Rule induction for subgroup discovery with CN2-SD

Nada Lavrač[1], Peter Flach[2], Branko Kavšek[1], and Ljupčo Todorovski[1]

[1] Institute Jožef Stefan, Jamova 39, 1000 Ljubljana, Slovenia
{Nada.Lavrac,Branko.Kavsek,Ljupco.Todorovski}@ijs.si
[2] University of Bristol, Bristol, UK
Peter.Flach@bristol.ac.uk

Abstract. Rule learning is typically used in solving classification and prediction tasks. However, learning of classification rules can be adapted also to subgroup discovery. This paper shows how this can be achieved by modifying the CN2 rule learning algorithm. Modifications include a new covering algorithm (weighted covering algorithm), a new search heuristic (weighted relative accuracy), probabilistic classification of instances, and a new measure for evaluating the results of subgroup discovery (area under ROC curve). The main advantage of the proposed approach is that each rule with high weighted accuracy represents a 'chunk' of knowledge about the problem, due to the appropriate tradeoff between accuracy and coverage, achieved through the use of the weighted relative accuracy heuristic. Moreover, unlike the classical covering algorithm, in which only the first few induced rules may be of interest as subgroup descriptors with sufficient coverage (since subsequently induced rules are induced from biased example subsets), the subsequent rules induced by the weighted covering algorithm allow for discovering interesting subgroup properties of the entire population. Experimental results on 17 UCI datasets are very promising, demonstrating big improvements in number of induced rules, rule coverage and rule significance, as well as smaller improvements in rule accuracy and area under ROC curve.

1 Introduction

Classical rule learning algorithms were designed to construct classification and prediction rules [5, 11]. In addition to this area of machine learning, referred to as *predictive induction*, developments in *descriptive induction* have recently gained much attention. These involve mining of association rules (e.g., the APRIORI association rule learning algorithm [1]), subgroup discovery (e.g., the MIDOS subgroup discovery algorithm [17]), and other approaches to non-classificatory induction.

The methodology presented in this paper can be applied to subgroup discovery. As in the MIDOS approach, a subgroup discovery task can be defined as follows: given a population of individuals and a property of those individuals we are interested in, find population subgroups that are statistically 'most

interesting', e.g., are as large as possible and have the most unusual statistical (distributional) characteristics with respect to the property of interest.

This paper investigates how to adapt classical classification rule learning approaches to subgroup discovery, by exploiting the information about class membership in training examples. This paper shows how this can be achieved by appropriately modifying the well-known CN2 rule learning algorithm [4, 5, 3], which we have implemented in Java and incorporated in the WEKA data mining environment [16]. The modified CN2 algorithm and its experimental evaluation in selected domains of the UCI Repository of Machine Learning Databases [12] are outlined. The experimental results are very promising, demonstrating big improvements in number of induced rules, rule coverage and rule significance, as well as smaller improvements in rule accuracy.

This paper is organized as follows. In Section 2 the background for this work is explained: the standard CN2 rule induction algorithm, including the covering algorithm and standard CN2 heuristics, as well as the weighted relative accuracy heuristic and probabilistic classification. Section 3 presents the modified CN2 algorithm, called CN2-SD, adapting the CN2 algorithm for subgroup discovery. Section 4 presents the experimental evaluation in selected UCI domains. Section 5 concludes by summarizing the results and presenting plans for further work.

2 Background

This section presents the backgrounds: classical CN2 rule induction algorithm, including the covering algorithm and standard CN2 heuristics, as well as the weighted relative accuracy heuristic, probabilistic classification and rule evaluation in the ROC space.

The CN2 Rule Induction Algorithm. CN2 is an algorithm for inducing propositional classification rules [4, 5]. CN2 consists of two main procedures: the search procedure that performs beam search in order to find a single rule and the control procedure that repeatedly executes the search.

The search procedure performs beam search using classification accuracy of the rule as a heuristic function. The accuracy of the propositional classification rule `if` *Cond* `then` *Class* is equal to the conditional probability of class *Class*, given that the condition *Cond* is satisfied: $Acc(\texttt{if}\ Cond\ \texttt{then}\ Class) = p(Class|Cond)$.

We replaced the accuracy measure with the weighted relative accuracy, defined in Equation 1 below. Furthermore, different probability estimates, like the Laplace [3] or the m-estimate [2, 6], can be used in CN2 for estimating the above probability and the probabilities in Equation 1. The standard CN2 algorithm used in this work uses the Laplace estimate.

Additionally, CN2 can apply a significance test to the induced rule. The rule is considered to be significant, if it locates regularity unlikely to have occurred by chance. To test significance, CN2 uses the likelihood ratio statistic [5] that

measures the difference between the class probability distribution in the set of examples covered by the rule and the class probability distribution in the set of all training examples. Empirical evaluation in [3] shows that applying a significance test reduces the number of induced rules (and also slightly reduces the predictive accuracy).

Two different control procedures are used in CN2: one for inducing an ordered list of rules and the other for the unordered case. When inducing an ordered list of rules, the search procedure looks for the best rule, according to the heuristic measure, in the current set of training examples. The rule predicts the most frequent class in the set of examples, covered by the induced rule. Before starting another search iteration, all examples covered by the induced rule are removed. The control procedure invokes a new search, until all the examples are covered.

In the unordered case, the control procedure is iterated, inducing rules for each class in turn. For each induced rule, only covered examples belonging to that class are removed, instead of removing all covered examples, like in the ordered case. The negative training examples (i.e., examples that belong to other classes) remain and positives are removed in order to prevent CN2 finding the same rule again.

The Weighted Relative Accuracy Heuristic. Weighted relative accuracy can be meaningfully applied both in the descriptive and predictive induction framework; in this paper we apply this heuristic for subgroup discovery.

We use the following notation. Let $n(Cond)$ stand for the number of instances covered by a rule $Class \leftarrow Cond$, $n(Class)$ stand for the number of examples of class $Class$, and $n(Class.Cond)$ stand for the number of correctly classified examples (true positives). We use $p(Class.Cond)$ etc. for the corresponding probabilities. We then have that rule accuracy can be expressed as $Acc(Class \leftarrow Cond) = p(Class|Cond) = \frac{p(Class.Cond)}{p(Cond)}$. Weighted relative accuracy [10, 15] is defined as follows.

$$WRAcc(Class \leftarrow Cond) = p(Cond).(p(Class|Cond) - p(Class)). \quad (1)$$

Weighted relative accuracy consists of two components: generality $p(Cond)$, and relative accuracy $p(Class|Cond) - p(Class)$. The second term, relative accuracy, is the accuracy gain relative to the fixed rule $Class \leftarrow true$. The latter rule predicts all instances to satisfy $Class$; a rule is only interesting if it improves upon this 'default' accuracy. Another way of viewing relative accuracy is that it measures the utility of connecting rule body $Cond$ with a given rule head $Class$. However, it is easy to obtain high relative accuracy with highly specific rules, i.e., rules with low generality $p(Cond)$. To this end, generality is used as a 'weight', so that weighted relative accuracy trades off generality of the rule ($p(Cond)$, i.e., rule coverage) and relative accuracy ($p(Class|Cond) - p(Class)$).

Probabilistic Classification. The induced rules can be ordered or unordered. Ordered rules are interpreted as a decision list [14] in a straight-forward manner:

when classifying a new example, the rules are sequentially tried and the first rule that covers the example is used for prediction.

In the case of unordered rule sets, the distribution of covered training examples among classes is attached to each rule. Rules of the form:

$$\text{if } Cond \text{ then } Class\ [ClassDistribution]$$

are induced, where numbers in the *ClassDistribution* list denote, for each individual class, how many training examples of this class are covered by the rule. When classifying a new example, all rules are tried and those covering the example are collected. If a clash occurs (several rules with different class predictions cover the example), a voting mechanism is used to obtain the final prediction: the class distributions attached to the rules are summed to determine the most probable class. If no rule fires, a default rule is invoked which predicts the majority class of uncovered training instances.

3 The CN2-SD Algorithm for Subgroup Discovery

The main modifications of the CN2 algorithm, making it appropriate for subgroup discovery, involve the implementation of the weighted covering algorithm, incorporation of example weights into the weighted relative accuracy heuristic, probabilistic classification also in the case of the 'ordered' induction algorithm, and area under ROC curve rule set evaluation.

The Weighted Covering Algorithm. In the classical covering algorithm only the first few induced rules may be of interest as subgroup descriptors with sufficient coverage, since subsequently induced rules are induced from biased example subsets, i.e., subsets including only positive examples not covered by previously induced rules. This bias constrains the population for subgroup discovery in a way that is unnatural for the subgroup discovery process which is, in general, aimed at discovering interesting properties of subgroups of the entire population. In contrast, the subsequent rules induced by the weighted covering algorithm allow for discovering interesting subgroup properties of the entire population.

The weighted covering algorithm is modified in such a way that covered positive examples are not deleted from the current training set. Instead, in each run of the covering loop, the algorithm stores with each example a count how many times (with how many rules induced so far) the example has been covered. Weights derived from these example counts then appear in the computation of *WRAcc*. We have implemented two approaches.

Multiplicative weights. In the first approach, weights decrease multiplicatively. For a given parameter $\gamma < 1$, weights of covered examples decrease as follows: $e(i) = \gamma^i$, where $e(i)$ is the weight of an example being covered i times. Note that the weighted covering algorithm with $\gamma = 1$ would result in finding the same rule over and over again, whereas with $\gamma = 0$ the algorithm would perform the same as the standard CN2 algorithm.

Additive weights. In the second approach, weights of covered examples are modified as follows: $e(i) = \frac{1}{i+1}$.

Modified WRAcc Heuristic with Example Weights. The modification of CN2 reported in [15] affected only the heuristic function: weighted relative accuracy was used as search heuristic, instead of the accuracy heuristic of the original CN2, while everything else stayed the same. In this work, the heuristic function was further modified to enable handling example weights, which provide the means to consider different parts of the instance space in each iteration of the weighted covering algorithm.

In the *WRAcc* computation (Equation 1) all probabilities are computed by relative frequencies. An example weight measures how important it is to cover this example in the next iteration. The initial example weight $e(0) = 1$ means that the example hasn't been covered by any rule, meaning 'please cover this example, it hasn't been covered before', while lower weights mean 'don't try too hard on this example'. The modified *WRAcc* measure is then defined as follows

$$WRAcc(Class \leftarrow Cond) = \frac{n'(Cond)}{N'}(\frac{n'(Class.Cond)}{n'(Cond)} - \frac{n'(Class)}{N'}). \quad (2)$$

where N' is the sum of the weights of all examples, $n'(Cond)$ is the sum of the weights of all covered examples, and $n'(Class.Cond)$ is the sum of the weights of all correctly covered examples.

Probabilistic classification. Each CN2 rule returns a class distribution in terms of numbers of examples covered, as distributed over classes. The CN2 algorithm uses class distribution in classifying unseen instances only in the case of unordered rule sets, where rules are induced separately for each class. In the case of ordered decision lists, the first rule that fires provides the classification. In our modified CN2-SD algorithm, the same probabilistic classification is used in both classifiers, due to overlapping rules. This means that the terminology 'ordered' and 'unordered', which in CN2 distinguished between decision list and rule set induction, has a different meaning in our setting: the 'unordered' algorithm refers to learning classes one by one, while the 'ordered' algorithm refers to finding best rule conditions and assigning the majority class in the head.

4 Experimental evaluation

We experimentally evaluated our approach on 17 data sets from the UCI Repository of Machine Learning Databases [12]. In Table 1, the selected data sets are summarised in terms of the number of attributes, the number of examples, and the percentage of examples of the majority class. These data sets have been widely used in other comparative studies. Since our re-implementation of CN2 currently does not support continuous attributes and can not handle missing

values, all continuous attributes have been discretised and data sets that contain no missing values have been chosen. The discretisation described in [8] was performed using the WEKA tool [16]. Moreover, all of the data sets have two classes, either originally or by selecting one class as 'positive' and joining all the other in a 'negative' class (in Table 1, the selected positive class is indicated by [{ClassName}]); this was done for the purpose of enabling the area under ROC curve evaluation.

Table 1. Characteristics of data sets used in the experiments.

#	Data set	#Attributes	#Examples	Majority class (%)
1	Anneal[{3}]	38	898	76.16
2	Australian	14	690	55.5
3	Balance[{L}]	4	625	46.08
4	Car[{unacc}]	6	1728	70.02
5	Credit-g	20	1000	70
6	Diabetes	8	768	65.1
7	Glass[{build wind non-float}]	9	214	35.51
8	Heart-stat	13	270	55.56
9	Ionosphere	34	351	64.1
10	Iris[{Iris-setosa}]	4	150	33.33
11	Lymph[{metastases}]	18	148	54.72
12	Segment[{brickface}]	19	2310	14.29
13	Sonar	60	208	53.36
14	Tic-tac-toe	9	958	65.34
15	Vehicle[{bus}]	18	846	25.77
16	Wine[{2}]	13	178	39.89
17	Zoo[{mammal}]	17	101	40.59

The performance of different variants of the CN2 rule induction algorithm was measured using 10-fold stratified cross-validation. In particular, we compared the CN2-SD subgroup discovery algorithm with the standard CN2 algorithm (*CN2-standard*, described in [4, 5, 3]) and the CN2 algorithm using *WRAcc* (*CN2-WRAcc*, described in [15]). All these variants of the CN2 algorithm were first re-implemented in the WEKA data mining environment [16], because the use of the same system makes the comparisons more impartial.

The results of these comparisons are presented in Tables 2 and 3, comparing *CN2-SD* with *CN2-standard* and *CN2-WRAcc* in terms of accuracy (Table 2), and size of the rule set (number of rules including the default rule), average example coverage and likelihood ratio[1] per rule (Table 3). Tables for the ordered algorithm are skipped due to space restrictions, and due to the fact that the unordered algorithm is better suited to the philosophy of subgroup discovery due to its aim at inducing independent individual rules. The results of the *CN2-SD* algorithm were computed using both the multiplicative weights (with $\gamma = 0.5, 0.7, 0.9$) and the additive weights. All other parameters of the CN2 algorithm were set to their default values (beam-size = 5, significance-threshold = 99%).

The experimental results show that *CN2-SD* achieves improvements across the board. Additive weights result in about half the number of rules on av-

[1] The likelihood ratio is used in CN2 for testing the significance of the induced rule [5]. For two-class problems this statistic is distributed approximately as χ^2 with one degree of freedom.

Table 2. Accuracy with standard deviation ($Acc \pm sd$) for different variants of the unordered algorithm.

#	**CN2 standard** $Acc \pm sd$	**CN2 WRAcc** $Acc \pm sd$	**CN2-SD** ($\gamma = 0.5$) $Acc \pm sd$	**CN2-SD** ($\gamma = 0.7$) $Acc \pm sd$	**CN2-SD** ($\gamma = 0.9$) $Acc \pm sd$	**CN2-SD (add. weight.)** $Acc \pm sd$
1	98.33 ± 0.11	94.54 ± 0.20	94.77 ± 0.19	95.21 ± 0.19	93.88 ± 0.21	94.65 ± 0.21
2	38.55 ± 0.53	85.51 ± 0.35	84.93 ± 0.35	84.93 ± 0.35	84.78 ± 0.35	84.93 ± 0.35
3	75.68 ± 0.39	81.76 ± 0.38	85.12 ± 0.38	86.40 ± 0.38	86.40 ± 0.38	83.68 ± 0.39
4	97.74 ± 0.11	95.08 ± 0.33	95.14 ± 0.33	90.28 ± 0.32	89.53 ± 0.33	85.53 ± 0.34
5	74.40 ± 0.43	69.90 ± 0.43	70.70 ± 0.43	70.80 ± 0.43	70.50 ± 0.43	69.90 ± 0.43
6	68.62 ± 0.45	72.79 ± 0.42	72.14 ± 0.42	73.18 ± 0.42	74.22 ± 0.42	72.92 ± 0.42
7	80.37 ± 0.38	79.91 ± 0.40	68.22 ± 0.46	69.16 ± 0.45	69.63 ± 0.45	68.69 ± 0.46
8	66.30 ± 0.47	71.85 ± 0.46	76.67 ± 0.41	78.52 ± 0.39	81.11 ± 0.39	78.15 ± 0.41
9	85.76 ± 0.33	85.76 ± 0.33	86.04 ± 0.33	86.89 ± 0.31	87.75 ± 0.31	83.48 ± 0.34
10	99.33 ± 0.05	99.33 ± 0.05	100.00 ± 0.07	99.33 ± 0.10	99.33 ± 0.10	98.00 ± 0.14
11	86.49 ± 0.33	75.68 ± 0.39	83.78 ± 0.37	83.11 ± 0.37	83.11 ± 0.37	81.08 ± 0.38
12	90.22 ± 0.26	87.88 ± 0.31	97.71 ± 0.13	97.71 ± 0.13	97.58 ± 0.15	97.53 ± 0.15
13	71.15 ± 0.49	61.06 ± 0.50	66.83 ± 0.49	67.79 ± 0.47	67.31 ± 0.48	65.38 ± 0.48
14	98.33 ± 0.08	70.56 ± 0.42	84.45 ± 0.38	85.07 ± 0.38	88.41 ± 0.37	83.92 ± 0.39
15	87.47 ± 0.29	80.73 ± 0.36	89.60 ± 0.33	89.95 ± 0.33	90.19 ± 0.33	88.89 ± 0.34
16	85.39 ± 0.33	91.57 ± 0.27	93.26 ± 0.25	93.82 ± 0.25	93.82 ± 0.25	92.13 ± 0.29
17	100.00 ± 0.00	100.00 ± 0.00	100.00 ± 0.00	100.00 ± 0.00	100.00 ± 0.00	100.00 ± 0.00
Average	**82.60 ± 0.30**	**82.58 ± 0.33**	**85.26 ± 0.31**	**85.42 ± 0.31**	**85.74 ± 0.31**	**84.05 ± 0.33**

Table 3. Average size (S), coverage (CVG) and likelihood ratio (LHR) of rules for different versions of the unordered algorithm.

#	**CN2 standard**			**CN2 WRAcc**			**CN2-SD** ($\gamma = 0.5$)			**CN2-SD** ($\gamma = 0.7$)			**CN2-SD** ($\gamma = 0.9$)			**CN2-SD (add. weight.)**		
	S	CVG	LHR	S	CVG	LHR	S	CVG	LHR	S	CVG	LHR	S	CVG	LHR	S	CVG	LHR
1	26	49.3	68.8	26	58.1	61.2	14	115.6	100.9	14	126.3	130.7	13	190.7	136.1	8	150.5	193.0
2	58	36.0	21.5	6	156.8	89.9	10	181.0	136.6	9	239.7	170.5	8	296.0	189.8	6	269.7	211.6
3	113	9.5	11.6	42	24.7	20.2	17	75.0	28.8	18	72.0	31.3	11	125.0	38.0	9	105.0	43.8
4	84	30.9	45.7	22	128.1	112.9	11	253.2	136.1	11	282.0	167.0	11	422.4	167.0	6	282.0	212.3
5	91	15.1	13.2	14	98.7	25.2	13	151.0	37.9	12	185.1	47.9	15	263.0	48.9	7	191.5	55.4
6	58	26.5	13.2	12	90.6	27.7	11	113.7	39.7	14	102.3	37.0	12	132.0	40.0	9	116.1	42.8
7	23	11.9	12.2	15	16.5	11.9	11	39.8	14.6	15	35.5	15.0	17	62.0	16.1	7	35.1	18.1
8	42	14.6	14.2	11	57.3	18.4	16	51.8	29.4	16	69.6	36.4	20	79.7	36.4	11	66.1	42.4
9	42	19.7	19.5	26	23.5	21.5	27	40.6	39.7	25	47.7	44.9	26	63.0	43.6	13	49.6	52.4
10	11	16.3	30.0	11	16.3	30.0	14	21.8	27.4	14	21.8	27.4	14	24.4	27.4	10	21.8	33.8
11	17	14.6	18.2	10	21.3	19.9	16	27.1	24.1	16	29.2	24.1	23	39.3	25.1	10	28.2	30.7
12	184	21.6	94.6	38	103.2	139.4	11	337.1	345.1	8	398.5	390.0	7	440.0	437.1	6	407.0	509.6
13	36	7.8	12.5	22	15.8	13.5	28	19.4	13.7	32	20.8	14.7	41	34.8	14.6	12	24.0	17.9
14	30	38.9	76.4	27	55.5	44.0	20	83.7	62.6	18	94.2	63.4	15	117.6	74.9	11	101.8	68.2
15	82	19.6	32.7	38	34.1	28.3	14	154.6	101.3	14	166.3	107.3	15	218.0	107.3	9	189.7	131.5
16	28	10.0	16.0	18	13.8	20.5	21	20.0	19.8	20	20.9	20.0	21	27.6	20.5	11	21.9	25.5
17	3	50.5	68.2	3	50.5	68.2	3	50.5	68.2	3	50.5	68.2	3	50.5	68.2	3	50.5	68.2
Avg	**54.6**	**23.1**	**33.5**	**20.0**	**56.8**	**44.3**	**15.1**	**102.1**	**72.2**	**15.2**	**115.5**	**82.1**	**16.0**	**152.1**	**87.8**	**8.7**	**124.2**	**103.4**

erage obtained by multiplicative weights. Average rule coverage is optimal for multiplicative weights with high γ, improving on the average coverage of *CN2-standard* rules with a factor of 6 and on *CN2-WRAcc* with a factor of 3. We conclude that both rules obtained with additive weights and with multiplicative weights with high γ are highly overlapping, due to the relatively modest decrease of example weights.

In addition, there is also a big increase in the average likelihood ratio: while the ratios achieved by *CN2-standard* are already significant at the 99% level,

this is further pushed up by *CN2-SD* with maximum values achieved by additive weights. An interesting question, to be verified with further experiments, is whether the weighted versions of the CN2 algorithm improve the significance of the induced subgroups also in the case when CN2 rules are induced without applying the significance test.

In summary, *CN2-SD* produces substantially smaller rule sets, where individual rules have higher coverage and significance. These three factors are important for subgroup discovery: smaller size enables better understanding, higher coverage means larger support, and rules should describe discovered subgroups that are significantly different from the entire population.

The increased accuracy of *CN2-SD* compared to *CN2-standard* and *CN2-WRAcc* (see Table 2) improves on the findings in [15], where the rule size decreased at the expense of a small drop in accuracy. It should be noted that the results of *CN2-standard* and *CN2-WRAcc* cannot be directly compared to those reported in [15] due to the following reasons: first, different datasets were selected in the two experiments, second, attribute discretisation was performed, third, minor differences in the algorithm implementations exist, and finally, results in this paper were obtained for binarised learning problems. Our hypothesis, that needs to be verified in further work, is that the improved results reported in this paper may be due to the binarised problem domains for which *WRAcc* may be better suited than for multi-class domains.

5 Conclusions

We have presented a novel approach to adapting standard classification rule learning to subgroup discovery. To this end we have appropriately adapted the covering algorithm, the search heuristics and the probabilistic classification procedure. Experimental results on 17 UCI datasets are very promising, demonstrating big improvements in number of induced rules, rule coverage and rule significance, as well as smaller improvements in rule accuracy.

In further work we will investigate the behaviour of *CN2-SD* in multi-class problems. We are also planning to evaluate the approach using the area under the ROC convex hull metric which is more appropriate for subgroup discovery than the standard accuracy metric. See the appendix for some ROC results. Finally, we plan to use our adapted procedure for subgroup discovery for solving practical problems, in which expert evaluations of induced subgroup descriptions will be of ultimate interest.

Acknowledgements

Thanks to Dragan Gamberger for inspiring the work on a weighted covering algorithm. The work reported in this paper was supported by the Slovenian Ministry of Education, Science and Sport, the IST-1999-11495 project Data Mining and Decision Support for Business Competitiveness: A European Virtual Enterprise, and the British Council project Partnership in Science PSP-18.

References

1. Agrawal, R., Mannila, H., Srikant, R., Toivonen, H., & Verkamo, A.I. (1996) Fast discovery of association rules. In U.M. Fayyad, G. Piatetski-Shapiro, P. Smyth and R. Uthurusamy (Eds.) *Advances in Knowledge Discovery and Data Mining* (pp. 307–328). AAAI Press.
2. B. Cestnik. Estimating probabilities: A crucial task in machine learning. In L. Aiello, editor, *Proceedings of the 9th European Conference on Artificial Intelligence*, pp. 147–149, Pitman, Stockholm, Sweden, 1990.
3. Clark, P. and Boswell, R. (1989). Rule induction with CN2: Some recent improvements. In Y. Kodratoff, editor, *Proceedings of the 5th European Working Session on Learning*, Springer-Verlag, 151–163.
4. P. Clark and T. Niblett. Induction in noisy domains. In I. Bratko and N. Lavrač, editors, *Progress in Machine Learning (Proceedings of the 2nd European Working Session on Learning)*, pp. 11–30, Sigma, Wilmslow, UK, 1987.
5. Clark, P. and Niblett, T. (1989). The CN2 induction algorithm. *Machine Learning*, Kluwer, 3(4):261–283.
6. Džeroski, S., Cestnik, B. and Petrovski, I. (1993) Using the m-estimate in rule induction. Journal of Computing and Information Technology, 1(1):37 – 46.
7. Ferri-Ramírez, C., Flach, P. and Hernandez-Orallo, J. (2002) Learning Decision Trees Using the Area Under the ROC Curve. *Proceedings of the 19th International Conference on Machine Learning*, Morgan Kaufmann, in press.
8. Fayyad, U.M. and Irani, K.B. (1993). Multi-interval discretisation of continuous-valued attributes for classification learning. In Bajcsy, R. (Ed.) *Proceedings of the 13th International Joint Conference on Artificial Intelligence*, Morgan Kaufmann, 1022–1027.
9. Kukar, M., Kononenko, I., Grošelj, C., Kralj, K., & Fettich, J.J. (1998) Analysing and improving the diagnosis of ischaemic heart disease with machine learning. *Artificial Intelligence in Medicine*, special issue on *Data Mining Techniques and Applications in Medicine*, 16, 25–50. Elsevier.
10. Lavrač, N., Flach, P. and Zupan, B. (1999) Rule Evaluation Measures: A Unifying View. In *Proceedings of the Ninth International Workshop on Inductive Logic Programming, volume 1634 of Lecture Notes in Artificial Intelligence*: 74–185. Springer-Verlag.
11. Michalski, R.S., Mozetič, I., Hong, J., & Lavrač, N. (1986) The multi-purpose incremental learning system AQ15 and its testing application on three medical domains. In *Proc. Fifth National Conference on Artificial Intelligence*, (pp. 1041–1045), Morgan Kaufmann.
12. Murphy, P.M. and Aha, D.W. (1994) *UCI repository of machine learning databases* [http://www.ics.uci.edu/ ˜mlearn/MLRepository.html]. Irvine, CA: University of California, Department of Information and Computer Science.
13. Provost, F. & Fawcett, T. (2001) Robust classification for imprecise environments. *Machine Learning*, 42(3), 203–231.
14. R. L. Rivest. Learning decision lists. *Machine Learning*, 2(3):229–246, Kluwer, 1987.
15. Todorovski, L., Flach, P. and Lavrač, N. (2000). Predictive Performance of Weighted Relative Accuracy. In Zighed, D.A., Komorowski, J. and Zytkow, J., editors, *Proceedings of the 4th European Conference on Principles of Data Mining and Knowledge Discovery (PKDD2000)*, Springer-Verlag, 255–264.

16. Witten, I.H. and Frank, E. (1999). *Data Mining: Practical Machine Learning Tools and Techniques with Java Implementations.* Morgan Kaufmann, San Francisco.
17. Wrobel, S. (1997) An algorithm for multi-relational discovery of subgroups. *Proc. First European Symposium on Principles of Data Mining and Knowledge Discovery*, (pp. 78–87), Springer.

Appendix: Area under ROC convex hull evaluation

A point on the *ROC curve* (ROC: Receiver Operating Characteristic) [9, 13] shows classifier performance in terms of false alarm or *false positive rate* $FPr = \frac{FP}{TN+FP}$ (plotted on the X-axis) that needs to be minimized, and sensitivity [2] or *true positive rate* $TPr = \frac{TP}{TP+FN}$ (plotted on the Y-axis) that needs to be maximized. In the ROC space, an appropriate tradeoff, determined by the expert, can be achieved by applying different algorithms, as well as by different parameter settings of a selected data mining algorithm or by taking into the account different misclassification costs. The ROC space is appropriate for measuring the success of subgroup discovery, since subgroups whose TPr/FPr tradeoff is close to the diagonal can be discarded as insignificant. The area under the ROC curve (AUC) can be used as a quality measure for comparing the success of different learners.

In subgroup discovery there are two ways in which a rule learner can give rise to a ROC curve.

AUC-Method-1. The first method treats each rule as a separate subgroup which is plotted in the ROC space with its true and false positive rates. We then calculate the convex hull of this set of points, selecting the subgroups which perform optimally under a particular range of operating characteristics. The area under this ROC convex hull (AUC) indicates the combined quality of the optimal subgroups.[3]

AUC-Method-2. The second method employs the combined probabilistic classifications of all subgroups, as indicated below. If we always choose the most likely predicted class, this corresponds to setting a fixed threshold 0.5 on the positive probability: if the positive probability is larger than this threshold we predict positive, else negative. A ROC curve can be constructed by varying this threshold from 1 (all predictions negative, corresponding to (0,0) in the ROC space) to 0 (all predictions positive, corresponding to (1,1) in

[2] *Sensitivity* measures the fraction of positive cases that are classified as positive, whereas *specificity* measures the fraction of negative cases classified as negative. If TP denotes true positives, TN true negatives, FP false positives, FN false negatives, Pos all positives, and Neg all negatives, then $Sensitivity = TPr = \frac{TP}{TP+FN} = \frac{TP}{Pos}$, and $Specificity = \frac{TN}{TN+FP} = \frac{TN}{Neg}$, and $FalseAlarm = FPr = 1 - Specificity = \frac{FP}{TN+FP} = \frac{FP}{Neg}$.

[3] In fact, we would have two convex hulls as some subgroups shift the distribution to the positive class and others shift it to the negative class. This method does not take account of overlapping subgroups.

the ROC space). This results in $n + 1$ points in the ROC space, where n is the total number of classified examples. Equivalently, we can order all the examples by decreasing predicted probability of being positive, and tracing the ROC curve by starting in (0,0), stepping up when the example is actually positive and stepping to the right when it is negative, until we reach (1,1).[4] The area under this ROC curve indicates the combined quality of all subgroups (i.e., the quality of the entire rules set). This method can be used with a test set or in cross-validation, but the resulting curve is not necessarily convex. A detailed description of this method can be found in [7].

Table 4. Area under the ROC curve with standard deviation ($AUC \pm sd$) for different variants of the unordered algorithm using 10-fold stratified cross-validation.

#	**CN2 standard** $AUC \pm sd$	**CN2 WRAcc** $AUC \pm sd$	**CN2-SD** ($\gamma = 0.5$) $AUC \pm sd$	**CN2-SD** ($\gamma = 0.7$) $AUC \pm sd$	**CN2-SD** ($\gamma = 0.9$) $AUC \pm sd$	**CN2-SD (add. weight.)** $AUC \pm sd$
1	99.41 ± 0.01	99.72 ± 0.00	99.24 ± 0.01	98.84 ± 0.01	98.51 ± 0.01	98.17 ± 0.01
2	35.10 ± 0.11	87.83 ± 0.05	83.15 ± 0.05	84.12 ± 0.04	84.32 ± 0.05	84.97 ± 0.04
3	86.22 ± 0.03	89.00 ± 0.03	93.89 ± 0.02	93.69 ± 0.02	93.56 ± 0.02	91.82 ± 0.03
4	99.93 ± 0.00	96.55 ± 0.02	94.67 ± 0.02	93.86 ± 0.02	93.00 ± 0.02	86.78 ± 0.02
5	70.10 ± 0.09	72.11 ± 0.06	71.38 ± 0.07	71.31 ± 0.07	72.68 ± 0.07	70.12 ± 0.06
6	69.52 ± 0.08	78.93 ± 0.05	79.89 ± 0.04	79.93 ± 0.05	80.14 ± 0.05	79.43 ± 0.05
7	68.23 ± 0.08	73.85 ± 0.12	70.71 ± 0.16	72.59 ± 0.15	72.91 ± 0.15	72.67 ± 0.14
8	74.75 ± 0.09	74.56 ± 0.07	82.96 ± 0.08	83.83 ± 0.11	86.16 ± 0.11	84.76 ± 0.09
9	93.81 ± 0.03	90.21 ± 0.06	90.66 ± 0.06	91.48 ± 0.06	91.80 ± 0.06	91.36 ± 0.05
10	100.00 ± 0.00	100.00 ± 0.00	100.00 ± 0.00	100.00 ± 0.00	100.00 ± 0.00	100.00 ± 0.00
11	94.34 ± 0.04	89.16 ± 0.08	88.15 ± 0.07	91.14 ± 0.06	90.76 ± 0.06	88.53 ± 0.08
12	99.73 ± 0.01	99.79 ± 0.00	98.99 ± 0.01	98.69 ± 0.02	98.19 ± 0.02	98.05 ± 0.02
13	65.32 ± 0.12	60.61 ± 0.10	69.35 ± 0.13	72.04 ± 0.15	71.19 ± 0.16	65.10 ± 0.16
14	100.00 ± 0.00	81.00 ± 0.08	92.97 ± 0.03	92.37 ± 0.04	91.96 ± 0.04	90.24 ± 0.04
15	97.27 ± 0.02	92.41 ± 0.03	94.38 ± 0.03	94.60 ± 0.02	94.18 ± 0.02	93.43 ± 0.02
16	94.14 ± 0.05	96.30 ± 0.06	95.39 ± 0.05	95.53 ± 0.05	95.53 ± 0.05	92.16 ± 0.09
17	100.00 ± 0.00	100.00 ± 0.00	100.00 ± 0.00	100.00 ± 0.00	100.00 ± 0.00	100.00 ± 0.00
Average	**85.17 ± 0.04**	**87.18 ± 0.05**	**88.58 ± 0.05**	**89.06 ± 0.05**	**89.11 ± 0.05**	**87.51 ± 0.05**

[4] In the case of ties, we make the appropriate number of steps up and to the right at once, drawing a diagonal line segment.