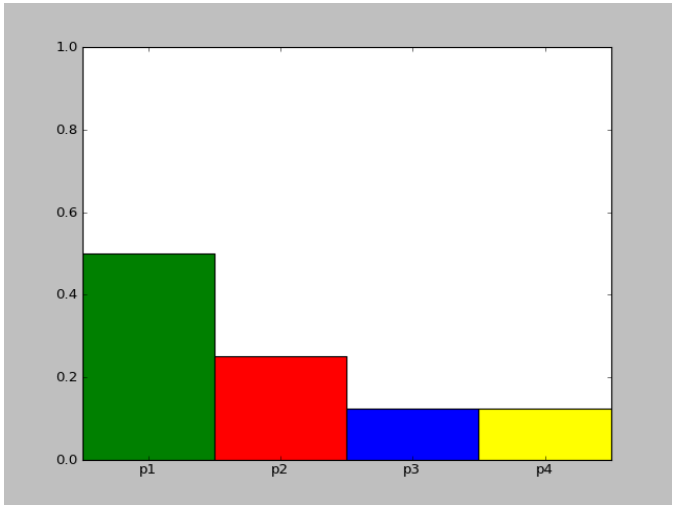


Number of bits required to encode a message

- Try this new encoding:



0 0 10 111 10 0 110 0 Total bits: 14



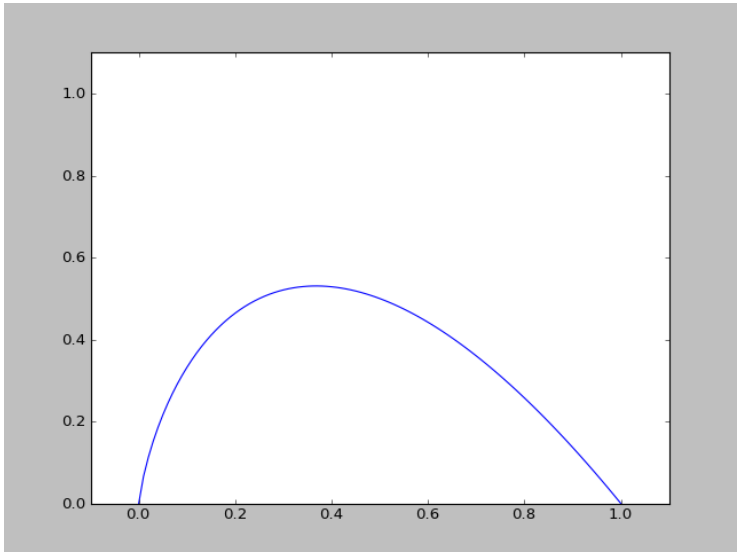
Expected number of bits:

$$0.5 \cdot 1 + 0.25 \cdot 2 + 0.125 \cdot 3 + 0.125 \cdot 3$$
$$= \quad 1.75 \quad (= 14/8)$$

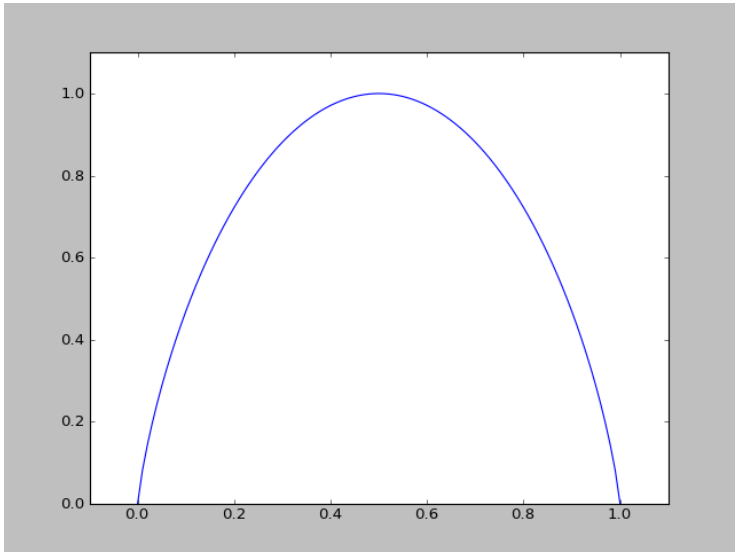
Indeed requires less bits!
Is this the optimum?

Entropy

- Claude Elwood *Shannon,* (*1916, †2001)
  - „A Mathematical Theory of Communication“ [Shannon 1948]
- Entropy of a set of probabilities $\{p_1, p_2, \ldots, p_n\}$
  - $H = -\sum_{i=1}^{n} p_i \log p_i \qquad \text{with} \quad \sum_{i=1}^{n} p_i = 1$



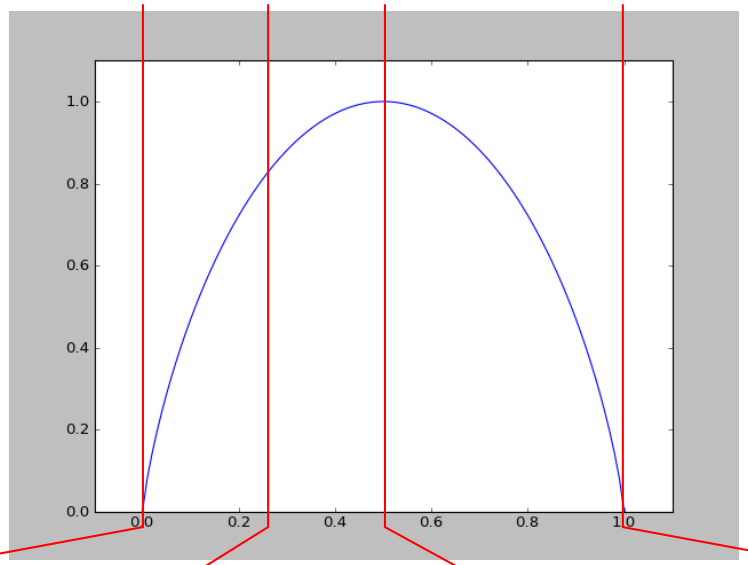
$-p \log p$



$-(p_1 \log p_1 + p_2 \log p_2) \text{ with } p_2 = 1 - p_1$

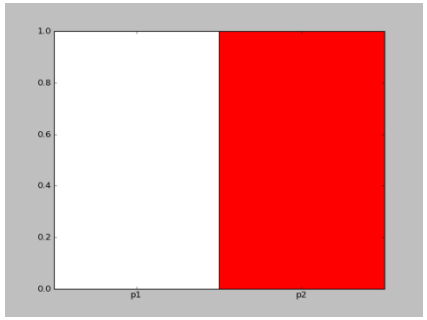
(Here: log base 2)

Entropy: Examples



$p_1 = 0.0,\ p_2 = 1.0$

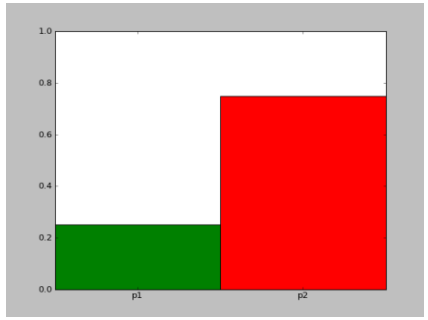
$H = 0$



Certain

$p_1 = 0.25,\ p_2 = 0.75$

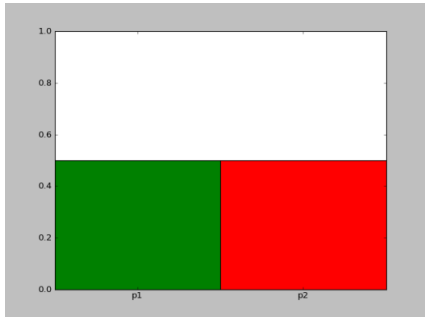
$H = 0.811$



Less certain

$p_1 = 0.5,\ p_2 = 0.5$

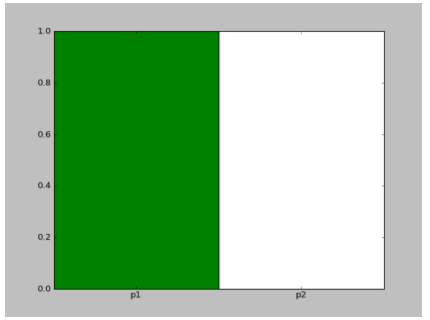
$H = 1$



Most uncertain

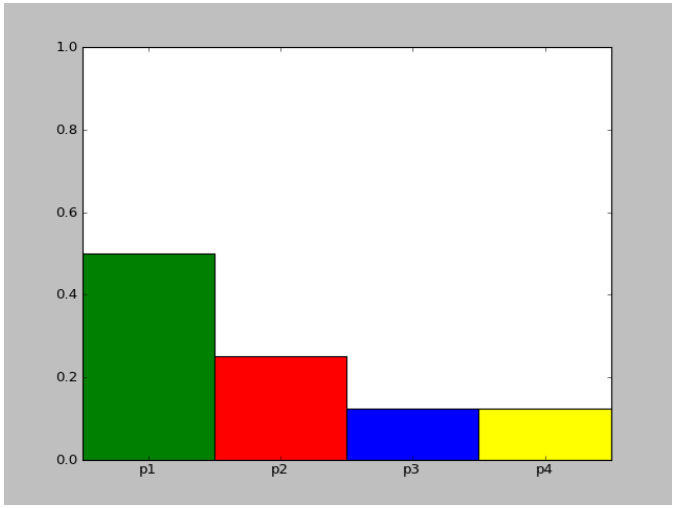
$p_1 = 1.0,\ p_2 = 0.0$

$H = 0$



Certain

Entropy
of the previous example



$$H = -\sum_{i=1}^{n} p_i \log p_i$$

$$= -\left(\frac{1}{2}\log\frac{1}{2} + \frac{1}{4}\log\frac{1}{4} + \frac{1}{8}\log\frac{1}{8} + \frac{1}{8}\log\frac{1}{8}\right)$$

$$= -\left(\frac{1}{2}(-1) + \frac{1}{4}(-2) + \frac{1}{8}(-3) + \frac{1}{8}(-3)\right)$$

$$= 1.75$$

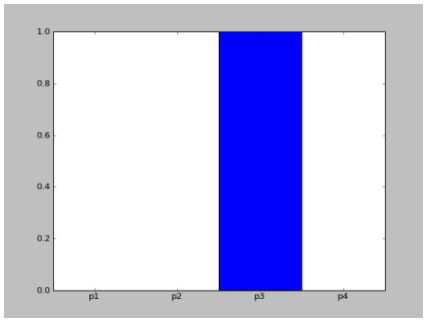
- This is exactly the expected number of bits of our previous encoding
- Entropy is the number of bits of an optimal encoding
- The number of bits of an actual encoding is larger or equal.

Entropy

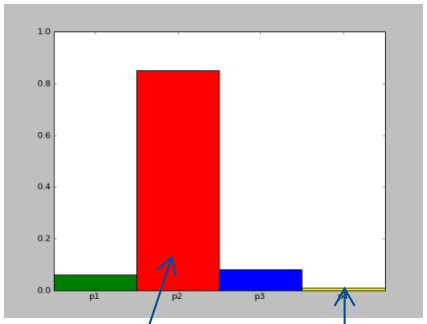
- Entropy is a measure for the information content
- Also can be seen as “average surprise”

0 → Maximum Entropy

H=0



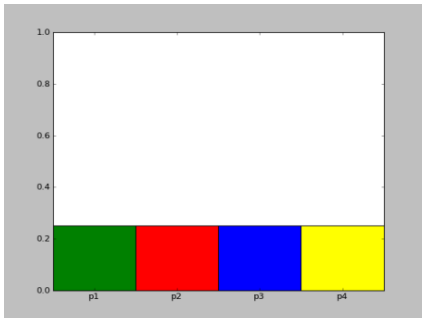
H=0.8



Little surprise

Large surprise

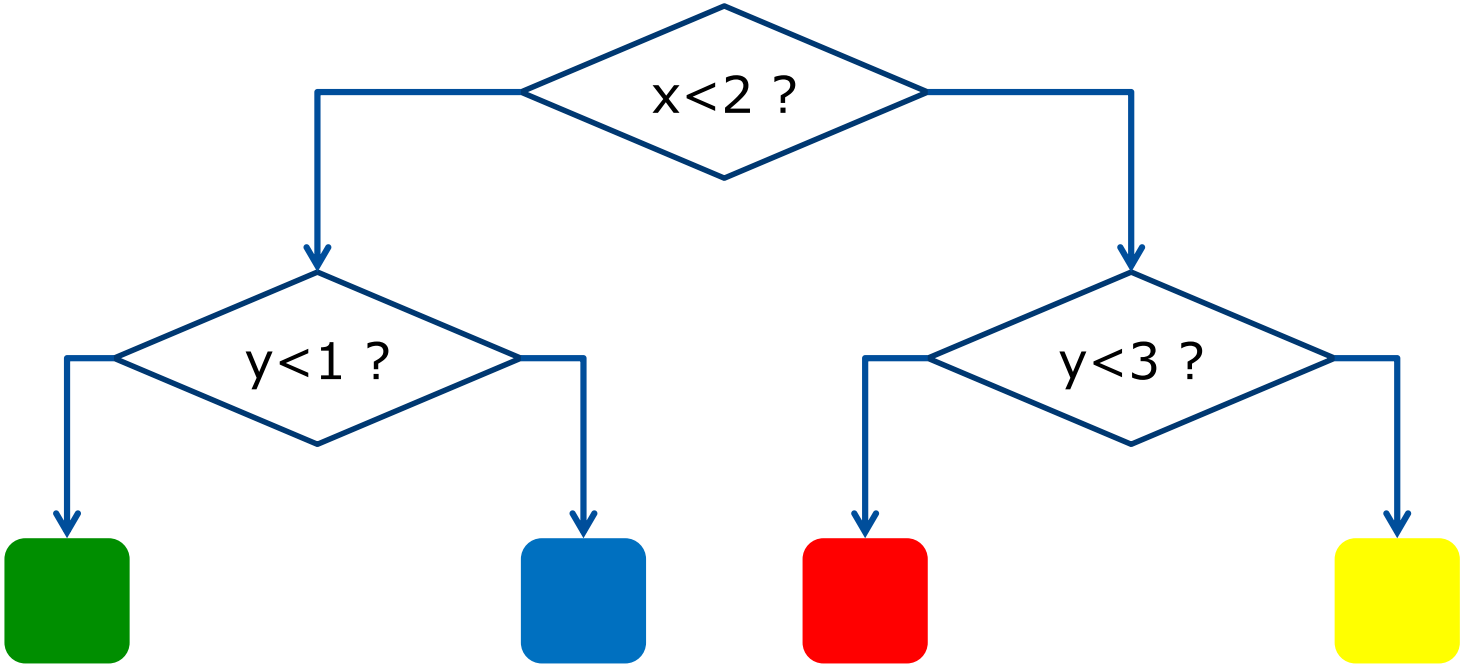
H=2



…back to decision trees

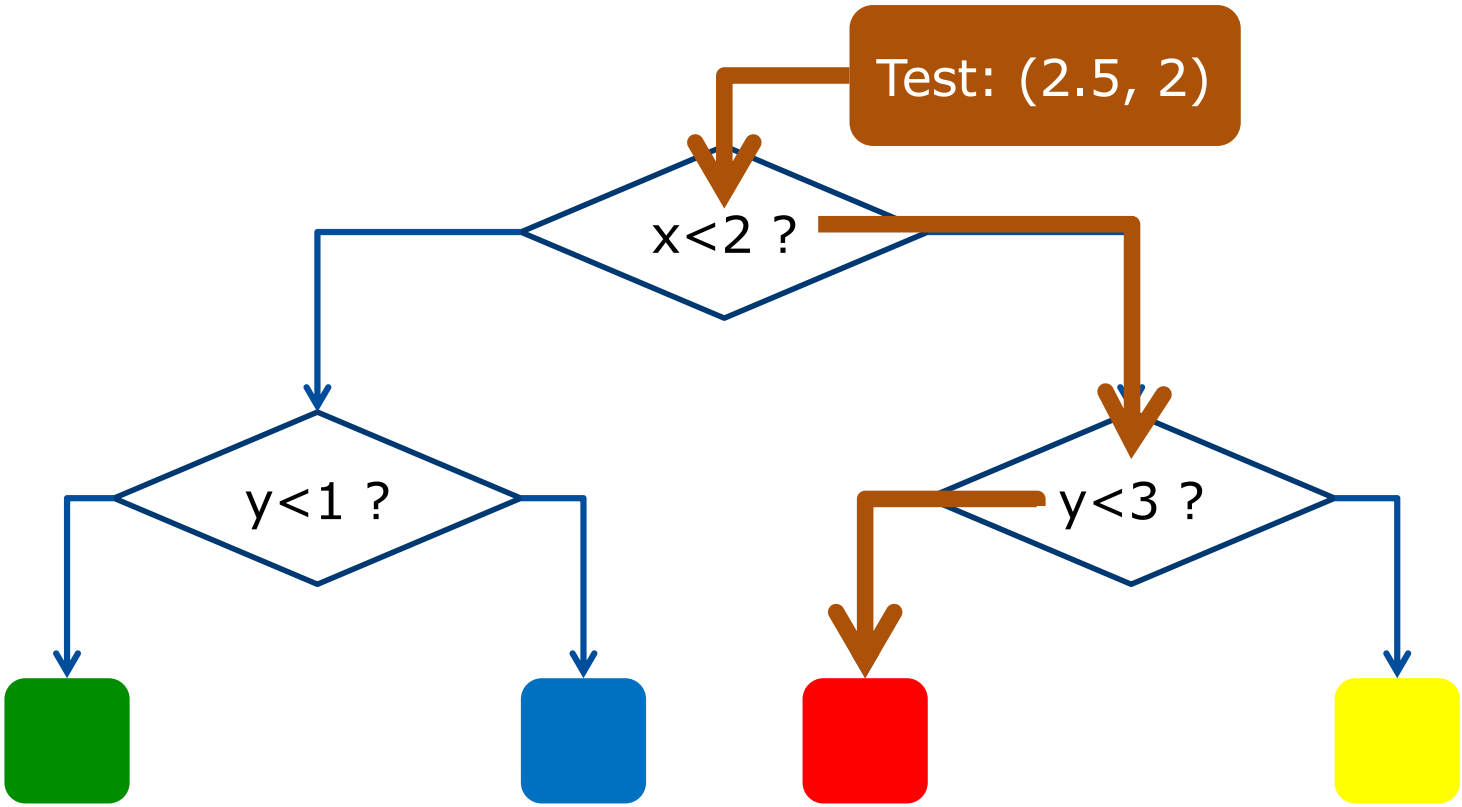
Decision tree

- Not necessarily, but often: binary tree
- Example: two attributes (x, y)



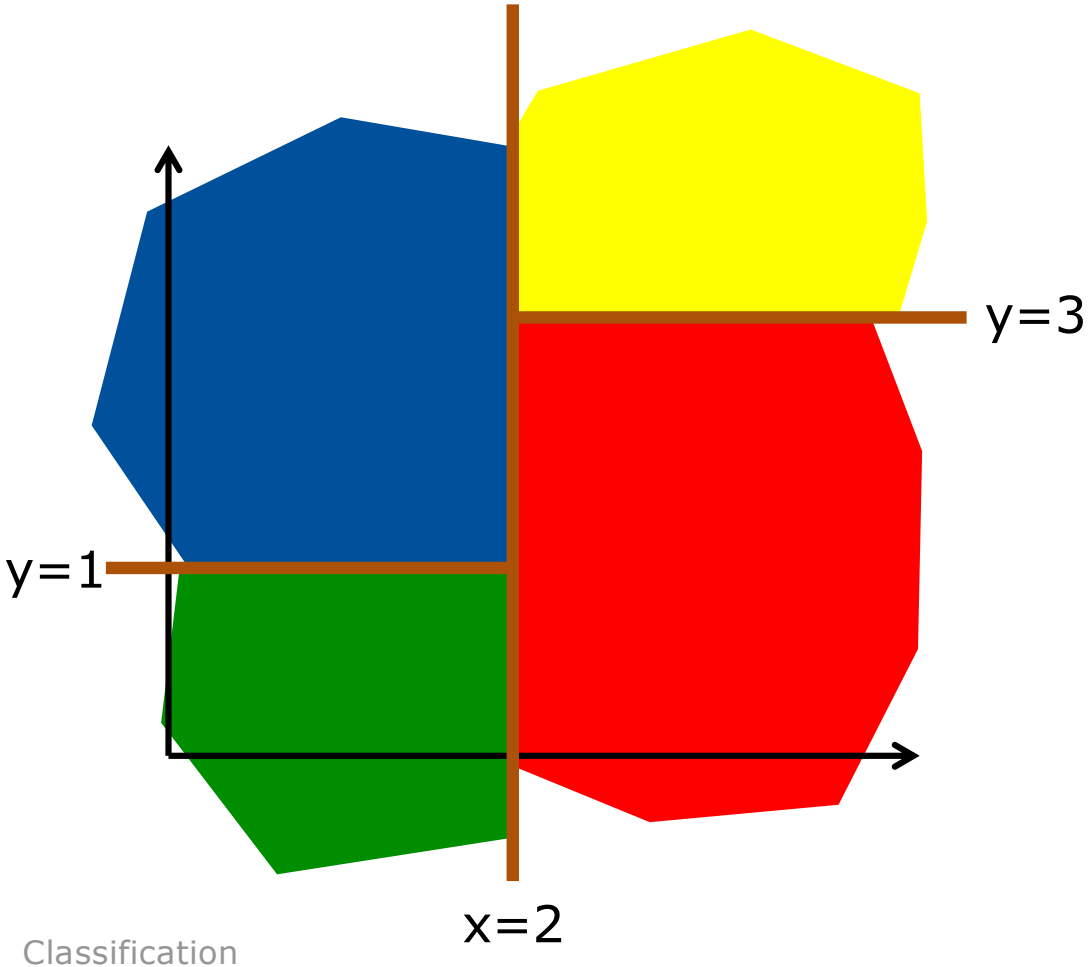
Decision tree

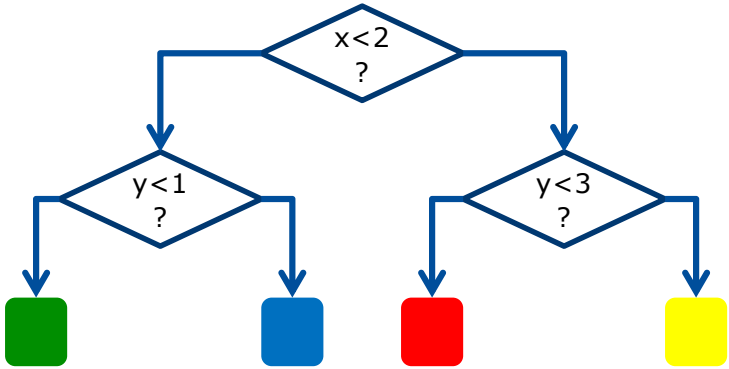
- Not necessarily, but often: binary tree
- Example: two attributes (x, y)



Decision tree

- In this case: subdivision of space using axis parallel lines





Decision tree

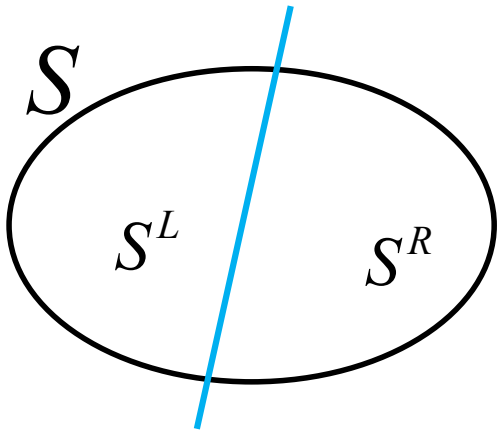
- Testing
  - At runtime, a feature vector (here: point) can be assigned to a leaf by walking through the tree from top (root) to bottom (leaf)
  - In each node, only a single attribute of the feature vector is tested
  - Very efficient

- Training
  - How can suitable tests (the nodes) be derived from the training data?
    - Which attributes? (x, y, …)
    - Which thresholds? (x<2, y<1, …)

Information Gain

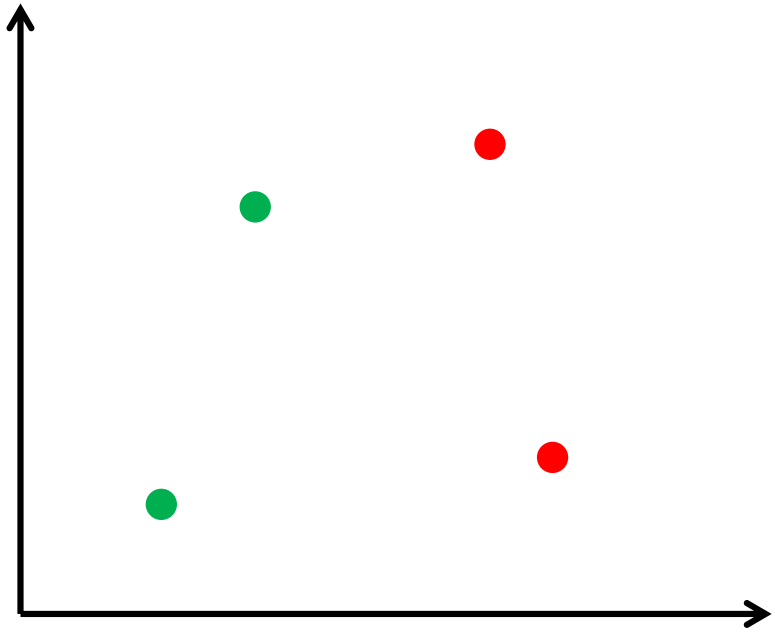
- The **information gain**, which results from the **subdivision** of a set, is computed using:
  - the entropy of the original set *minus*
  - the entropy of each subset, weighted by the number of elements.

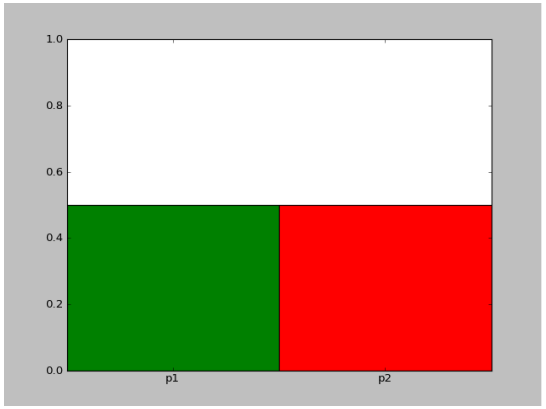
$$I = H(S) - \sum_{i=L,R} \frac{|S^i|}{|S|} H(S^i)$$



Example (two-dimensional features)

- Use relative frequencies (instead of probabilities)

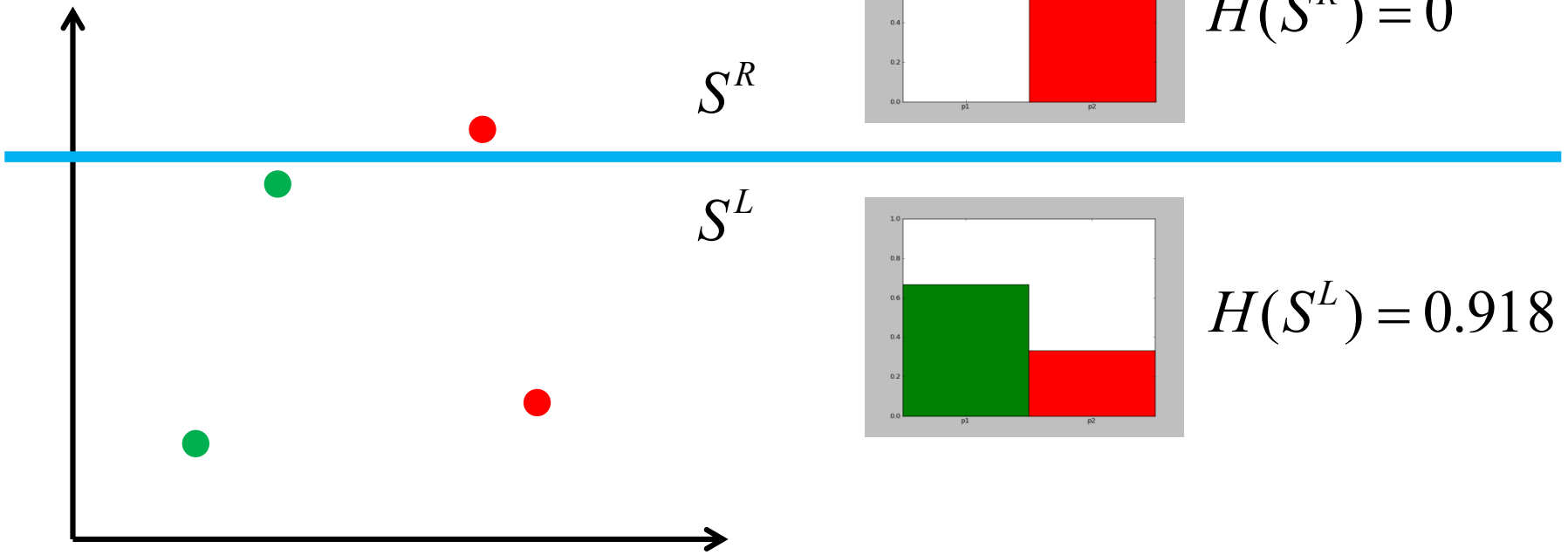


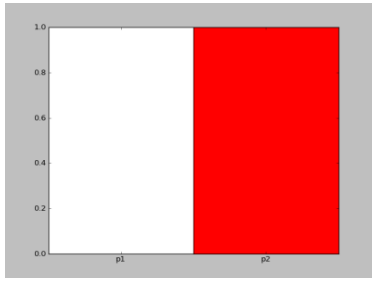


$$H(S) = 1$$

Information gain

- One possible subdivision





$$H(S^R) = 0$$

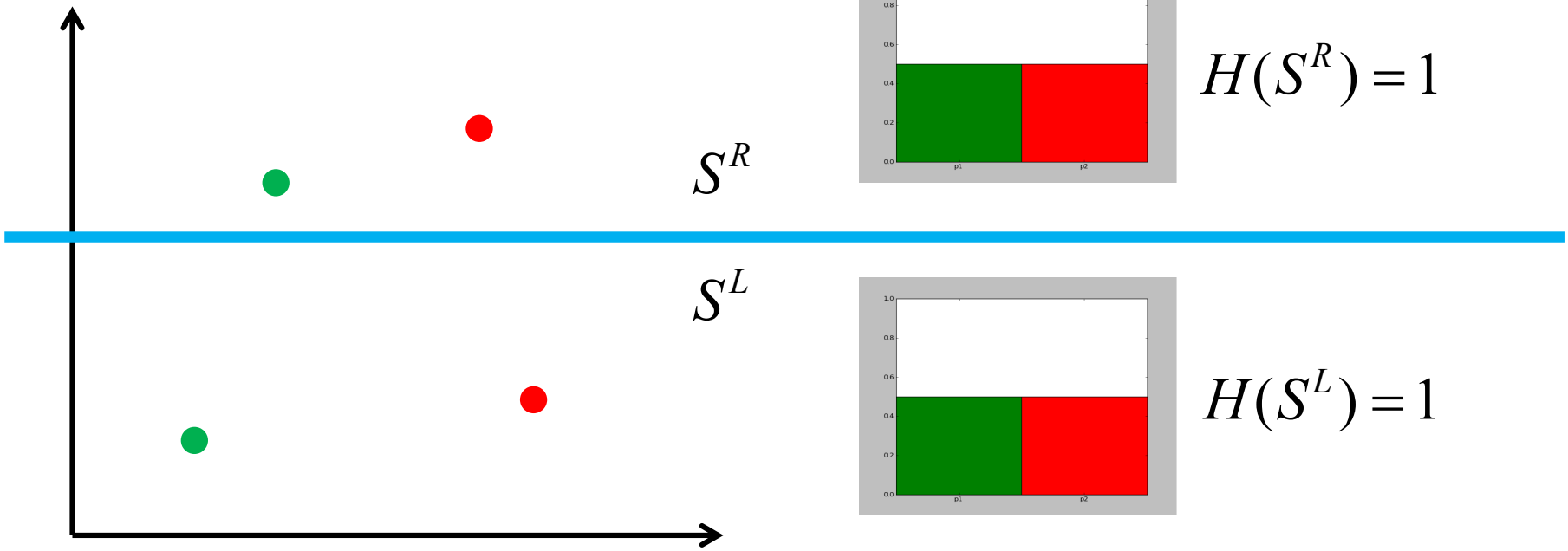
$$H(S^L) = 0.918$$

$$I = H(S) - \sum_{i=L,R} \frac{|S^i|}{|S|} H(S^i)$$

$$= 1 - \left(\frac{3}{4} \cdot 0.918 + \frac{1}{4} \cdot 0 \right) = 0.311$$

Information gain

- Another possible subdivision



$$H(S^R) = 1$$

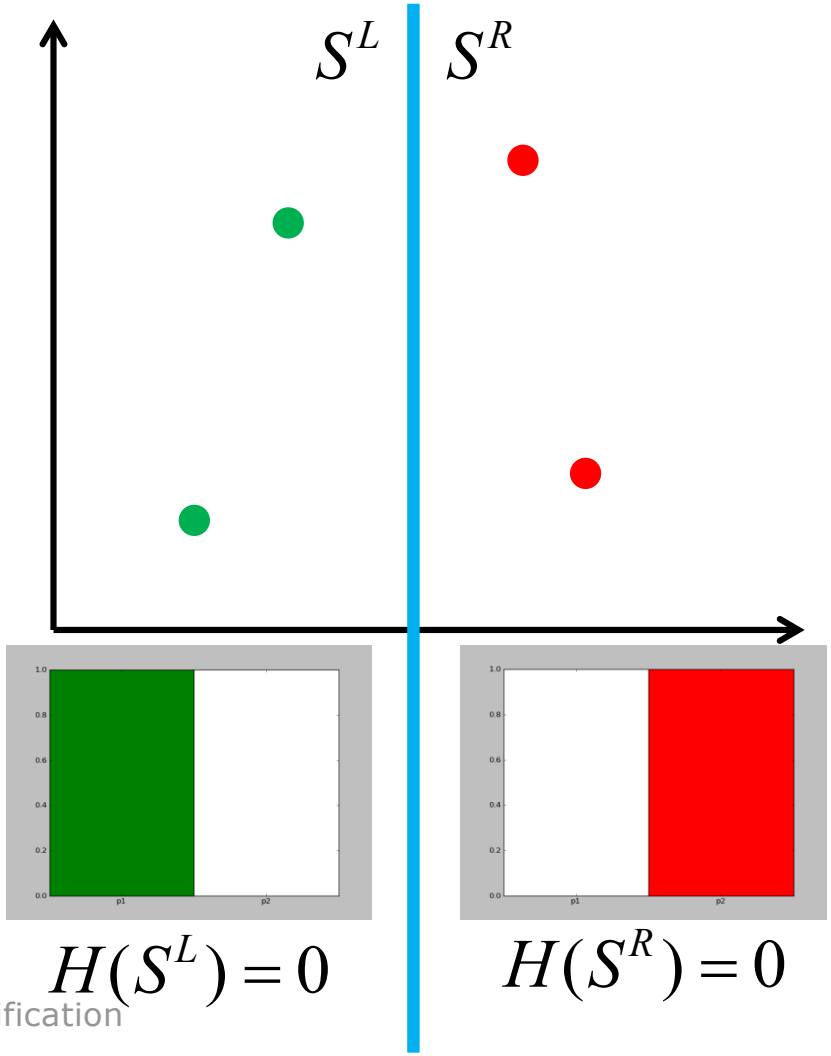
$$H(S^L) = 1$$

$$I = 1 - \left(\frac{2}{4} \cdot 1 + \frac{2}{4} \cdot 1 \right) = 0$$

(no gain)

Information gain

- Another possible subdivision

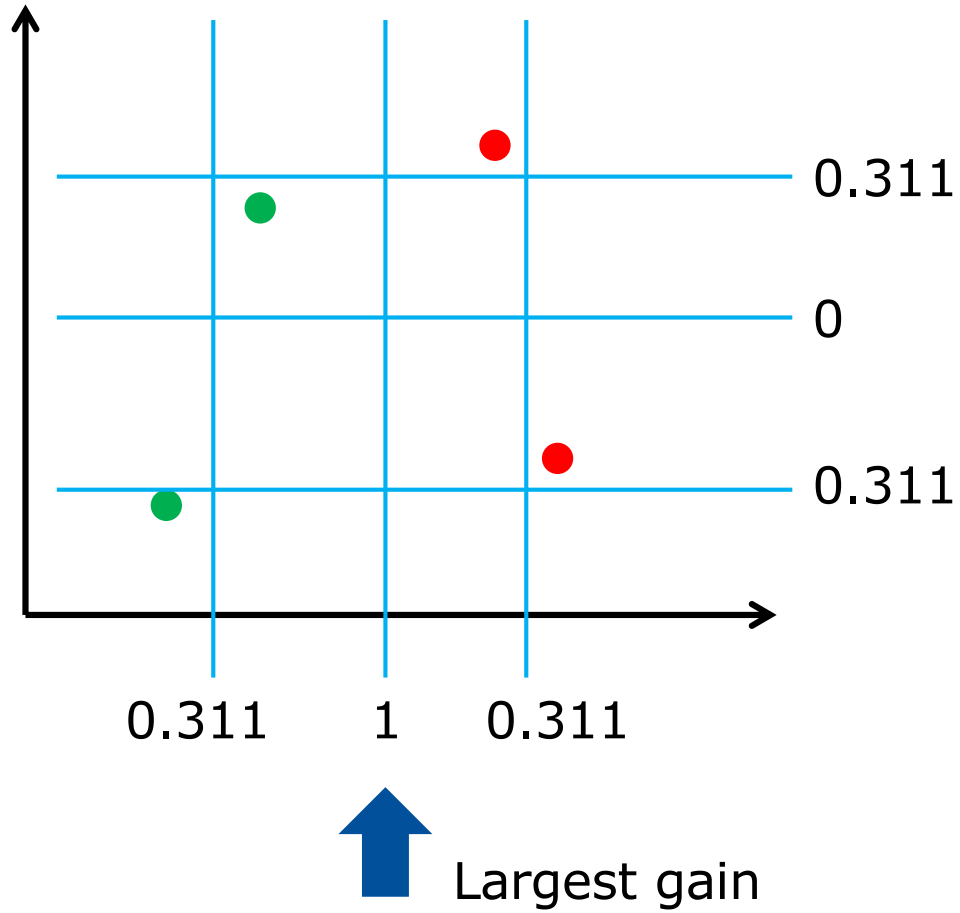


$$I = 1 - \left(\frac{2}{4} \cdot 0 + \frac{2}{4} \cdot 0 \right) = 1$$

(maximum gain)

Information gain

- Information gain for all possible (axis parallel) subdivisions



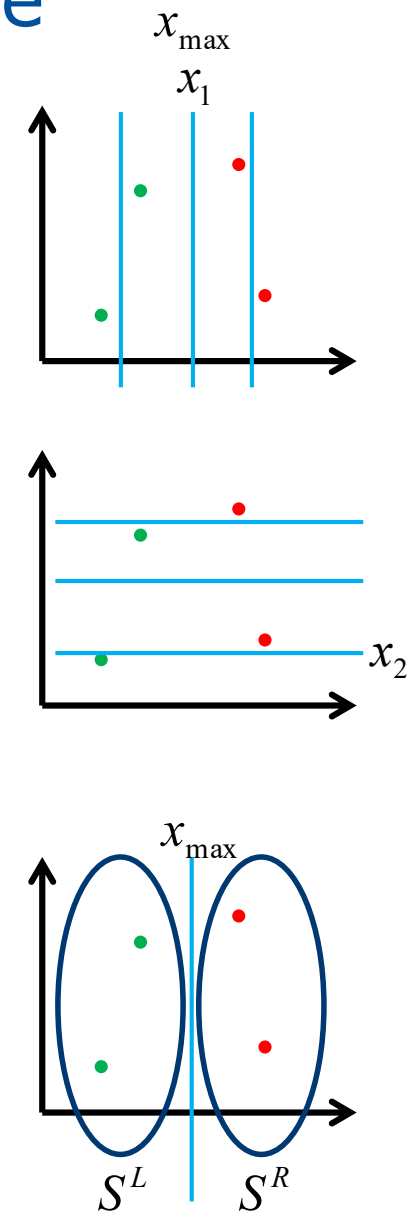
Algorithm for generating a decision tree

- Given: training data set S

```
Generate_Tree( S ):
for dim = 1 … n:
    determine the subdivision x_dim along
     the dimension which leads to the largest
     information gain
```

$$I = H(S) - \sum_{i=L,R} \frac{|S^i|}{|S|} H(S^i)$$

```
of those, select the best subdivision x_max
   (among all dimensions)
Subdivide along x_max into S^L and S^R
Generate_Tree( S^L )
Generate_Tree( S^R )
```

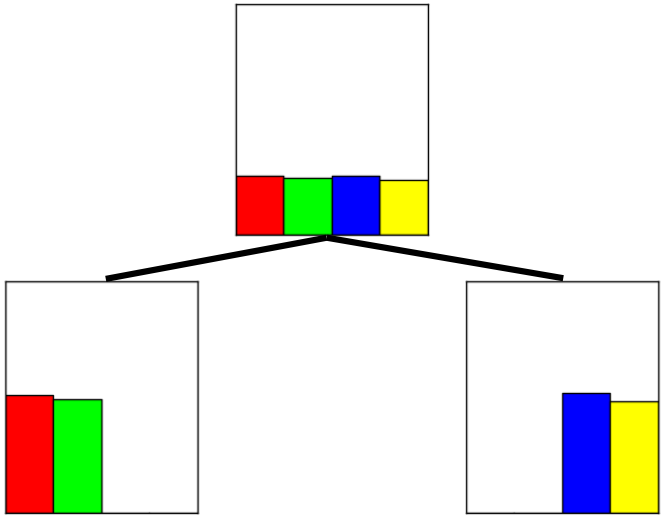


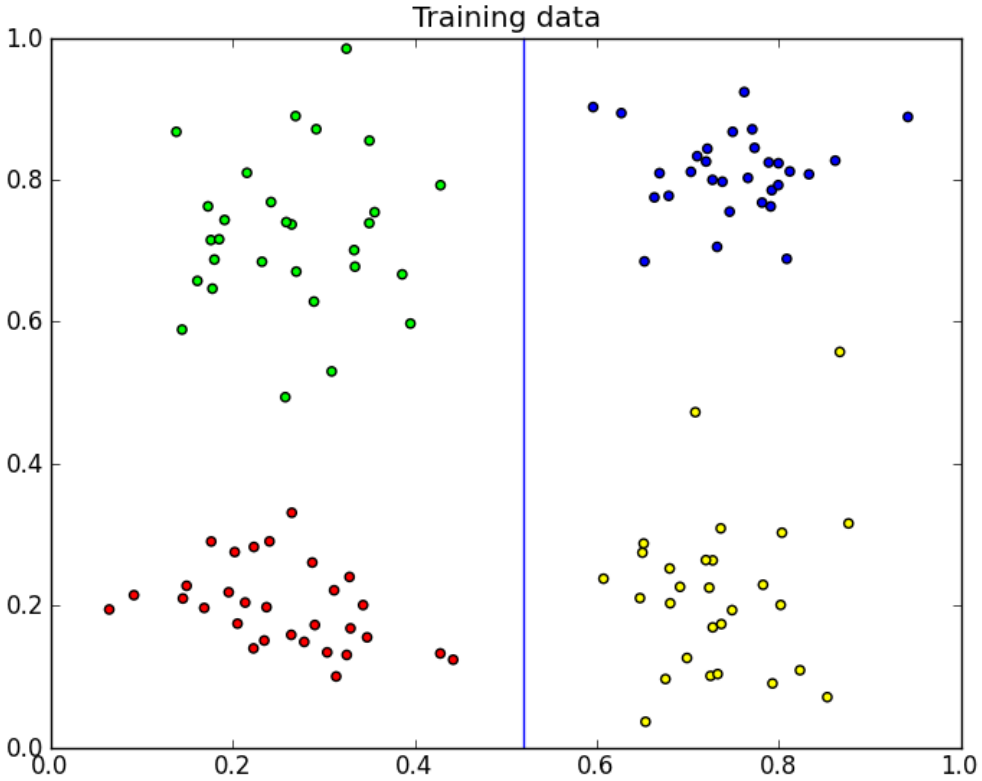
Termination criteria

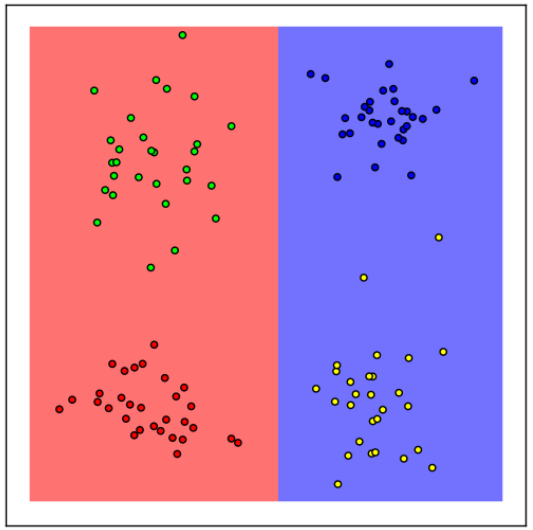
- When shall the algorithm terminate?
  - This is missing in the algorithm on the previous slide
- Possible criteria
  - After the decision tree reached a certain depth
  - If the set *S*, resulting from subdivision, is too small
  - If the information gain is below a threshold

Example: 2D, 4 classes

- Max. tree depth = 2

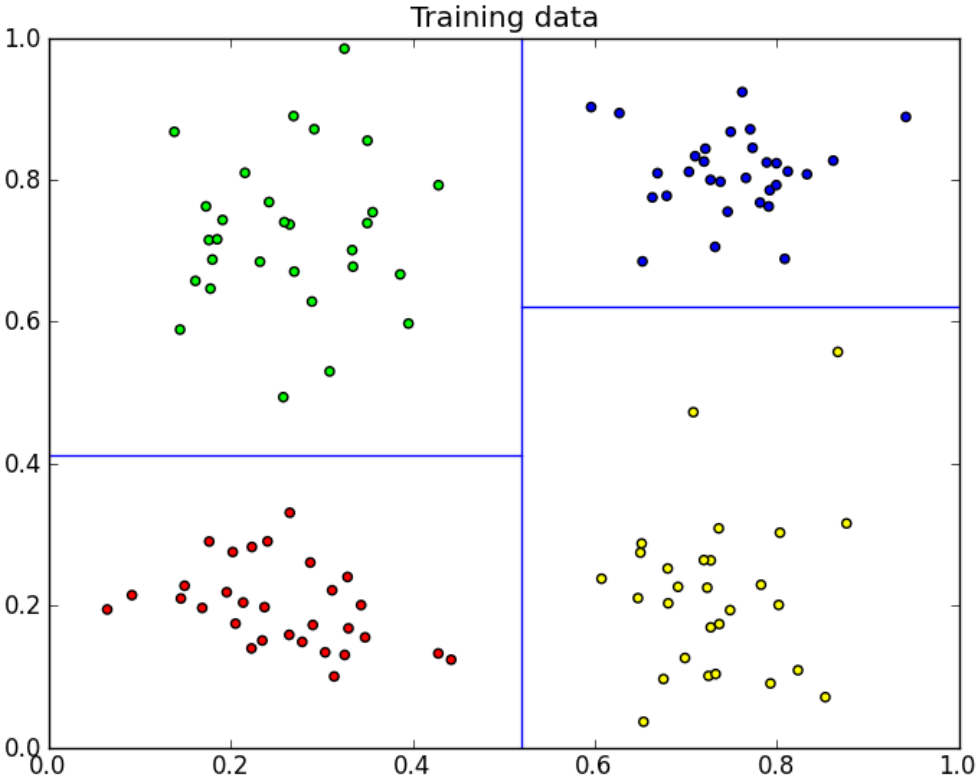


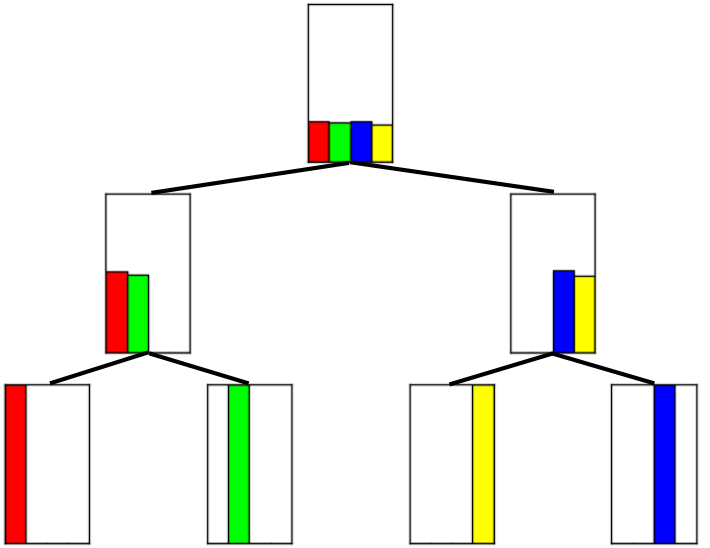


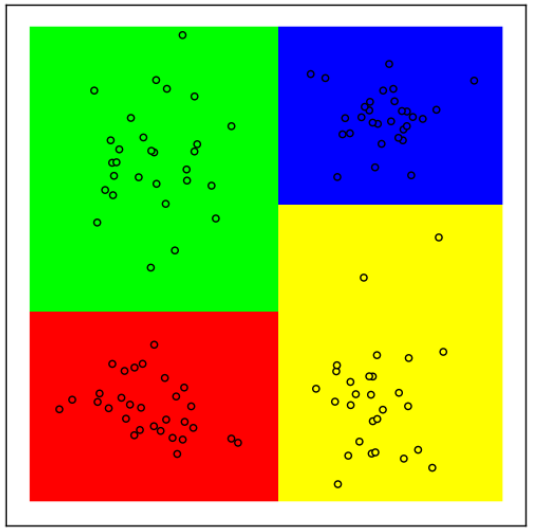


Example: 2D, 4 classes

- Max. tree depth = 3
- Tree depth > 3: no improvement

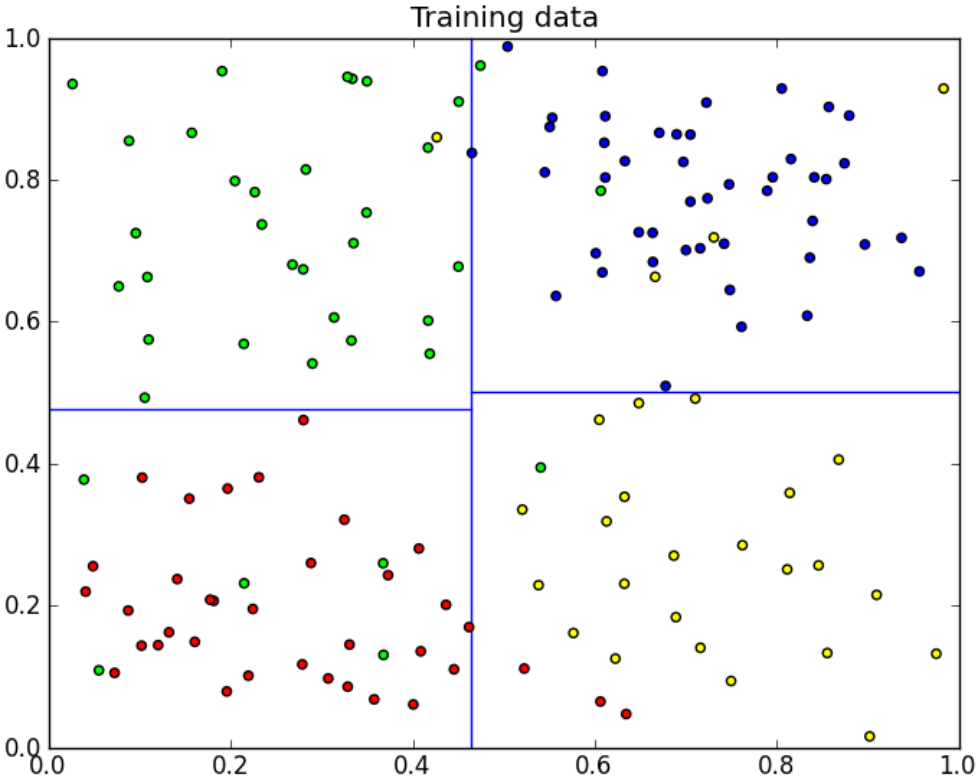


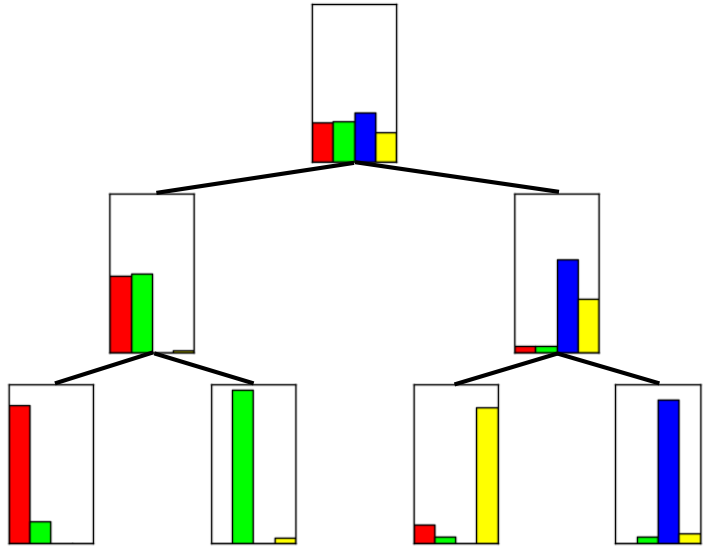


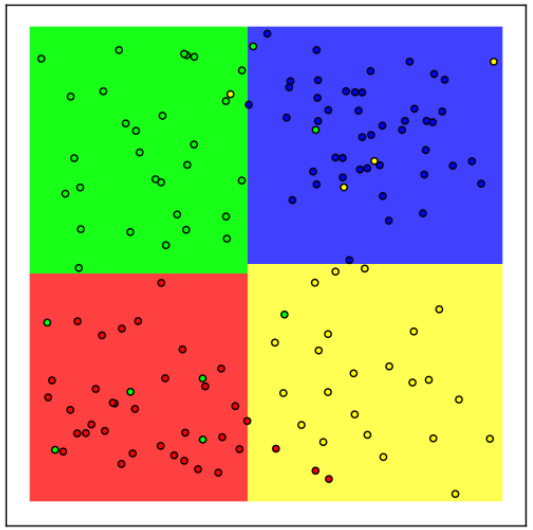


Example: 2D, 4 classes

- Now: overlapping training data
- Max. tree depth = 3

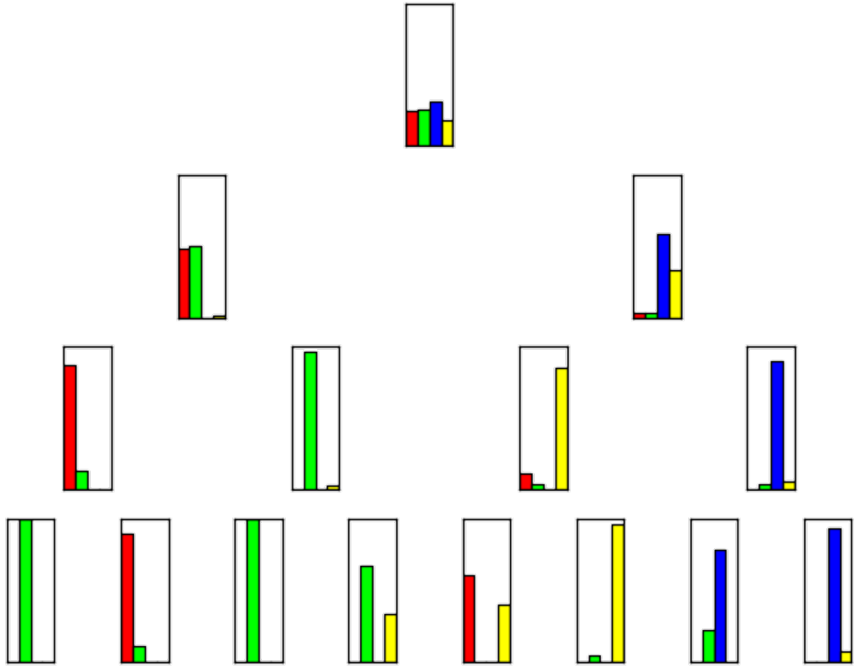


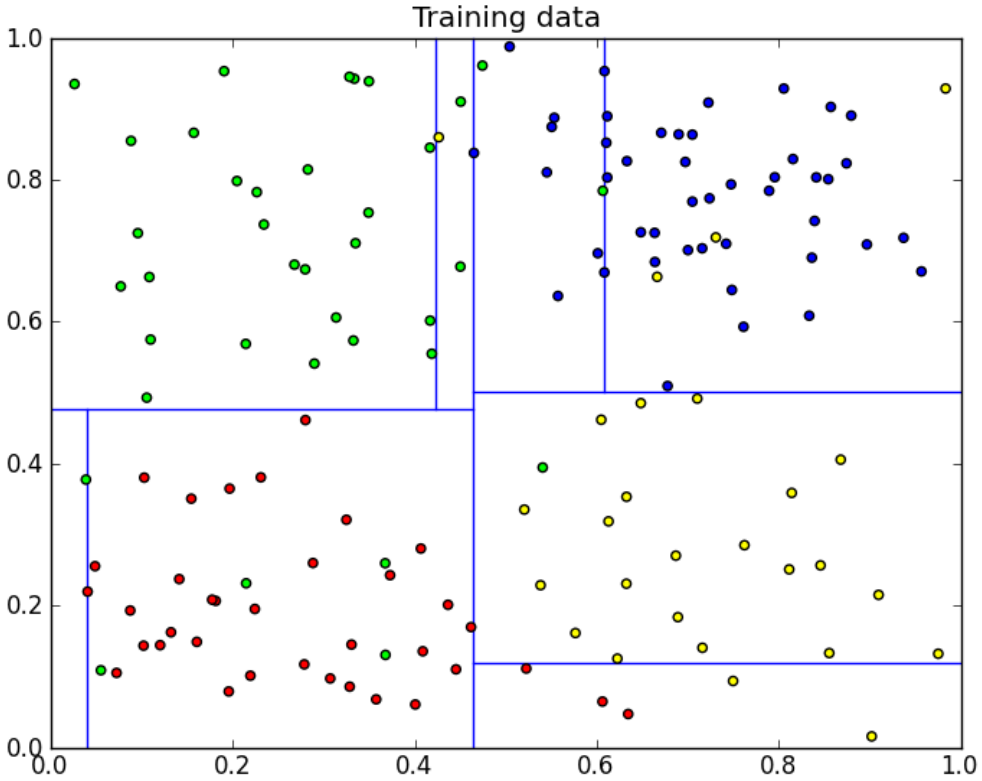


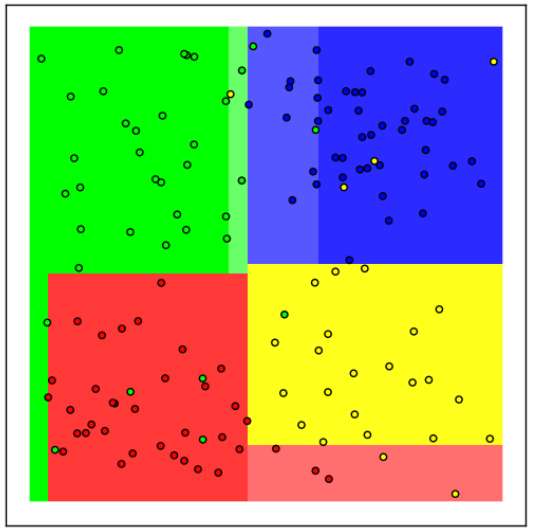


Example: 2D, 4 classes

- Max. tree depth = 4

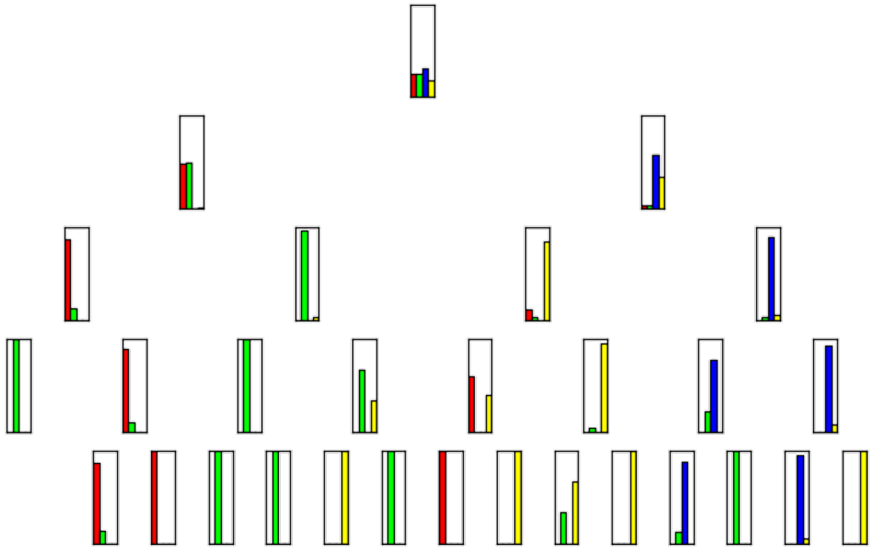


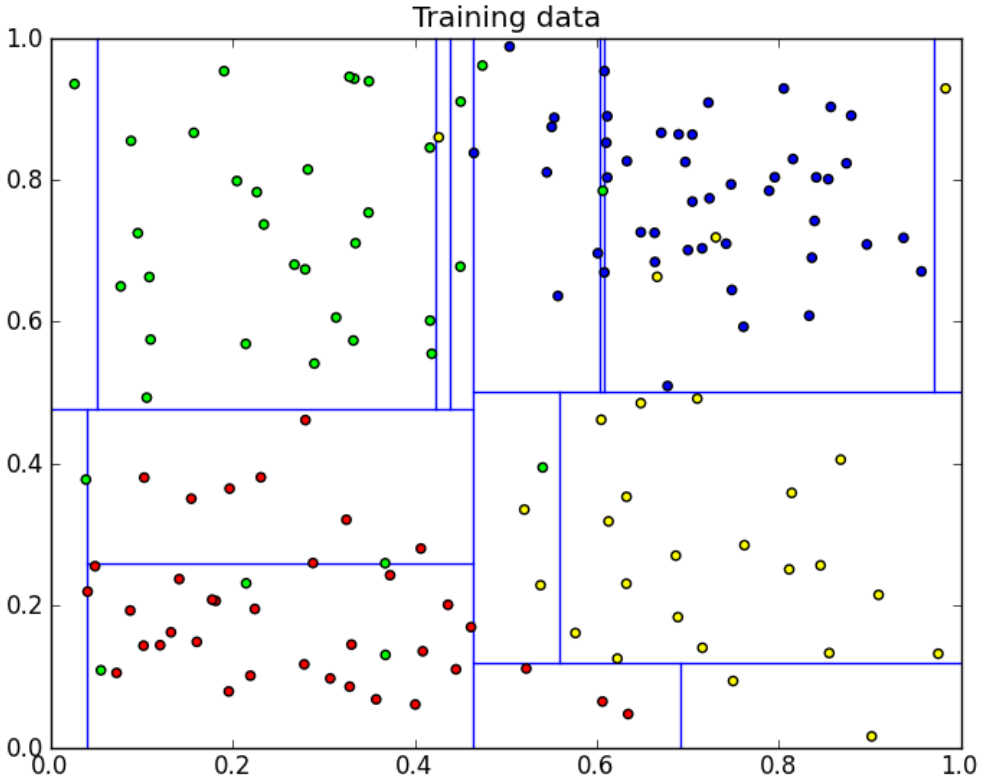


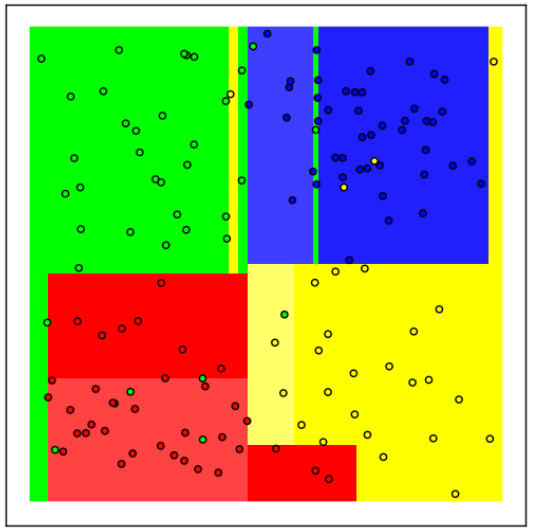


Example: 2D, 4 classes

- Max. tree depth = 5

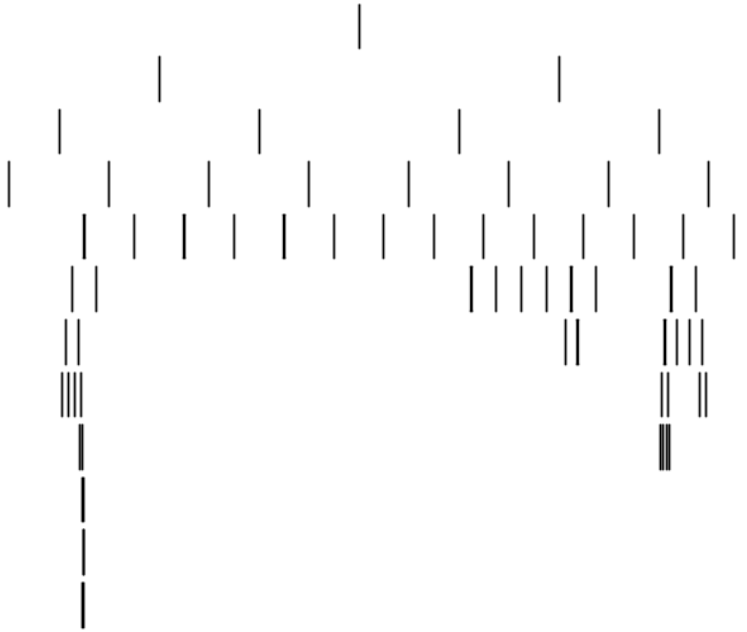


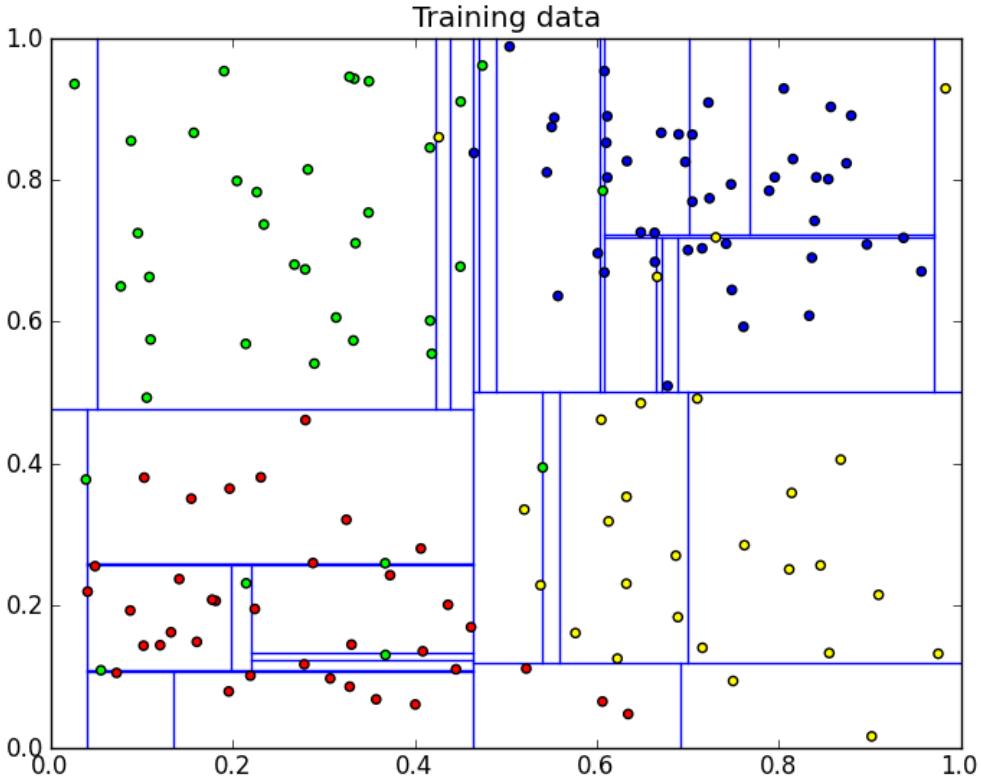


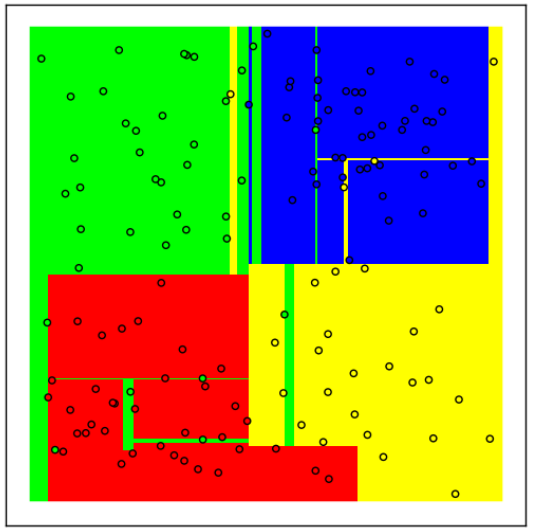


Example: 2D, 4 classes

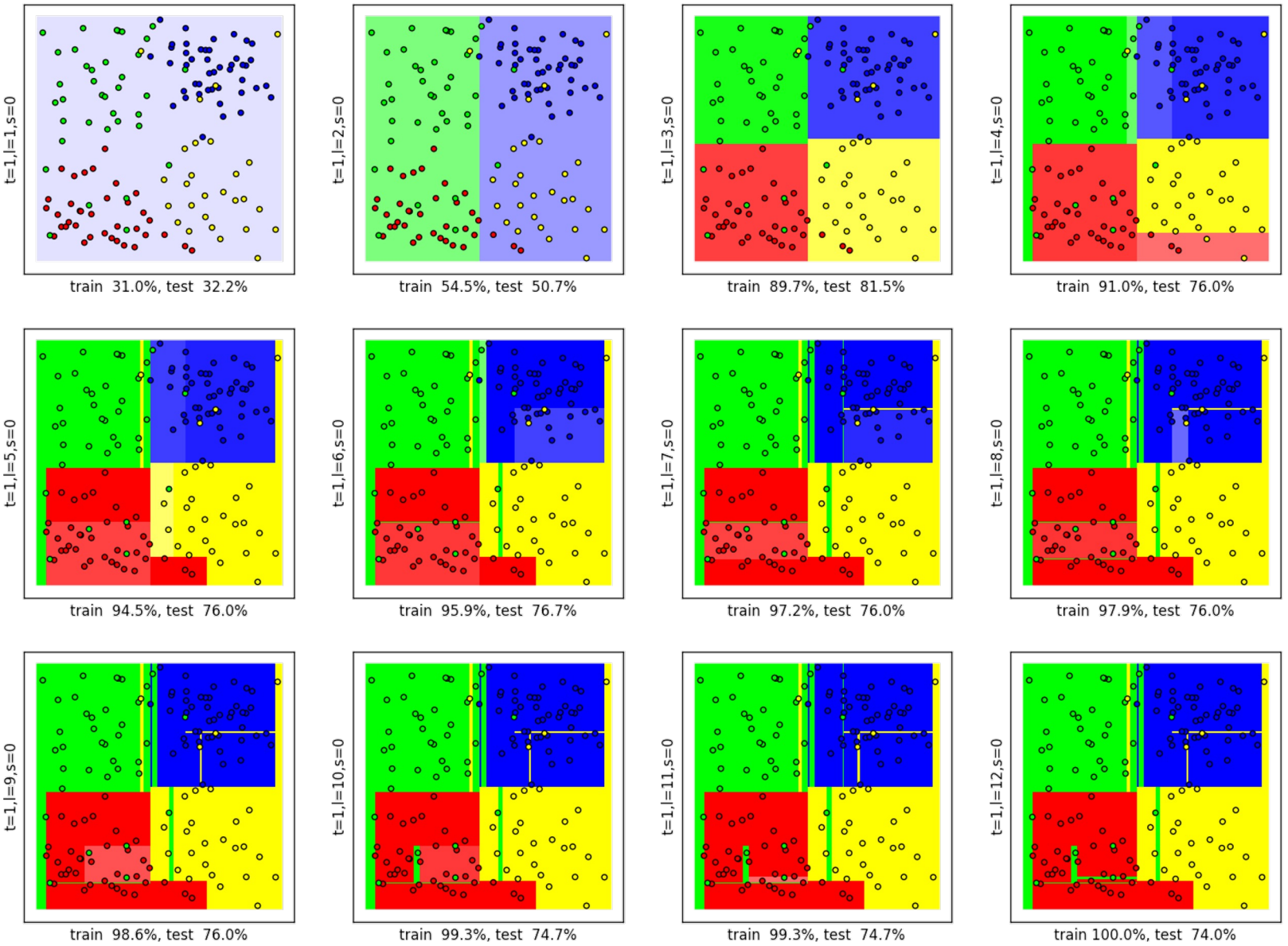
- Max. tree depth = 12





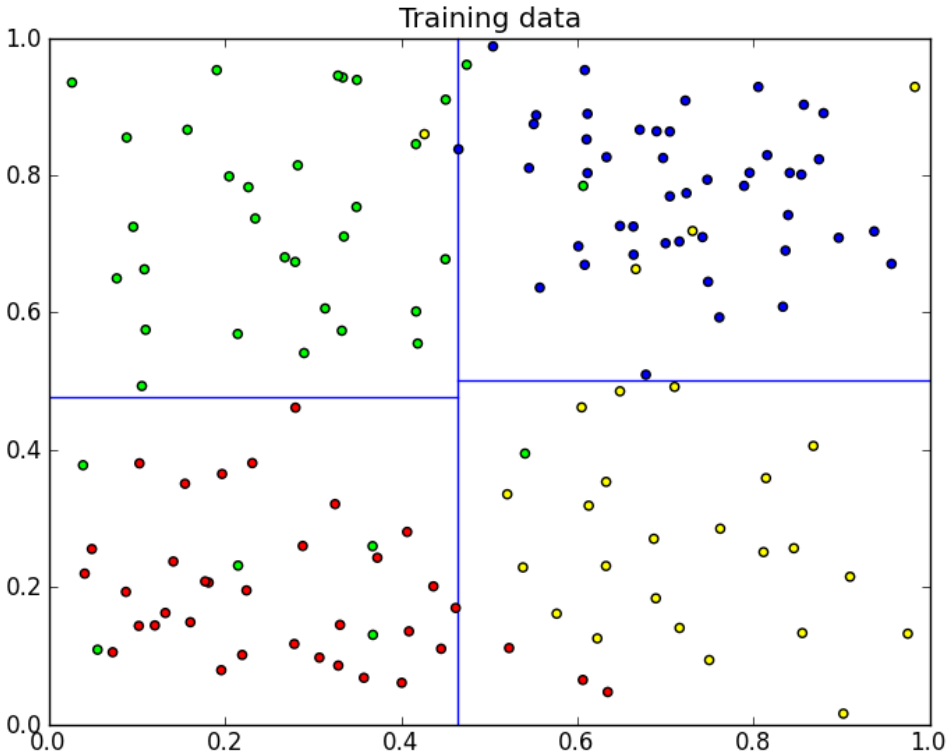


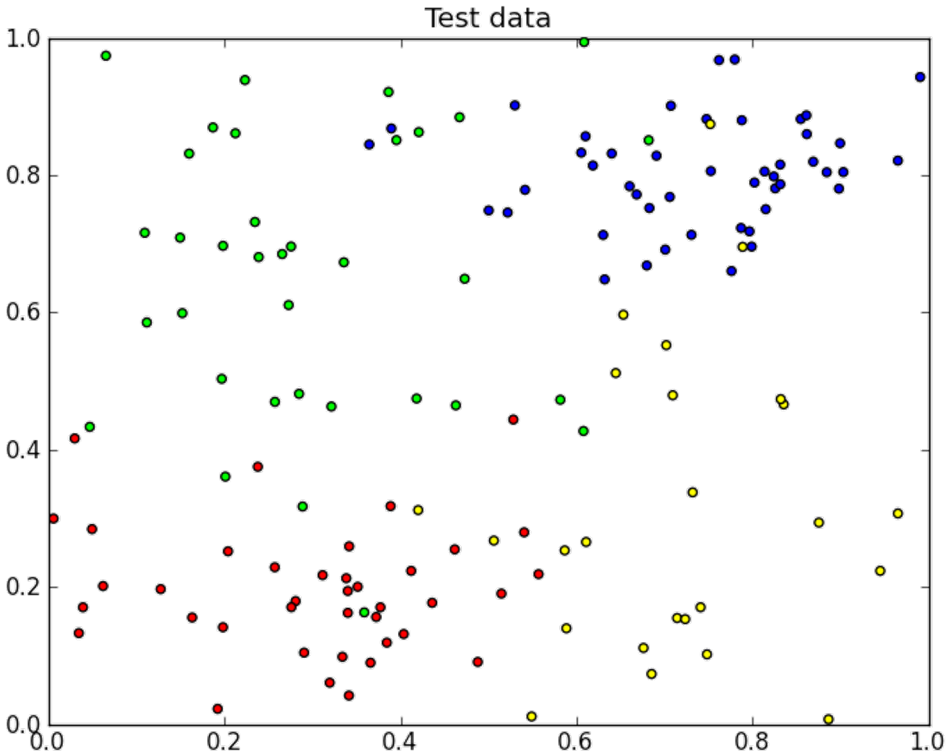
Variation of the tree depth from 1 to 12



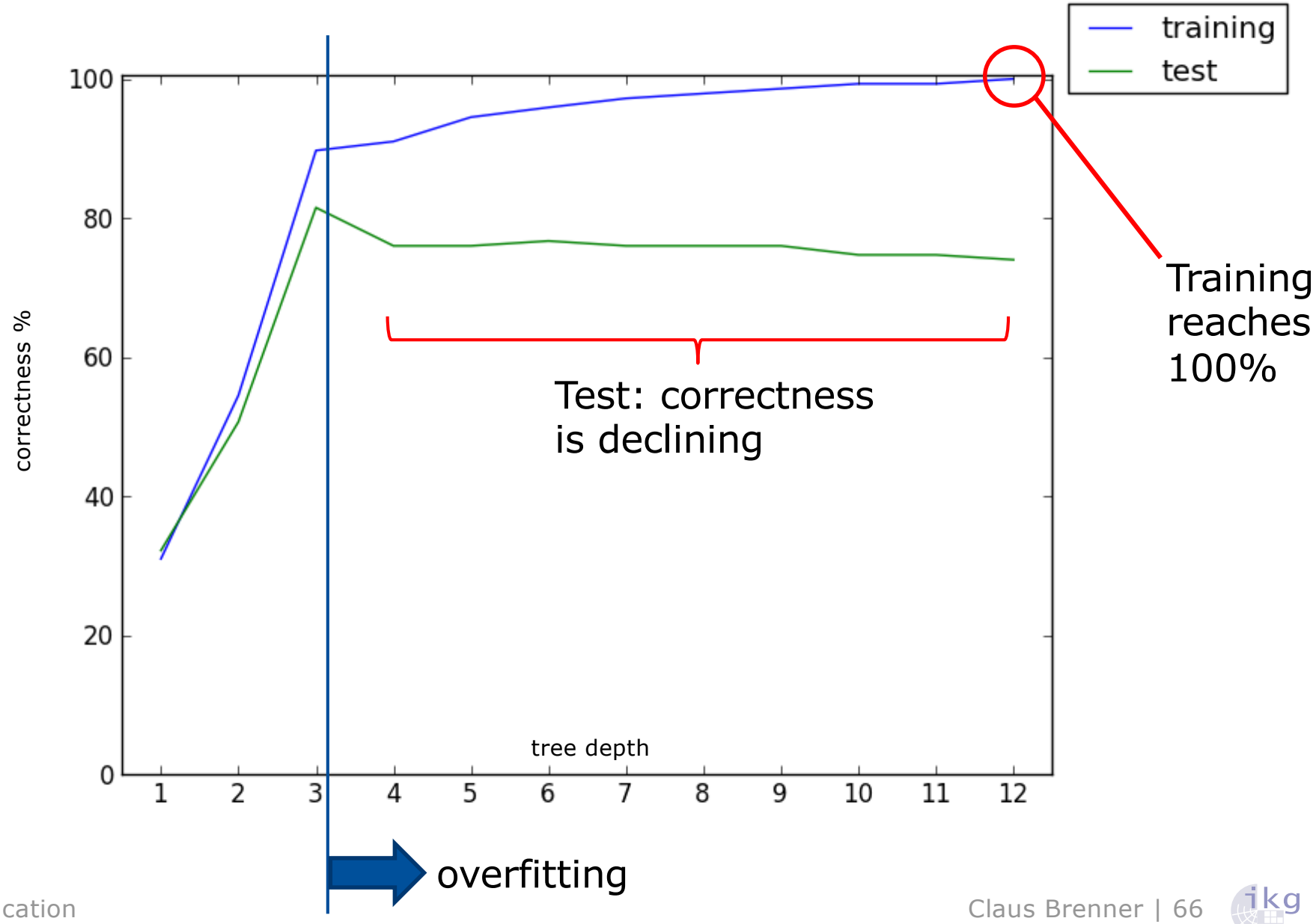
Test: using a test dataset

- Test dataset is drawn from the same distributions as the training dataset
- Determination of the percentage of correctly classified instances





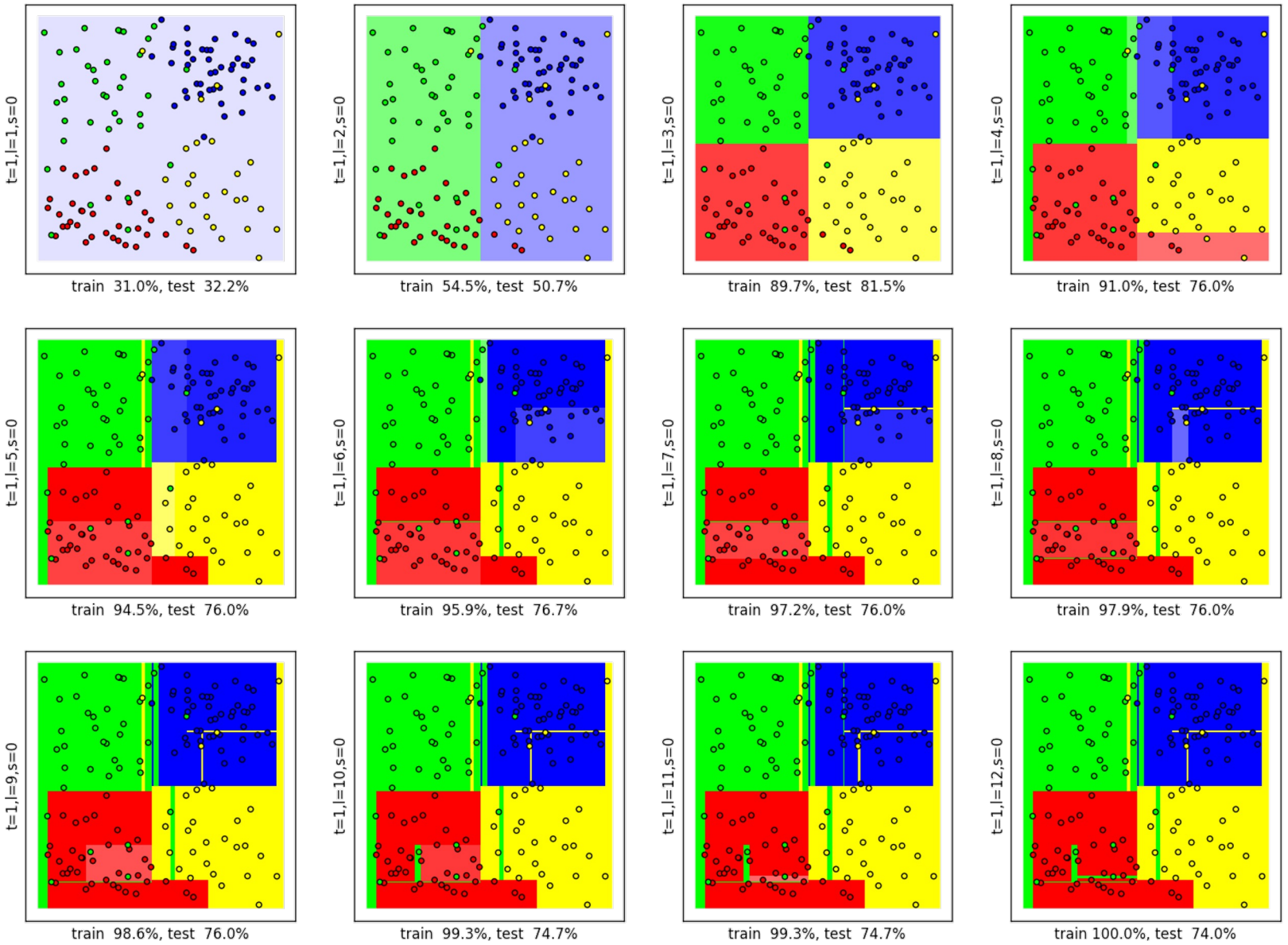
Training and test: correctness vs. number of tree levels



Overfitting

- With increasing tree depth, the model increases in complexity
  - The tree has the ability to describe many small regions
- At tree depth 12, the tree has learned the training data by heart
  - Note: “depth 12” means a maximum of 2^{11} leaf nodes, which corresponds to 2048 regions
- One also says that the classifier “does not generalize well”
- This can be seen from the application to test data
  - At depth 3: training 89.7%, test 81.5%
  - At depth 12: training 100%, test 74%

Revisited:
Variation of the tree depth from 1 to 12



References

- Criminisi, A., Shotton, J., Konukoglu, E. (2011): Decision Forests for Classification, Regression, Density Estimation, Manifold Learning and Semi-Supervised Learning. Microsoft Research technical report TR-2011-114
- Christopher M. Bishop (2007): Pattern Recognition and Machine Learning, Springer.
- Quinlan, J. R. (1993): C4.5: Programs for Machine Learning. Morgan Kaufmann Publishers
- Geurts, P., Ernst, D., Wehenkel, L. (2006): Extremely Randomized Trees, Machine Learning 36(1), 3-42.
- Shannon, C. E. (1948): A Mathematical Theory of Communication. The Bell System Technical Journal, Vol. 27, July/October, 379-423/623-656