Compression-based Pruning of Decision Lists

Bernhard Pfahringer

Department of Computer Science, University of Waikato, New Zealand
email: bernhard@cs.waikato.ac.nz

Abstract. We define a formula for estimating the coding costs of decision lists for propositional domains. This formula allows for multiple classes and both categorical and numerical attributes. For artificial domains the formula performs quite satisfactory, whereas results are rather mixed and inconclusive for natural domains. Further experiments lead to a principled simplification of the original formula which is robust in both artificial and natural domains. Simple hill-climbing search for the most compressive decision list significantly reduces the complexity of a given decision list while not impeding and sometimes even improving its predictive accuracy.

1 Introduction

The *Minimum Description Length (MDL) Principle* (Rissanen 86), also called the *Minimum Message Length (MML) Principle* (Georgeff & Wallace 84), has been successfully applied in Machine Learning for a broad variety of problems (for a just a few selected papers see Quinlan & Rivest 89, Quinlan 93a, Forsyth et al. 94, Pfahringer 95a, Muggleton et al. 92). But recently also some problems with the MDL principle have been discovered (Quinlan 94, Quinlan 95). The problematic formula described in these papers is used by the C4.5RULES system for pruning the rule-sets of each class separately. Despite these theoretical concerns, in our own experimental experience (see e.g. results reported in Pfahringer 95a) we have found C4.5RULES to be a rather robust and reliable learner. The predictive accuracy of the induced ordered rule-sets (i.e. decision lists) is usually quite good, yet the rule-sets appear to be overly complex.

In this paper we introduce an alternative MDL-based formula for decision lists based on the ideas of Kononenko 95. This formula will be described in section 2. Section 3 will describe the algorithmic usage of the new formula and the experimental setup. In section 4 we first report on good results in artificial domains. But for some of the natural domains results using this original formula are just unacceptable. A hypothesis and a few experiments to explain these unpleasant findings then lead to a simplified version of the original formula which is broadly successful. Section 5 summarizes conclusions and discusses open problems and further research directions.

2 Estimating coding lengths of decision lists

Empirical induction is always faced with the problem of *overfitting* the data, especially in the presence of noise or irrelevant attributes. The MDL principle is a possible solution as it simultaneously judges both the simplicity and the accuracy of a particular induced theory. This is done by summing up both the estimated coding length of the theory and the estimated coding length of the training data given the theory:

```
Total coding length = Coding length to describe the model +
                      Coding length to describe the data,
                       given the model.
```

A theory is called *compressive*, if and only if its coding length is smaller than the coding length of the original training data (using the *null* theory). In Pfahringer 95b we have tried to pinpoint the origin of the problems described in Quinlan 94 and we have defined an alternative formula yielding some improvement. The main idea was to replace the global exception coding mechanism with a local one based on the sets of examples covered by each rule. In the following we will introduce an improved version of this formula using both better (i.e. smaller) coding length estimates in general and extending its applicability to multi-class problems and to both symbolic and numerical attributes.

In Pfahringer 95b we used the entropy of the distribution of symbols in a string to estimate its coding cost, which is valid in a information-theoretic sense, but inefficient for finite-length strings, as is exemplified in Kononenko 95. That paper defines the following formula for estimating the coding length of strings of finite length N constructed from k different possible symbols with respective absolute frequencies $c_1..c_k$:

$$cl(string) = log_2 \binom{N+k-1}{k-1} + log_2 \binom{N}{c_1..c_k} \tag{1}$$

The first part encodes the frequencies of the different symbols, which is sufficient information for constructing the decoder for the compressed string. The size of the compressed string itself is estimated by the second part of the formula.

Using this formula we can now define an MDL-based coding length estimate for decision lists. We assume that both the sender and the receiver know all examples and therefore all possible attributes, their types and values, and also the different possible classes. The sender only wants to transmit the correct classification of each example to the receiver. Furthermore we assume that the last rule of the decision-list has an empty set of conditions, therefore it acts as a kind of default rule, assigning some sensible classification to examples not covered by any previous rule. We use the majority class of all training examples not covered by any rule. Alternatively one could use the global majority class, especially in situations where the number of uncovered examples is very small thus leading to an unreliable class estimate.

Let us now define the total coding length of a decision-list as being just the sum of the coding lengths of its rules:

$$cl(dl) = \Sigma_i cl(rule_i) \tag{2}$$

The coding length estimate of a single rule is the sum of the estimate for encoding the conditions of this rule and the estimate for the class-label string of the examples covered by this rule (using formula 1). Note that we do not explicitly encode the class being predicted by this rule, we just assume that the majority class of the covered examples will be assigned:

$$cl(rule) = cl(tests) + cl(class\_label\_string) \tag{3}$$

The coding length estimate for all tests of a rule is a bit complicated due to the different possible kinds of tests. Categorical attributes involve testing for membership in some subset of all possible values, numerical attributes are tested for being either greater-equal or less-than some threshold. Therefore the coding length estimate for all tests of a single rule is the sum of these three different possible kinds of tests:

$$cl(tests) = cl(subset\_tests) + cl(greaterequal\_tests) + cl(lessthan\_tests) \tag{4}$$

Subset tests are estimated by first selecting those attributes actually involved in subset tests out of all categorical attributes, and then encoding the respective subsets:

$$cl(subsettests) = cl(choose(involved, all\_cat)) + \Sigma cl(subset) \tag{5}$$

The coding length of binary choices such as which attributes are actually used, or which subset of all possible values is tested, is always computed using formula 1. Additionally, it should be noted that this encoding schema assigns equal cost to both testing a categorical attribute for a specific single value (`attrX = valueY`, encoded as a subset-test for just this value) and to testing whether a categorical attribute is different from a specific single value (`attrX <> valueY`, encoded as a subset-test for all other possible values). This is due to the symmetry properties of formula 1.

Numerical tests are encoded in a similar fashion by first selecting the relevant subset of numerical attributes to be tested and than by encoding the respective thresholds. These thresholds are simply chosen from the set of all different values of the respective attribute actually occurring in training data passed down to this rule. This means we ignore possible values occurring only in examples already covered by earlier rules of the decision list. If there are n possible values, we can choose between $n-1$ possible thresholds and therefore need $log_2(n-1)$ bits for encoding a single threshold:

$$cl(lessthan\_tests) = cl(choose(involved, all\_num)) + \Sigma cl(threshold) \tag{6}$$
$$cl(greaterequal\_tests) = cl(choose(involved, all\_num)) + \Sigma cl(threshold) \tag{7}$$
$$cl(threshold) = log_2(n-1) \tag{8}$$

This concludes the specification of the new formula (MDL0). To reiterate, its improvements are a better estimate based on the finite-length property of all encoded strings (as proposed in Kononenko 95), and its extended applicability. Now multiple classes as well as both categorical and numerical attributes are allowed in a domain.

3 Algorithmic usage

For evaluating the formula introduced in the previous section, we have implemented the following simple greedy hill-climbing search for pruning a given decision list (figure 1. The initial search starting point is the decision list returned by C4.5RULES having set all options of both C4.5 and C4.5RULES to their default values, except for enabling *subsetting* of categorical attributes. This results in a decision list being of exactly the syntax expected by our previously defined formula (see section 2). This initial decision list will also be used for comparison purposes later on.

```
mdl_prune(decision_list,training_examples)
{
  find best_basic_step
  if cl(pruned(decision_list,best_basic_step),training_examples) ≤
     cl(decision_list,training_examples)
     then mdl_prune(pruned(decision_list,best_basic_step),training_examples)
     else return decision_list
}
cl(decision_list,training_examples)
{ see section 2
}
```

Fig. 1. Pseudo-code for the pruning of decision lists.

Two basic operations are used for pruning a decision list: deleting a single test from the conditions of one rule or deleting a complete rule by itself. As long as the the total coding length does not increase, always the most compressive basic pruning step is chosen. Even though one might think that pruning complete rules is logically redundant as a basic step (a rule could be pruned by iteratively pruning all its tests), it can still make a big difference in greedy hill-climbing search. Pruning a single test effectively generalizes a rule thereby possibly increasing its coverage. This causes *fewer* examples to be passed on to rules further down the list. Quite contrary, deleting a complete rule causes *more* examples to be passed on, namely all those that used to be covered by the now deleted rule.

Additionally, as we allow for subset tests, one might even add a third pruning operator, namely deleting values from subsets. This would result in even more fine-grained pruning possibilities. We plan to implement such a pruning operator for further experiments.

4 Experimental Results

Ten complete 10-fold stratified cross-validation runs were carried out in each experiment reported below. We always list the mean predictive accuracies and the mean decision list sizes (simply measured as the total number of tests used) for both C4.5RULES and MDL_PRUNE.

4.1 Artificial Domains

Our initial experiments were carried out using two artificial data-sets: the parity of 5 boolean attributes in the presence of an additional 5 irrelevant boolean attributes and the 2-4 multiplexor in the presence of an additional 4 irrelevant attributes. Experiments involved different levels of class noise, where class noise is simulated by flipping the class bit of $N\%$ of the examples drawn at random. In all experiments all possible 1024 (2^{10}) examples were used.

Domain	Noise	Error		Size	
		C4	MDL0	C4	MDL0
Parity5-5	0%	0.00	0.00	160.0	80.0
	5%	8.59	7.96	222.3	148.4
	10%	18.70	22.38	305.6	262.6
	20%	33.29	43.58	259.4	154.6
Mux2-4-4	0%	0.00	0.00	46.4	22.4
	5%	6.39	5.66	70.7	59.6
	10%	11.53	11.39	80.5	52.3
	20%	25.39	24.31	97.9	16.6

Table 1. PARITY5-5 and MUX2-4-4: Average predictive error rates and average rule-set sizes for C4.5RULES and MDL_PRUNE at various levels of class noise.

Table 1 shows the average predictive errors and the average rule-set sizes for both C4.5RULES and MDL_PRUNE. We see that MDL_PRUNE performs quite satisfactorily. Rule-sets are always significantly smaller, sometimes even producing a five-fold reduction in size. And except for Parity5-5 at the 20% class noise level error rates are never dramatically worse, usually their difference is not statistically significant (as judged by a paired t-test).

4.2 Natural Domains

For experiments involving more natural domains we have chosen a few standard databases available from the UCI repository (Merz & Murphy 96). These selected databases range from small to mid-size with regard to number of examples. Some databases comprise solely categorical attributes, whereas others comprise solely numerical attributes, and finally there are a few comprising both categorical and numerical attributes. The selected databases also show a good mix of two-class and multi-class problems.

Domain	Cases	Classes	Cat	Num	Error		Size	
					C4	MDL0	C4	MDL0
Audiology	226	24	70	0	84.82	53.49	8.3	4.1
Breast-w	699	2	0	9	4.56	5.43	21.4	11.1
Credit-a	690	2	9	6	15.59	14.56	35.1	27.2
Diabetes	768	2	0	8	27.15	29.97	50.0	30.3
Glass	214	6	0	9	32.76	55.14	44.3	34.7
Hypo	3163	2	18	7	0.86	0.89	18.5	13.1
Labor	57	2	8	8	14.91	33.33	7.5	1.52
Lymph	148	4	18	0	21.62	32.29	20.5	9.4
Mush	8124	2	22	0	0.00	0.00	19.3	16.8
Sick	3772	2	22	7	1.71	1.76	49.5	29.6
Sonar	208	2	0	60	29.66	35.91	20.6	10.1
Soybean	683	19	35	0	7.80	27.75	63.7	35.5
Vote	435	2	16	0	4.71	4.89	16.7	13.1

Table 2. Properties, average predictive error rates and average rule-set sizes for C4.5RULES and MDL_PRUNE for various natural domains.

The properties of and the experimental results in these natural domains are listed in table 2. Even though we see the expected reduction in rule-set sizes just as we have experienced in artificial domains, the predictive error rates are less satisfactory. Rule-set size reduction is accompanied by a major degradation of predictive performance in some of the domains (see e.g. `soybean`, `labor`, `lymph`, or `glass`).

What is the reason of this at times abysmal performance? The first hypothesis was the suspicion that our coding schema for numerical tests (cf. equations 6 to 8) was indeed overly simplistic and too inefficient, thus leading to overly general rule-sets. But mediocre performance in solely categorical domains (e.g. `soybean` and `lymph`) ruled out this hypothesis as a single cause.

4.3 Modifying the coding length estimator

Inspecting the well-performing data-sets we see that these include all data-sets with more than 1000 training examples overall. Therefore we may conclude that a small number of available training examples might be one of the culprits. To clarify the impact of a small training set samples we devised the following experiment: can we increase the number of available examples in a sensible way? Simple-minded duplication of examples might seriously distort experimental results, because identical examples could be present in both the training and the test set, thus leading to a measure of rote learning capability only. More sophisticated, one could just duplicate the examples once after they have been assigned to the different cross-validation folds. Instead of actually duplicating examples, the same effect can easily be simulated inside the compression formula by modifying equation 3 in the following way:

$$cl(rule) = cl(tests) + f_{dup} * cl(class\_label\_string) \tag{9}$$

By simply multiplying the original class-label string with a duplication factor f_{dup} we can easily simulate even vast artificial training sets. It is interesting to note that most practical systems using some MDL-based formula either internally weigh the two summands differently (Quinlan 93a) or even allow the user to specify weights explicitly (Cohen 95, Oliveira & Sangiovanni-Vincentelli 95).

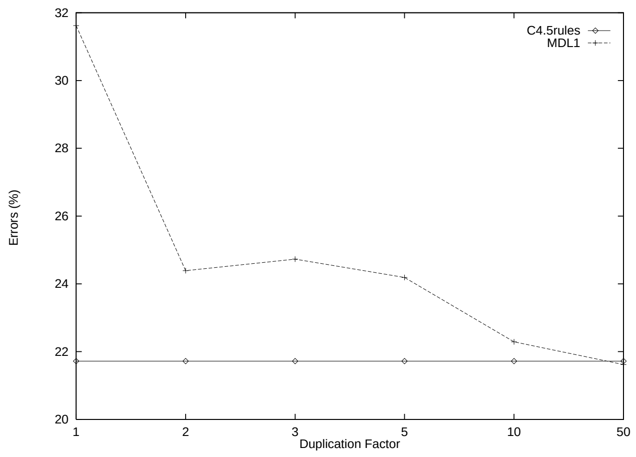

Fig. 2. Lymphgraphy: average error rates for various duplication factors.

Figures 2 and 3 depict the average predictive error rates and decision lists sizes respectively for various settings of f_{dup} in the `lymph` domain in comparison to the base error rates and sizes achieved by `C4.5rules`. We can clearly see that the predictive error steadily decreases until it is statistically insignificant for f_{dup} being 10. And even a much larger setting for f_{dup} still results in a significant almost two-fold reduction of the original rule-set's size.

How can this behaviour be explained? Certainly, for a duplication factor equal to 50, the explicit theory coding cost as computed by equations 4 to 8 is

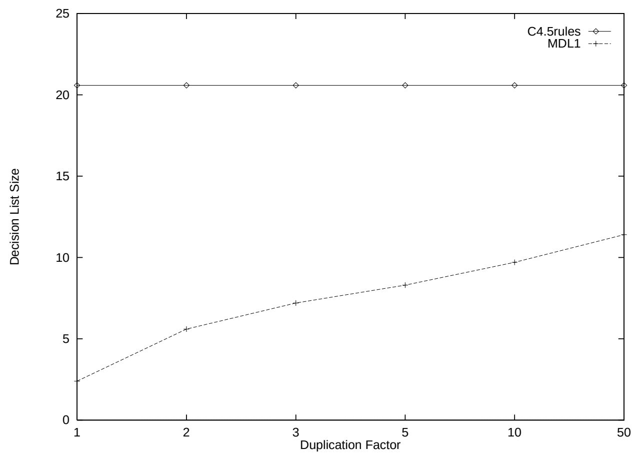

Fig. 3. Lymphgraphy: average rule-set sizes for various duplication factors.

$$cl(dl) = \Sigma_i cl(rule_i)$$
$$cl(rule) = cl(class\_label\_string)$$
$$cl(string) = log_2 \binom{N+k-1}{k-1} + log_2 \binom{N}{c_1..c_k}$$

Fig. 4. The final simplified coding length estimation formula (MDL1).

neglectably small. But the experimental results are very reasonable, still. How can this be the case? Why is the resulting theory not overfitting the data?

One of the reasons is simply the fact that we start with the decision list produced by `C4.5rules`, and that we only prune this list. So the original list forms an upper bound on the syntactic complexity, the pruned list can never be more complex. But there is also an additional, more subtle explanation.

If one investigates the way the size of the class-label string of a single rule is estimated by equation 1, one notices that even absolutely correct rules are assigned some non-zero cost. If all but one of the absolute class frequencies c_i are zero, equation 1 can be simplified to:

$$cl(string) = log_2 \binom{N+k-1}{k-1} \tag{10}$$

Therefore, even in the optimal case of absolutely correct rules we get some reasonable implicit accounting for the size an induced theory.

This immediately suggests the following simplification of our rule-set compression estimation. We can replace the original coding length estimation of single rules (equation 3) by simply:

$$cl(rule) = cl(class\_label\_string) \tag{11}$$

Domain	Error		Size	
	C4	MDL1	C4	MDL1
(Mu0) Mux24 0%	0.00	0.00	46.4	24.0
(Mu1) Mux24 10%	10.74	10.85	92.0	59.0
(Mu2) Mux24 20%	22.87	23.39	98.4	80.8
(Pa0) Par5 0%	0.00	0.00	160.0	80.0
(Pa1) Par5 10%	17.99	16.85	295.7	167.1
(Pa2) Par5 20%	29.72	27.21	298.7	202.2
(AU) Audiology	84.69	73.19	8.1	6.0
(BW) Breast-w	4.55	4.69	20.8	12.3
(CR) Credit-a	15.26	15.24	37.0	24.7
(DI) Diabetes	26.80	27.15	49.0	39.8
(GL) Glass	33.83	33.08	44.6	28.1
(HY) Hypo	0.88	0.89	18.2	10.1
(LA) Labor	14.91	14.56	7.7	3.9
(LY) Lymph	21.62	21.96	20.2	11.2
(MU) Mush	0.00	0.00	19.2	8.8
(SI) Sick	1.69	1.24	49.5	31.4
(SO) Sonar	29.18	29.51	21.5	16.4
(SY) Soybean	7.54	7.26	64.3	39.2
(VO) Vote	4.80	4.69	16.1	9.0

Table 3. Average predictive error rates and average rule-set sizes for C4.5RULES and the modified MDL1_PRUNE for various domains.

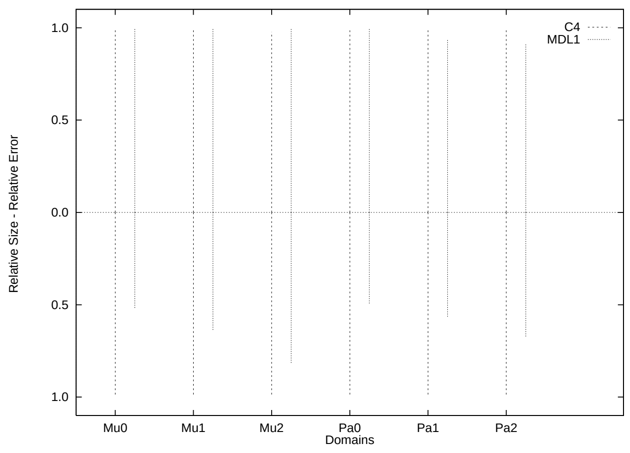

Fig. 5. Error rates and rule-set sizes for artificial domains.

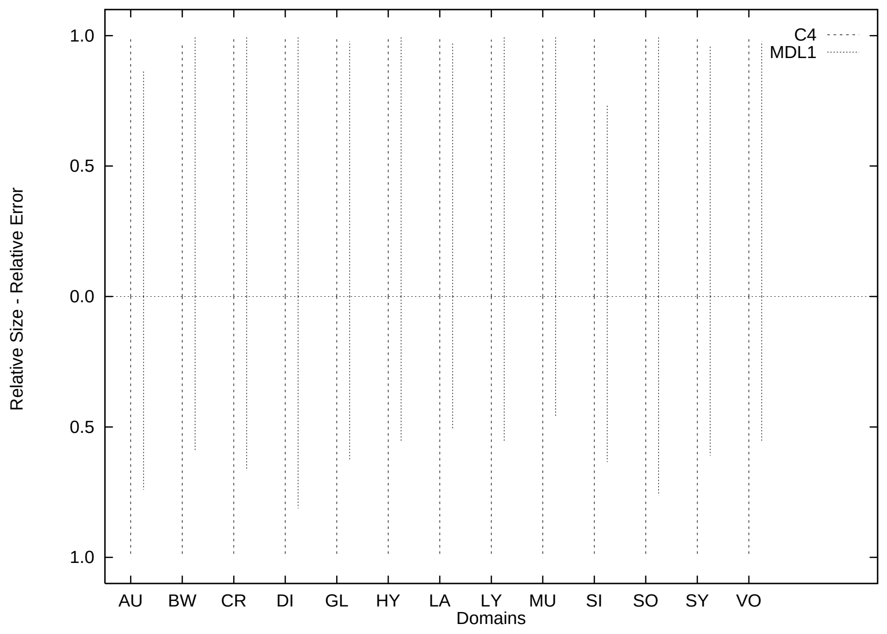

Fig. 6. Error rates and rule-set sizes for natural domains.

This means that we do not compute theory costs explicitly anymore! Therefore equations 4 to 8 are superfluous now, too. All the cost estimates are now just based on the number of subsets of training examples being produced by a given decision list and on how different classes are distributed over these subsets. Figure 4 summarizes this simplified version which will be called MDL1 in the following.

We have rerun all experiments using this simplified formula `MDL1` and report the final results in both table 3 and figures 5 and 6. The figures are provided for simplified relative comparisons. Both error rates and decision list sizes are mapped into the range of $[0, 1]$ by dividing all values by the respective maximal value. Error rates are depicted upwards from zero, whereas decision list sizes printed downwards. Both are grouped together for each domain and both `C4.5rules` and `MDL1`. Smaller impulses indicate smaller error rates and smaller decision list sizes respectively.

The simplified formula `MDL1` seems to do pretty well across all domains. Predictive accuracy is *never* significantly worse and in a few domains even improves somewhat. More importantly, the reduction in terms of size is still quite satisfactory, the pruned decision lists are approximately half as complex in most domains. Only for the two artificial domains there is some indication of overfitting in the presence of noise, as the sizes of the pruned decision lists tend to converge to the upper bound provided by C4.5RULES.

To give an idea for the improved intelligibility of induction results due to the

```
IF pension = none
THEN outcome = bad

IF longterm-disability-assistance = no
THEN outcome =  bad

IF contribution-to-health-plan = none
THEN outcome = bad

IF wage-increase-first-year > 2.3
AND wage-increase-first-year < 2.65
THEN outcome = bad

IF pension <> none
AND longterm-disability-assistance <> no
AND contribution-to-health-plan <> none
THEN outcome = good

IF default
THEN outcome = bad
```

Fig. 7. An initial decision list for `labor` negotiation settlements as returned by C4.5RULES.

```
IF pension <> none
AND longterm-disability-assistance <> no
AND contribution-to-health-plan <> none
THEN outcome = good

IF default
THEN outcome = bad
```

Fig. 8. The pruned decision list for `labor` negotiation settlements as returned by the modified MDL1_PRUNE.

additional compression achievable over C4.5RULES when using the new formula, we list an example rule-set from the **`labor`** negotiation domain. This is a rather small database of final settlements in labor negotiations in Canadian industry. The initial decision-list induced by C4.5RULES is depicted in figure 7.

The new formula allows to considerably prune and simplify this rule-set to a much smaller rule-set of comparable predictive accuracy being depicted in figure 8. It is interesting to note that part of the simplification could have been achieved by simply removing logical redundancies. In the **`labor`** example the first three rules of the original set taken together are just the negation of the single rule predicting the opposite class. The supplied default rule would have predicted correctly anyway in these cases. On the other hand, pruning of the remaining fourth original rule for class `bad` is certainly not simply a removal of some logical redundancy.

So one might speculate that part of the additional amount of simplification gained is due to the global view on decision list simplification inherent in our approach. Whereas `C4.5rules` only optimizes sets of rules predicting the same class, we simplify decision lists as a whole. Thereby the interactions between the different subsets of rules and the default rule are taken into account in an implicit manner.

5 Conclusions

We have successfully engineered a new coding lengths formula for decision-lists, which allows for significant simplification of decision lists without impeding their predictive performance. Simpler decision lists are usually more intelligible as exemplified by an example given in the previous section. The final, simplified formula (figure 4) does not take into account theory cost explicitly, but it does so implicitly to a limited degree. This is one of the probable reasons impeding its original formulation. It also sets limits to the applicability of such a scheme to problems different from propositional rule-set pruning. When inducing and pruning a propositional decision-list in C4.5, a set of rather strong biases is involved in the search for an accurate decision list. Every single rule is of limited complexity, because its number of test-conditions is bounded by the total number of attributes. Additionally, the only disjunctions being induced are so-called internal value disjunctions. This prevents from having the complete theory complexity being shifted into a single rule of high explicit, but low implicit cost. However in the context of constructive induction, such a shift is possible. Therefore one would have to extend the formula to cope with arbitrarily complex attributes possibly derived by constructive induction. Application of the new formula in an ILP context might also be problematic. Although the number of possible conditions in a rule is usually bounded in an ILP system, too, these bounds tend to be much looser in general. More directly applicable scenarios being on our list for further research include variations on the search process itself like:

- Starting directly from a (pruned or unpruned) decision tree.
- Inducing a decision list from scratch using the new formula as its search heuristic.

Another promising direction should be the construction of a similar formula applicable to the prediction of continuous class values. This seems to be a prevalent problem in practise and has therefore got some attention in the Machine Learning community recently (Kramer 96, Quinlan 93b, Weiss & Indurkhya 95).

Instead of relying on the probably overly simple MDL1 formula of figure 4, one could try to estimate the appropriate value for f_{dup} of equation 9 for a given domain. Cross validation in a kind of wrapper approach (Kohavi 95) should be the right tool for this task.

Comparison to other pruning methods which are not based on a variation of the MDL principle are also necessary. A good starting point for this endeavour should be the work described in Fürnkranz 96.

Acknowledgements I would like to thank the ML group at Waikato University for providing an inspiring working environment. This research was sponsored by the Austrian Fonds zur Förderung der Wissenschaftlichen Forschung (FWF) by means of the following grant: Schrödingerstipendium Nr.J01269-MAT.

References

Cohen W.W.: Fast Effective Rule Induction, in Prieditis A. and Russell S.(eds.), *Proceedings of the 12th International Conference on Machine Learning (ML95)*, Morgan Kaufmann, San Francisco, 115-123, 1995.

Forsyth R.S., Clarke D.D., Wright R.L.: Overfitting Revisited: An Information-Theoretic Approach to Simplifying Discrimination Trees, *JETAI Journal of Experimental and Theoretical Artificial Intelligence*, 6,3, 1994.

Fürnkranz J.: Pruning Algorithms for Rule Learning, Österreichisches Forschungsinstitut für Artificial Intelligence, Wien, TR-96-07, 1996.

Georgeff M.P., Wallace C.S.: A General Selection Criterion for Inductive Inference, in O'Shea T.(ed.), *Proceedings of the Sixth European Conference on Artificial Intelligence (ECAI-84)*, Elsevier, Amsterdam, 1984.

Kohavi R.: Wrappers for Performance Enhancement and Oblivious Decision Graphs, Computer Science Dept., Stanford University, Stanford, CA 94305, USA, PhD Dissertation, 1995.

Kononenko I.: On Biases in Estimating the Multi-Valued Attributes, in Mellish C.S.(ed.), *Proceedings of the 14th International Joint Conference on Artificial Intelligence*, Morgan Kaufmann, San Mateo, CA, pp.1034-1040, 1995.

Kramer S.: Structural Regression Trees, in *Proceedings of the Thirteenth National Conference on Artificial Intelligence*, AAAI Press/MIT Press, Cambridge, MA, pp.812-819, 1996.

Merz C.J., Murphy P.M.: UCI Repository of machine learning databases, University of California, Department of Information and Computer Science, Irvine, CA, 1996. [http://www.ics.uci.edu/ mlearn/MLRepository.html]

Muggleton S., Srinivasan A., Bain M.: Compression, Significance, and Accuracy, in Sleeman D. and Edwards P.(eds.), *Machine Learning: Proceedings of the Ninth International Workshop (ML92)*, Morgan Kaufmann, San Mateo, CA, pp.338-347, 1992.

Oliveira A., Sangiovanni-Vincentelli A.: Inferring Reduced Ordered Decision Graphs of Minimal Description Length, in Prieditis A. and Russell S.(eds.), *Proceedings of the 12th International Conference on Machine Learning (ML95)*, Morgan Kaufmann, San Francisco, 1995.

Pfahringer B.: Practical Uses of the Minimum Description Length Principle in Inductive Learning, Institut für Med.Kybernetik u. AI, Technische Universität Wien, Dissertation, 1995.

Pfahringer B.: A New MDL Measure for Robust Rule Induction (Extended Abstract), in Lavrac N. and Wrobel S.(eds.), *Machine Learning: ECML-95*, Springer, Berlin Heidelberg New York, pp.331-334, 1995.

Quinlan, J.R.: Simplifying decision trees. Proc Workshop on Knowledge Acquisition for Knowledge-based Systems, Banff, Canada.(1986)

Quinlan J.R., Rivest R.L.: Inferring Decision Trees Using the Minimum Description Length Principle, *Information and Computation*, 80:227-248, 1989.

Quinlan, J.R.: C4.5: Programs for Machine Learning. Morgan Kaufmann Publishers, 1993.

Quinlan J.R.: Combining Instance-Based and Model-Based Learning, in *Proceedings of the Tenth International Conference on Machine Learning*, Morgan Kaufmann, San Mateo, CA, pp.236-243, 1993.

Quinlan J.R.: The Minimum Description Length Principle and Categorical Theories, in Cohen W.W. and Hirsh H.(eds.), *Machine Learning*, Rutgers University, New Brunswick, NJ, pp.233-241, 1994.

Quinlan J.R.: MDL and Categorical Theories (Continued), in Prieditis A. and Russell S.(eds.), *Proceedings of the 12th International Conference on Machine Learning (ML95)*, Morgan Kaufmann, San Francisco, 1995.

Rissanen J.: Stochastic Complexity and Modeling, in *The Annals of Statistics*, 14(3),p.1080-1100, 1986.

Weiss S.M., Indurkhya N.: Rule-based Machine Learning Methods for Functional Prediction, *Journal of Artificial Intelligence Research 3 (1995)*, 1995.

This article was processed using the LaTeX macro package with LLNCS style