An Analysis of Stopping and Filtering Criteria for Rule Learning

Johannes Fürnkranz[1] and Peter Flach[2]

[1] TU Darmstadt, Knowledge Engineering Group
Hochschulstraße 10, D-64289 Darmstadt, Germany
fuernkranz@informatik.tu-darmstadt.de
[2] Department of Computer Science, University of Bristol
Merchant Venturers Building, Woodland Road, Bristol BS8 1UB, UK
Peter.Flach@bristol.ac.uk

Abstract. In this paper, we investigate the properties of commonly used pre-pruning heuristics for rule learning by visualizing them in PN-space. PN-space is a variant of ROC-space, which is particularly suited for visualizing the behavior of rule learning and its heuristics. On the one hand, we think that our results lead to a better understanding of the effects of stopping and filtering criteria, and hence to a better understanding of rule learning algorithms in general. On the other hand, we uncover a few shortcomings of commonly used heuristics, thereby hopefully motivating additional work in this area.

1 Introduction

Noisy data and the problem of overfitting theories is typically addressed by biasing the learner towards simpler concepts. In the area of inductive rule learning, there are two fundamental approaches for achieving this goal: *pre-pruning* in the form of stopping and filtering criteria that decide when a rule should no longer be specialized, and *post-pruning*, which simplifies overfitting rules by generalizing them as long as their performance on an independent part of the training set increases [6]. The predominant strategy is pruning, which is mostly due to the success of Ripper [3], arguably the most powerful rule learning algorithm available today. Pre-pruning approaches, such as those used in CN2 [2, 1] or Foil [14], turn out to be inferior in practice.

In this paper, we analyze commonly used pre-pruning heuristics. Our main analytical tool is visualization in PN-space, a variant of ROC-space that is particularly suited for rule learning. We will briefly review PN-spaces in Section 2. There are two slightly different approaches to pre-pruning in rule learning, namely filtering of irrelevant candidate rules and early stopping of the refinement process. We will discuss their differences on the examples of Foil [14] and CN2 [2] in Section 3 before we turn to the main part of our analysis in Section 4. Parts of this paper also appear in [9].

2 PN-space

Our main tool of analysis will be PN-spaces as introduced in [8]. In brief, *PN-space* is quite similar to ROC-space, the main differences being that PN-spaces work with absolute numbers of covered positive and negative examples, whereas ROC-spaces work

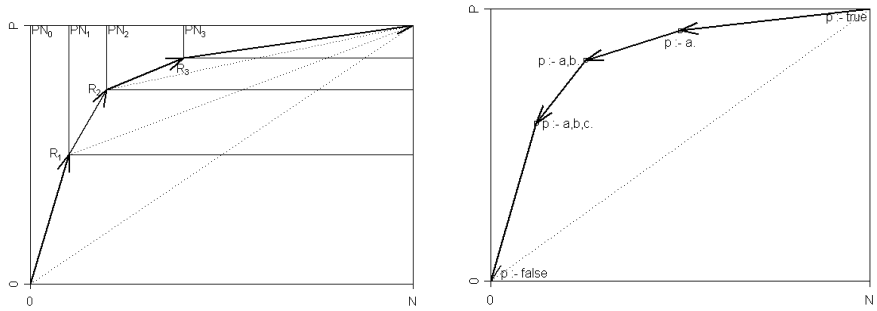

Fig. 1. Schematic depiction of the PN-paths for (left) the covering strategy of learning a theory by adding one rule at a time and (right) greedy specialization of a single rule.

with true positive and false positive rates. A rule that covers p out of a total of P positive examples and n out of N negative examples is represented as a point in PN-space with the co-ordinates (n, p).

The covering or separate-and-conquer strategy for rule learning [7] proceeds by learning one rule at a time. Adding a rule to a rule set means that more examples are classified as positive, i.e., it increases the coverage of the rule set. All positive examples that are uniquely covered by the newly added rule contribute to an increase of the true positive rate on the training data. Conversely, covering additional negative examples may be viewed as increasing the false positive rate on the training data. Therefore, adding rule r_{i+1} to rule set R_i effectively moves from point $R_i = (n_i, p_i)$ (corresponding to the number of negative and positive examples that are covered by previous rules), to a new point $R_{i+1} = (n_{i+1}, p_{i+1})$ (corresponding to the examples covered by the new rule set). Moreover, R_{i+1} will typically be closer to (N, P) and farther away from $(0, 0)$ than R_i.

Consequently, learning a rule set one rule at a time may be viewed as a path through PN-space, where each point on the path corresponds to the addition of a rule to the theory. Such a *PN-path* starts at $(0, 0)$, which corresponds to the empty theory that does not cover any examples. Figure 1 shows the PN-path for a theory with three rules. Each point R_i represents the rule set consisting of the first i rules. Adding a rule moves to a new point in PN-space, corresponding to a theory consisting of all rules that have been learned so far. Removing the covered examples has the effect of moving to a subspace of the original PN-space, using the last rule as the new origin. Thus the path may also be viewed as a sequence of nested PN-spaces PN_i. After the final rule has been learned, one can imagine adding yet another rule with a body that is always true. Adding such a rule has the effect that the theory now classifies *all* examples as positive, i.e., it will take us to the point $\tilde{R} = (N, P)$. Even this theory might be optimal under some cost assumptions.

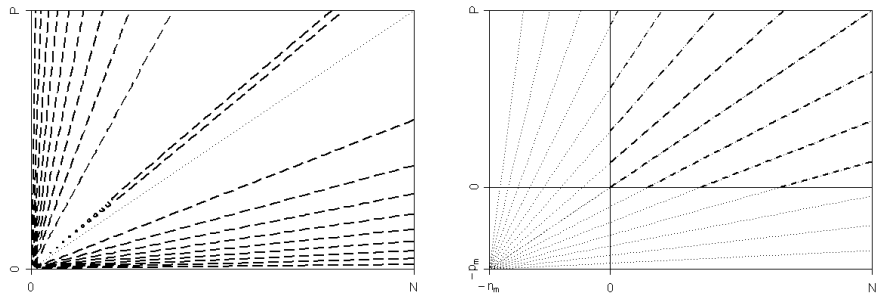

Fig. 2. Isometrics for entropy (left) and the m-estimate (right).

For finding individual rules, the vast majority of algorithms use a heuristic top-down hill-climbing[3] or beam search strategy, i.e., they search the space of possible rules by successively specializing the current best rule [7]. Rules are specialized by greedily adding the condition which promises the highest gain according to some *evaluation metric*. Like with adding rules to a rule set, this successive refinement describes a path trough PN-space (Figure 1, right). However, in this case, the path starts at the upper right corner (covering all positive and negative examples), and successively proceeds towards the origin (which would be a rule that is too specific to cover any example).

PN-spaces are also well-suited for visualizing the behavior of evaluation metrics. For this purpose, we look at their *isometrics*, i.e., the lines that connect the points that are evaluated equal by the used heuristic [8]. In this paper, we employ this methodology for visualizing stopping and filtering criteria of rule learning algorithms.

3 Stopping vs. Filtering

In addition to their regular evaluation metric, many rule learning algorithms employ separate criteria to filter out uninteresting candidates and/or to fight overfitting. There are two slightly different approaches: *stopping criteria* determine when the refinement process should stop whereas *filtering criteria* determine regions of acceptable performance.

As an illustration, let us compare the strategies used by Foil and CN2. CN2 evaluates rules on an absolute scale, using either entropy or the m-estimate. Figure 2 shows their isometrics. For 2-class problems, entropy is more or less equivalent to precision, i.e., its isometrics rotate around the origin $(0, 0)$. The only difference is that entropy is symmetric around the 45 degree line ($p = n$). The reason is that early versions of CN2

[3] The term "top-down hill-climbing" may seem somewhat contradictory: hill-climbing refers to the process of greedily moving towards a (local) optimum of the evaluation function, whereas top-down refers to the fact that the search space is searched by successively specializing the candidate rules, thereby moving downards in the generalization hierarchy induced by the rules.

assigned the class label after the rule has been learned. If a rule is in the area above this line, it will be assigned the positive label, below this line the negative class.

The basic idea of the m-estimate is to assume that each rule covers a certain number of examples *a priori*. It computes a precision estimate, but assumes that it has already observed a total of m examples distributed according to the prior probabilities $(N/(P+N), P/(P+N))$ of the domain. For example, in the special case of the Laplace estimate, both the positive and negative coverage of a rule are initialized with 1 (thus assuming an equal prior distribution). This has the effect that the rotation point of the precision estimate is moved to a point $(-n_m, -p_m)$, where $n_m = p_m = 1$ in the case of the Laplace heuristic, and $p_m = m * P/(P+N)$ and $n_m = m - p_m = m * N/(P+N)$ for the m-estimate. Note that for $m \to \infty$, the isometrics of the m-estimate become increasingly parallel to the diagonal of the PN-space. This is the defining characteristic of weighted relative accuracy [11], which has also been tested in a CN2-like algorithm [16]. Thus, the m-estimate may be considered as a means for trading off precision and WRA.

CN2 evaluates all possible candidate refinements of a rule according to one of these measures. As the evaluation function does not change during the search, the evaluations of all searched rules are comparable. As a consequence, CN2 continues to search until no further refinements are possible, and the *best* rule (the one with the highest evaluation) encountered during this search is returned.

On the other hand, Foil, which forms the basis of many rule learning algorithms, most notably Ripper [3], does not evaluate rules on an absolute scale but relative to their respective predecessors. Hence, the evaluation of two rules with different predecessors are not directly comparable. For this reason, Foil-like algorithms always return the last rule searched. This process is illustrated in Figure 3. In the upper left graph, we see the initial situation: the starting rule covers all examples, i.e., it lies on the upper right corner of PN-space. Foil evaluates its rules using a heuristic based on information gain. This heuristic has value 0 when a refinement has no gain over its predecessor rule. In the beginning, all such rules lie on the the diagonal. In the area above the 0-gain line, the isometrics are almost parallel to this line, i.e., in this region the heuristic behaves almost like a cost-weighted version of accuracy [8]. The area below the 0-gain area (the gray area in the graphs of Figure 3) is not interesting because these are rules that have an information loss, i.e., their quality is worse than the quality of their predecessor.

When Foil adds a condition to a rule, the resulting rule becomes the new starting point for the next refinement step. Consequently, the 0-gain line is rotated until it goes through the point corresponding to this rule. The isometrics of the heuristic rotate with this line and become steeper and steeper the closer the rule moves towards the upper left corner of PN-space. This rotation of the 0-gain line around the origin has an interesting parallel to the use of precision as a search heuristic. The main difference is that, due to the (almost) parallel lines above the 0-gain line, information gain has a tendency to prefer more general refinements, thus trying to stay in the upper regions of PN-space. This may also be interpreted as a method for implementing "patient" rule induction, as advocated in [4].

Because of this cost rotation, Foil will (almost) always find an improvement over its predecessor, because the point (0.1), covering no negative examples and a single

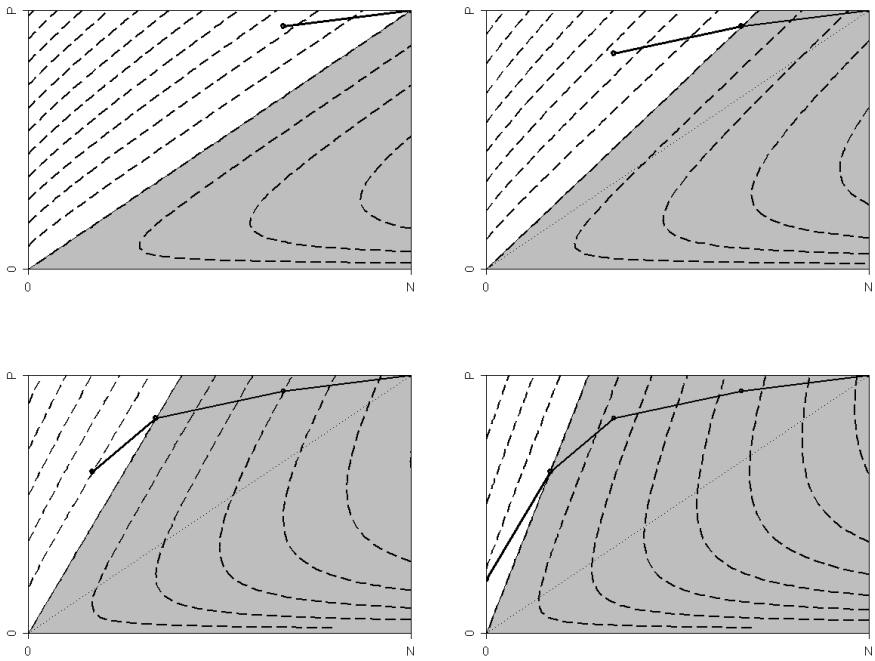

Fig. 3. A typical PN-path for *Foil*.

positive example, is an improvement over all predecessor rules that cover at least 1 negative example. Therefore, Foil's refinement process will usually continue until a pure rule is found. As this typically leads to overfitting, it is crucial to have a good *stopping* criterion, which determines when the refinement process should terminate. For CN2-type algorithms this is not so crucial, because the rule returned is not necessarily the last rule searched, but the rule with the highest evaluation encountered during the search. In this case, the role of a stopping criterion is replaced with a *filtering* criterion that filters out unpromising candidates, but does not directly influence the choice of the best rule. This observation was already made in [1]. As we will see in the next session, filtering criteria typically define an area of PN-space that is ignored by the search space.

Filtering and stopping criteria are closely related. In particular, filtering criteria can also be used as stopping criteria: If no further rule can be found within the acceptable region of a filtering criterion, the learned theory is considered to be complete. Basically the same technique is also used for refining single rules: if no refinement is in the acceptable region, the rule is considered to be complete, and the specialization process stops. For this reason, we will often use the term stopping criterion instead of filtering criterion because this is the more established terminology.

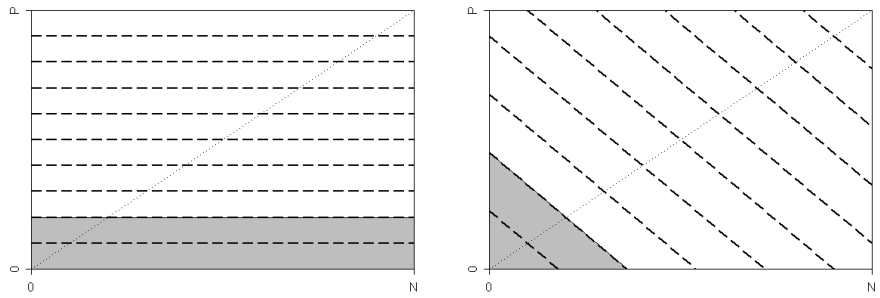

Fig. 4. Thresholds on minimum coverage of positive examples (left) and total number of examples (right).

4 Analysis of Common Criteria

In the following, we illustrate prominent filtering and stopping criteria for greedy specialization: minimum coverage constraints, support and confidence, significance tests, encoding length restrictions, and correlation cutoff. We use PN-space to analyze stopping criteria, by visualizing regions of acceptable hypotheses.

4.1 Minimum Coverage Constraints

The simplest form of overfitting avoidance is to disregard rules with low coverage. For example, one could require that a rule covers a certain minimum number of examples or a minimum number of positive examples. These two cases are illustrated in Figure 4. The graph on the left shows the requirement that a minimum fraction (here 20%) of the positive examples in the training set are covered by the rule. All rules in the gray area are thus excluded from consideration. The right graph illustrates the case where a minimum fraction (here 20%) of examples needs to be covered by the rule, regardless of whether they are positive or negative. Changing the size of the fraction will cut out different slices of the PN-space, each delimited with a coverage isometric (-45 degrees lines). Clearly, in both cases the goal is to fight overfitting by filtering out rules whose quality cannot be reliably estimated because of the small number of training examples they cover. Notice that different misclassification costs can be modeled in this framework by changing the slope of the coverage isometrics.

4.2 Support and Confidence

There is no reason why a single measure should be used for filtering out unpromising rules. The most prominent example for combining multiple estimates are the thresholds on support and confidence that are used mostly in association rule mining algorithms,

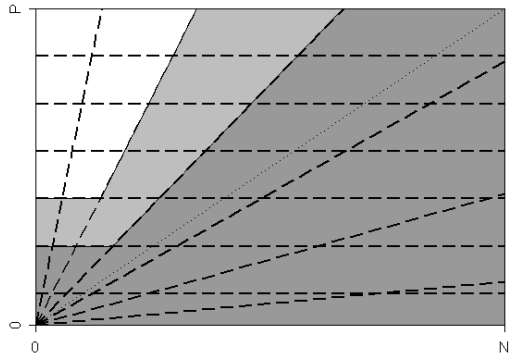

Fig. 5. Filtering rules with minimum support and minimum confidence.

but also in classification algorithms that obtain the candidate rules for the covering loop in an association rule learning framework [12, 13, 10].

Figure 5 illustrates the effect of thresholds on support and confidence in PN-space. Together, they specify an area for valid rules around the $(0, P)$-point in ROC-space. Rules in the gray areas will be filtered out. The dark gray region shows a less restrictive combination of the two thresholds, the light gray region a more restrictive setting. In effect, confidence constrains the quality of the rules, whereas support aims at ensuring a minimum reliability by filtering out rules whose confidence estimate originates from too few positive examples.

4.3 Foil's Encoding Length Restriction

Foil uses a criterion based on the minimum description length (MDL) principle [15] for deciding when to stop refining the current rule. For explicitly indicating the p positive examples covered by the rule, one needs h_{MDL} bits:

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

For the purposes of our analysis, we interpret h_{MDL} as a heuristic that is compared to a variable threshold the size of which depends on the length of the rule $l(r)$. If $h_{MDL}(r) < l(r)$, i.e., if the encoding of the rule is longer than the encoding of the examples themselves, the rule is rejected. As $l(r)$ depends solely on the encoding length (in bits) of the current rule, Foil's stopping criterion depends on the size of the training set: the same rule that is too long for a smaller training set might be good enough for a larger training set, in which it covers more examples.

Figure 6 shows the behavior of h_{MDL} in PN-space. The isometric landscape is equivalent to the minimum support criterion, namely parallel lines to the N-axis. This

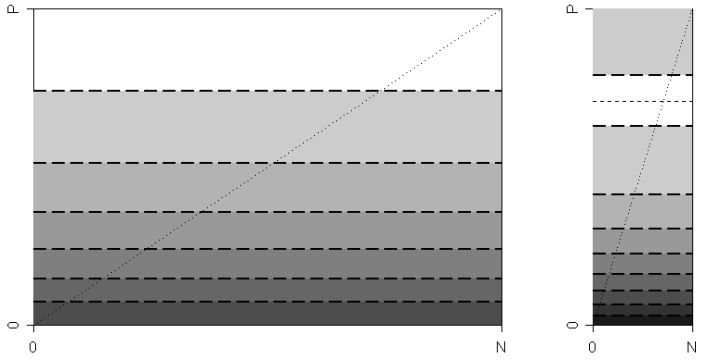

Fig. 6. Illustration of Foil's encoding length restriction for domains with $P < N$ (left) and $P > N$ (right). Lighter shades of gray correspond to larger encoding lengths for the rule.

is not surprising, considering that h_{MDL} is independent of n, the number of covered negative examples.

For $P < N$, h_{MDL} is monotonically increasing with p. If we see this in connection with Foil's search heuristic (Figure 3), we note that while the rule refinement process rotates the 0-gain line towards the right, the MDL metric steadily increases a minimum support constraint. Thus, the combined effect is the same as the effect of support and confidence constraints (Figure 5) with the difference that Foil chooses the thresholds dynamically based on the quality and length of the rules.

Particularly interesting is the case $P > N$ (right graph of Figure 6): the isometric landscape is still the same, but the labels are no longer monotonically increasing. In fact, h_{MDL} has a maximum at the point $p = (P + N)/2$. Below this line (shown dashed in Figure 6), the function is monotonically increasing (as above), but above this line it starts to decrease again. Thus, for skewed class distributions with many positive examples, there might be cases where a rule r is acceptable, while a rule r' that has the same encoding length ($l(r') = l(r)$), covers the same number or fewer negative examples ($n(r') \leq n(r)$), but more positive examples ($p(r') > p(r)$) is *not* acceptable. For example, in the example shown on the right of Figure 6, for a certain rule length l, only rules that cover between 65% and 80% of the positive examples are acceptable. A rule of the same length that covers all positive and no negative examples would not be acceptable. This is very counter-intuitive and sheds some doubts upon the suitability of Foil's encoding length restriction for such domains.

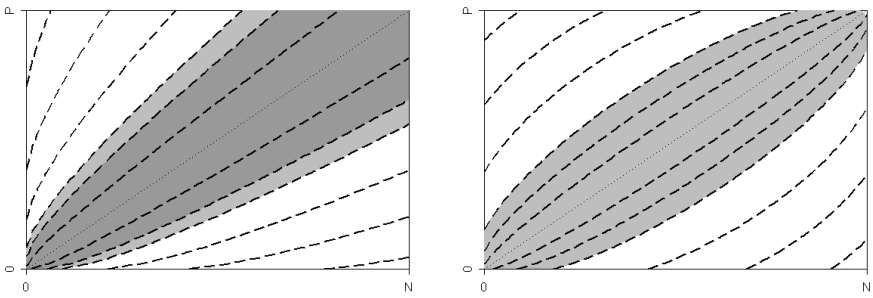

Fig. 7. Left: CN2's significance test, with the regions that would not pass a 95% (dark gray) and a 99% (light gray) significant test.
Right: Fossil's cutoff criterion. The gray region corresponds to the cutoff threshold 0.3.

4.4 CN2's Significance Test

CN2 filters rules for which there is no statistically significant difference between distribution of the covered examples and the distribution of examples in the full data set. To this end, it computes the *likelihood ratio statistic*:

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

where $e_p = (p+n)\frac{P}{P+N}$ and $e_n = (p+n)\frac{N}{P+N} = (p+n) - e_p$ are the number of positive and negative examples one could expect the rule to cover if the $p+n$ examples covered by the rule were distributed in the same way as the $P+N$ examples in the full data set.

The left graph of Figure 7 illustrates CN2's filtering criterion. The dark gray area shows the location of the rules that will be filtered out because it can not be established with 95% confidence that their distribution is different from the distribution in the full dataset. The light gray area shows the set of rules that will be filtered out if 99% confidence in the difference is required. The area is symmetric around the diagonal, which represents all rules that have the same distribution as the full example set.

Note that the likelihood ratio isometrics do not depend on the size of the training set. Any graph corresponding to a bigger training set with the same class distribution will contain this graph in its lower left corner. In other words, whether a rule covering p positive and n negative examples is significant according to h_{lrs} depends on the class distribution of the examples it covers, and on the absolute number of positive and negative examples covered by a rule. Thus, similar to Foil's case the same rule that is not significant for one dataset, might be good enough for a larger training set, in which it has a larger empirical support.

The isometric structure of this heuristic is quite similar to those of precision or the m-heuristic, with the difference that the isometrics are not linear but tilted towards

the origin. Thus, the significance test has a tendency to prefer purer rules. As small pure rules are often the result of overfitting, it is questionable whether this strategy is reasonable if the primary goal of the significance test is to counter overfitting. The main purpose is to filter out uninteresting rules, i.e., rules for which the class distribution of the covered examples does not differ significantly from the *a priori* distribution.

This similarity between search and stopping heuristic also throws up the question why different heuristics are used. It seems to be unlikely that CN2's behavior would be much different if the likelihood statistics is directly used as a search heuristic. Conversely, thresholds upon the m-heuristic would have a similar effect than CN2's significant test (although the semantics of the associated significance levels would be lost).

4.5 Fossil's Correlation Cutoff

Fossil [5] imposes a threshold upon the correlation heuristic h_{corr}.

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

Only rules that evaluate above this threshold are admitted. The left graph of Figure 7 shows the isometric landscape of the correlation heuristic and the effect of a cutoff of 0.3, which appears to perform fairly well over a variety of domains [6]. Fossil uses this criterion as a stopping criterion. Like Foil, it does not return the rule with the highest evaluation, but it continues to add conditions until the stopping criterion fires. Thus, the cutoff line shown in Figure 7 may be viewed as a minimum quality line: learning stops as soon as the path of the learner crosses this line (from the acceptable region to the non-acceptable region), and the last rule above this line is returned.[4]

It can be clearly seen that, like CN2, Fossil focuses upon filtering out uninteresting rules, i.e., rules whose example distribution does not deviate much from the example distribution in the full training set. Similar to CN2, rules that cover few negative examples are preferred by the bended shape of the isometric lines. A major difference between these two approaches is that h_{corr} is independent of the size of the training set, i.e., it always fits the same isometric landscape into PN-space. As a result, the evaluation of a point (n, p) depends on its relative location $(n/N, p/P)$. In this case, the same rule will always be evaluated in the same way, as long as it covers the same fraction of the training data. This differs from the behavior of CN2 and Foil, where the filtering of a rule depends on its absolute location (n, p). It is still an open question which approach is preferable in practice, but there is considerable evidence that Foil's and CN2's pre-pruning heuristic are inefficient in preventing overfitting in the presence of noisy data [5, 16].

[4] The reason for this is that as in Foil, the heuristic is only used for determining the best refinement of the currently searched rule, and not for finding an optimum among all candidate rules. For this type of algorithms, the stopping criterion is particularly crucial. Later versions of Fossil switched to a global evaluation, but no strict empirical comparison of the approaches was performed.

5 Conclusions

In this paper we continued our analysis of rule learning heuristics. While previous work [8] concentrated on learning heuristics, this work focused on pre-pruning heuristics, in particular those that are used in the Foil and CN2 rule learning algorithms. Our main results are

- Foil's MDL-based filtering is quite similar to support/confidence filtering with a dynamic adjustment of the thresholds.
- Foil's criterion has an erratic behavior for class distributions with a majority of positive examples. In these cases, rules that cover more positive and fewer negative examples than the current, acceptable rule may be rejected.
- Fossil's cutoff criterion and CN2's significance test aim at filtering out uninteresting rules (rules for which the distribution of the covered examples does not deviate much from the *a priori* distribution), but do not explicitly address overfitting.
- Whether a rule is filtered by Foil's or CN2's filtering criterion depends on the *absolute* number of covered examples, whereas Fossil filters based on their *relative* number. This may explain the ineffectiveness of Foil's and CN2's pre-pruning heuristics, which is known from several independent studies.

Overall, we believe that our analysis has shown that we are still far from a systematic understanding of pre-pruning heuristics. The fact that, unlike decision-tree algorithms, most state-of-the-art rule learning algorithms use post-pruning for noise-handling may not necessarily be a strong indicator for the superiority of this approach, but may also be interpreted as an indicator of the inadequacy of currently used stopping and filtering criteria.

For this reason, we hope that our work will also support future efforts on designing novel stopping and filtering criteria. Working in PN-space allows one to simply draw the region of PN-space that is intended to be filtered out and find a function that approximates this decision boundary. For stopping criteria, more dynamic criteria can be imagined. For example, a simple criterion would be to stop the generation process whenever the PN-path of the rule set crosses the line that starts in the upper left corner $(0, P)$ and decreases with an angle of -45 degrees. This line represents the points where $p + n = P$, i.e., where the number of examples that are predicted positive equals the number of examples that are actually positive. Thus, a theory that is near this line will adhere to the prior probability of the positive class. Eventually, our work should lead to a broad empirical comparison of different stopping criteria, both new and old, which is currently in preparation.

References

[1] Peter Clark and Robin Boswell. Rule induction with CN2: Some recent improvements. In *Proceedings of the 5th European Working Session on Learning (EWSL-91)*, pages 151–163, Porto, Portugal, 1991. Springer-Verlag.

[2] Peter Clark and Tim Niblett. The CN2 induction algorithm. *Machine Learning*, 3(4):261–283, 1989.

[3] William W. Cohen. Fast effective rule induction. In A. Prieditis and S. Russell, editors, *Proceedings of the 12th International Conference on Machine Learning (ML-95)*, pages 115–123, Lake Tahoe, CA, 1995. Morgan Kaufmann.

[4] Jerome H. Friedman and Nicholas I. Fisher. Bump hunting in high-dimensional data. *Statistics and Computing*, 9(2):1–20, 1999.

[5] Johannes Fürnkranz. FOSSIL: A robust relational learner. In F. Bergadano and L. De Raedt, editors, *Proceedings of the 7th European Conference on Machine Learning (ECML-94)*, volume 784 of *Lecture Notes in Artificial Intelligence*, pages 122–137, Catania, Italy, 1994. Springer-Verlag.

[6] Johannes Fürnkranz. Pruning algorithms for rule learning. *Machine Learning*, 27(2):139–171, 1997.

[7] Johannes Fürnkranz. Separate-and-conquer rule learning. *Artificial Intelligence Review*, 13(1):3–54, February 1999.

[8] Johannes Fürnkranz and Peter Flach. An analysis of rule evaluation metrics. In T. Fawcett and N. Mishra, editors, *Proceedings of the 20th International Conference on Machine Learning (ICML-03)*, pages 202–209, Washington, DC, 2003. AAAI Press.

[9] Johannes Fürnkranz and Peter Flach. Roc 'n' rule learning - towards a better understanding of covering algorithms. *Machine Learning*, To appear.

[10] Viktor Jovanoski and Nada Lavrač. Classification rule learning with APRIORI-C. In P. Brazdil and A. Jorge, editors, *Proceedings of the 10th Portuguese Conference on Artificial Intelligence (EPIA 2001)*, pages 44–51, Porto, Portugal, 2001. Springer-Verlag.

[11] Nada Lavrač, Peter Flach, and Blaz Zupan. Rule evaluation measures: A unifying view. In S. Džeroski and P. Flach, editors, *Proceedings of the 9th International Workshop on Inductive Logic Programming (ILP-99)*, volume 1634 of *Lecture Notes in Artificial Intelligence*, pages 174–185. Springer-Verlag, 1999.

[12] Bing Liu, Wynne Hsu, and Yiming Ma. Integrating classification and association rule mining. In R. Agrawal, P. Stolorz, and G. Piatetsky-Shapiro, editors, *Proceedings of the 4th International Conference on Knowledge Discovery and Data Mining (KDD-98)*, 1998.

[13] Bing Liu, Yiming Ma, and Ching-Kian Wong. Improving an exhaustive search based rule learner. In D. A. Zighed, H. J. Komorowski, and J. M. Zytkow, editors, *Proceedings of the 4th European Conference on Principles and Practice of Knowledge Discovery in Databases (PKDD-2000)*, pages 504–509, Lyon, France, 2000.

[14] J. Ross Quinlan. Learning logical definitions from relations. *Machine Learning*, 5:239–266, 1990.

[15] J. Rissanen. Modeling by shortest data description. *Automatica*, 14:465–471, 1978.

[16] Ljupčo Todorovski, Peter Flach, and Nada Lavrač. Predictive performance of weighted relative accuracy. In D.A. Zighed, J. Komorowski, and J. Zytkow, editors, *Proceedings of the 4th European Conference on Principles of Data Mining and Knowledge Discovery (PKDD-2000)*, pages 255–264, Lyon, France, 2000. Springer-Verlag.