Machine Learning: Symbolische Ansätze

TECHNISCHE UNIVERSITÄT DARMSTADT

Learning Rule Sets

- Introduction
  - Learning Rule Sets
  - Terminology
  - Coverage Spaces

- Separate-and-Conquer Rule Learning
  - Covering algorithm
  - Top-Down Hill-Climbing
  - Rule Evaluation Heuristics
  - Overfitting and Pruning
  - Multi-Class Problems
  - Bottom-Up Hill-Climbing

Learning Rule Sets

TECHNISCHE UNIVERSITÄT DARMSTADT

- many datasets cannot be solved with a single rule
  - not even the simple weather dataset
  - they need a rule set for formulating a target theory
- finding a computable generality relation for rule sets is not trivial
  - adding a condition to a rule specializes the theory
  - adding a new rule to a theory generalizes the theory
- practical algorithms use different approaches
  - covering or separate-and-conquer algorithms
  - based on heuristic search

A sample task

TECHNISCHE UNIVERSITÄT DARMSTADT

Temperature	Outlook	Humidity	Windy	Play Golf?
hot	sunny	high	false	no
hot	sunny	high	true	no
hot	overcast	high	false	yes
cool	rain	normal	false	yes
cool	overcast	normal	true	yes
mild	sunny	high	false	no
cool	sunny	normal	false	yes
mild	rain	normal	false	yes
mild	sunny	normal	true	yes
mild	overcast	high	true	yes
hot	overcast	normal	false	yes
mild	rain	high	true	no
cool	rain	normal	true	no
mild	rain	high	false	yes

- Task:
  - Find a rule set that correctly predicts the dependent variable from the observed variables

A Simple Solution

TECHNISCHE UNIVERSITÄT DARMSTADT

IF	T=hot	AND	H=high	AND	O=overcast	AND	W=false	THEN yes
IF	T=cool	AND	H=normal	AND	O=rain	AND	W=false	THEN yes
IF	T=cool	AND	H=normal	AND	O=overcast	AND	W=true	THEN yes
IF	T=cool	AND	H=normal	AND	O=sunny	AND	W=false	THEN yes
IF	T=mild	AND	H=normal	AND	O=rain	AND	W=false	THEN yes
IF	T=mild	AND	H=normal	AND	O=sunny	AND	W=true	THEN yes
IF	T=mild	AND	H=high	AND	O=overcast	AND	W=true	THEN yes
IF	T=hot	AND	H=normal	AND	O=overcast	AND	W=false	THEN yes
IF	T=mild	AND	H=high	AND	O=rain	AND	W=false	THEN yes

- The solution is
  - a set of rules
  - that is complete and consistent on the training examples

→ it must be part of the version space

- but it does not generalize to new examples!

The Need for a Bias

- rule sets can be generalized by
  - generalizing an existing rule (as in (Batch-)Find-S)
  - introducing a new rule (this is new)
- a minimal generalization could be
  - introduce a new rule that covers only the new example
- Thus:
  - The solution on the previous slide will be found as a result of the FindS algorithm
  - FindG (or Batch-FindG) are less likely to find such a bad solution because they prefer general theories
- But in principle this solution is part of the hypothesis space and also of the version space

  ⇒ *we need a search bias to prevent finding this solution!*

A Better Solution

TECHNISCHE UNIVERSITÄT DARMSTADT

IF	Outlook = overcast	THEN	yes
IF	Humidity = normal AND Outlook = sunny	THEN	yes
IF	Outlook = rainy AND Windy = false	THEN	yes

Recap: Batch-Find

TECHNISCHE UNIVERSITÄT DARMSTADT

- Abstract algorithm for learning a single rule:

1. Start with an empty theory T and training set E
2. Learn a single (*consistent*) rule R from E and add it to T
3. return T

- Problem:
  - the basic assumption is that the found rules are complete, i.e., they cover all positive examples
  - What if they don't?
- Simple solution:
  - If we have a rule that covers part of the positive examples:
  - add some more rules that cover the remaining examples

Separate-and-Conquer Rule Learning

TECHNISCHE UNIVERSITÄT DARMSTADT

- Learn a set of rules, one rule after the other

1. Start with an empty theory T and training set E
2. Learn a single (*consistent*) rule R from E and add it to T
3. If T is satisfactory (*complete*), return T
4. Else:
   - *Separate:* Remove examples explained by R from E
   - *Conquer:* goto 2.

- One of the oldest family of learning algorithms
  - goes back to AQ (Michalski, 60s)
  - FRINGE, PRISM and CN2: relation to decision trees (80s)
  - popularized in ILP (FOIL and PROGOL, 90s)
  - RIPPER brought in good noise-handling
- Different learners differ in how they find a single rule

Relaxing Completeness and Consistency

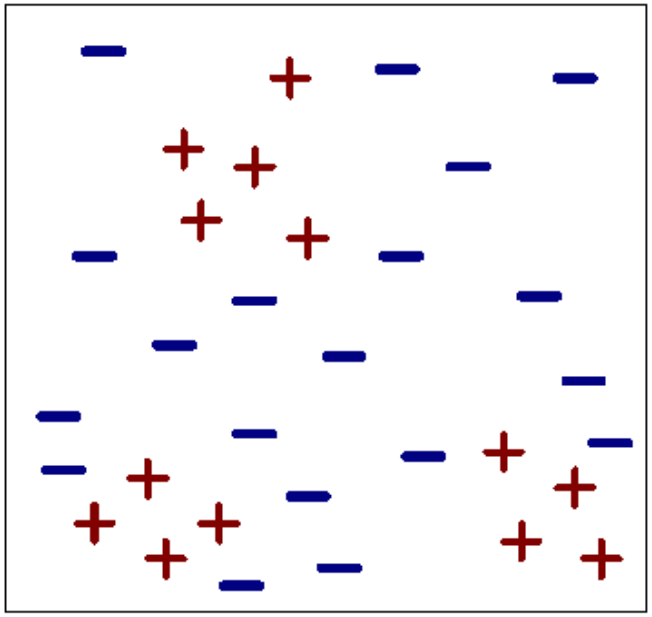
- So far we have always required a learner to learn a complete and consistent theory
  - e.g., one rule that covers all positive and no negative examples
- This is not always a good idea (→ overfitting)
- Example:

  Training set with 200 examples, 100 positive and 100 negative
  - **Theory A** consists of 100 complex rules, each covering a single positive example and no negatives

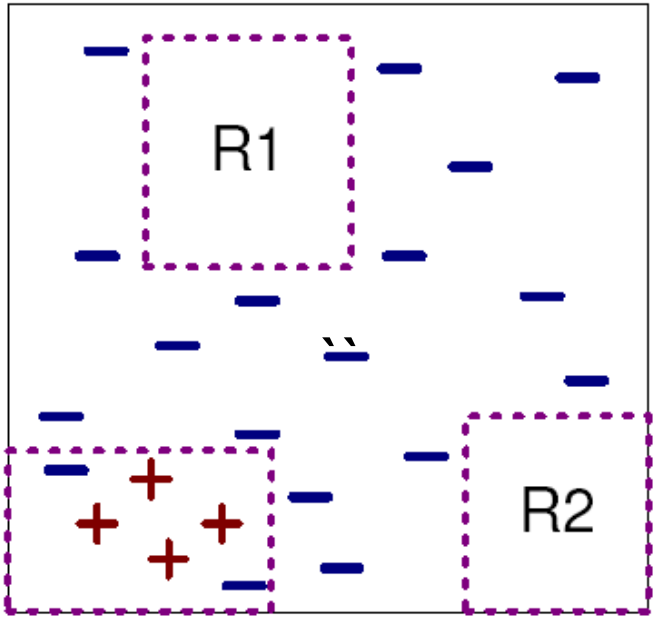
    → Theory A is complete and consistent on the training set
  - **Theory B** consists of a single rule, covering 99 positive and 1 negative example

    → Theory B is incomplete and incosistent on the training set
- Which one will generalize better to unseen examples?

Separate-and-Conquer Rule Learning



(i) Original Data



(iv) Step 3

Quelle für Grafiken: http://www.cl.uni-heidelberg.de/kurs/ws03/einfki/KI-2004-01-13.pdf

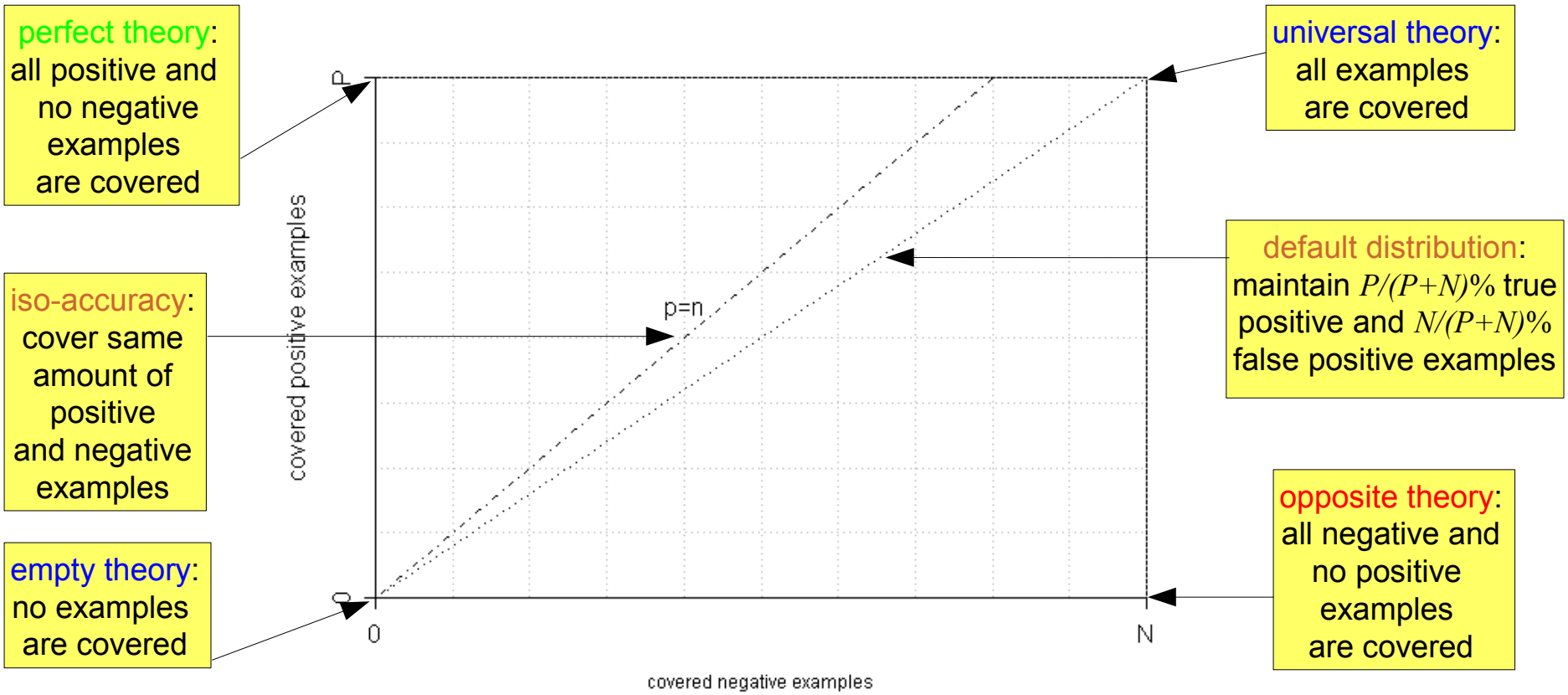
Terminology

- 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
	$p + n$	$P+N-(p+n)$	$P+N$

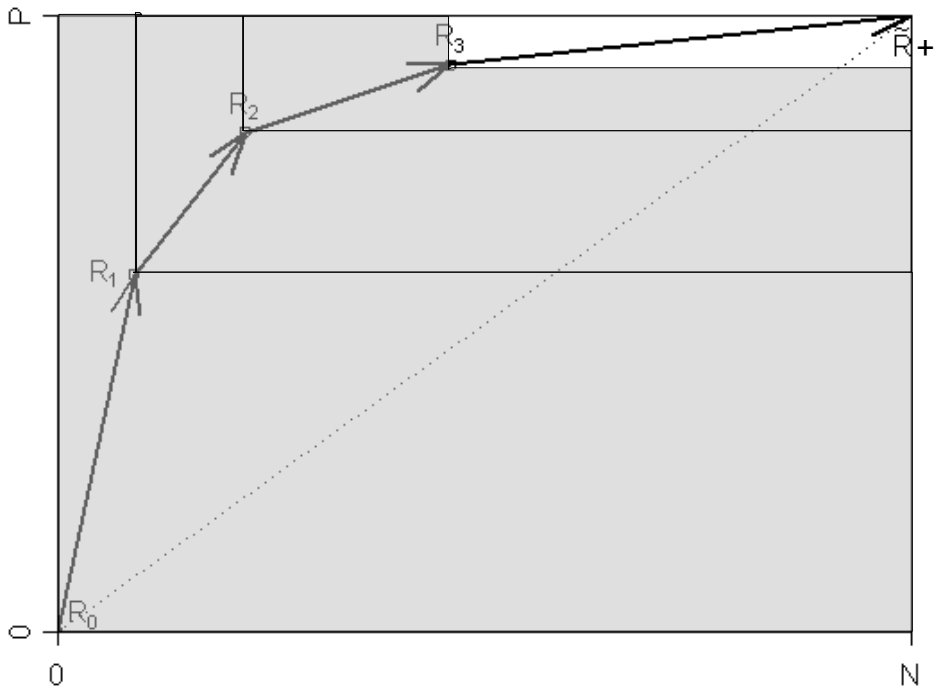
Coverage Spaces

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



Covering Strategy

- Covering or Separate-and-Conquer rule learning learning algorithms learn one rule at a time
- This corresponds to a path in coverage space:
  - The empty theory R_0 (no rules) corresponds to (0,0)
  - Adding one rule never decreases p or n because adding a rule covers *more* examples (generalization)
  - The universal theory R+ (all examples are positive) corresponds to (N,P)



Top-Down Hill-Climbing

TECHNISCHE UNIVERSITÄT DARMSTADT

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

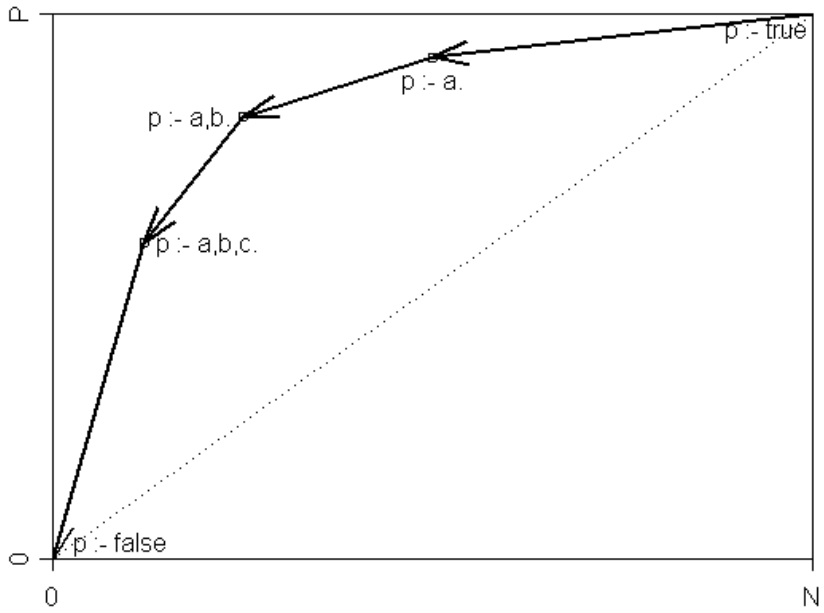
1. Start with the universal rule R that covers all examples
2. Evaluate all possible ways to add a condition to R
3. Choose the best one (according to some heuristic)
4. If R is satisfactory, return it
5. Else goto 2.

- Almost all greedy s&c rule learning systems use this strategy

Top-Down Hill-Climbing

TECHNISCHE UNIVERSITÄT DARMSTADT

- 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 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 $p \to \infty$
    - grows with $n \to 0$
  - Example: Precision $h_{Prec} = \frac{p}{p+n}$
    - is not a good heuristic. Why?

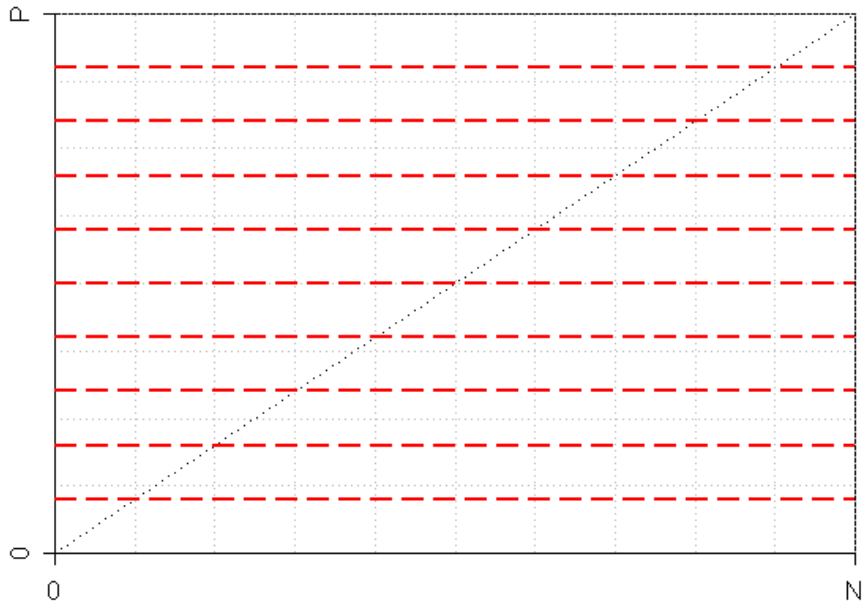
Example

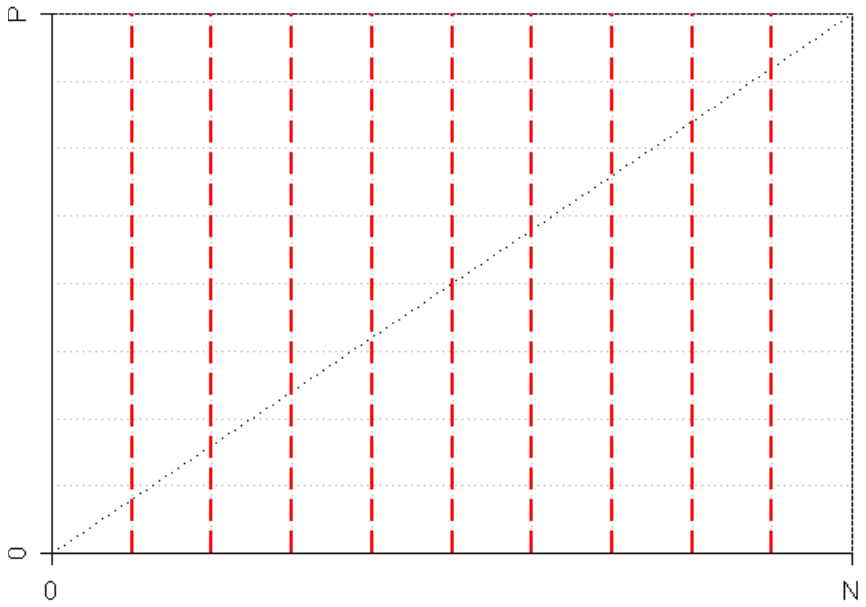
Condition		p	n	Precision	Laplace	p-n
Temperature =	Hot	2	2	0.5000	0.5000	0
	Mild	3	1	0.7500	0.6667	2
	Cold	4	2	0.6667	0.6250	2
Outlook =	Sunny	2	3	0.4000	0.4286	-1
	Overcast	4	0	1.0000	0.8333	4
	Rain	3	2	0.6000	0.5714	1
Humidity =	High	3	4	0.4286	0.4444	-1
	Normal	6	1	0.8571	0.7778	5
Windy =	True	3	3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

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 $h_n = -n$

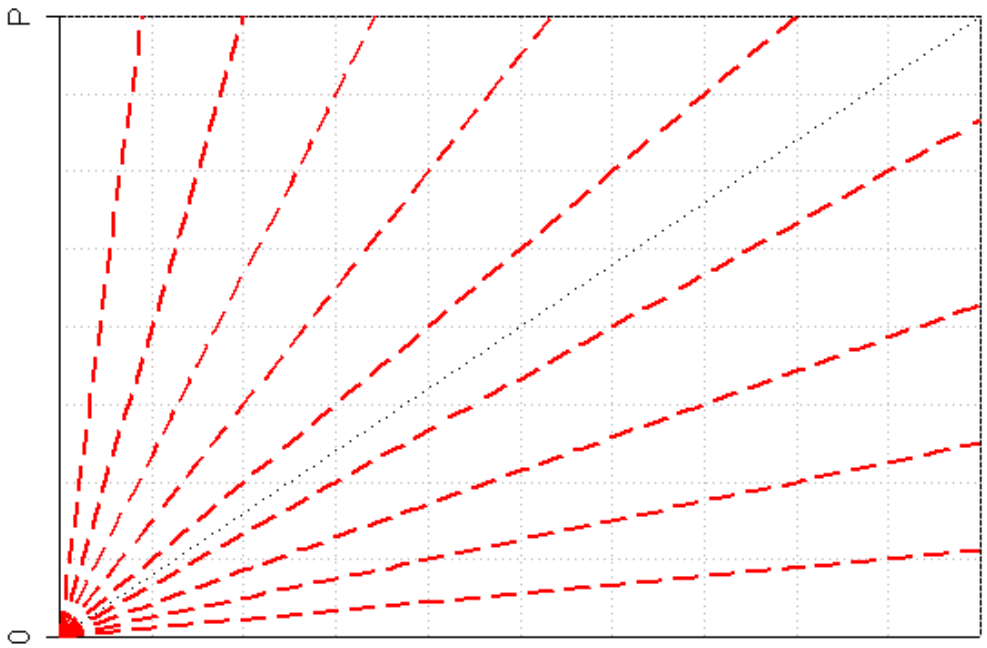




Precision (Confidence)

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

- *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



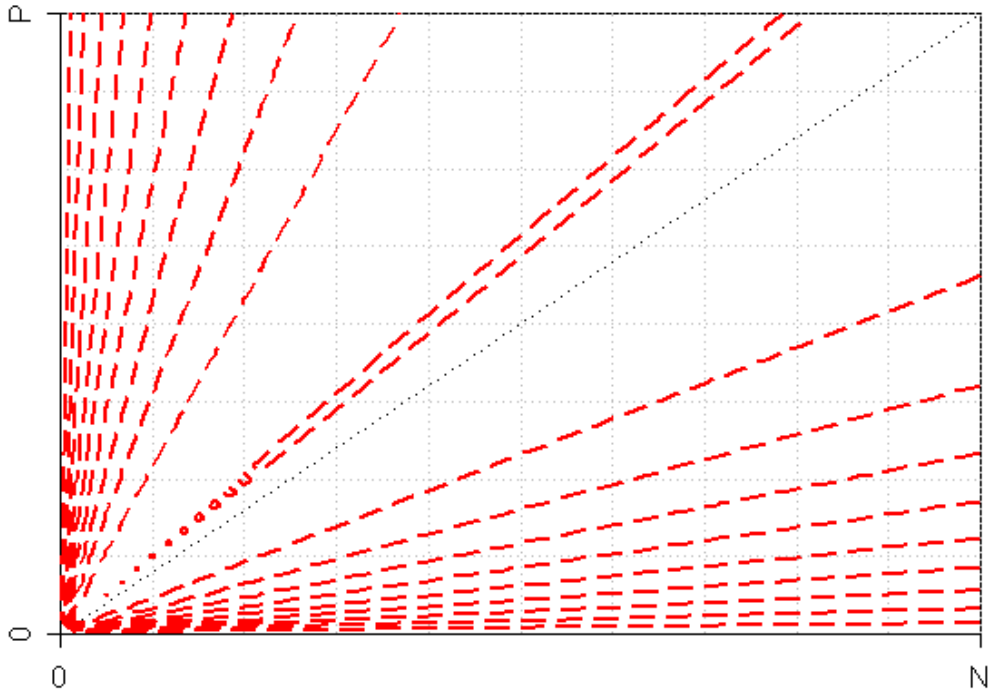
Entropy and Gini Index

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

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

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

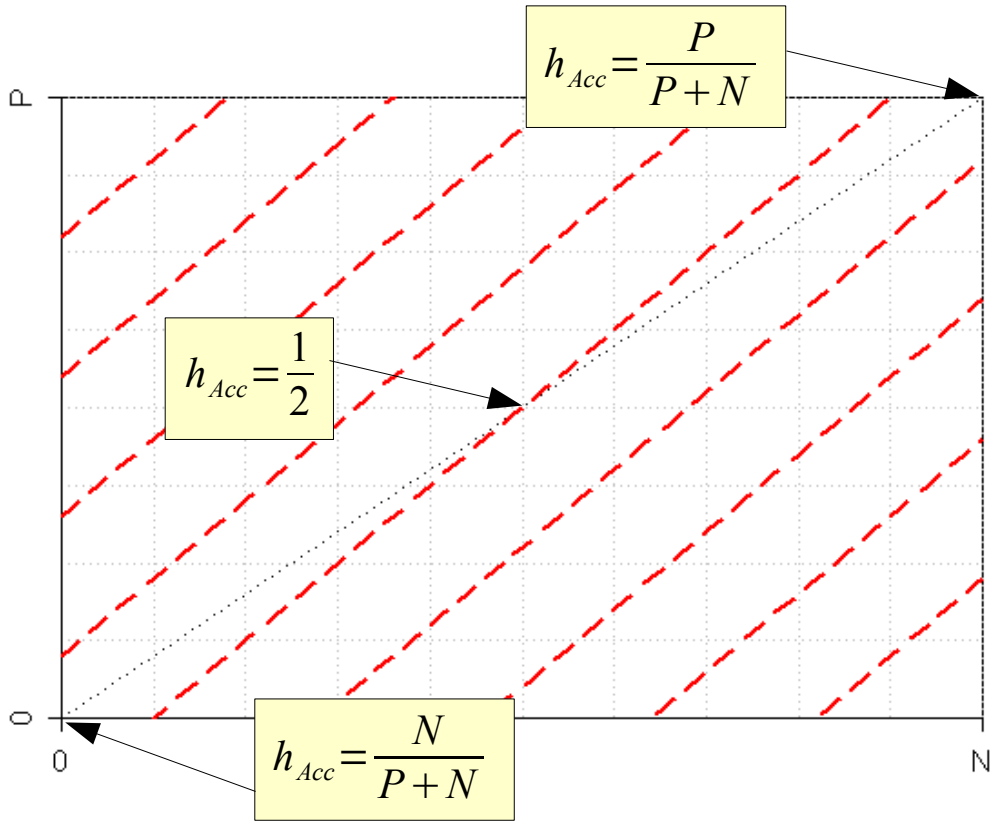


Accuracy

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

Why are they equivalent?

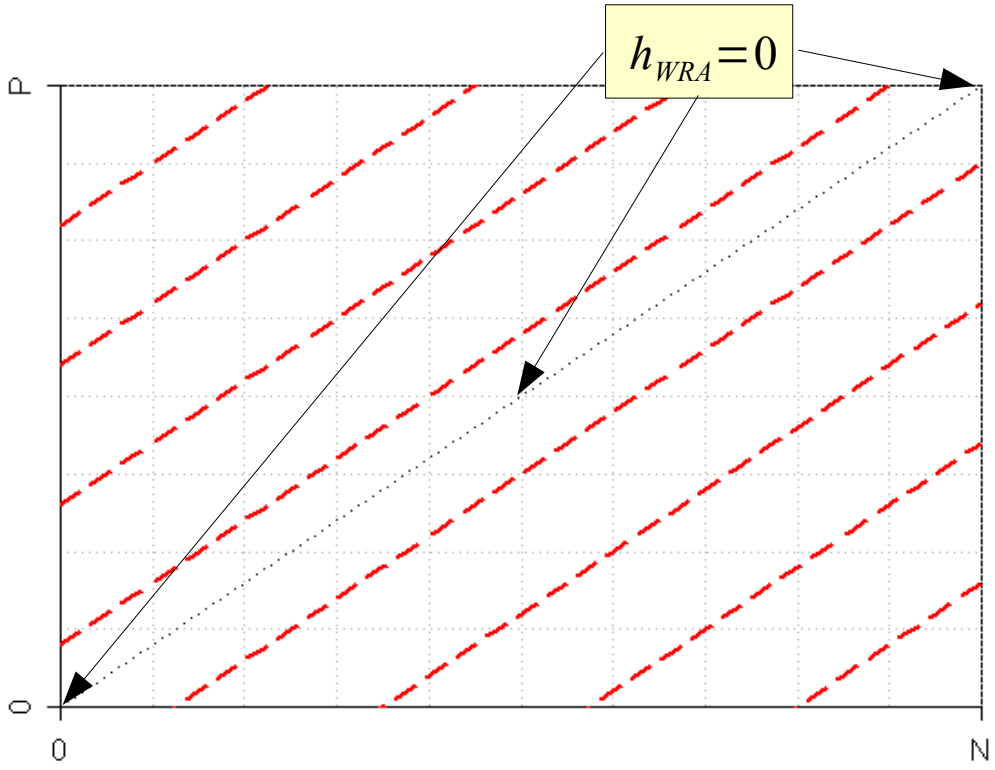
- *basic idea:* percentage of correct classifications (*covered positives* plus *uncovered negatives*)
- *effects:*
  - isometrics are parallel to 45° line
  - covering one positive example is as good as not covering one negative example



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

TECHNISCHE UNIVERSITÄT DARMSTADT

- 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

TECHNISCHE UNIVERSITÄT DARMSTADT

- 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

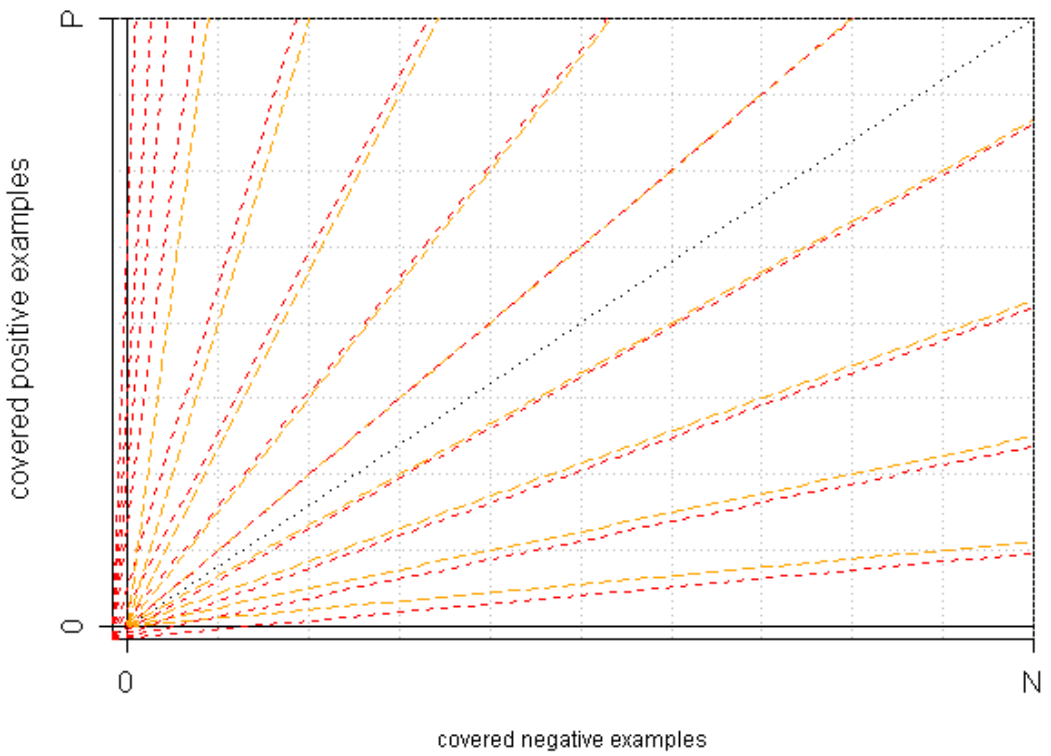
- The general form is then $h_{rcost} = c \cdot \frac{p}{P} - (1-c) \cdot \frac{n}{N}$
  - the isometrics of h_{cost} are parallel lines with slope $(1\text{-}c)/c$

- 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

- *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

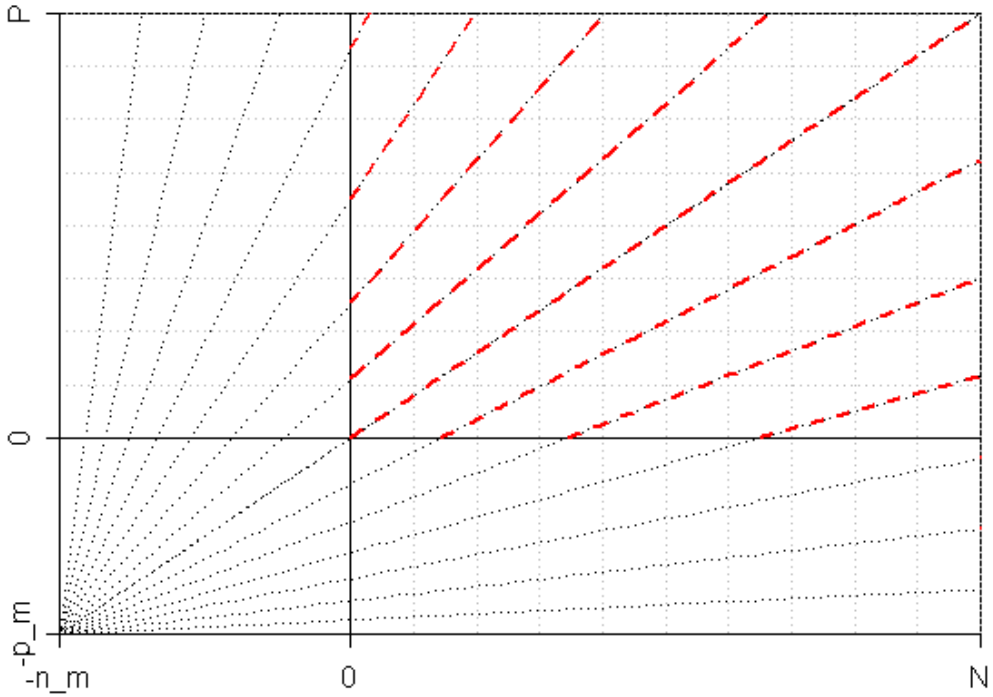
$$h_{Lap} = \frac{p+1}{(p+1)+(n+1)} = \frac{p+1}{p+n+2}$$



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

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



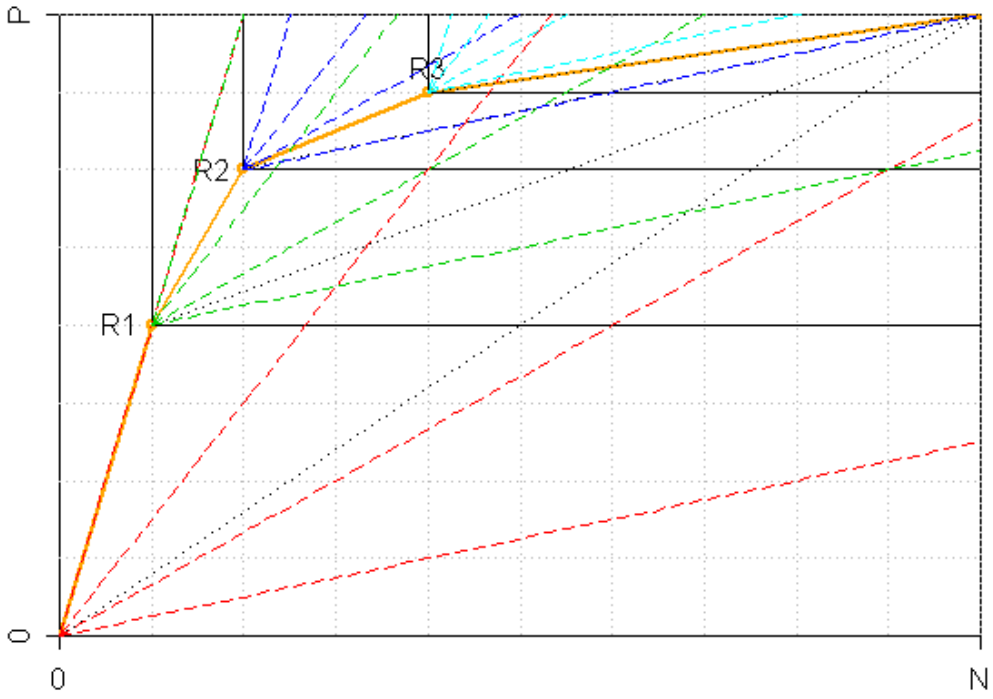
Generalized m-Estimate

TECHNISCHE UNIVERSITÄT DARMSTADT

- 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

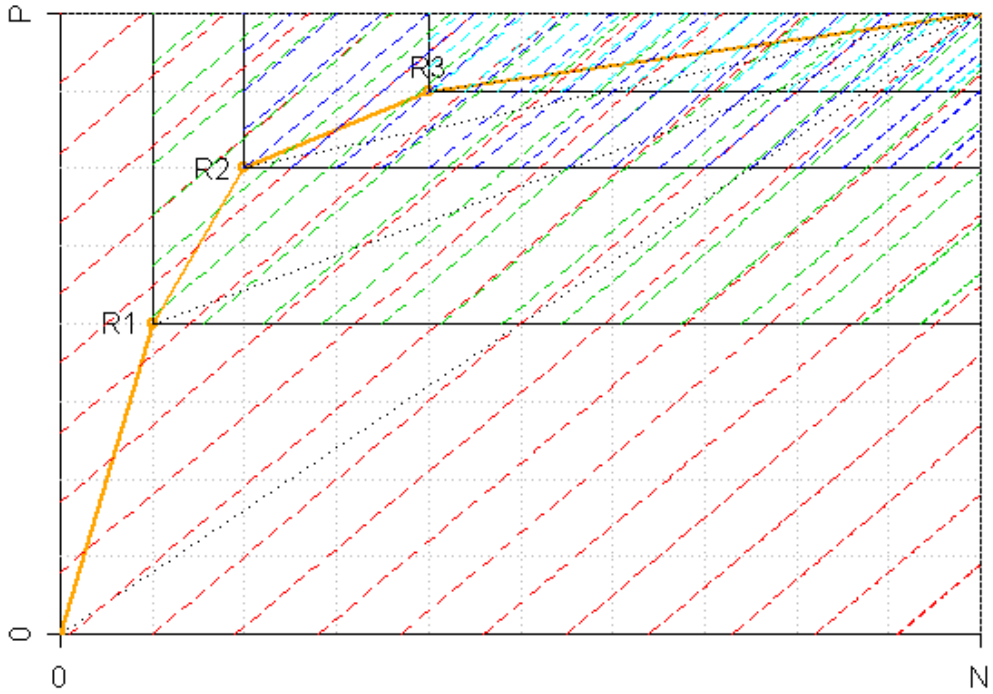
Optimizing Precision

- Precision tries to pick the steepest continuation of the curve
  - tries to maximize the area under this curve
    (→ AUC: Area Under the ROC Curve)
  - no particular angle of isometrics is preferred, i.e. no preference for a certain cost model



Optimizing Accuracy

- Accuracy assumes the same costs in all subspaces
  - a local optimum in a sub-space is also a global optimum in the entire space



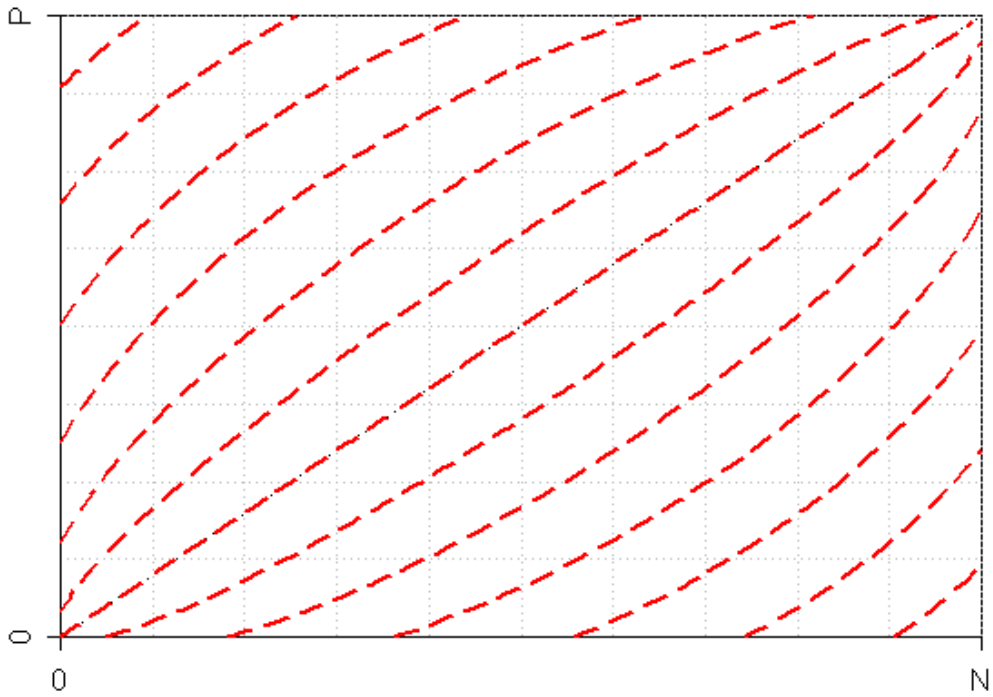
Summary of Linear Rule Learning Heuristics

- There are two basic types of (linear) heuristics.
  - *precision*: rotation around the origin
  - *cost metrics*: parallel lines
- They have different goals
  - *precision* picks the steepest continuation for the curve for unkown costs
  - *linear cost metrics* pick the best point according to known or assumed costs
- The m-heuristic may be interpreted as a trade-off between the two prototypes
  - parameter c chooses the *cost model*
  - parameter m chooses the *"degree of parallelism"*

Correlation

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

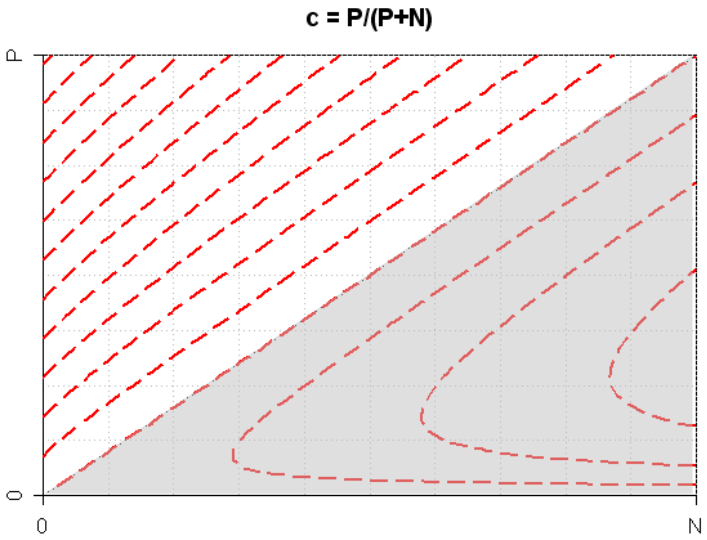
- *basic idea:*
  measure correlation coefficient of predictions with target
- *effects:*
  - non-linear isometrics
  - in comparison to WRA
    - prefers rules near the edges
    - steepness of connection of intersections with edges increases
  - equivalent to χ^2

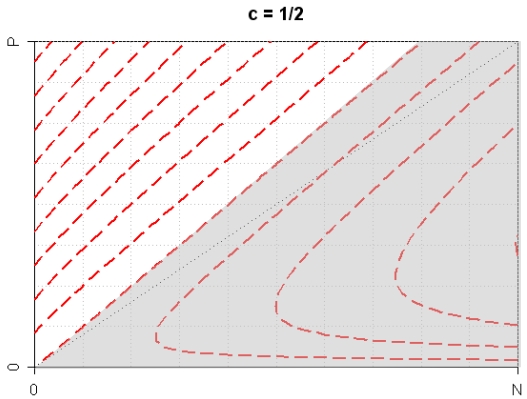


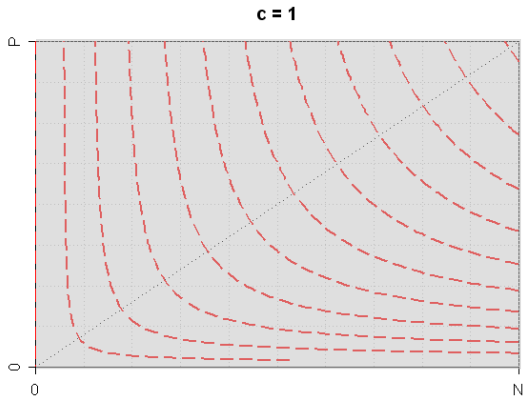
Foil Gain

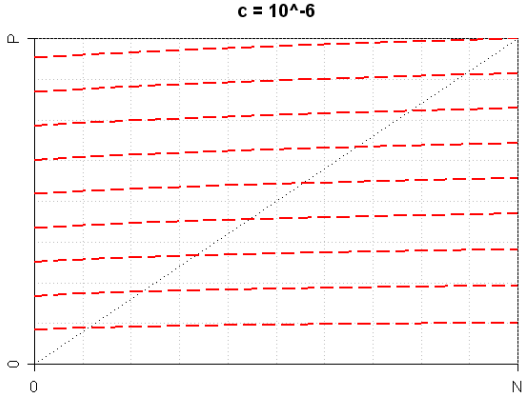
$$h_{foil} = -p\left(\log_2 c - \log_2 \frac{p}{p+n}\right)$$

(c is the precision of the parent rule)









Which Heuristic is Best?

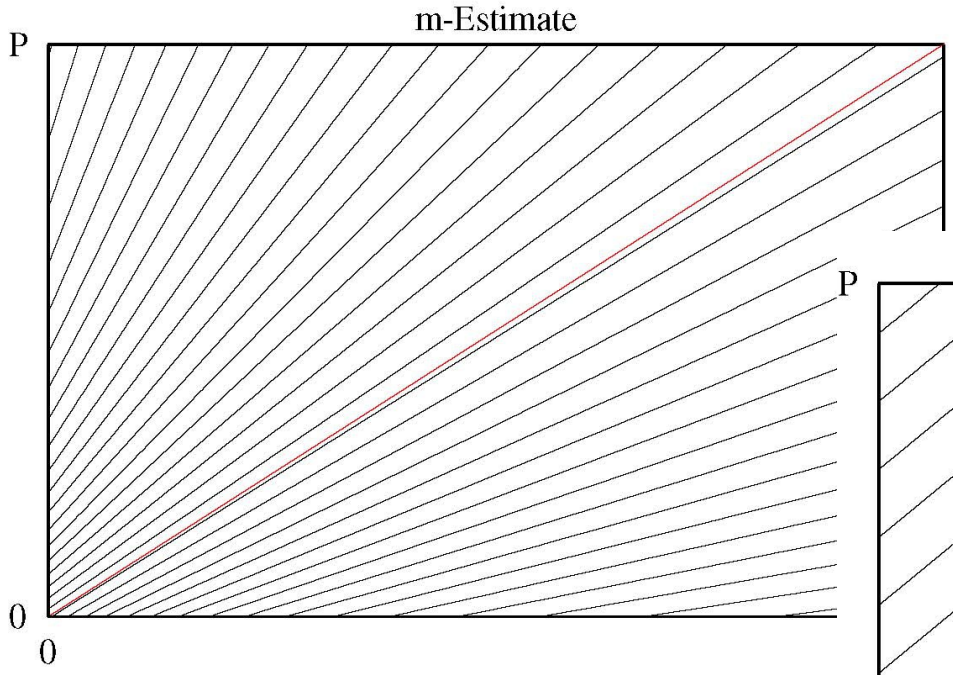
TECHNISCHE UNIVERSITÄT DARMSTADT

- There have been many proposals for different heuristics
  - and many different justifications for these proposals
  - some measures perform better on some datasets, others on other datasets

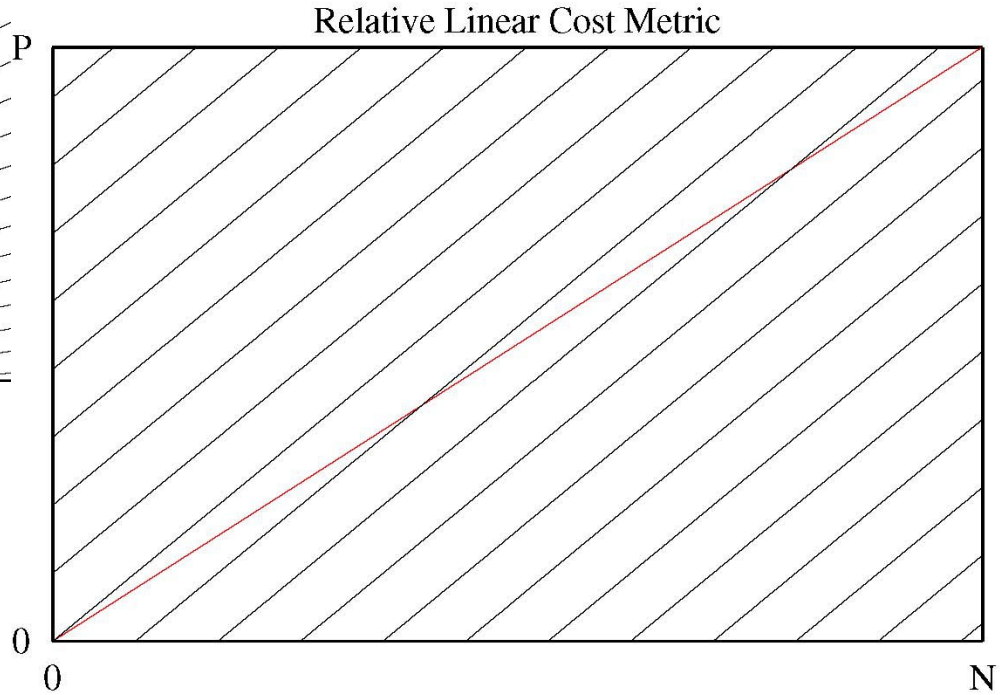
- Large-Scale Empirical Comparison:
  - 27 training datasets
    - on which parameters of the heuristics were tuned)
  - 30 independent datasets
    - which were not seen during optimization
  - Goals:
    - see which heuristics perform best
    - determine good parameter values for parametrized functions

Best Parameter Settings

for m-estimate: $m = 22.5$



for relative cost metric: $c = 0.342$



Empirical Comparison of Different Heuristics

TECHNISCHE UNIVERSITÄT DARMSTADT

Heuristic	Training Datasets Accuracy	# Conditions	Independent Datasets Accuracy	#Conditions
Ripper (JRip)	84,96	16,93	78,97	12,20
Relative Cost Metric (c =0.342)	85,63	26,11	78,87	25,30
m-Estimate (m = 22.466)	85,87	48,26	78,67	46,33
Correlation	83,68	37,48	77,54	47,33
Laplace	82,28	91,81	76,87	117,00
Precision	82,36	101,63	76,22	128,37
Linear Cost Metric (c = 0.437)	82,68	106,30	76,07	122,87
WRA	82,87	14,22	75,82	12,00
Accuracy	82,24	85,93	75,65	99,13

- Ripper is best, but uses pruning (the others don't)
- the optimized parameters for the m-estimate and the relative cost metric perform better than all other heuristics
  - also on the 30 datasets on which they were not optimized
- some heuristics clearly overfit (bad performance with large rules)
- WRA over-generalizes (bad performance with small rules)

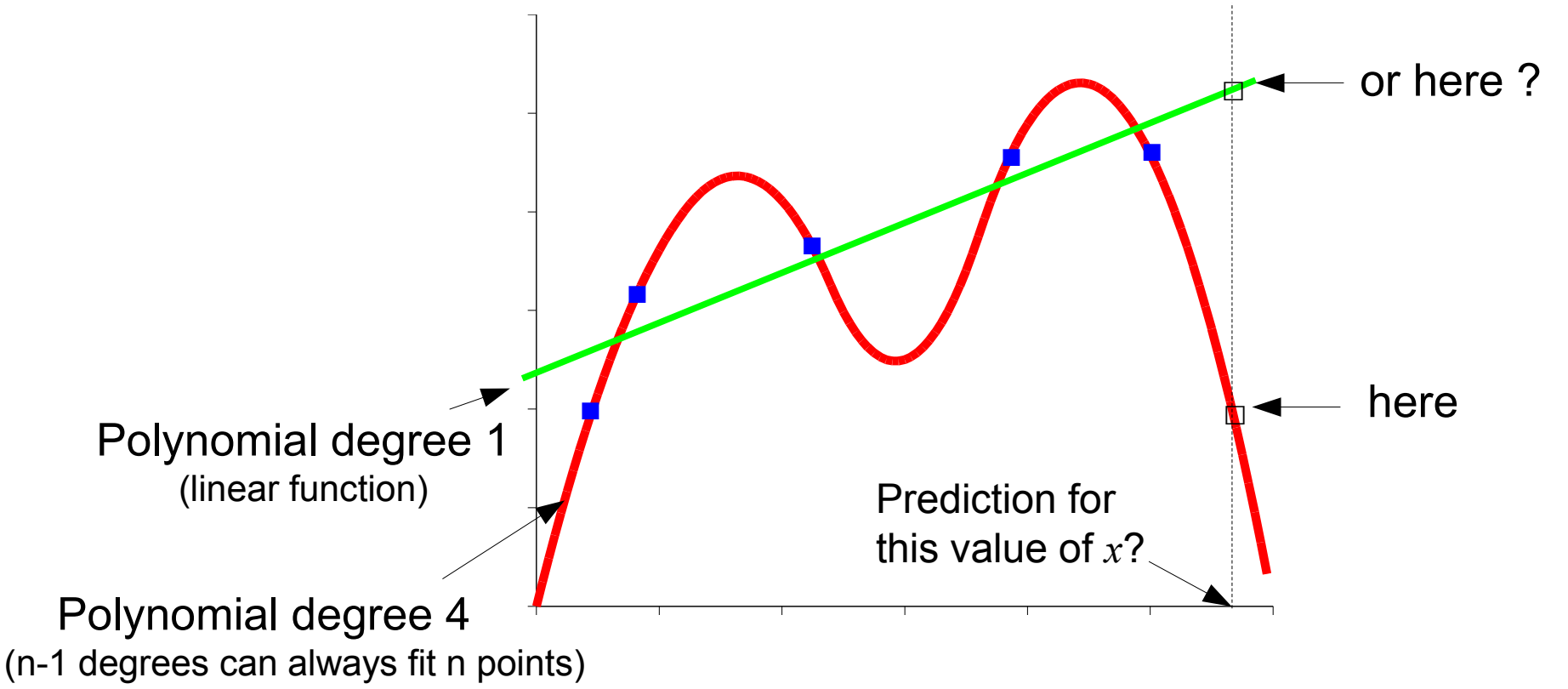
Overfitting

TECHNISCHE UNIVERSITÄT DARMSTADT

- Overfitting
  - Given
    - a fairly general model class
    - enough degrees of freedom
  - you can always find a model that explains the data
    - even if the data contains error (noise in the data)
    - in rule learning: each example is a rule
- Such concepts do not generalize well!

  → Pruning

Overfitting - Illustration

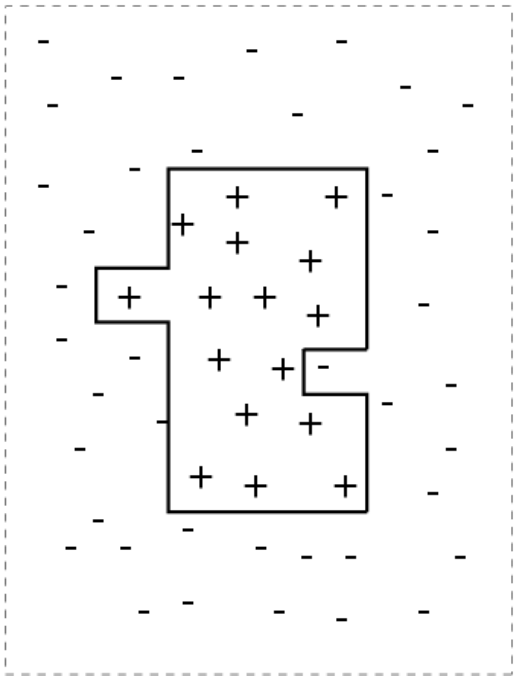


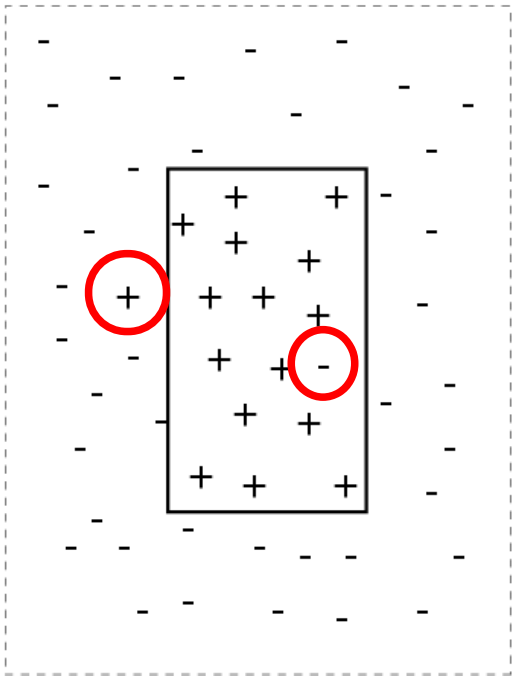
Overfitting Avoidance

TECHNISCHE UNIVERSITÄT DARMSTADT

- A perfect fit to the data is not always a good idea
  - data could be imprecise
    - e.g., random noise
  - the hypothesis space may be inadequate
    - a perfect fit to the data might not even be possible
    - or it may be possible but with bad generalization properties (e.g., generating one rule for each training example)
- Thus it is often a good idea to avoid a perfect fit of the data
  - fitting polynomials so that
    - not all points are exactly on the curve
  - learning concepts so that
    - not all positive examples have to be covered by the theory
    - some negative examples may be covered by the theory

Overfitting Avoidance





- learning concepts so that
  - not all positive examples have to be covered by the theory
  - some negative examples may be covered by the theory

Complexity of Concepts

TECHNISCHE UNIVERSITÄT DARMSTADT

- For simpler concepts there is less danger that they are able to overfit the data
  - for a polynomial of degree n one can choose $n+1$ parameters in order to fit the data points

→ many learning algorithms focus on learning simple concepts

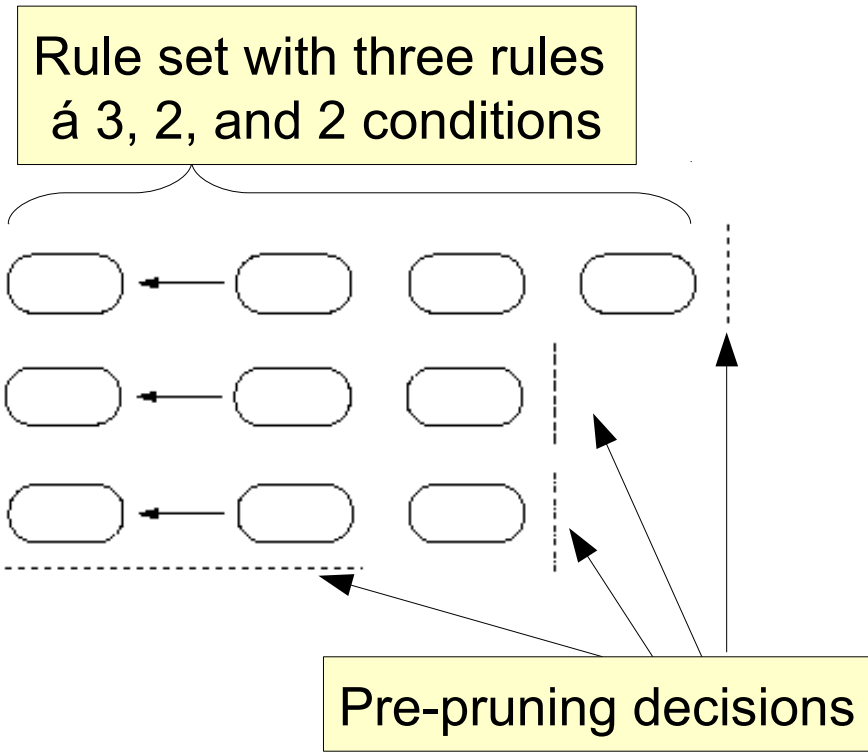
  - a short rule that covers many positive examples (but possibly also a few negatives) is often better than a long rule that covers only a few positive examples

- Pruning: Complex rules will be simplified
  - **Pre-Pruning:**
    - during learning
  - **Post-Pruning:**
    - after learning

Pre-Pruning

TECHNISCHE UNIVERSITÄT DARMSTADT

- keep a theory simple *while* it is learned
  - decide when to stop adding conditions to a rule (*relax consistency* constraint)
  - decide when to stop adding rules to a theory (*relax completeness* constraint)
  - efficient but not accurate



Pre-Pruning Heuristics

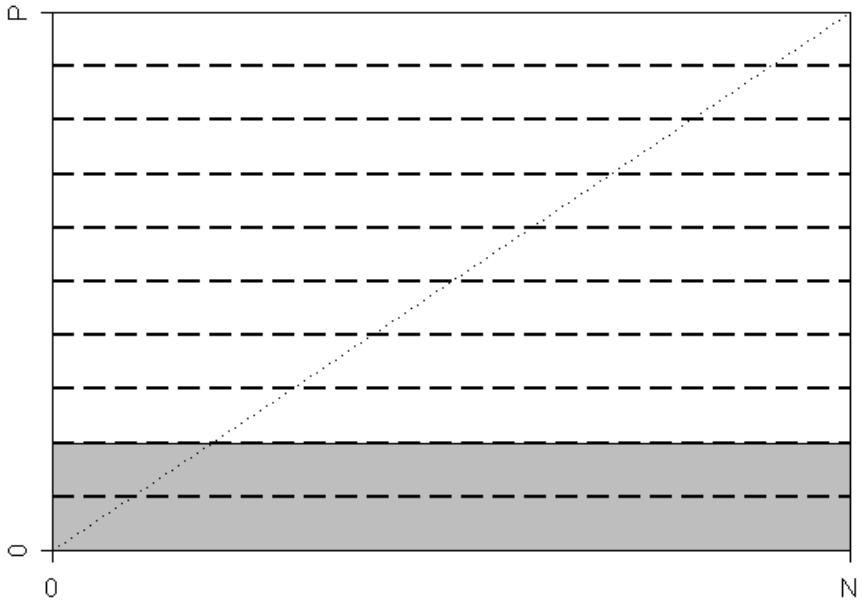
TECHNISCHE UNIVERSITÄT DARMSTADT

1. Thresholding a heuristic value
   - require a certain minimum value of the search heuristic
   - e.g.: Precision > 0.8.
2. Foil's Minimum Description Length Criterion
   - the length of the theory plus the exceptions (misclassified examples) must be shorter than the length of the examples by themselves
   - lengths are measured in bits (information content)
3. CN2's Significance Test
   - tests whether the distribution of the examples covered by a rule deviates significantly from the distribution of the examples in the entire training set
   - if not, discard the rule

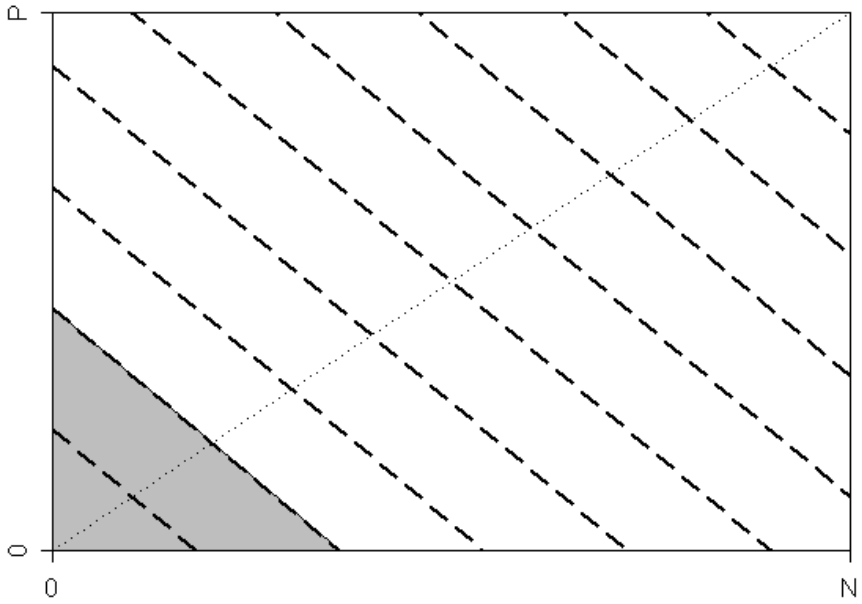
Minimum Coverage Filtering

filter rules that do not cover a minimum number of

positive examples (support)

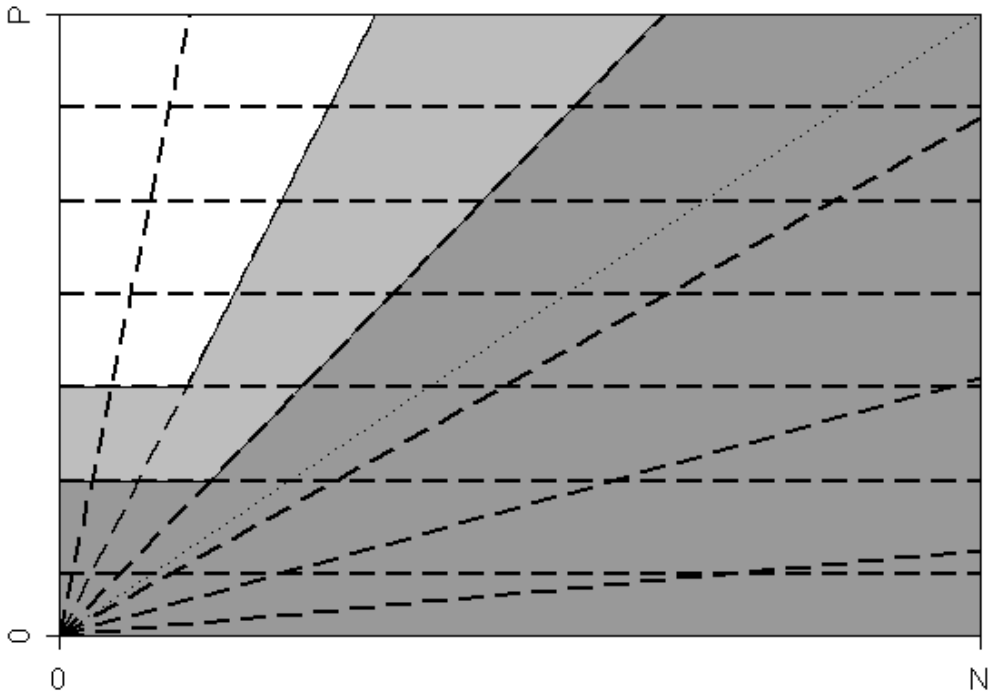


all examples (coverage)



Support/Confidence Filtering

- *basic idea:*
  filter rules that
  - cover not enough positive examples ($p < supp_{min}$)
  - are not precise enough ($h_{prec} < conf_{min}$)
- *effects:*
  - all but a region around $(0,P)$ is filtered



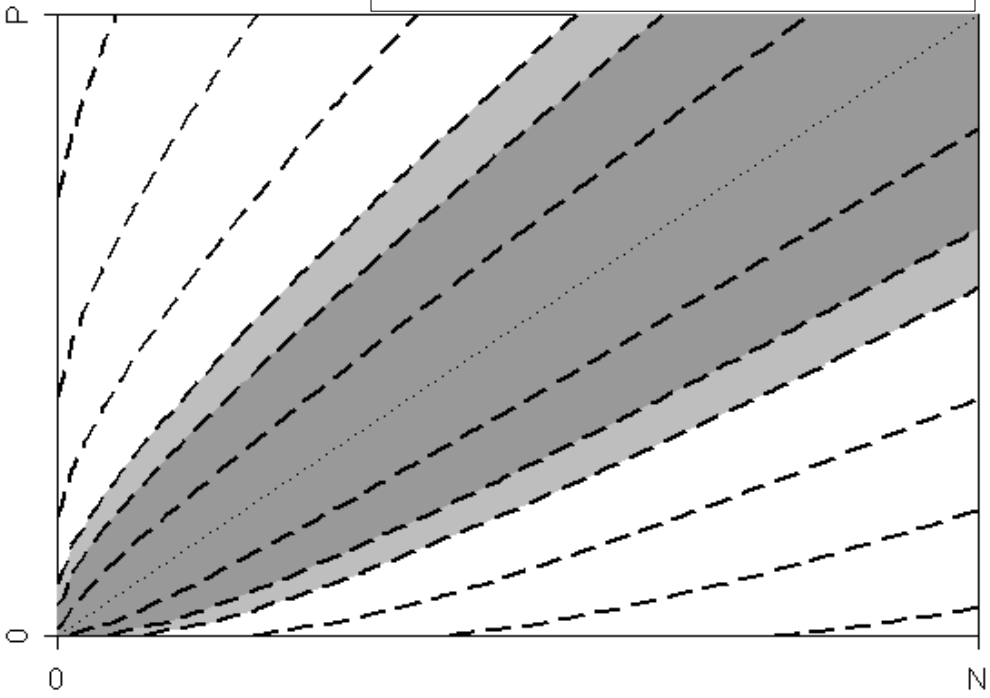
→ we will return to support/confidence in the context of association rule learning algorithms!

CN2's likelihood ratio statistics

$$h_{LRS} = 2\left(p \log \frac{p}{e_p} + n \log \frac{n}{e_n}\right)$$

$e_p = (p+n)\frac{P}{P+N}$; $e_n = (p+n)\frac{N}{P+N}$ are the expected number of positive and negative example in the $p+n$ covered examples.

- *basic idea:*
  measure significant deviation from prior probability distribution
- *effects:*
  - non-linear isometrics
    - similar to m-estimate
    - but prefer rules near the edges
  - distributed χ^2
  - significance levels 95% (dark) and 99% (light grey)

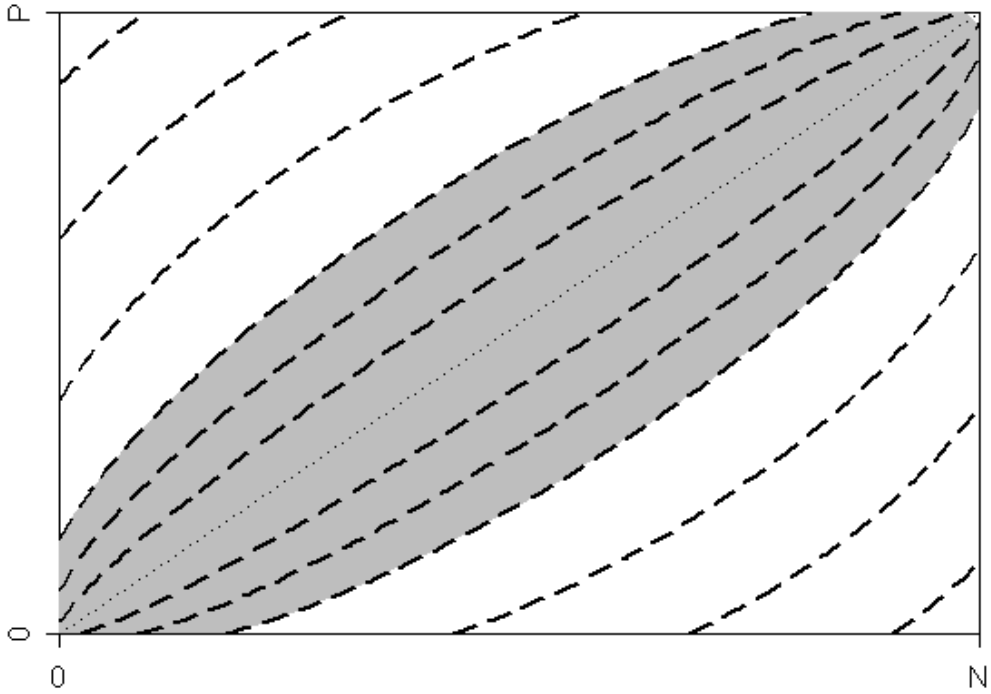


Correlation

TECHNISCHE UNIVERSITÄT DARMSTADT

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

- *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
  - grey area = cutoff of 0.3



MDL-Pruning in Foil

TECHNISCHE UNIVERSITÄT DARMSTADT

- based on the Minimum Description Length-Principle (MDL)
  - is it more effective to transmit the rule or the covered examples?
    - compute the information contents of the rule (in bits)
    - compute the information contents of the examples (in bits)
    - if the rule needs more bits than the examples it covers, on can directly transmit the examples → no need to further refine the rule
  - Details → (Quinlan, 1990)
- doesn't work all that well
  - if rules have expections (i.e., are inconsistent), the negative examples must be encoded as well
    - they must be transmitted, otherwise the receiver could not reconstruct which examples do not conform to the rule
  - finding a minimal encoding (in the information-theoretic sense) is practically impossible

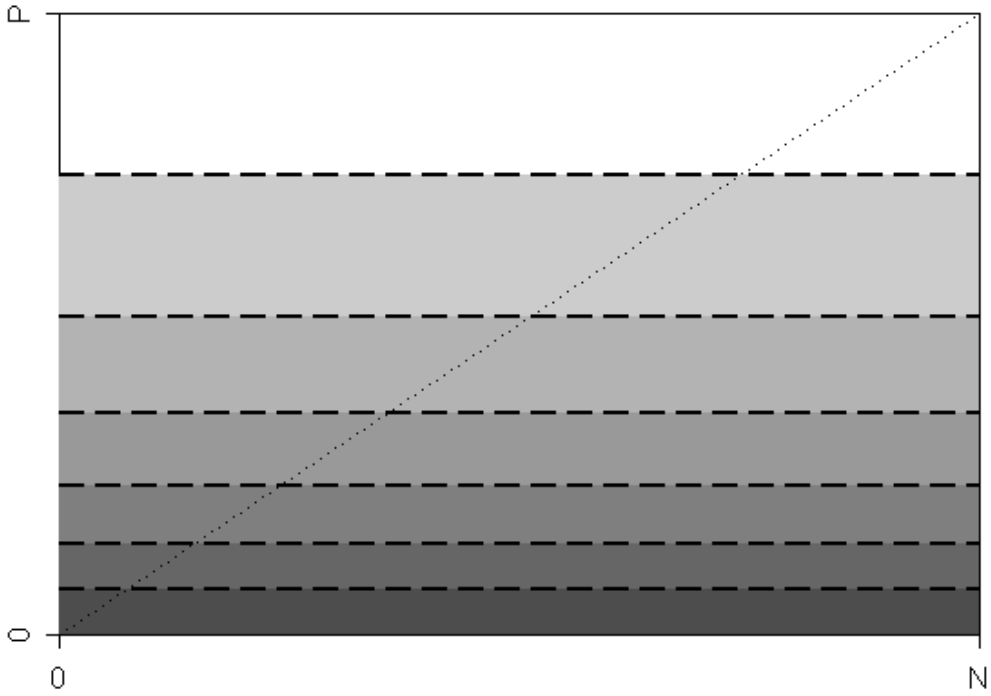
Foil's MDL-based Stopping Criterion

costs for transmitting how many examples we have (can be ignored)

$$h_{MDL} = \log_2(P+N) + \log_2\binom{P+N}{p}$$

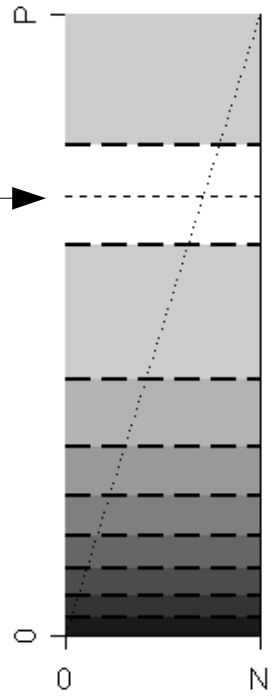
costs for transmitting which of the $P+N$ examples are covered and positive

- *basic idea:*
  compare the encoding length of the rule $l(r)$ to the encoding length h_{MDL} of the example.
  - we assume $l(r) = c$ constant
- *effects:*
  - equivalent to filtering on support
  - because function only depends on p



Anomaly of Foil's Stopping criterion

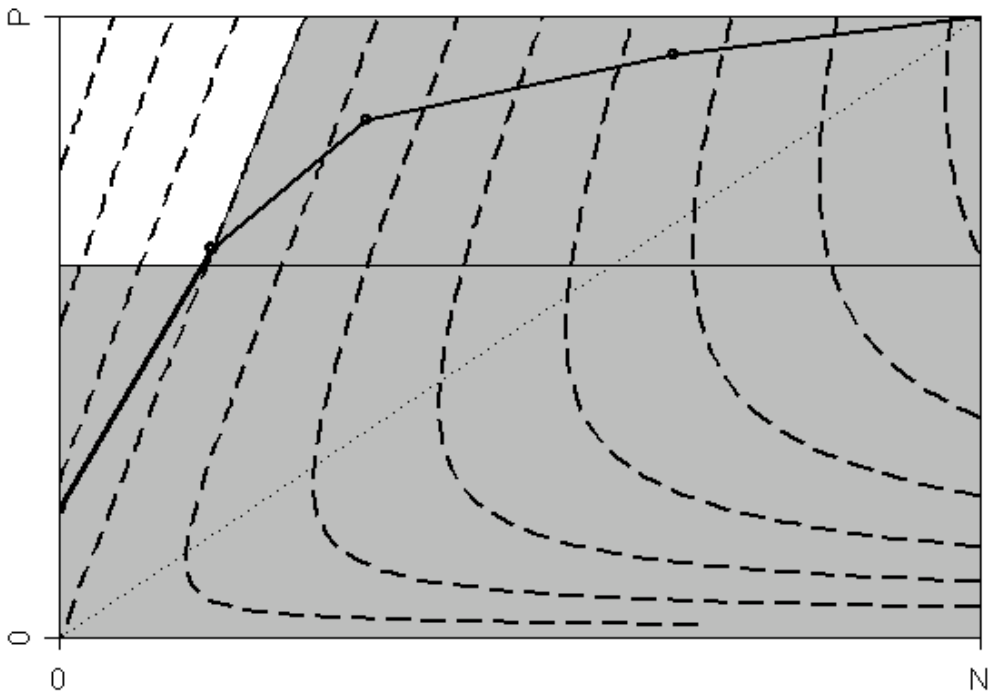
- We have tacitly assumed $N > P$...
- h_{MDL} assumes its maximum at $p = (P+N)/2$
  - thus, for $P > N$, the maximum is not on top!
- there may be rules
  - of equal length
  - covering the same number of negative examples
  - so that the rule covering fewer positive examples is acceptable
  - but the rule covering more positive examples is not!



How Foil Works

→ Foil (almost) implements Support/Confidence Filtering
(will be explained later → association rules)

- filtering of rules with no information gain
  - after each refinement step, the region of acceptable rules is adjusted as in precision/ confidence filtering
- filtering of rules that exceed rule length
  - after each refinement step, the region of acceptable rules adjusted as in support filtering

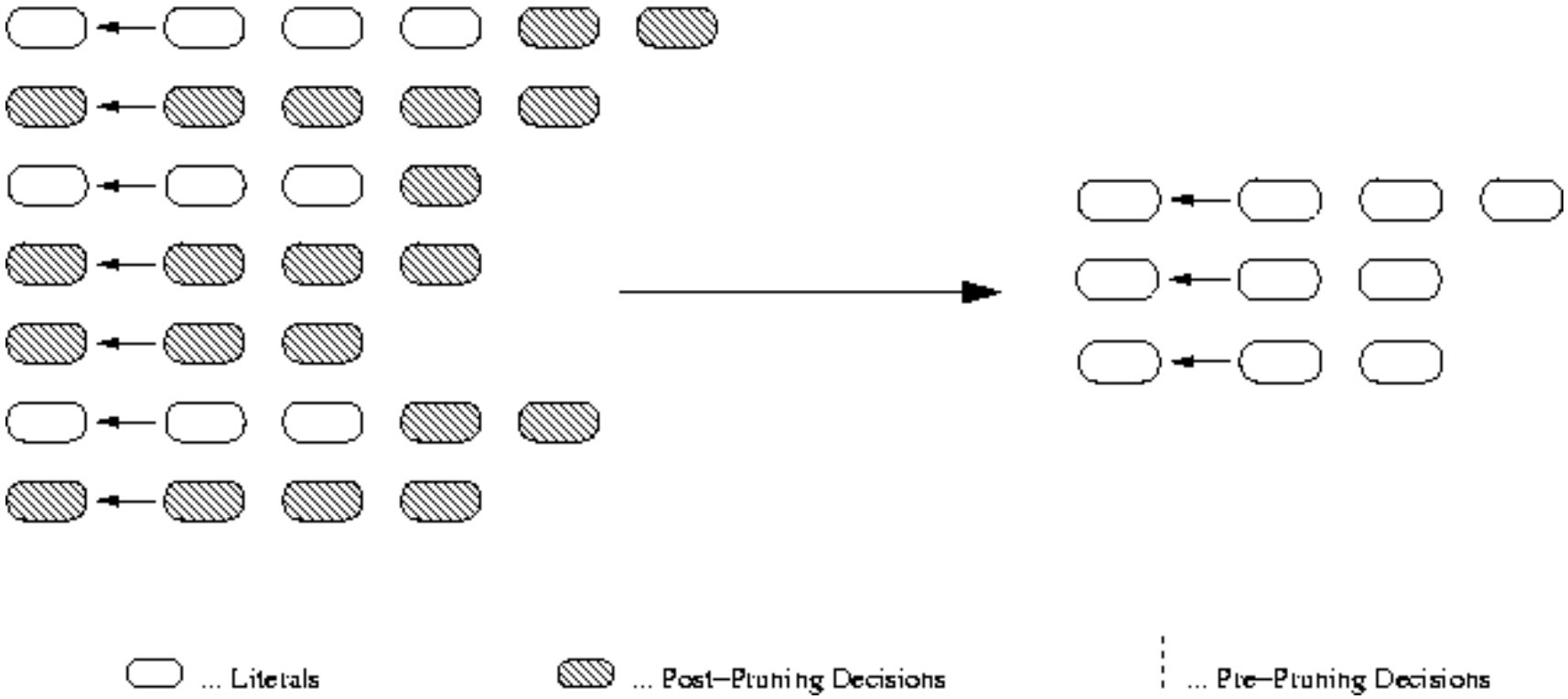


Pre-Pruning Systems

TECHNISCHE UNIVERSITÄT DARMSTADT

- Foil:
  - Search heuristic: Foil Gain
  - Pruning: MDL-Based
- CN2:
  - Search heuristic: Laplace
  - Pruning: Likelihood Ratio
- Fossil:
  - Search heuristic: Correlation
  - Pruning: Threshold

Post Pruning



Post-Pruning: Example

TECHNISCHE UNIVERSITÄT DARMSTADT

IF	T=hot	AND	H=high	AND	O=sunny	AND	W=false	THEN	no
IF	T=hot	AND	H=high	AND	O=sunny	AND	W=true	THEN	no
IF	T=hot	AND	H=high	AND	O=overcast	AND	W=false	THEN	yes
IF	T=cool	AND	H=normal	AND	O=rain	AND	W=false	THEN	yes
IF	T=cool	AND	H=normal	AND	O=overcast	AND	W=true	THEN	yes
IF	T=mild	AND	H=high	AND	O=sunny	AND	W=false	THEN	no
IF	T=cool	AND	H=normal	AND	O=sunny	AND	W=false	THEN	yes
IF	T=mild	AND	H=normal	AND	O=rain	AND	W=false	THEN	yes
IF	T=mild	AND	H=normal	AND	O=sunny	AND	W=true	THEN	yes
IF	T=mild	AND	H=high	AND	O=overcast	AND	W=true	THEN	yes
IF	T=hot	AND	H=normal	AND	O=overcast	AND	W=false	THEN	yes
IF	T=mild	AND	H=high	AND	O=rain	AND	W=true	THEN	no
IF	T=cool	AND	H=normal	AND	O=rain	AND	W=true	THEN	no
IF	T=mild	AND	H=high	AND	O=rain	AND	W=false	THEN	yes

Post-Pruning: Example

TECHNISCHE UNIVERSITÄT DARMSTADT

IF H=high AND O=sunny THEN no

IF O=rain AND W=true THEN no

ELSE yes

Reduced Error Pruning

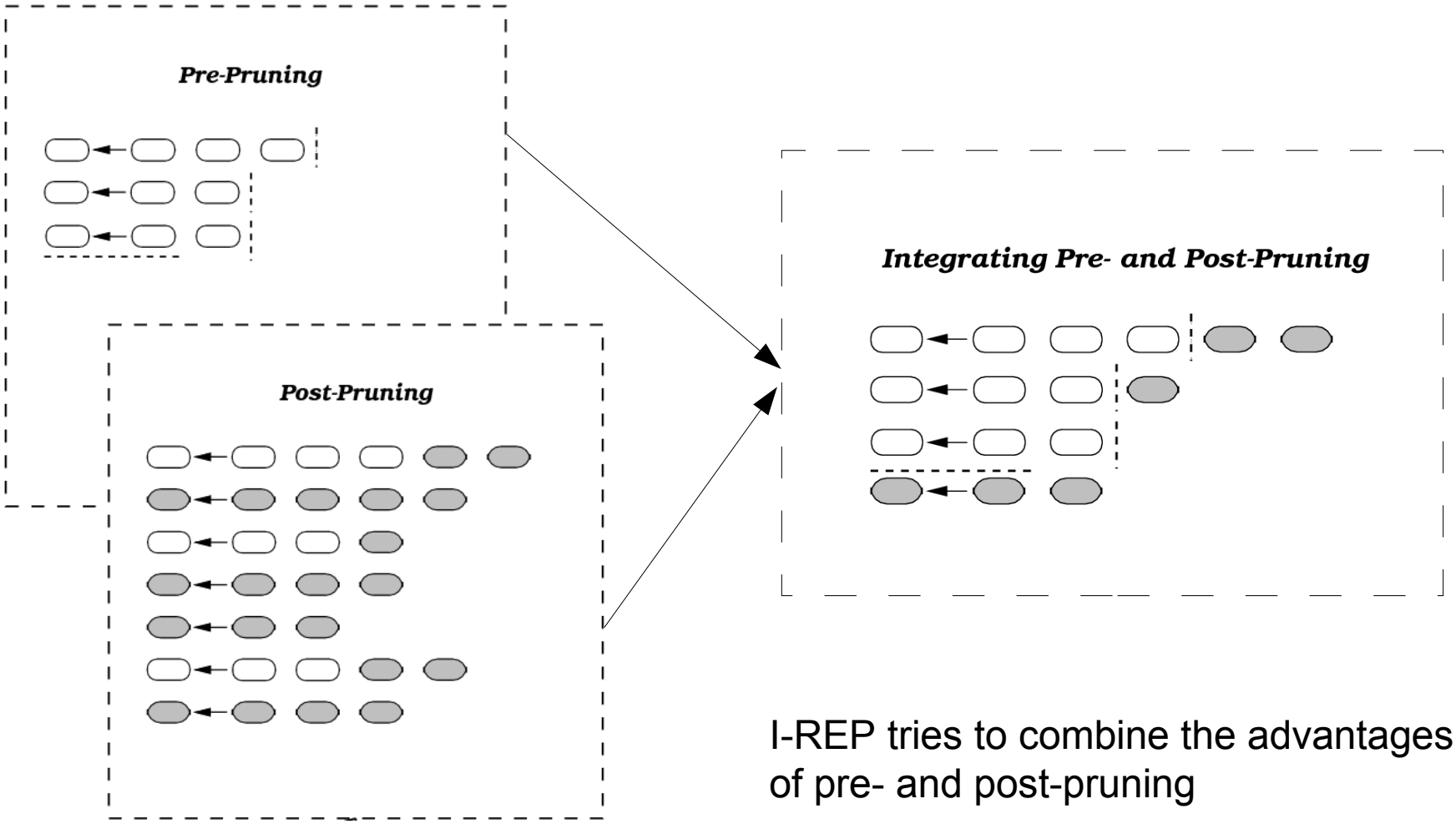
TECHNISCHE UNIVERSITÄT DARMSTADT

- basic idea
  - optimize the accuracy of a rule set on a separate pruning set

1. split training data into a growing and a pruning set
2. learn a complete and consistent rule set covering all positive examples and no negative examples
3. as long as the error on the pruning set does not increase
   - delete condition or rule that results in the largest reduction of error on the pruning set
4. return the remaining rules

- REP is accurate but not efficient
  - $O(n^4)$

Incremental Reduced Error Pruning



I-REP tries to combine the advantages of pre- and post-pruning

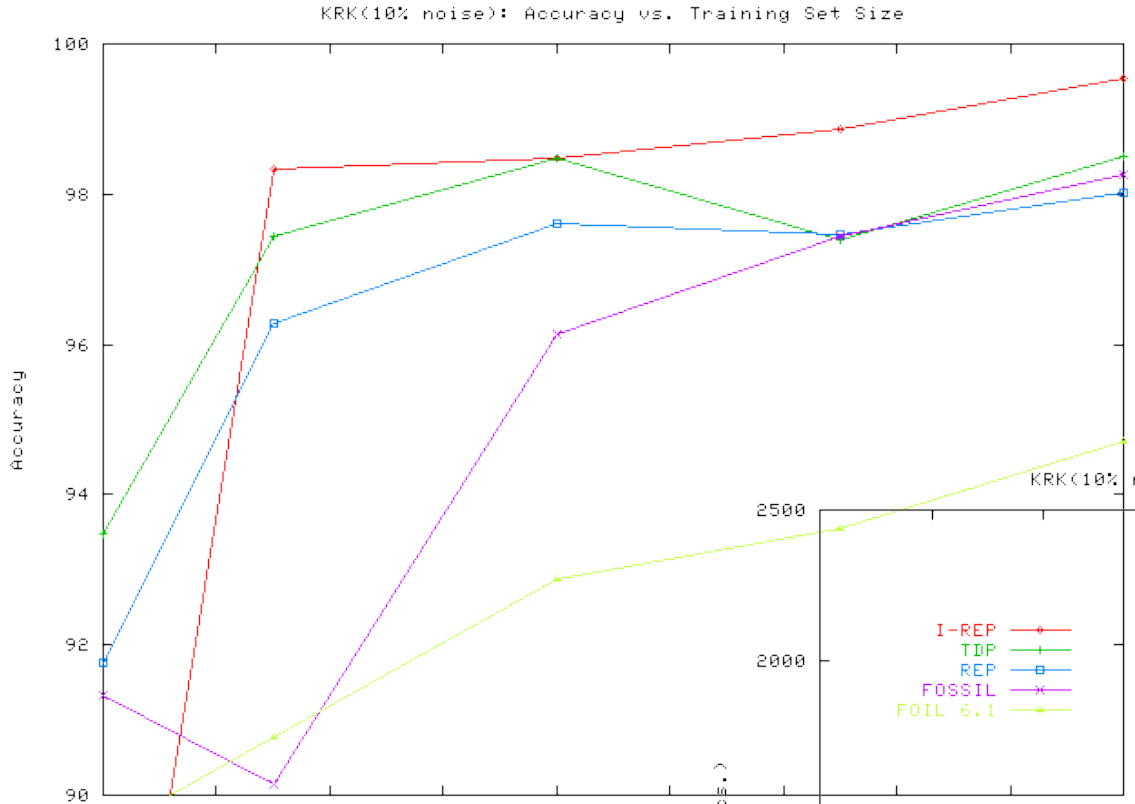
Incremental Reduced Error Pruning

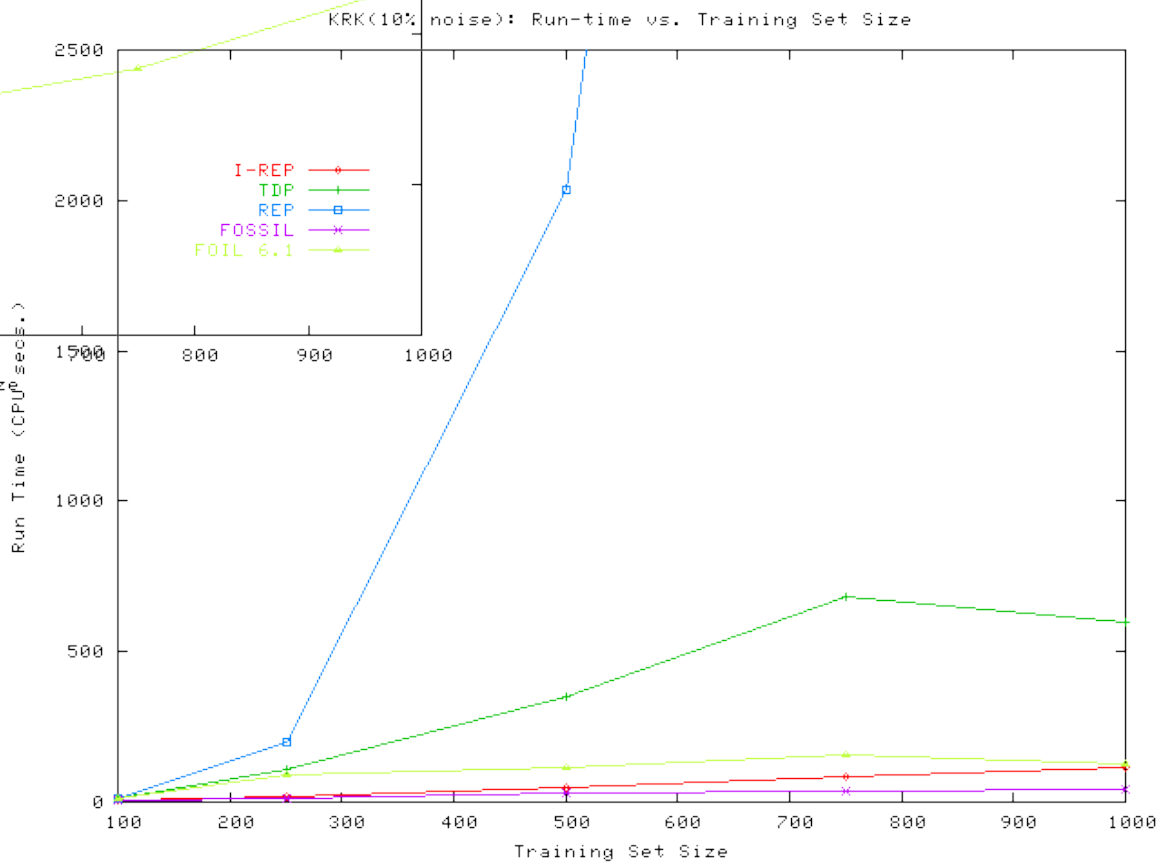
TECHNISCHE UNIVERSITÄT DARMSTADT

- Prune each rule right after it is learned:

1. split training data into a growing and a pruning set
2. learn a consistent rule covering only positive examples
3. delete conditions as long as the error on the pruning set does not increase
4. if the rule is better than the default rule
   - add the rule to the rule set
   - goto 1.

- More accurate, much more efficient
  - because it does not learn overly complex intermediate concept
  - REP: $O(n^4)$ I-REP: $O(n \log^2 n)$
- Subsequently used in RIPPER rule learner (Cohen, 1995)
  - JRip in Weka

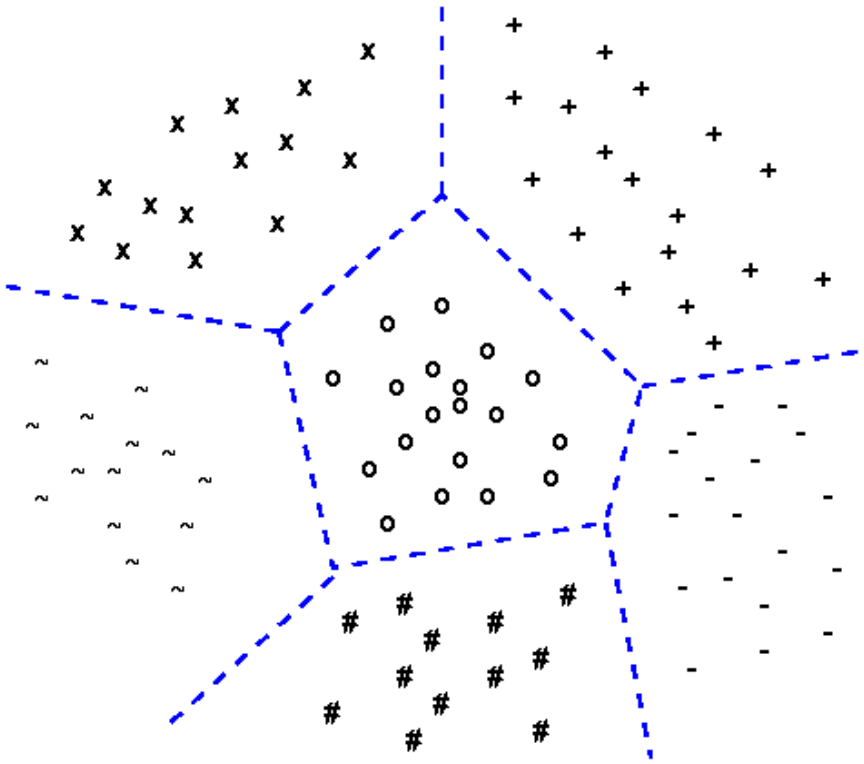




Empirical comparison of accuracy and run-time of various pruning algorithms on a dataset with 10% noise

Multi-class problems

- GOAL: discriminate c classes from each other
- PROBLEM: many learning algorithms are only suitable for binary (2-class) problems
- SOLUTION:
  "Class binarization":
  Transform an c-class problem into a series of 2-class problems

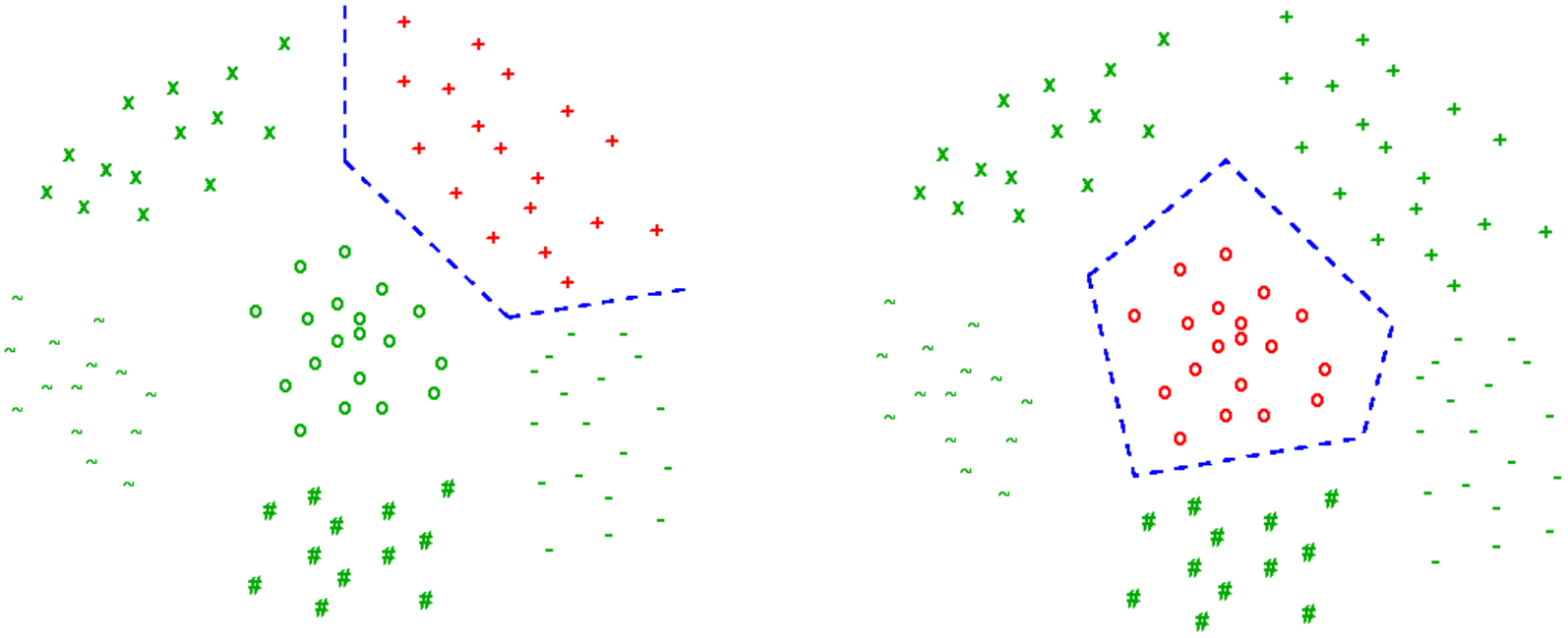


Class Binarization for Rule Learning

TECHNISCHE UNIVERSITÄT DARMSTADT

- None
  - class of a rule is defined by the majority of covered examples
  - decision lists, CN2 (Clark & Niblett 1989)
- One-against-all / unordered
  - foreach class c: label its examples positive, all others negative
  - CN2 (Clark & Boswell 1991), Ripper -a unordered
- Ordered
  - sort classes - learn first against rest - remove first - repeat
  - Ripper (Cohen 1995)
- Error Correcting Output Codes (Dietterich & Bakiri, 1995)
  - generalized by (Allwein, Schapire, & Singer, JMLR 2000)

One-against-all binarization



Treat each class as a separate concept:

- c binary problems, one for each class
- label examples of one class positive, all others negative

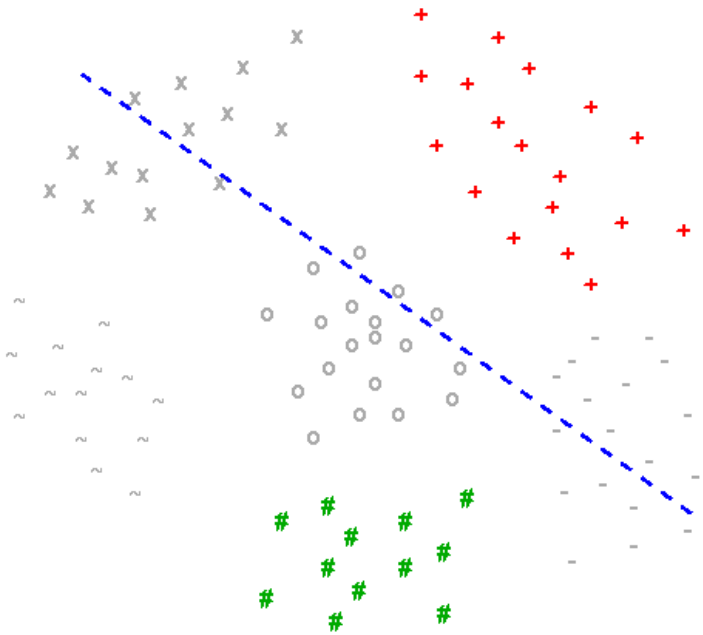
Prediction

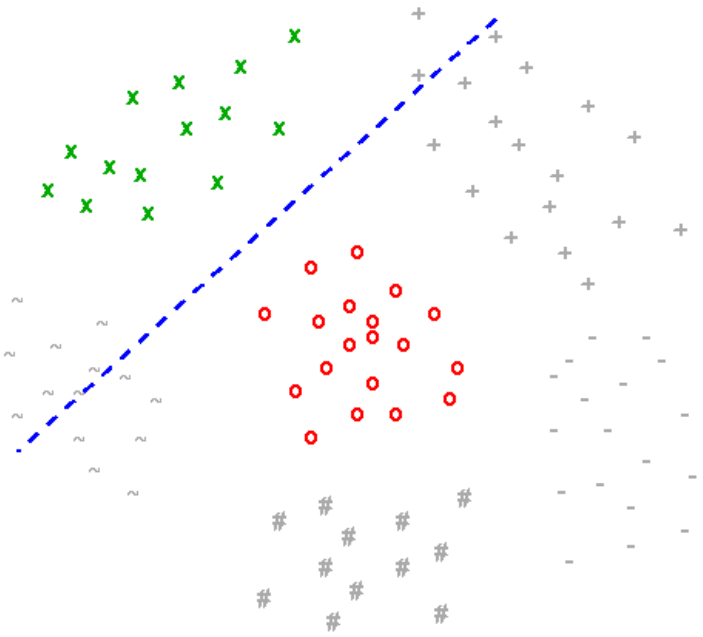
TECHNISCHE UNIVERSITÄT DARMSTADT

- It can happen that multiple rules fire for a class
  - no problem for concept learning (all rules say +)
  - but problematic for multi-class learning
    - because each rule may predict a different class
  - Typical solution:
    - use rule with the highest (Laplace) precision for prediction
  - more complex approaches are possible: e.g., voting
- It can happen that no rule fires on a class
  - no problem for concept learning (the example is then -)
  - but problematic for multi-class learning
    - because it remains unclear which class to select
  - Typical solution: predict the largest class
  - more complex approaches:
    - e.g., rule stretching: find the most similar rule to an example

Pairwise Classification

- $c(c\text{-}1)/2$ problems
- each class against each other class





- ✔ smaller training sets
- ✔ simpler decision boundaries
- ✔ larger margins

Prediction

TECHNISCHE UNIVERSITÄT DARMSTADT

- Voting:
  - as in a sports tournament:
    - each class is a player
    - each player plays each other player, i.e., for each pair of classes we get a prediction which class „wins“
    - the winner receives a point
    - the class with the most points is predicted
      - tie breaks, e.g., in favor of larger classes
- Weighted voting:
  - the vote of each theory is proportional to its own estimate of its correctness
  - e.g., proportional to proportion of examples of the predicted class covered by the rule that makes the prediction

Accuracy

one-vs-all pairwise

dataset	Ripper unord.	Ripper ordered	R^3	ratio	<
abalone	81.03	82.18	72.99	*0.888*	++
covertype	35.37	38.50	33.20	*0.862*	++
letter	15.22	15.75	7.85	*0.498*	++
sat	14.25	17.05	11.15	*0.654*	++
shuttle	0.03	0.06	0.02	*0.375*	=
vowel	64.94	53.25	53.46	*1.004*	=
car	5.79	12.15	2.26	*0.186*	++
glass	35.51	34.58	25.70	*0.743*	++
image	4.15	4.29	3.46	*0.808*	+
lr spectrometer	64.22	61.39	53.11	*0.865*	++
optical	7.79	9.48	3.74	*0.394*	++
page-blocks	2.85	3.38	2.76	*0.816*	++
solar flares (c)	15.91	15.91	15.77	*0.991*	=
solar flares (m)	4.90	5.47	5.04	*0.921*	=
soybean	8.79	8.79	6.30	*0.717*	++
thyroid (hyper)	1.25	1.49	1.11	*0.749*	+
thyroid (hypo)	0.64	0.56	0.53	*0.955*	=
thyroid (repl.)	1.17	0.98	1.01	*1.026*	=
vehicle	28.25	30.38	29.08	*0.957*	=
yeast	44.00	42.39	41.78	*0.986*	=
average	21.80	21.90	18.52	*0.770*	

- error rates on 20 datasets with 4 or more classes
  - 10 significantly better ($p > 0.99$, McNemar)
  - 2 significantly better ($p > 0.95$)
  - 8 equal
  - never (significantly) worse

Advantages of the Pairwise Approach

TECHNISCHE UNIVERSITÄT DARMSTADT

- Accuracy
  - better than one-against-all (also in independent studies)
  - improvement appr. on par with 10 boosting iterations
- Example Size Reduction
  - subtasks might fit into memory where entire task does not
- Stability
  - simpler boundaries/concepts with possibly larger margins
- Understandability
  - similar to pairwise ranking as recommended by Pyle (1999)
- Parallelizable
  - each task is independent of all other tasks
- Modularity
  - train binary classifiers once
  - can be used with different combiners
- Ranking ability
  - provides a ranking of classes for free
- Complexity?
  - we have to learn a quadratic number of theories...
  - but with fewer examples

Training Complexity of PC

Lemma: The total number of training examples for all binary classifiers in a pairwise classification ensemble is $(c-1)\cdot n$

Proof:

- each of the n training examples occurs in all binary tasks where its class is paired with one of the other $c-1$ classes

Theorem: For learning algorithms with at least linear complexity, pairwise classification is more efficient than one-against-all.

Proof Sketch:

- one-against-all binarization needs a total of $c\cdot n$ examples
- fewer training examples are distributed over more classifiers
- more small training sets are faster to train than few large training sets
- for complexity $f(n) = n^o$ $(o > 1)$: $o>1 \rightarrow \sum n_i^o < (\sum n_i)^o$

Myopy of Top-Down Hill-Climbing

TECHNISCHE UNIVERSITÄT DARMSTADT

- 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

TECHNISCHE UNIVERSITÄT DARMSTADT

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

a fully specialized — a single example

1. Start with ~~an empty~~ rule R that covers ~~all examples~~ delete
2. Evaluate all possible ways to ~~add~~ 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

TECHNISCHE UNIVERSITÄT DARMSTADT

	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

TECHNISCHE UNIVERSITÄT DARMSTADT

- 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)

Algorithm	Language Bias: Static: Selectors	Literals	Synt. Restr.	Rel. Clichés	Rule Models	Dyn.: Lang. Hier.	Constr. Ind.	Search Bias: Algorithm: Hill-Climbing	Beam Search	Best First	Stochastic	Strategy: Top-Down	Bottom-Up	Bidirectional	Overfitting Avoidance: Pre-Pruning	Post-Pruning	Integrated
AQ	×							×	×			×					
AQ15	×							×	×			×				×	
AQ17	×						×	×	×			×					
ATRIS	×							×			×			×		×	
BEXA	×							×	×			×			×	×	
CHAMP	×	×	×				×	×	×			×			×		
C1PF	×						×	×				×				×	
CN2	×							×	×			×			×		
CN2-MCI	×						×	×	×			×			×		
CLASS	×									×		×					
DLG	×							×	×				×				
FOCL	×	×		×				×				×			×		
FOIL	×	×	×					×				×			×		
FOSSIL	×	×	×	×				×				×			×		
GA-SMART	×	×		×	×						×	×			×		
GOLEM		×	×					×					×				
GREEDY3	×							×				×				×	
GRENDEL					×			×				×					
GROW	×							×				×				×	
HYDRA	×	×						×				×					
IBL-SMART	×	×		×						×				×	×		
INDUCE	×	×						×	×			×					
I-REP, I^2-REP	×	×	×	×				×				×					×
JOJO	×	×						×						×			
m-FOIL	×	×	×					×	×			×			×		
MDL-FOIL	×	×	×					×				×			×	×	
MILP	×	×	×								×	×			×		
ML-SMART	×	×		×				×	×	×		×			×		
NINA					×	×		×					×				
POSEIDON	×							×	×			×				×	
PREPEND	×							×				×					
PRISM	×							×				×					
PROGOL	×	×	×							×		×					
REP	×	×		×				×				×				×	
RIPPER	×							×				×				×	×
RDT					×			×				×					
SFOIL	×										×	×			×		
SIA	×										×		×			×	
SMART+	×	×		×	×			×	×	×	×	×			×		
SWAP-1	×							×						×		×	
TDP	×	×	×	×				×				×			×	×	

TECHNISCHE UNIVERSITÄT DARMSTADT

- language bias:
  - which type of conditions are allowed (static)
  - which combinations of condictions are allowed (dynamic)
- search bias:
  - search heuristics
  - search algorithm (greedy, stochastic, exhaustive)
  - search strategy (top-down, bottom-up)
- overfitting avoidance bias:
  - pre-pruning (stopping criteria)
  - post-pruning