Data Mining and Machine Learning

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
  - Inverted heuristics

A Sample Database

TECHNISCHE UNIVERSITÄT DARMSTADT

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. $\mathbf{r}_{best} = \mathbf{r}$ = most general hypothesis / rule in H
   F = set of all possible features

II. while r covers negative examples
   I. for each possible feature $f \in C$
      a) $\mathbf{r}' = \mathbf{r} \cup \{f\}$
      b) if $\mathbf{r}'$ covers
         - all positive examples
         - and fewer negative examples than $\mathbf{r}_{best}$

         then $\mathbf{r}_{best} = \mathbf{r}'$
   II. $\mathbf{r} = \mathbf{r}_{best}$

III. return $\mathbf{r}_{best}$

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
- If V is not empty, it 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 $\mathbf{r}$ that covers all examples
2. Evaluate all possible ways to add a condition to $\mathbf{r}$
3. Choose the best one (according to some heuristic)
4. If $\mathbf{r}$ is satisfactory, return it
5. Else goto 2.

- Most greedy rule learning systems use a top-down strategy

Beam Search:

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

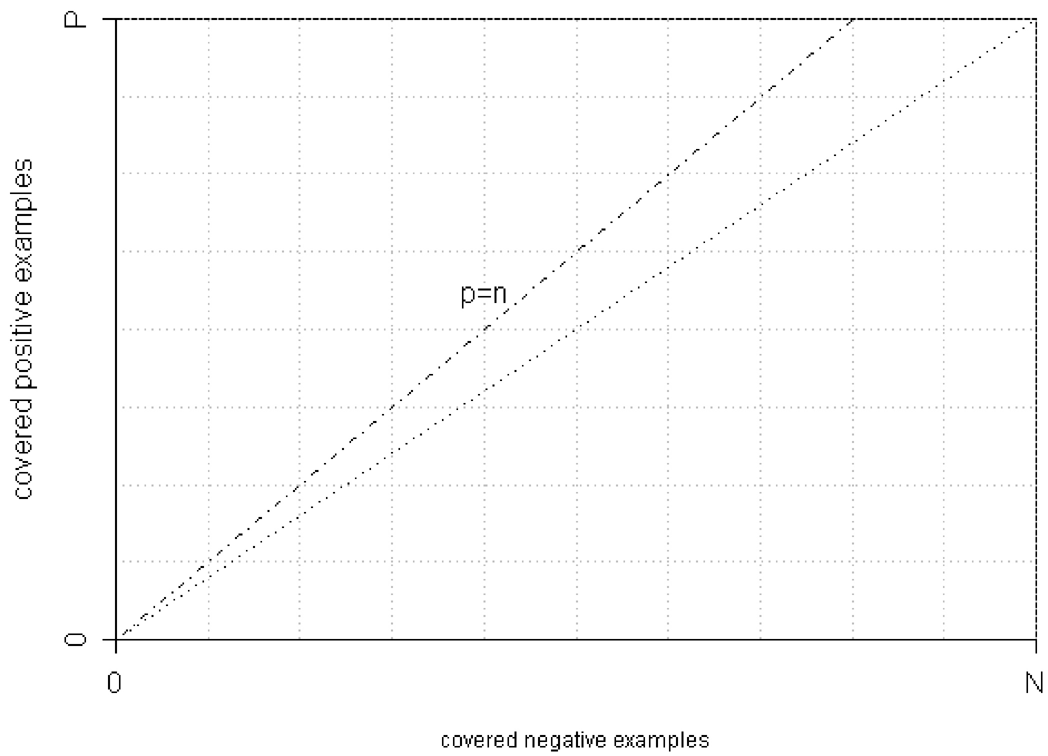
Terminology

- 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



Coverage Spaces

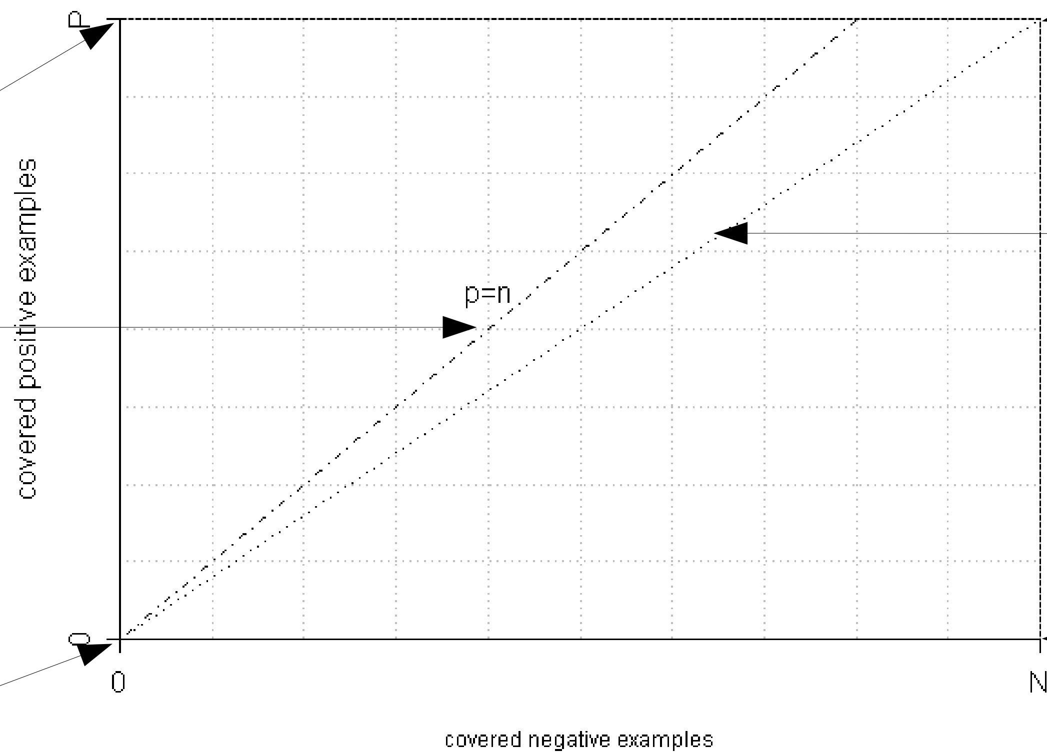
TECHNISCHE UNIVERSITÄT DARMSTADT

- 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)



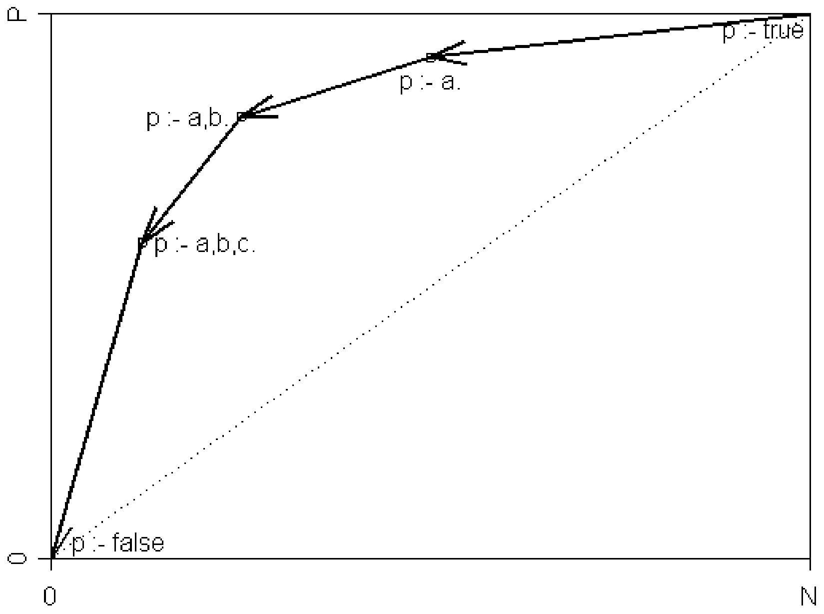
universal theory: all examples are covered (most general)

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

TECHNISCHE UNIVERSITÄT DARMSTADT

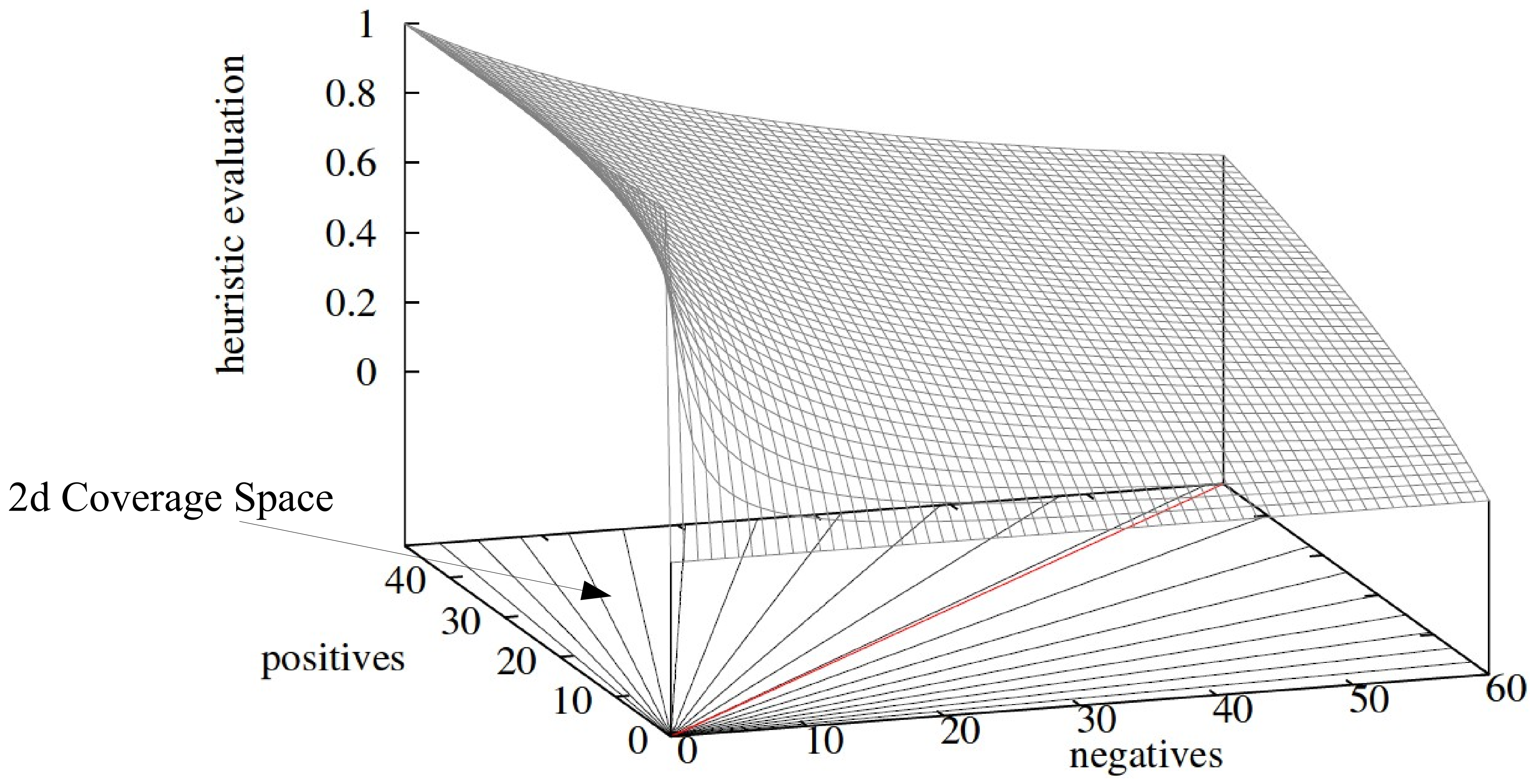
- Adding a condition to a rule should
  - decrease the number of covered negative examples n as much as possible (increase *consistency*)
  - decrease the number of covered positive p examples as little as possible (do not decrease *completeness*)
- An evaluation heuristic should therefore trade off these two extremes $\rightarrow$ prefer rules with larger p and smaller n
  - 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
`Education =`	`Primary`	2	2	0.5000	0.5000	0
	`University`	3	1	0.7500	0.6667	2
	`Secondary`	4	2	0.6667	0.6250	2
`Marital Status =`	`Single`	2	3	0.4000	0.4286	-1
	`Married`	4	0	1.0000	0.8333	4
	`Divorced`	3	2	0.6000	0.5714	1
`Sex =`	`Male`	3	4	0.4286	0.4444	-1
	`Female`	6	1	0.8571	0.7778	5
`Children =`	`Yes`	3	3	0.5000	0.5000	0
	`No`	6	2	0.7500	0.7000	4

- Heuristics Precision and Laplace
  - add the condition **`Outlook= Overcast`** to the (empty) rule
- Heuristic Accuracy / $p - n$
  - adds **`Sex = Female`**

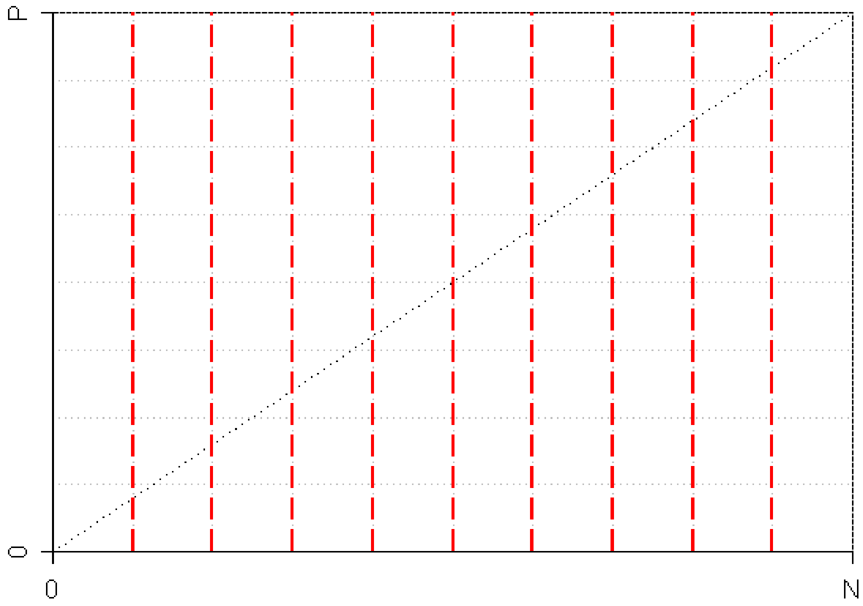
3d-Visualization of Precision



Isometrics in Coverage Space

TECHNISCHE UNIVERSITÄT DARMSTADT

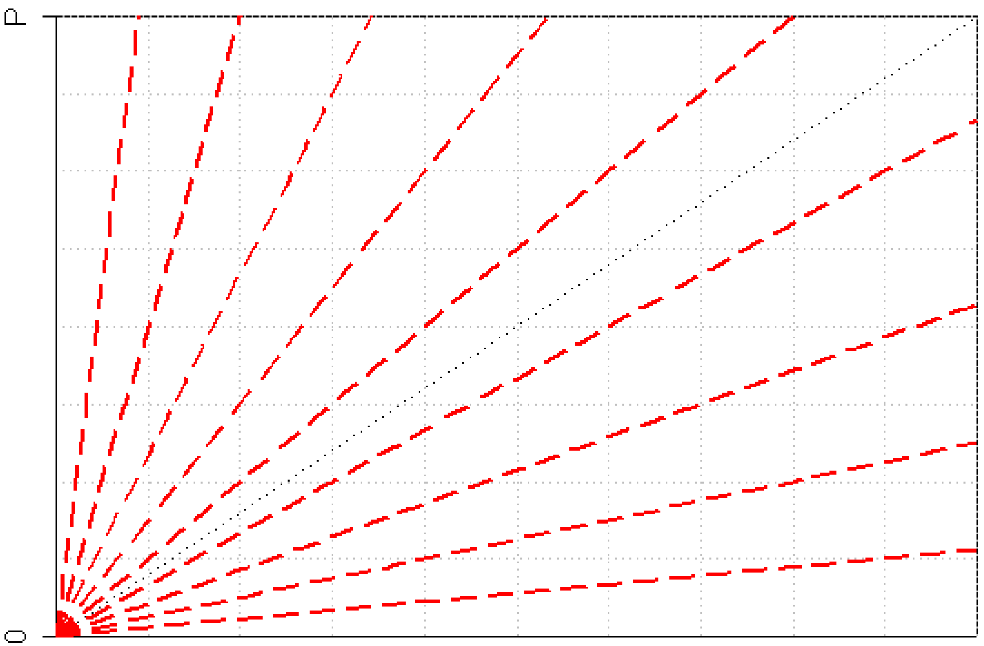
- Isometrics are lines that connect points for which a function in p and n has equal values
  - *Examples*:
    Isometrics for heuristics $h_p = p$ and $h_n = -n$



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



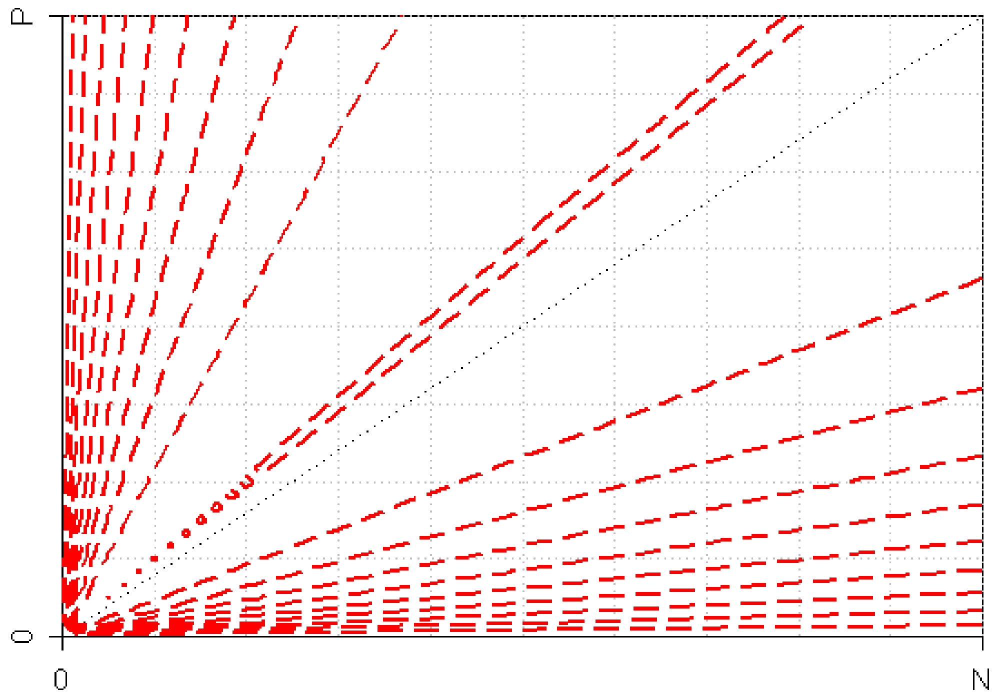
Entropy and Gini Index

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

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

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

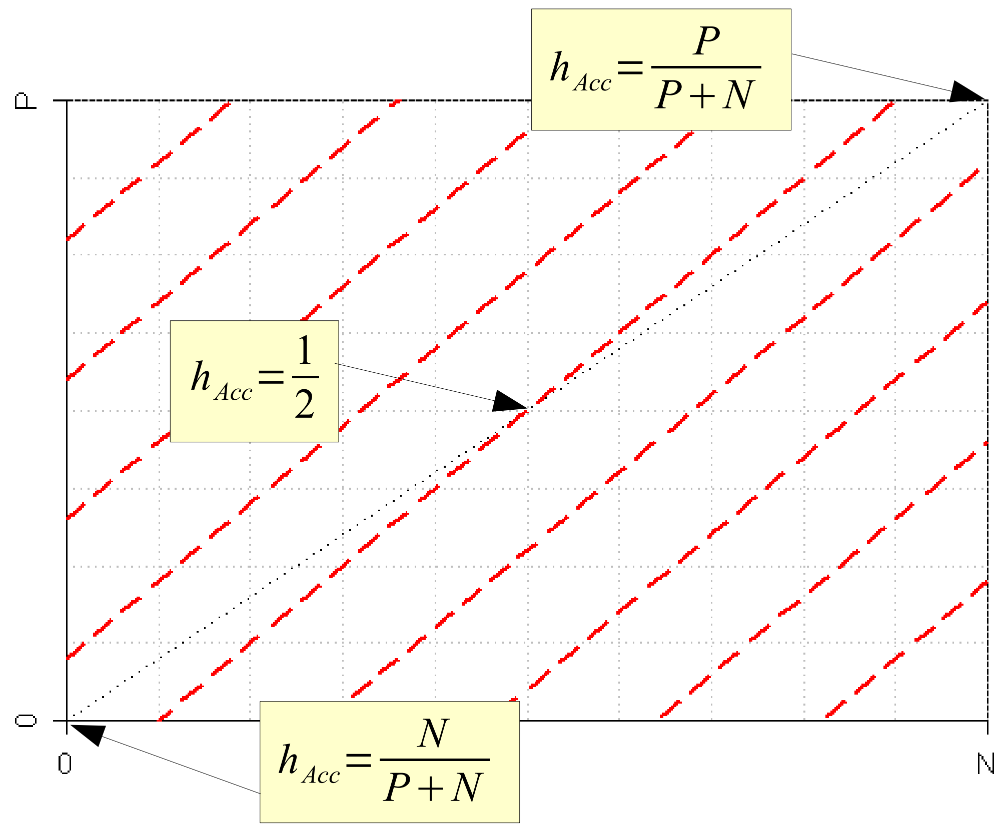


Accuracy

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

Why are they equivalent?

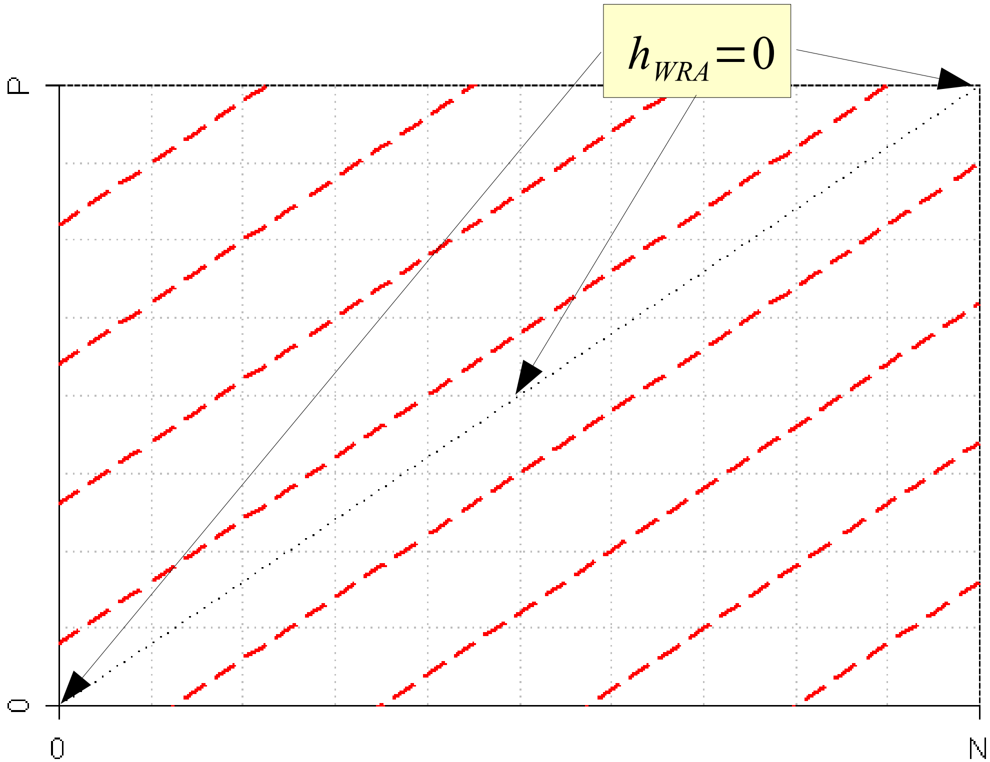
- *basic idea:*
  percentage of correct classifications (*covered positives* plus *uncovered negatives*)
- *effects:*
  - isometrics are parallel to 45° line
  - covering one positive example is as good as not covering one negative example



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 $c/(1\text{-}c)$ 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 $(1\text{-}c)/c$

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 $(1\text{-}c)/c$

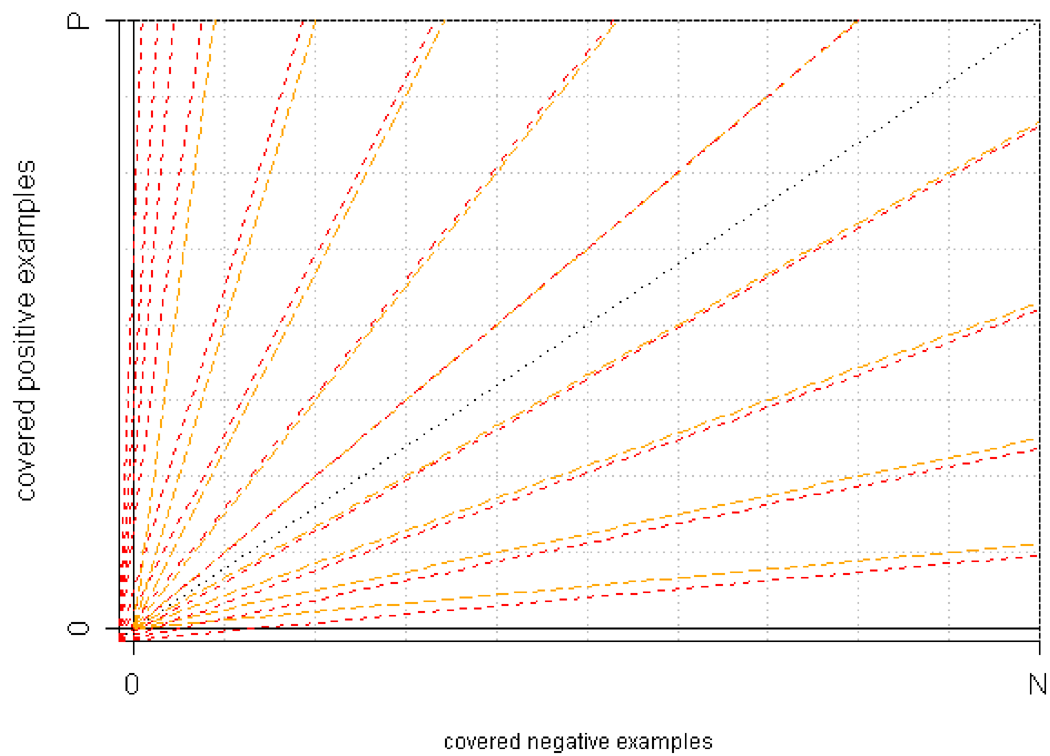
- The plots look the same as for the linear cost metric
  - but the semantics of the c value is different:
    - for h_{cost} it does not include the example distribution
    - for h_{rcost} it includes the example distribution

Laplace-Estimate

TECHNISCHE UNIVERSITÄT DARMSTADT

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

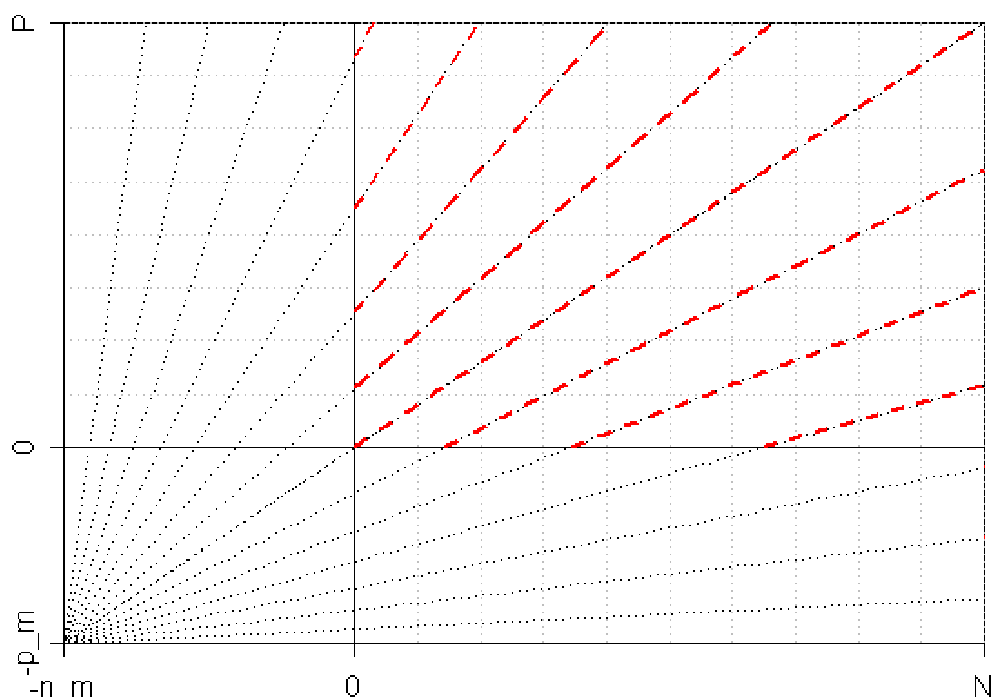
- *basic idea:*
  precision, but count coverage for positive and negative examples starting with 1 instead of 0
- *effects:*
  - origin at $(-1,-1)$
  - different values on $p=0$ or $n=0$ axes
  - not equivalent to precision



m-Estimate

- *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

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



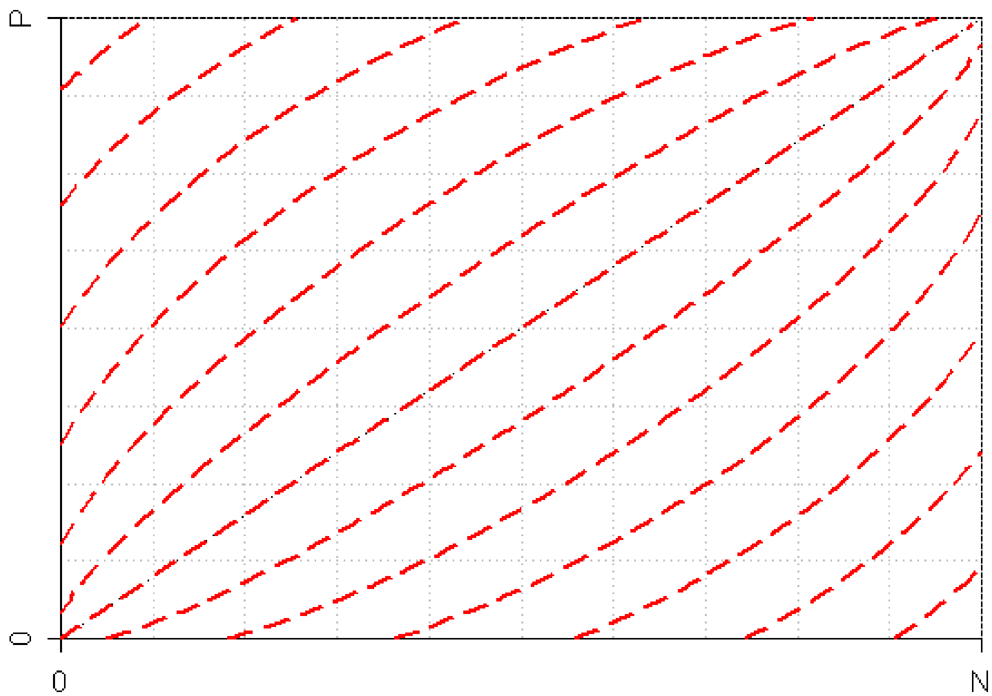
Generalized m-Estimate

- 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)
    - $m \to \infty$: 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

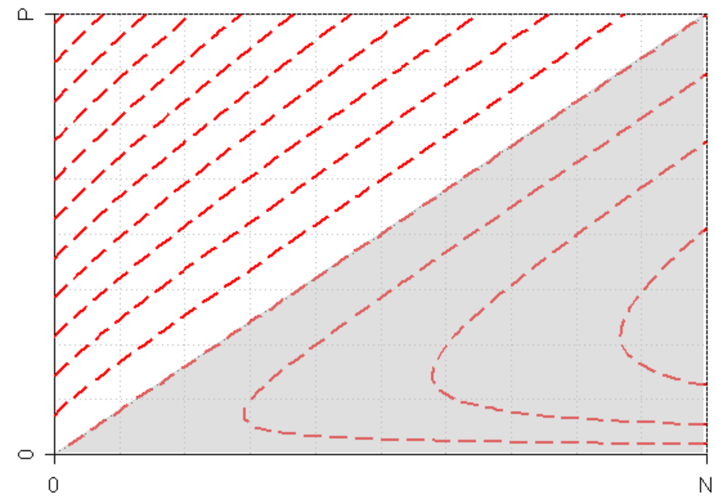


Foil Gain

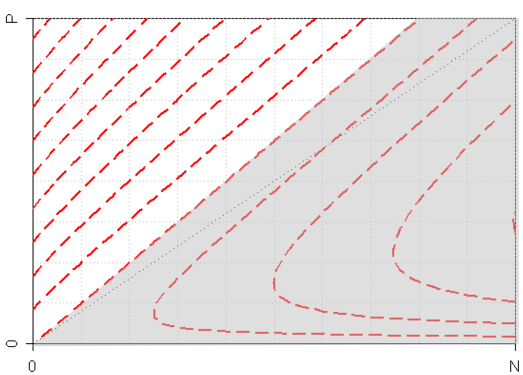
$$h_{foil} = -p\left(\log_2 c - \log_2 \frac{p}{p+n}\right)$$

(c is the precision of the parent rule)

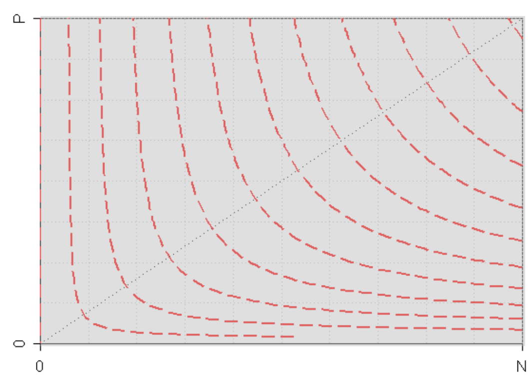
c = P/(P+N)



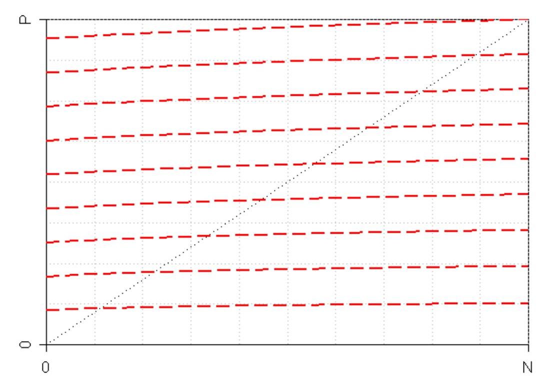
c = 1/2



c = 1



c = 10^-6



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

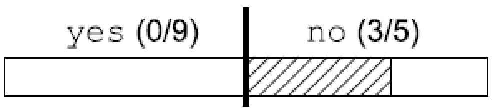
TECHNISCHE UNIVERSITÄT DARMSTADT

- Definition

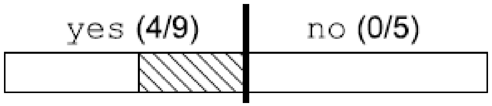
> "Given a population of individuals and a property of those individuals that we are interested in, **find population subgroups** that are statistically 'most interesting', e.g., are as large as possible and have the most unusual distributional characteristics with respect to the property of interest"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

- Examples

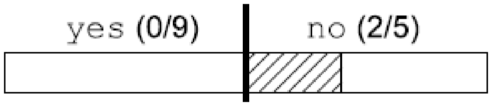
```
IF   MaritalStatus = single
 AND Sex = male
THEN Approved = no
```




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



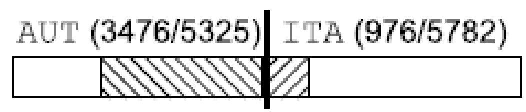

```
IF   MaritalStatus = divorced
 AND HasChildren = yes
THEN Approved = no
```



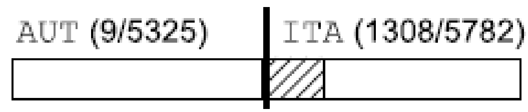
Application Study: Life Course Analysis

- 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:

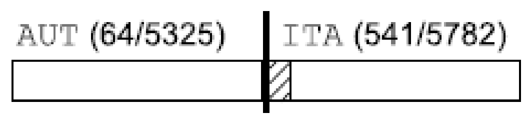
```
IF   LeftHome < Marriage
THEN AUT
```




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




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

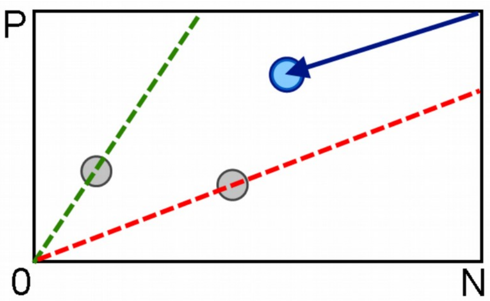


Rule Length and Comprehensibility

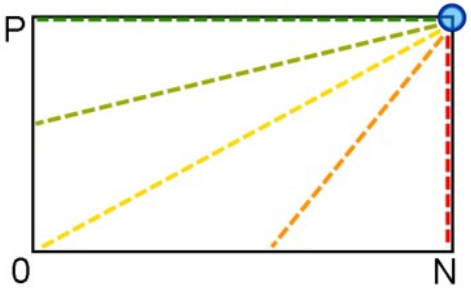
- Some Heuristics tend to learn longer rules
  - If there are conditions that can be added without decreasing coverage, they heuristics will add them first (before adding discriminative conditions)
- Typical intuition:
  - long rules are less understandable, therefore short rules are preferable
  - short 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...

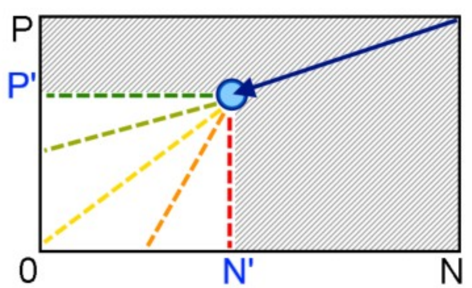
Inverted Heuristics – Motivation

- While the search proceeds top-down
- the evaluation of refinements happens from the point of view of the origin (bottom-up)



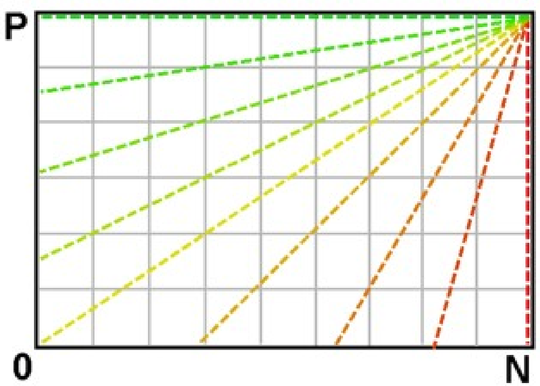
- Instead, we want to evaluate the refinement from the point of view of the predecessor

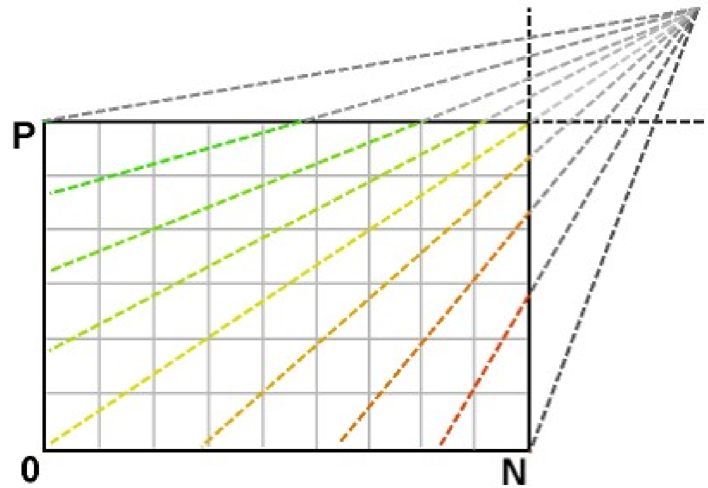




Inverted Heuristics

- Many heuristics can be “inverted” by replacing changing their angle point from the origin to the current rule





$$h'_{precision}(p,n,P,N)=\frac{N-n}{(P+N)-(p+n)}$$

$$h'_{m-Estimate}(p,n,P,N)=\frac{N-n+m\cdot\frac{P}{P+N}}{(P+N)-(p+n-m)}$$

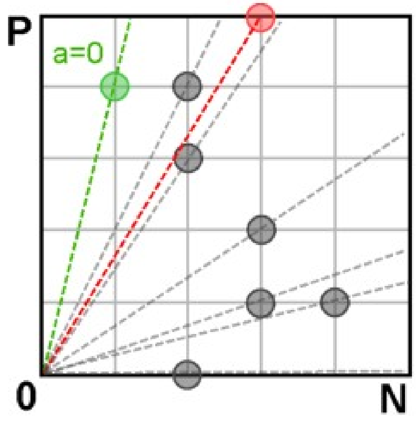
- **Note:** not all heuristics can be inverted
  - e.g. WRA is invariant w.r.t. inversion (because of symmetry)

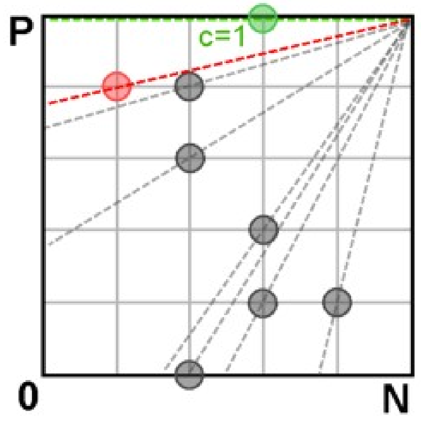
Inverted Heuristics – Example

First refinement step in small example dataset

- 4 Attributes, 10 data points, binary-class

a	b	c	d	*C*
0	1	1	1	+
0	1	1	1	+
0	0	1	0	-
1	1	1	0	-
1	0	0	1	-
0	1	1	0	+
0	0	1	1	+
1	1	1	0	-
1	0	1	1	+
1	0	0	1	-





Inverted heuristic function (right image) selects preferable refinement condition $c=1$ with coverage of $(p,n)=(5,3)$

Implementation

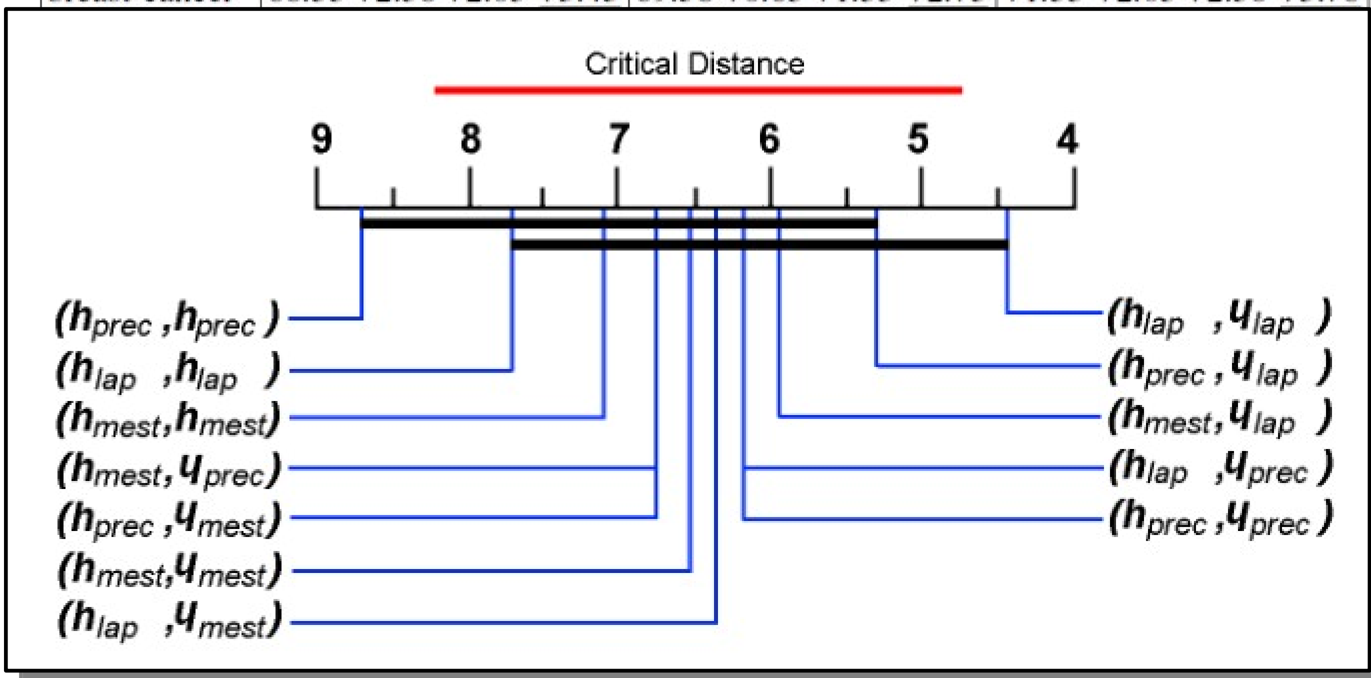
- Modification of a conventional covering algorithm
  - CN2-like
  - No pruning, no significance test
- Rule refinement proceeds with inverted heuristics
  - In each iteration, the best condition is added to the rule until the rule covers no more examples
- Rule selection proceeds with regular heuristics
  - Among all refinements on the path, the best rule is selected using a regular heuristic

Results:
Inverted heuristics tend to work better

Dataset	$(h_{prec}, .)$				$(h_{lap}, .)$				$(h_{mest}, .)$			
	h_{prec}	ɥ_{prec}	ɥ_{lap}	ɥ_{mest}	h_{lap}	ɥ_{prec}	ɥ_{lap}	ɥ_{mest}	h_{mest}	ɥ_{prec}	ɥ_{lap}	ɥ_{mest}
breast-cancer	68.53	72.38	72.03	73.43	69.58	70.63	71.33	72.73	71.33	72.03	72.38	73.78
car	90.10	90.34	90.51	88.66	90.45	91.20	91.73	91.20	89.64	90.45	90.28	87.91
contact-lenses	79.17	87.50	87.50	83.33	79.17	87.50	87.50	83.33	87.50	87.50	87.50	83.33
futebol	28.57	64.29	57.14	42.88	28.57	64.29	57.14	42.88	50.00	64.29	57.14	42.86
glass	56.54	65.89	68.69	62.15	61.22	65.89	68.69	62.15	69.16	67.29	71.50	63.55
hepatitis	78.07	79.36	80.00	76.77	78.71	79.36	80.00	76.74	78.07	79.36	80.00	76.77
hypothyroid	98.23	98.61	98.74	98.83	98.39	98.61	98.74	98.83	98.80	98.61	98.74	98.83
horse-colic	72.01	79.35	79.35	77.99	70.65	79.35	80.16	77.99	77.45	79.35	78.80	77.99
idh	62.07	82.76	75.86	75.86	62.07	82.76	75.86	75.86	68.97	82.76	75.86	75.86
iris	92.67	93.33	95.33	94.67	94.00	93.33	95.33	94.67	94.00	93.33	95.33	94.67
ionosphere	95.16	82.62	83.19	89.46	94.87	82.62	93.19	89.46	91.74	82.91	83.19	91.17
labor	91.23	80.70	82.46	89.47	91.23	80.70	82.46	89.47	85.97	80.70	82.46	89.47
lymphography	83.78	77.70	84.46	83.11	85.14	77.70	84.46	83.11	75.00	76.35	81.08	83.78
mushroom	100.0	100.0	100.0	100.0	100.0	100.0	100.0	100.0	100.0	100.0	100.0	100.0
monk3	87.71	82.79	82.79	84.43	88.53	85.25	84.43	86.89	81.15	79.51	81.15	82.79
primary-tumor	33.63	39.23	35.10	30.97	32.45	39.23	35.99	30.38	33.92	37.76	34.51	30.68
soybean	90.04	91.51	92.24	91.36	90.34	91.80	92.39	90.63	91.51	90.92	90.48	91.36
tic-tac-toe	97.39	98.02	97.60	97.81	97.60	98.02	97.60	97.91	98.12	98.02	97.60	97.81
vote	94.94	93.56	94.25	94.48	95.40	94.25	94.25	94.94	93.33	93.56	94.71	96.09
zoo	84.16	88.12	92.08	90.01	86.14	88.12	92.08	90.10	89.11	88.12	92.08	90.10
average rank	3.075	2.400	1.975	2.550	3.000	2.500	1.975	2.525	2.700	2.625	2.225	2.450

Results:
Inverted heuristics tend to work better

Dataset	(h_{prec}, .)				(h_{lap}, .)				(h_{mest}, .)			
	h_{prec}	ɥ$_{prec}$	ɥ$_{lap}$	ɥ$_{mest}$	h_{lap}	ɥ$_{prec}$	ɥ$_{lap}$	ɥ$_{mest}$	h_{mest}	ɥ$_{prec}$	ɥ$_{lap}$	ɥ$_{mest}$
breast-cancer	68.53	72.38	72.03	73.43	69.58	70.63	71.33	72.73	71.33	72.03	72.38	73.78
tic-tac-toe	97.39	98.02	97.60	97.81	97.60	98.02	97.60	97.91	98.12	98.02	97.60	97.81
vote	94.94	93.56	94.25	94.48	95.40	94.25	94.25	94.94	93.33	93.56	94.71	96.09
zoo	84.16	88.12	92.08	90.01	86.14	88.12	92.08	90.10	89.11	88.12	92.08	90.10
average rank	3.075	2.400	1.975	2.550	3.000	2.500	1.975	2.525	2.700	2.625	2.225	2.450



Inverted Heuristics – Rule Length

- Inverted Heuristics tend to learn longer rules
  - If there are conditions that can be added without decreasing coverage on the positive examples, inverted heuristics will add them first (before adding discriminative conditions)

	(h_{lap}, h_{lap})		(h_{lap}, h'_{lap})			(h_{lap}, h_{lap})		(h_{lap}, h'_{lap})	
Dataset	R	L	R	L	**Dataset**	R	L	R	L
breast-cancer	25	67	38	173	ionosphere	17	25	8	42
car	107	495	107	506	labor	5	7	3	12
contact-lenses	5	14	5	15	lymphography	18	42	11	47
futebol	4	7	2	5	monk3	13	38	11	32
glass	50	103	14	83	mushroom	11	13	7	35
hepatitis	13	26	7	46	primary-tumor	80	319	72	518
horse-colic	44	114	19	111	soybean	62	134	45	195
hypothyroid	27	65	9	69	tic-tac-toe	22	84	16	69
iris	7	15	5	17	vote	13	48	12	58
idh	4	5	2	5	zoo	19	19	6	14
averages						28.2	85.6	20.6	106.2

Discriminative Rules

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

- Example:

Discriminative Rules

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

- Example:

Characteristic Rules

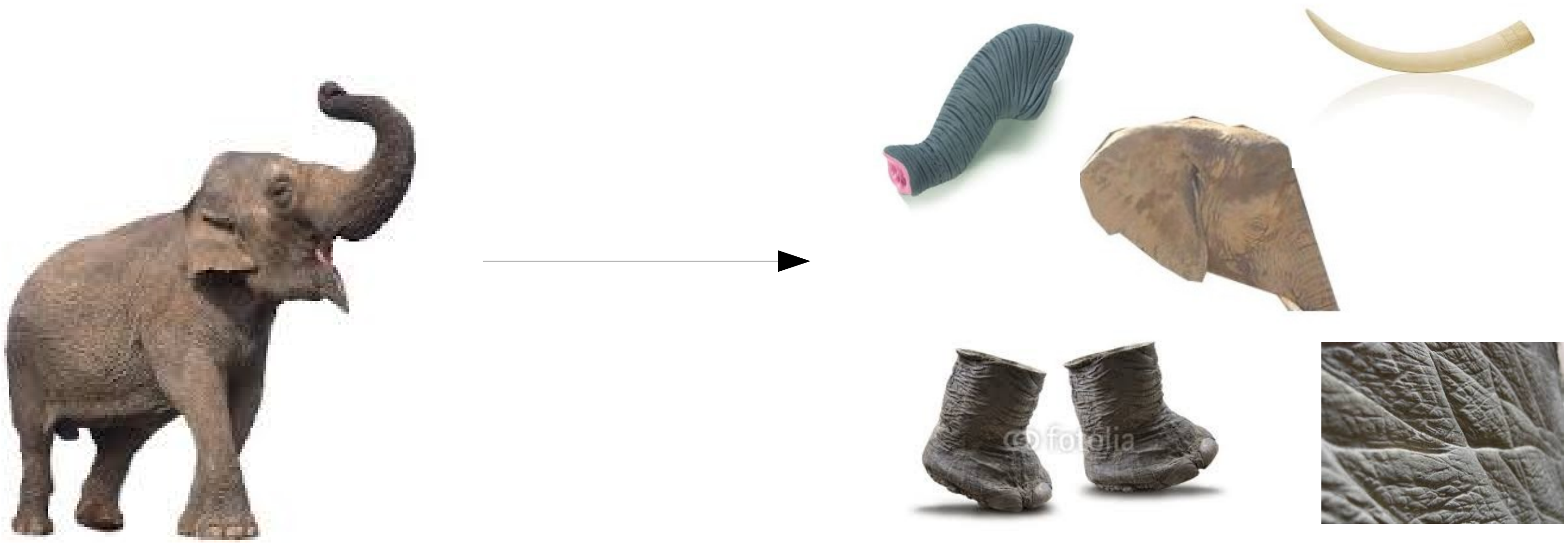
- Allow to characterize an object of a category
- Focus is on all properties that are typical for objects of that category

- 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