Data Mining and Machine Learning
(Machine Learning: Symbolische Ansätze)

TECHNISCHE UNIVERSITÄT DARMSTADT

Learning Individual Rules and Subgroup Discovery

- Introduction
  - Batch Learning
  - Terminology
  - Coverage Spaces
- Algorithms
  - Top-Down Hill-Climbing
  - Bottom-Up Hill-Climbing
- Rule Evaluation Heuristics
  - Linear
  - Non-linear
- Descriptive vs. Predictive Rule Learning
  - Characteristic vs discriminative rules

A Sample Database

No.	Education	Marital S.	Sex.	Children?	Approved?
1	Primary	Single	M	N	-
2	Primary	Single	M	Y	-
3	Primary	Married	M	N	+
4	University	Divorced	F	N	+
5	University	Married	F	Y	+
6	Secondary	Single	M	N	-
7	University	Single	F	N	+
8	Secondary	Divorced	F	N	+
9	Secondary	Single	F	Y	+
10	Secondary	Married	M	Y	+
11	Primary	Married	F	N	+
12	Secondary	Divorced	M	Y	-
13	University	Divorced	F	Y	-
14	Secondary	Divorced	M	N	+

Property of Interest
(“class variable”)

Batch induction

- So far our algorithms looked at
  - all theories at the same time (implicitly through the version space)
  - and processed examples incrementally
- We can turn this around:
  - work on the theories incrementally
  - and process all examples at the same time
- Basic idea:
  - try to quickly find a complete and consistent rule
  - need not be in either S or G (but in the version space)

→ We can define an algorithm similar to FindG:
  - successively refine rule by adding conditions:
    - evaluate all refinements and pick the one that looks best
  - until the rule is consistent

Algorithm Batch-FindG

I. h = most general hypothesis in H
   C = set of all possible conditions
II. while h covers negative examples
   I. $h_{best} = h$
   II. for each possible condition $c \in C$
      a) $h' = h \cup \{c\}$
      b) if h' covers
         - all positive examples
         - and fewer negative examples than h_{best}

         then $h_{best} = h'$
   III. $h = h_{best}$
III. return h_{best}

Scan through all examples in database:
- count covered positives
- count covered negatives

Evaluation of a rule by # covered positive and # covered negative examples

Properties

- General-to-Specific (Top-Down) Search
  - similar to FindG:
    - **FindG** makes an arbitrary selection among possible refinements, taking the risk that it may lead to an inconsistency later
    - **Batch-FindG** selects next refinement based on all training examples
- Heuristic algorithm
  - among all possible refinements, we select the one that leads to the fewest number of covered negatives
    - IDEA: the more negatives are excluded with the current condition, the less have to be excluded with subsequent conditions
- Converges towards some theory in V
  - not necessarily towards a theory in G
- Not very efficient, but quite flexible
  - criteria for selecting conditions could be exchanged

Algorithms for Learning a Single Rule

Objective:

- Find the best rule according to some measure h

Algorithms

- Greedy search
  - top-down hill-climbing or beam search
    - successively add conditions that increase value of h
    - most popular approach
- Exhaustive search
  - efficient variants
    - avoid to search permutations of conditions more than once
    - exploit monotonicity properties for pruning of parts of the search space
- Randomized search
  - genetic algorithms etc.

Top-Down Hill-Climbing

Top-Down Strategy: A rule is successively *specialized*

1. Start with the universal rule R that covers all examples
2. Evaluate all possible ways to add a condition to R
3. Choose the best one (according to some heuristic)
4. If R is satisfactory, return it
5. Else goto 2.

- Most greedy s&c rule learning systems use a top-down strategy

Beam Search:

- Always remember (and refine) the best b solutions in parallel

Terminology

TECHNISCHE UNIVERSITÄT DARMSTADT

- training examples
  - P: total number of positive examples
  - N: total number of negative examples
- examples covered by the rule (predicted positive)
  - true positives p: positive examples covered by the rule
  - false positives n: negative examples covered by the rule
- examples not covered the rule (predicted negative)
  - false negatives P-p: positive examples not covered by the rule
  - true negatives N-n: negative examples not covered by the rule

	predicted +	predicted -	
class +	p **(true positives)**	P-p **(false negatives)**	P
class -	n **(false positives)**	N-n **(true negatives)**	N
	$p + n$	$P+N-(p+n)$	$P+N$

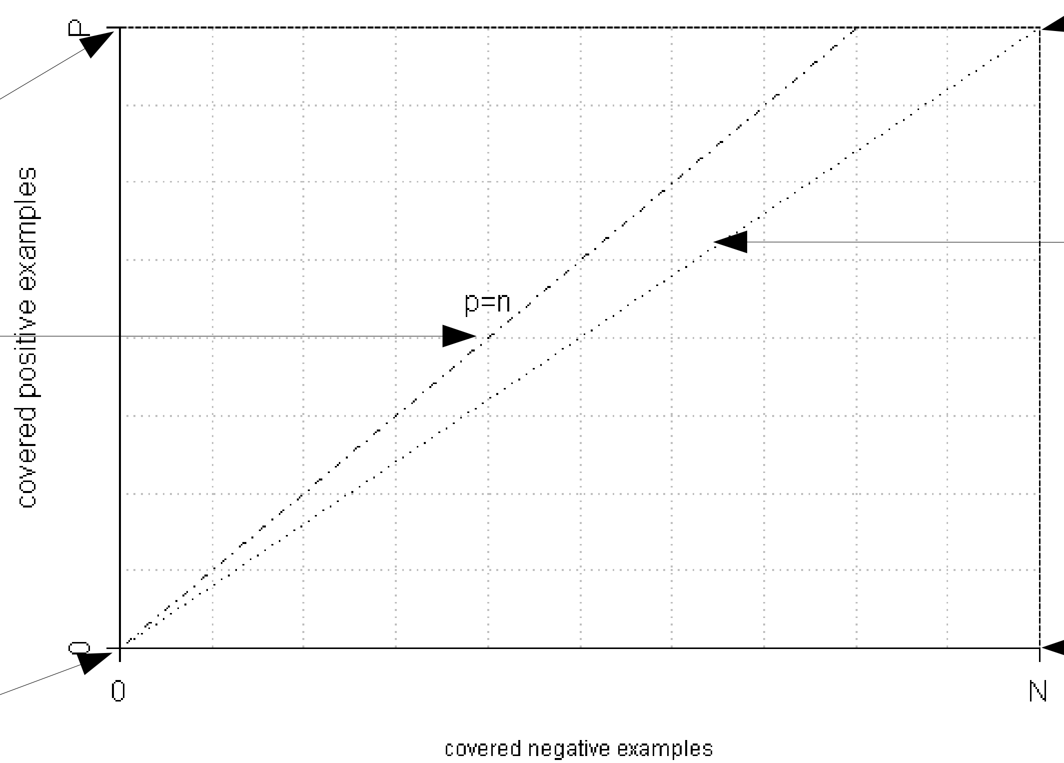
Coverage Spaces

- good tool for visualizing properties of covering algorithms
  - each point is a theory covering p positive and n negative examples

perfect theory: all positive and no negative examples are covered

iso-accuracy: cover same amount of positive and negative examples

empty theory: no examples are covered (most specific)



universal theory: all examples are covered (most general)

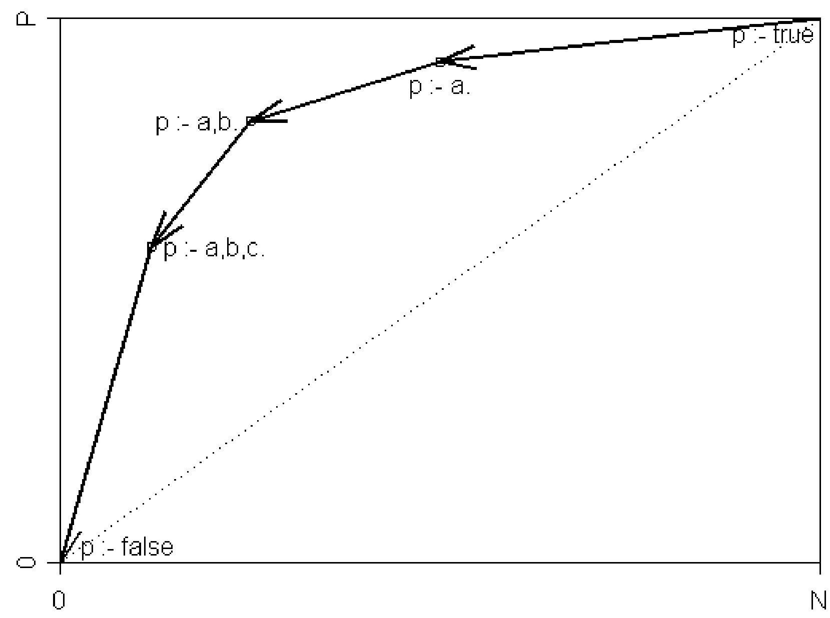
Random theories: maintain $P/(P+N)\%$ true positive and $N/(P+N)\%$ false positive examples,

opposite theory: all negative and no positive examples are covered

Top-Down Hill-Climbing in Coverage Space

- successively extends a rule by adding conditions

- This corresponds to a path in coverage space:
  - The rule `p:-true` covers all examples (universal theory)
  - Adding a condition never increases *p* or *n* (specialization)
  - The rule `p:-false` covers no examples (empty theory)



- which conditions are selected depends on a *heuristic function* that estimates the quality of the rule

Rule Learning Heuristics

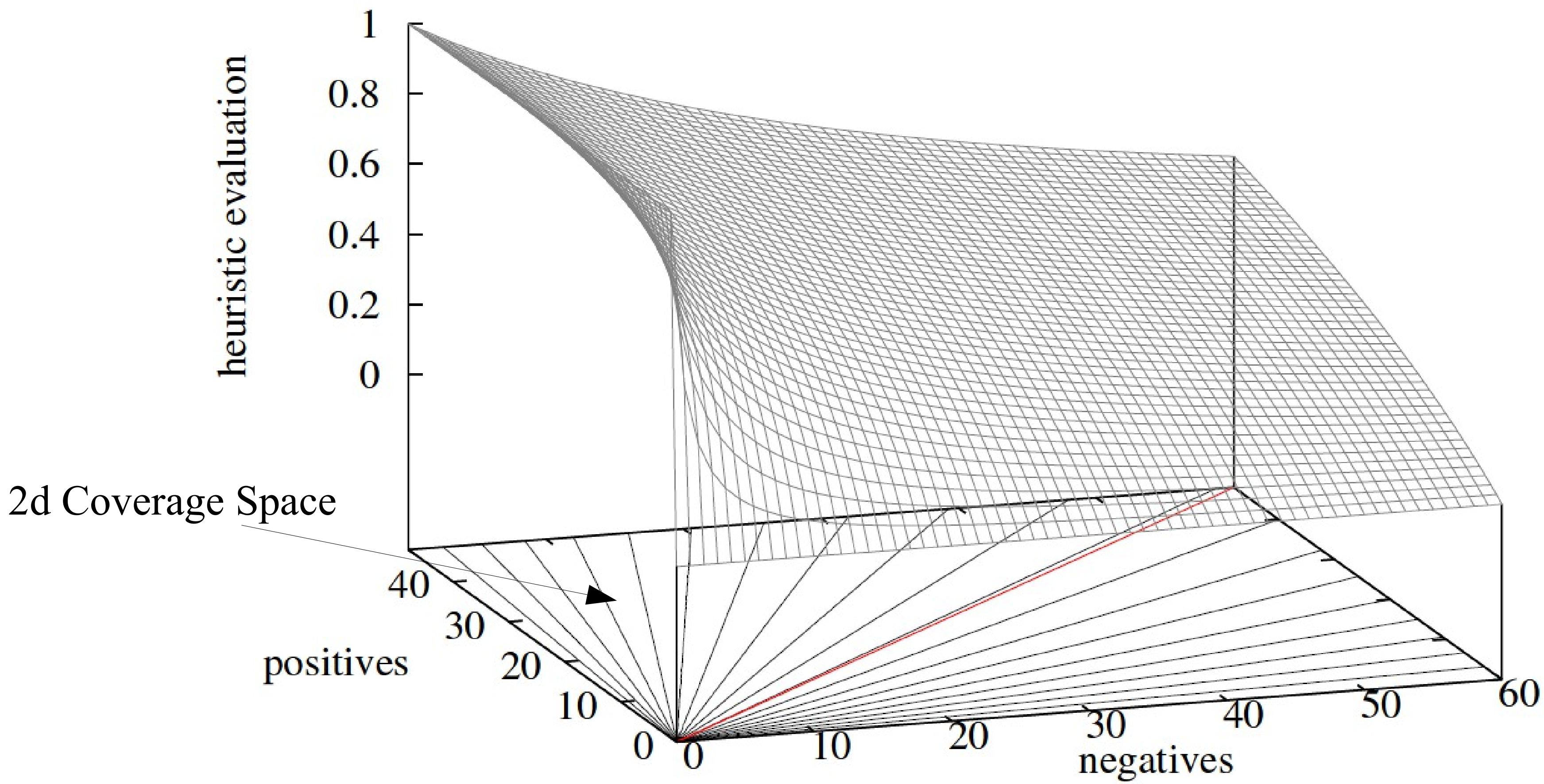
- Adding a rule should
  - increase the number of covered negative examples as little as possible (do not decrease *consistency*)
  - increase the number of covered positive examples as much as possible (increase *completeness*)
- An evaluation heuristic should therefore trade off these two extremes
  - Example: Laplace heuristic $h_{Lap} = \frac{p+1}{p+n+2}$
    - grows with $p \rightarrow \infty$
    - grows with $n \rightarrow 0$
  - Example: Precision $h_{Prec} = \frac{p}{p+n}$
    - is not a good heuristic. Why?

Example

Condition		p	n	Precision	Laplace	p-n
Temperature =	Hot	2	2	0.5000	0.5000	0
	Mild	3	1	0.7500	0.6667	2
	Cold	4	2	0.6667	0.6250	2
Outlook =	Sunny	2	3	0.4000	0.4286	-1
	Overcast	4	0	1.0000	0.8333	4
	Rain	3	2	0.6000	0.5714	1
Humidity =	High	3	4	0.4286	0.4444	-1
	Normal	6	1	0.8571	0.7778	5
Windy =	True	3	3	0.5000	0.5000	0
	False	6	2	0.7500	0.7000	4

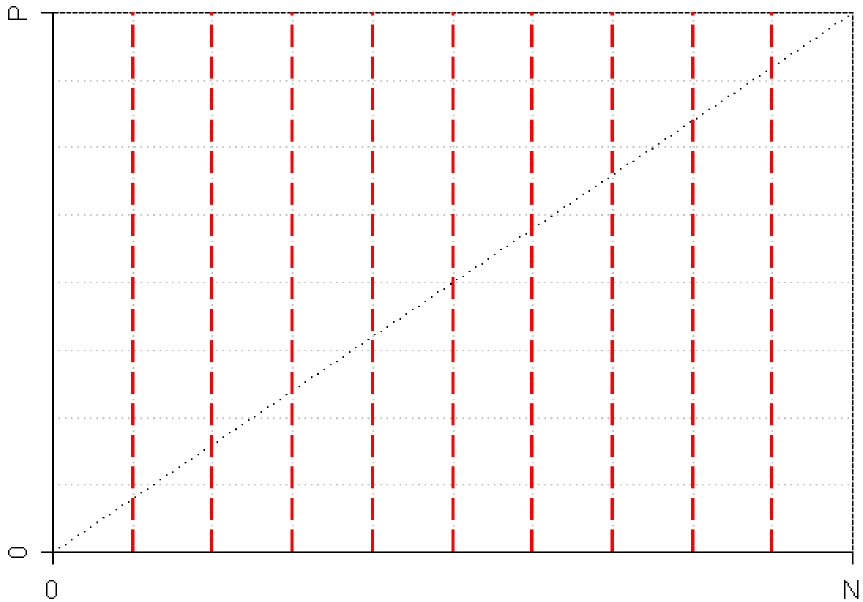
- Heuristics Precision and Laplace
  - add the condition Outlook= Overcast to the (empty) rule
  - stop and try to learn the next rule
- Heuristic Accuracy / $p - n$
  - adds Humidity = Normal
  - continue to refine the rule (until no covered negative)

3d-Visualization of Precision



Isometrics in Coverage Space

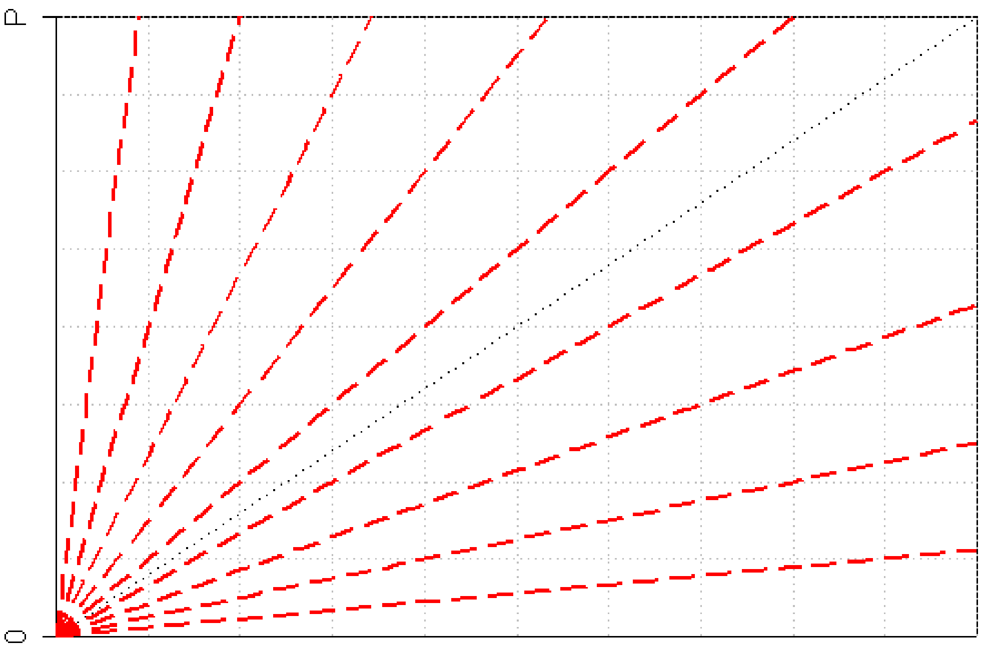
- Isometrics are lines that connect points for which a function in p and n has equal values
  - *Examples*:
    Isometrics for heuristics $h_p = p$ and $h_n = -n$



Precision (Confidence)

$$h_{Prec} = \frac{p}{p+n}$$

- *basic idea:* percentage of positive examples among covered examples
- effects:
  - rotation around origin (0,0)
  - all rules with same angle equivalent
  - in particular, all rules on P/N axes are equivalent



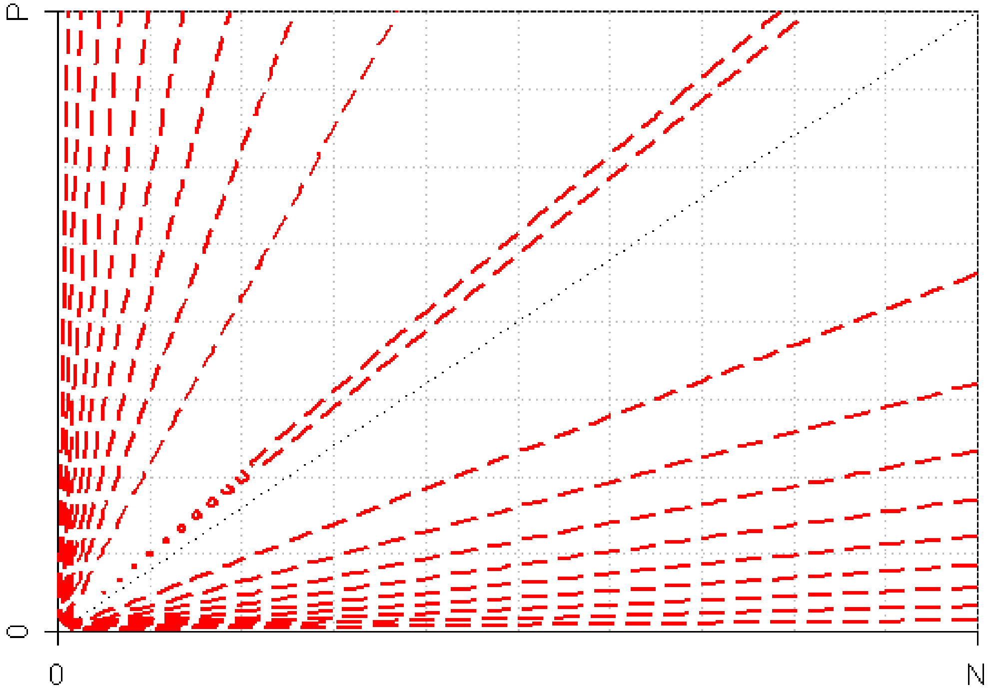
Entropy and Gini Index

$$h_{Ent} = -\left(\frac{p}{p+n}\log_2\frac{p}{p+n} + \frac{n}{p+n}\log_2\frac{n}{p+n}\right)$$

$$h_{Gini} = 1 - \left(\frac{p}{p+n}\right)^2 - \left(\frac{n}{p+n}\right)^2 \simeq \frac{pn}{(p+n)^2}$$

These will be explained later (decision trees)

- *effects:*
  - entropy and Gini index are equivalent
  - like precision, isometrics rotate around (0,0)
  - isometrics are symmetric around 45º line
  - a rule that only covers negative examples is as good as a rule that only covers positives

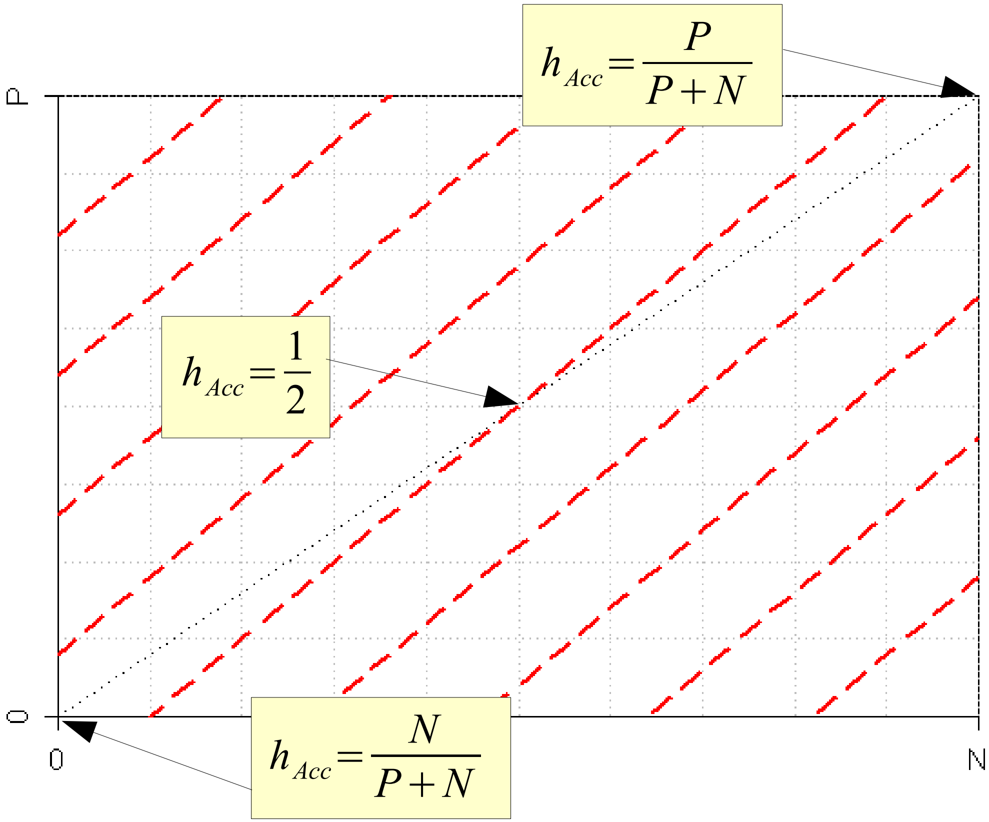


Accuracy

$$h_{Acc} = \frac{p + (N - n)}{P + N} \simeq p - n$$

Why are they equivalent?

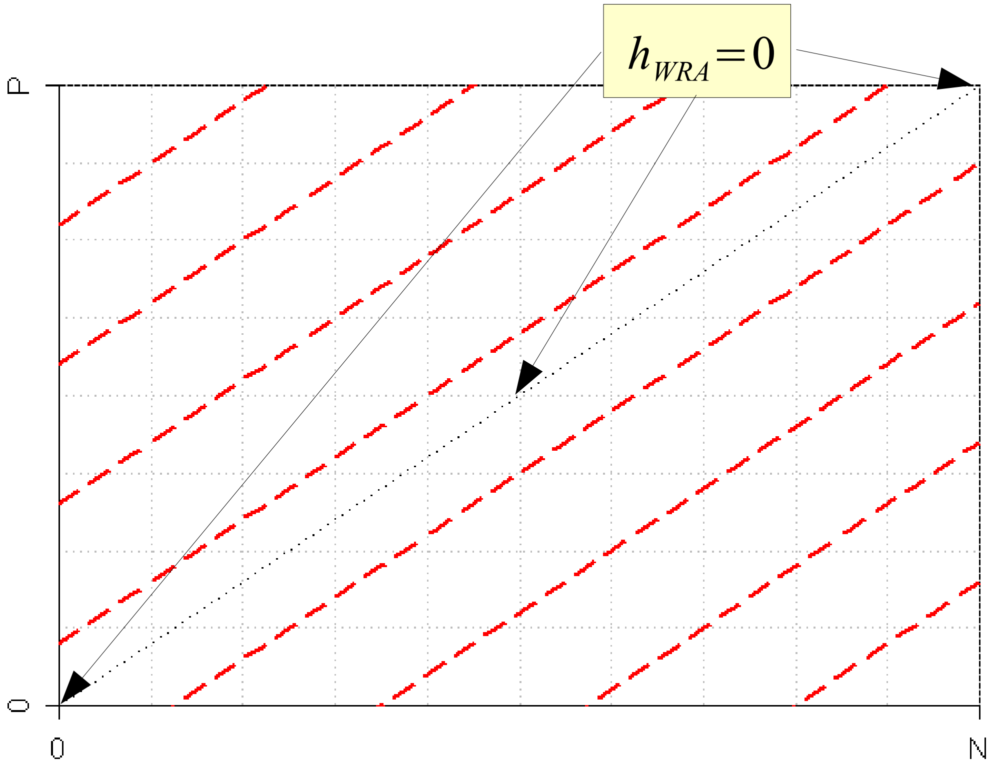
- *basic idea:*
  percentage of correct classifications
  (*covered positives* plus *uncovered negatives*)
- *effects:*
  - isometrics are parallel to 45° line
  - covering one positive example is as good as not covering one negative example



Weighted Relative Accuracy

$$h_{WRA} = \frac{p+n}{P+N}\left(\frac{p}{p+n} - \frac{P}{P+N}\right) \simeq \frac{p}{P} - \frac{n}{N}$$

- *basic idea:*
  normalize accuracy with the class distribution
- *effects:*
  - isometrics are parallel to diagonal
  - covering $x\%$ of the positive examples is considered to be as good as not covering $x\%$ of the negative examples



Weighted Relative Accuracy

- Two Basic ideas:
  - **Precision Gain:** compare precision to precision of a rule that classifies all examples as positive

$$\frac{p}{p+n}-\frac{P}{P+N}$$

  - **Coverage:** Multiply with the percentage of covered examples

$$\frac{p+n}{P+N}$$

- Resulting formula:

$$h_{WRA}=\frac{p+n}{P+N}\cdot\left(\frac{p}{p+n}-\frac{P}{P+N}\right)$$

  - one can show that sorts rules in exactly the same way as

$$h_{WRA}'=\frac{p}{P}-\frac{n}{N}$$

Linear Cost Metric

- Accuracy and weighted relative accuracy are only two special cases of the general case with linear costs:
  - costs c mean that covering 1 positive example is as good as not covering $c/(1-c)$ negative examples

c	*measure*
½	accuracy
$N/(P+N)$	weighted relative accuracy
0	excluding negatives at all costs
1	covering positives at all costs

- The general form is then $h_{cost} = c \cdot p - (1-c) \cdot n$
  - the isometrics of h_{cost} are parallel lines with slope $(1-c)/c$

Relative Cost Metric

- Defined analogously to the Linear Cost Metric
- Except that the trade-off is between the normalized values of p and n
  - between true positive *rate* p/P and false positive *rate* n/N

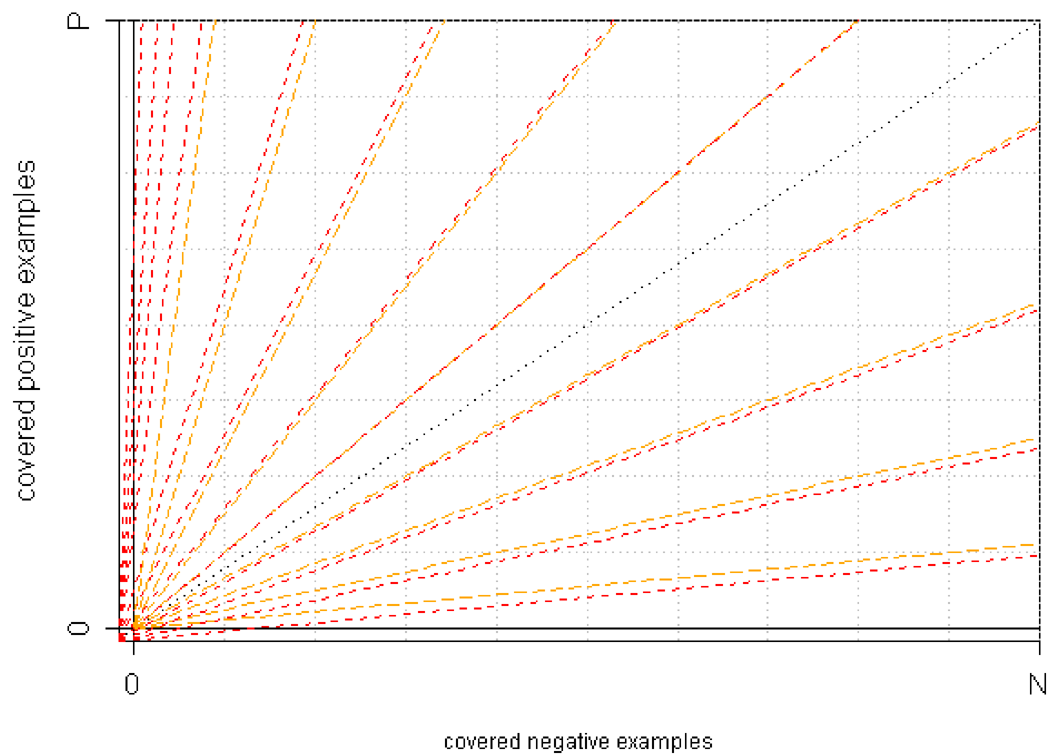
- The general form is then $h_{rcost} = c \cdot \frac{p}{P} - (1-c) \cdot \frac{n}{N}$
  - the isometrics of h_{cost} are parallel lines with slope $(1\text{-}c)/c$

- The plots look the same as for the linear cost metric
  - but the semantics of the c value is different:
    - for h_{cost} it does not include the example distribution
    - for h_{rcost} it includes the example distribution

Laplace-Estimate

$$h_{Lap} = \frac{p+1}{(p+1)+(n+1)} = \frac{p+1}{p+n+2}$$

- *basic idea:*
  precision, but count coverage for positive and negative examples starting with 1 instead of 0
- *effects:*
  - origin at $(-1,-1)$
  - different values on $p=0$ or $n=0$ axes
  - not equivalent to precision

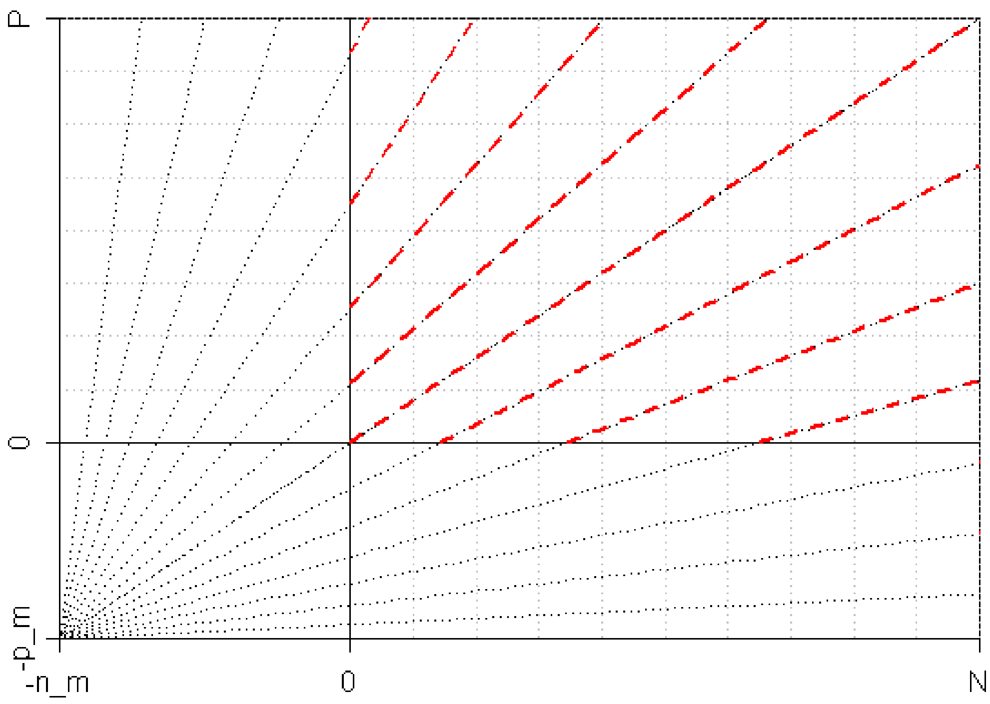


m-Estimate

TECHNISCHE UNIVERSITÄT DARMSTADT

- *basic idea:*
  initialize the counts with m examples in total, distributed according to the prior distribution $P/(P+N)$ of p and n.
- *effects:*
  - origin shifts to $(-mP/(P+N), -mN/(P+N))$
  - with increasing m, the lines become more and more parallel
  - can be re-interpreted as a trade-off between WRA and precision/confidence

$$h_m = \frac{p + m\frac{P}{P+N}}{\left(p + m\frac{P}{P+N}\right) + \left(n + m\frac{N}{P+N}\right)} = \frac{p + m\frac{P}{P+N}}{p+n+m}$$



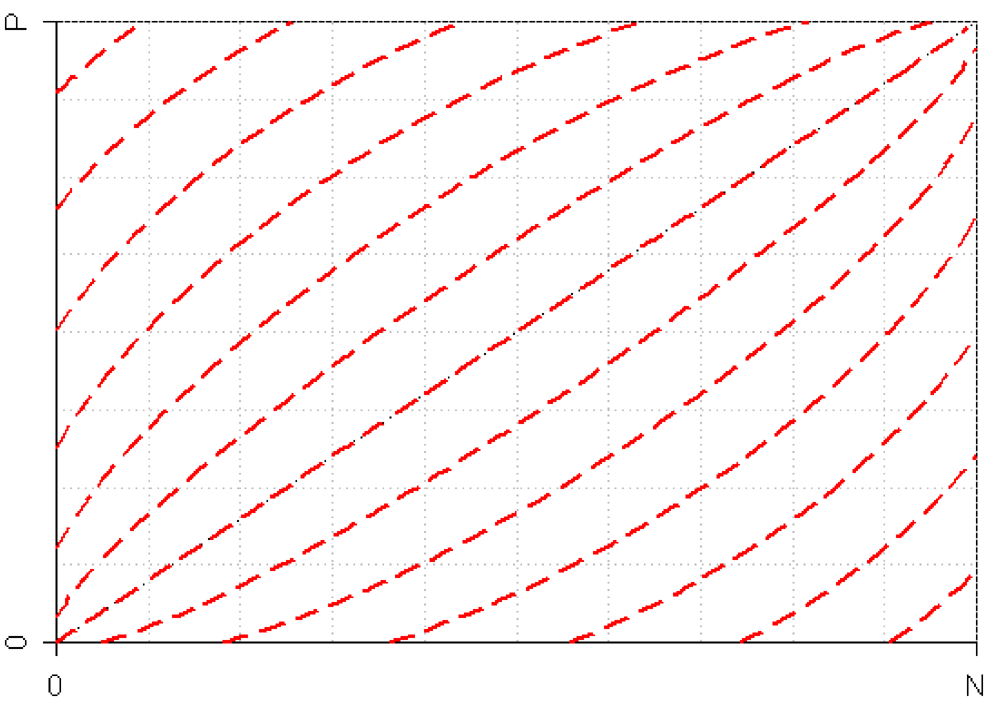
Generalized m-Estimate

- One can re-interpret the m-Estimate:
  - Re-interpret $c = N/(P+N)$ as a cost factor like in the general cost metric
  - Re-interpret m as a trade-off between precision and cost-metric
    - $m = 0$: precision (independent of cost factor)
    - $m \rightarrow \infty$: the isometrics converge towards the parallel isometrics of the cost metric
- Thus, the generalized m-Estimate may be viewed as a means of trading off between precision and the cost metric

Correlation

$$h_{Corr} = \frac{p(N-n)-(P-p)n}{\sqrt{PN(p+n)(P-p+N-n)}}$$

- *basic idea:*
  measure correlation coefficient of predictions with target
- *effects:*
  - non-linear isometrics
  - in comparison to WRA
    - prefers rules near the edges
    - steepness of connection of intersections with edges increases
  - equivalent to χ^2

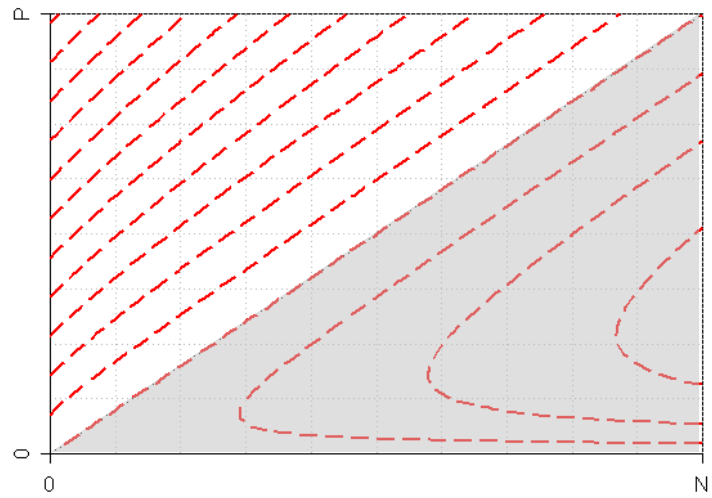


Foil Gain

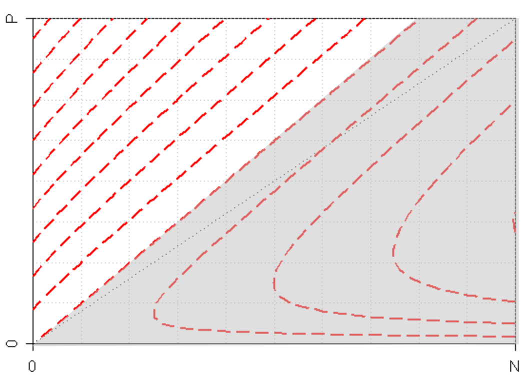
$$h_{foil} = -p\left(\log_2 c - \log_2 \frac{p}{p+n}\right)$$

(c is the precision of the parent rule)

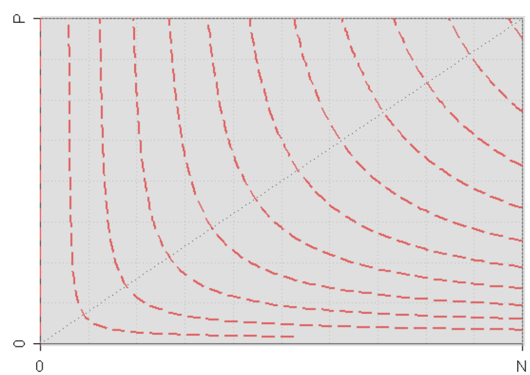
c = P/(P+N)



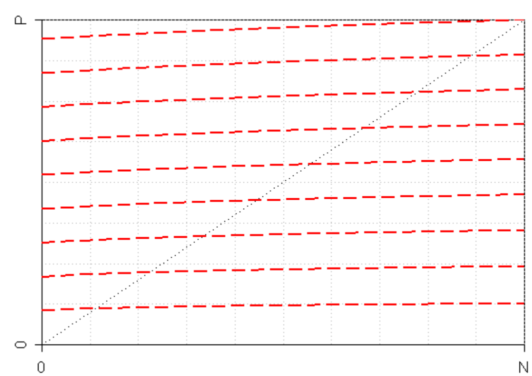
c = 1/2



c = 1



c = 10^-6



Myopy of Top-Down Hill-Climbing

- Parity problems (e.g. XOR)
  - r relevant binary attributes
  - s irrelevant binary attributes
  - each of the $n = r + s$ attributes has values 0/1 with probability ½
  - an example is positive if the number of 1's in the relevant attributes is even, negative otherwise
- Problem for top-down learning:
  - by construction, each condition of the form $a_i = 0$ or $a_i = 1$ covers approximately 50% positive and 50% negative examples
  - irrespective of whether a_i is a relevant or an irrelevant attribute
    - ➔ top-down hill-climbing cannot learn this type of concept
- Typical recommendation:
  - use *bottom-up learning* for such problems

Bottom-Up Hill-Climbing

- Simple inversion of top-down hill-climbing
- A rule is successively *generalized*

1. Start with ~~an empty~~ a fully specialized rule R that covers ~~all examples~~ a single example
2. Evaluate all possible ways to ~~add~~ delete a condition to R
3. Choose the best one
4. If R is satisfactory, return it
5. Else goto 2.

A Pathology of Bottom-Up Hill-Climbing

	att1	*att2*	*att3*
+	1	1	1
+	1	0	0
–	0	1	0
–	0	0	1

- Target concept $att1 = 1$ is not (reliably) learnable with bottom-up hill-climbing
  - because no generalization of any seed example will increase coverage
  - Hence you either stop or make an arbitrary choice (e.g., delete attribute 1)

Bottom-Up Rule Learning Algorithms

- AQ-type:
  - select a seed example and search the space of its generalizations
  - BUT: search this space top-down
  - Examples: AQ (Michalski 1969), Progol (Muggleton 1995)
- based on least general generalizations (lggs)
  - greedy bottom-up hill-climbing
  - BUT: expensive generalization operator (*lgg/rlgg* of *pairs* of seed examples)
  - Examples: Golem (Muggleton & Feng 1990), DLG (Webb 1992), RISE (Domingos 1995)
- Incremental Pruning of Rules:
  - greedy bottom-up hill-climbing via deleting conditions
  - BUT: start at point previously reached via top-down specialization
  - Examples: I-REP (Fürnkranz & Widmer 1994), Ripper (Cohen 1995)

Descriptive vs. Predictive Rules

TECHNISCHE UNIVERSITÄT DARMSTADT

- **Descriptive Learning**
  - Focus on discovering patterns that describe (parts of) the data
- **Predictive Learning**
  - Focus on finding patterns that allow to make predictions about the data

- **Rule Diversity and Completeness:**
  - Predictive rules need to be able to make a prediction for every possible instance
- **Predictive Evaluation:**
  - It is important how well rules are able to predict the dependent variable on new data
- **Descriptive Evaluation:**
  - “insight” delivered by the rule

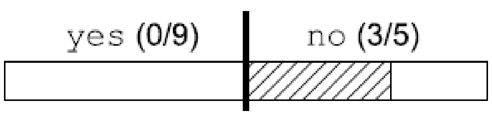
Subgroup Discovery

- Definition

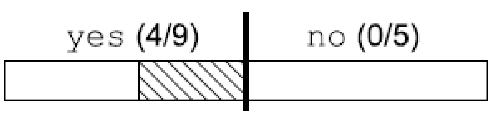
> "Given a population of individuals and a property of those individuals that we are interested in, **find population subgroups** that are statistically 'most interesting', e.g., are as large as possible and have the most unusual distributional characteristics with respect to the property of interest"
>
> (Klösgen 1996; Wrobel 1997)

- Examples

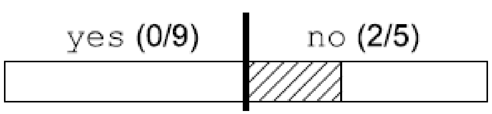
```
IF   MaritalStatus = single
 AND Sex = male
THEN Approved = no
```




```
IF   MaritalStatus = married
THEN Approved = yes
```



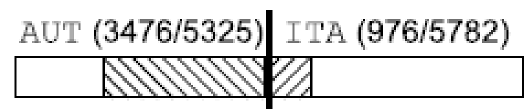

```
IF   MaritalStatus = divorced
 AND HasChildren = yes
THEN Approved = no
```



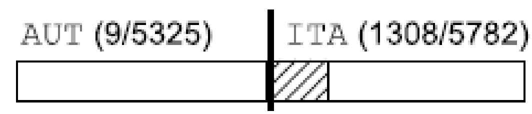
Application Study: Life Course Analysis

- Data:
  - Fertility and Family Survey 1995/96 for Italians and Austrians
  - Features based on general descriptors and variables that describes whether (quantum), at which age (timing) and in what order (sequencing) typical life course events have occurred.
- Objective:
  - Find subgroups that capture typical life courses for either country
- Examples:

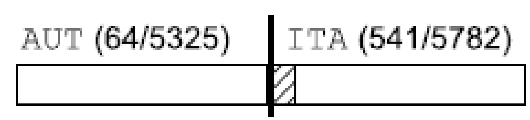
```
IF   LeftHome < Marriage
THEN AUT
```




```
IF   Union = Marriage
 AND Education <= 14
THEN ITA
```




```
IF   Union = Marriage
 AND Education >= 22
THEN ITA
```



Rule Length and Comprehensibility

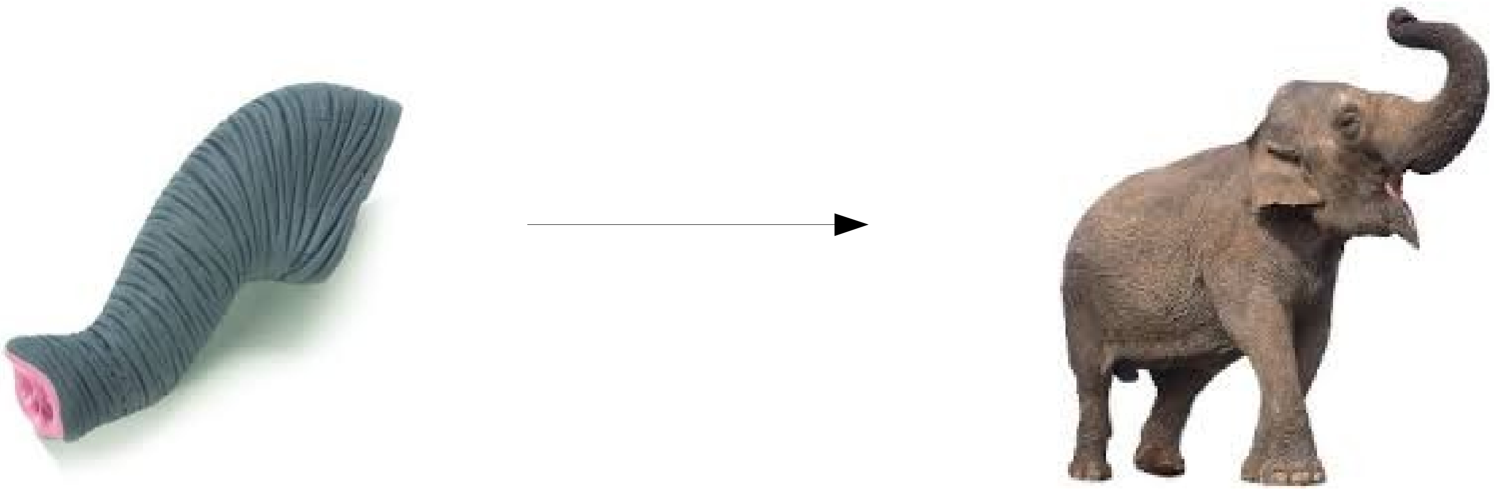
TECHNISCHE UNIVERSITÄT DARMSTADT

- Some Heuristics tend to learn longer rules
  - If there are conditions that can be added without decreasing coverage, they heuristics will add them first (before adding discriminative conditions)
- Typical intuition:
  - long rules are less understandable, therefore short rules are preferable
  - short rules are more general, therefore (statistically) more reliable
- Should shorter rules be preferred?
  - Not necessarily, because longer rules may capture more information about the object
  - Related to concepts in FCA, closed vs. free itemsets, discriminative rules vs. characteristic rules
  - Open question...

Discriminative Rules

- Allow to quickly discriminate an object of one category from objects of other categories
- Typically a few properties suffice

- Example:



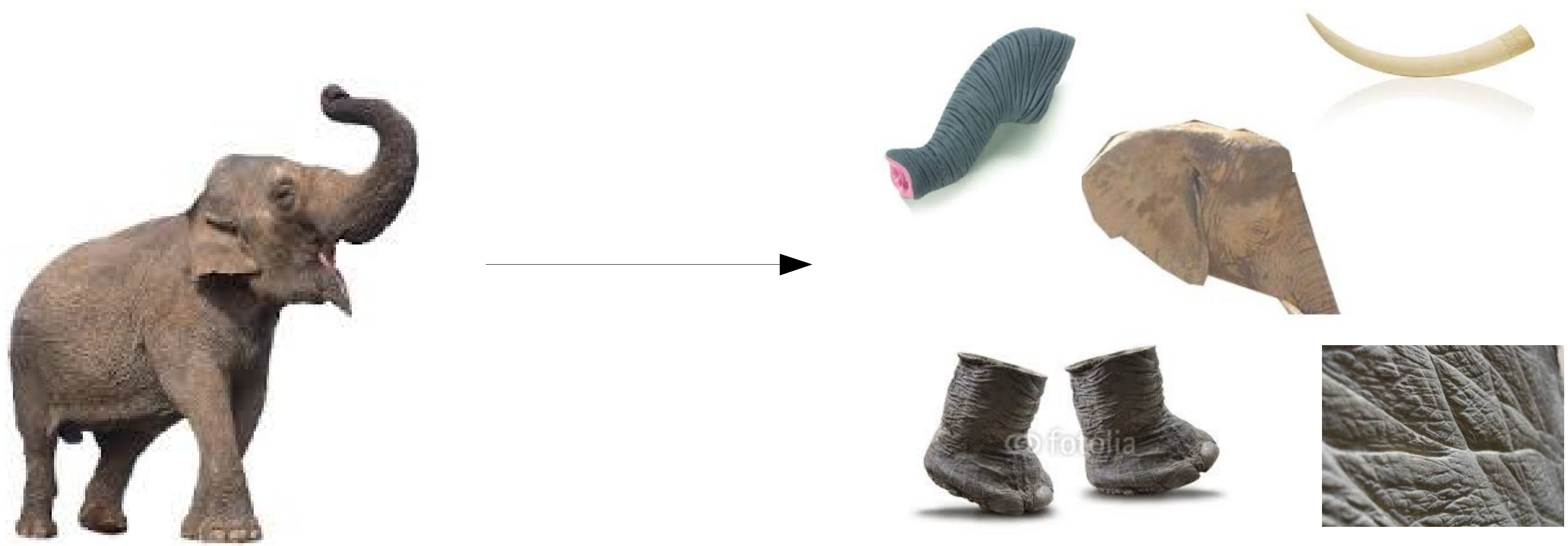
Characteristic Rules

- Allow to characterize an object of a category
- Focus is on all properties that are typical for objects of that category
- Example:

Characteristic Rules

- An alternative view of characteristic rules is to invert the implication sign
- All properties that are implied by the category

- Example:



Example: Mushroom dataset

- The best three rules learned with conventional heuristics

```
IF odor = f          THEN poisonous (2160,0)
IF gill-color = b    THEN poisonous (1152,0)
IF odor = p          THEN poisonous (256,0)
```

- The best three rules learned with inverted heuristics

```
IF veil-color = w, gill-spacing = c, bruises? = f,
   ring-number = o, stalk-surface-above-ring = k
THEN poisonous (2192,0)
IF veil-color = w, gill-spacing = c, gill-size = n,
   population = v, stalk-shape = t
THEN poisonous (864,0)
IF stalk-color-below-ring = w, ring-type = p,
   stalk-color-above-ring = w, ring-number = o,
   cap-surface = s, stalk-root = b, gill-spacing = c
THEN poisonous (336,0)
```

Summary

TECHNISCHE UNIVERSITÄT DARMSTADT

- Single Rules can be learned in batch mode from data by searching for rules that optimize a trade-off between covered positive and negative examples
- Different heuristics can be defined for optimizing this trade-off
- Coverage spaces can be used to visualize the behavior or such heuristics
  - precision-like heuristics tend to find the steepest ascent
  - accuracy-like heuristics assume a cost ratio between positive and negative examples
  - m-heuristic may be viewed as a trade-off between these two
- Subgroup Discovery is a task of its own ...
  - where typically the found description is the important result
- … but subgroups may also be used for prediction
  - → learning rule sets to ensure completeness