Applied Artificial Intelligence, 20:543–583

ISSN: 0883-9514 print/1087-6545 online
DOI: 10.1080/08839510600779688

Taylor & Francis
Taylor & Francis Group

# APRIORI-SD: ADAPTING ASSOCIATION RULE LEARNING TO SUBGROUP DISCOVERY

**Branko Kavšek** □ *Jožef Stefan Institute, Ljubljana, Slovenia*

**Nada Lavrač** □ *Nova Gorica Polytechnic, Nova Gorica, Slovenia*

□ *This paper presents a subgroup discovery algorithm APRIORI-SD, developed by adapting association rule learning to subgroup discovery. The paper contributes to subgroup discovery, to a better understanding of the weighted covering algorithm, and the properties of the weighted relative accuracy heuristic by analyzing their performance in the ROC space. An experimental comparison with rule learners CN2, RIPPER, and APRIORI-C on UCI data sets demonstrates that APRIORI-SD produces substantially smaller rulesets, where individual rules have higher coverage and significance. APRIORI-SD is also compared to subgroup discovery algorithms CN2-SD and SubgroupMiner. The comparisons performed on U.K. traffic accident data show that APRIORI-SD is a competitive subgroup discovery algorithm.*

Standard rule learning algorithms are designed to construct classification and prediction rules (Michalski et al. 1986; Clark amd Niblett 1989; Cohen 1995). In addition to this area of machine learning, referred to as *supervised learning* or *predictive induction*, developments in *descriptive induction* have recently gained much attention, in particular *association rule learning* (Agrawal et al. 1993), *subgroup discovery* (Wrobel 1997; 2001), and other approaches to non-classificatory induction.

This paper considers the task of *subgroup discovery* defined as follows (Wrobel 1997; 2001) given a population of individuals and a specific property of the individuals that we are interested in, find population subgroups that are statistically "most interesting," e.g., are as large as possible and have the most unusual statistical (distributional) characteristics with respect to the property of interest.

The work reported in this paper was supported by the Slovenian Ministry of Education, Science and Sport. We acknowledge also the support of the cInQ (Consortium on Discovering Knowledge with Inductive Queries) project, funded by the European Commission under contract IST-2000-26469.

Address correspondence to Branko Kavšek, Jožef Stefan Institute, Jamova 39, Ljubljana 1000, Slovenia. E-mail: branko.kavsek@ijs.si

While the goal of standard classification/prediction rule learning is to generate models, one for each class, inducing class characteristics in terms of properties occuring in the descriptions of training examples, in contrast, subgroup discovery aims at discovering individual "patterns" of interest. In this sense, subgroup discovery is a form of descriptive induction. However, as in subgroup discovery, we restrict the form of patterns to individual rules of the form $X \rightarrow Y$, and we limit the scope of investigation to patterns with a certain property of interest, which is the goal of investigation (the target class, $Y$) that appears in the rule consequent. In the selected rule formalism, the antecedent ($X$) is a conjunction of features (attribute-value pairs) selected from the features describing the training examples. Consequently, subgroup discovery is also a form of supervised predictive induction, as individual rules are induced from labeled training examples (labeled positive if the property of interest holds, and negative otherwise), and the process of subgroup discovery is targeted to uncovering properties of a selected target population of individuals with a given property of interest $Y$. In summary, subgroup discovery is a task at the intersection of predictive and descriptive induction.

Some of the questions on how to adapt standard classification rule learning to subgroup discovery have already been addressed in the development of the CN2-SD subgroup discovery algorithm (Lavrač et al. 2004), in which the CN2 classification rule learner (Clark and Niblett 1989; Clark and Boswell 1991) was adapted to subgroup discovery. In this paper we adapt association rule learning to subgroup discovery, following some of the guidelines from (Lavrač et al. 2004). This was achieved by first developing the APRIORI-C classification rule learner (Jovanoski and Lavrač 2001), which was further enhanced by a novel post-processing mechanism using example weighting incorporated into the covering algorithm and into the modified weighted relative accuracy measure of rule quality, a probabilistic classification scheme, and the use of the ROC space (Provost and Fawcett 2001) for the evaluation of discovered rules in terms of the area under the ROC curve. The latter evaluation criterion is used also for ruleset evaluation, in addition to the standard evaluation criteria: ruleset size, coverage, and accuracy.

This paper presents the APRIORI-SD subgroup discovery algorithm, the analysis of its ingredients in the ROC space (Provost and Fewcett 2001), its experimental evaluation on selected data sets of the UCI Repository of Machine Learning Databases (Murphy and Aha 1994), as well as its application to the U.K. Traffic challenge data set. Experimental comparisons with rule learners CN2 (Clark and Niblett 1989; Clark and Boswell 1991), RIPPER (Cohen 1995), and APRIORI-C (Jovanoski and Lavrač 2001) demonstrate that subgroup discovery algorithm APRIORI-SD produces substantially smaller rulesets, where individual rules have higher coverage,

significance, and unusualness. These factors are important for subgroup discovery: smaller size enables better understanding, higher coverage means larger subgroups, higher significance and unusualness mean that rules describe subgroups whose class distribution is significantly different from the entire population. This is achieved by virtually no loss in terms of the area under the ROC curve and accuracy. Moreover, the application of APRIORI-SD to the U.K. traffic accident data and the comparison with two other state-of-the-art subgroup discovery algorithms, CN2-SD (Lavrač et al. 2002) and SubgroupMiner (Klösgen and May 2002), show that APRIORI-SD is a competitive subgroup discovery algorithm. The comparisons show that, in terms of performance measured in the ROC space, APRIORI-SD outperforms CN2-SD slightly. APRIORI-SD and CN2-SD outperform SubgroupMiner in finding descriptions (subgroups) for minority classes, while SubgroupMiner is better in finding subgroups describing the majority class.

## RELATED WORK IN SUBGROUP DISCOVERY

Some well-known systems in the field of subgroup discovery are EXPLORA (Klösgen 1996; 1999), MIDOS (Wrobel 1997; 2001), and SubgroupMiner (Klösgen and May 2002). EXPLORA treats the learning task as a single relation problem, i.e., all the data are assumed to be available in one table (relation), while MIDOS and SubgroupMiner perform subgroup discovery from multiple relational tables. The most important features of these systems, related to this paper, concern the definition of the learning task and the use of heuristics for subgroup discovery. Recent approaches to subgroup discovery, SD (Gamberger and Lavrač 2002), CN2-SD (Lavrač et al. 2004) and RSD [35], aim to overcome the problem of inappropriate bias of the standard covering algorithm. They use a weighted covering algorithm and modify search heuristics by example weights. SD and CN2-SD are propositional, while RSD is a relational subgroup discovery algorithm.

The related work described in this section focuses on the CN2-SD and SubgroupMiner subgroup discovery algorithms, the state-of-the-art subgroup discovery algorithms used in the experimental comparisons with APRIORI-SD.

### SubgroupMiner

SubgroupMiner (Klösgen and May 2002) is an advanced subgroup mining system enabling the exploration of very large databases by efficient database integration, multirelational hypotheses, visualization-based interaction options, and the discovery of causal subgroup structures. It is an

extension of older subgroup discovery systems EXPLORA (Klösgen 1996) and MIDOS (Wrobel 1997).

SubgroupMiner discovers subgroups in the form of rules $X \rightarrow Y$, where $X$ is a conjunction of features (attribute-value pairs) and $Y$ is the target class. SubgroupMiner uses (interactive) beam search in the space of possible solutions. It uses a quality function to rank the rules during the beam search. In addition, SubgroupMiner uses a special post-processing approach to eliminate redundant subgroups.

Statistical significance of a subgroup is evaluated by a quality function, which has to satisfy some basic monotonicity axioms, hold symmetry and equivalence properties, and can be arranged in families to adjust to user preferences (Klösgen 2002; 1999). Possible quality functions depend on the type of the subgroup pattern. As a standard quality function $Q(R)$ used to evaluate rule $R$ of the form $X \rightarrow Y$, SubgroupMiner uses the classical binomial test to verify if the target share in a subgroup is significantly different than in the entire population.

SubgroupMiner uses the same approach as EXPLORA (Klösgen 1996) to eliminate redundant subgroups. This approach is called subgroup suppression. The algorithm suppresses subgroups that are worse than, but not too different from another subgroup. A subgroup that is dissimilar to the other ones is retained, while better ones may be discarded because they are very similar to others that are a little bit better. A subgroup is evaluated as redundant relative to a subgroup with a higher significance when a constraint balancing overlap degree and significance difference is satisfied (Gebhardt 1991). Let $R_i$ be two rules of the form $X_i \rightarrow Y$. Suppression is defined as follows: $R_1$ *suppresses* $R_2$ if:

$$Q(R_2) < \mathit{Affinity}(R_2, R_1) \cdot Q(R_1), \quad \text{where } \mathit{Affinity}(R_2, R_1) = \left(\frac{n(X_1 \cdot X_2)}{n(X_2)}\right)^{\alpha}$$

In this definition, $n(X_i)$ stands for the number of examples covered by rule $X_i \rightarrow Y$, and $n(X_1 \cdot X_2)$ for the number of examples covered by both rules. Parameter $\alpha$ (with default value 1) can be used to control the number of suppressions. The user can increase (or decrease) $\alpha$ to get fewer (or more) resulting subgroups.

### CN2-SD

Algorithm CN2-SD (Lavrač et al. 2004) adapts classical classification rule learning algorithm CN2 (Clark and Niblett 1989; Clark and Boswell 1991) to subgroup discovery. CN2 uses the *covering algorithm* for ruleset construction. In the covering algorithm only the first few induced rules may be

of interest as subgroup descriptors with sufficient coverage, while 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 the *weighted covering algorithm* used in CN2-SD, positive examples covered by the induced rule are not deleted from the current training set. Instead, their weights are modified so that the probability that an example with a modified weight will be covered by subsequent rules is decreased. Example weights are also taken into account in the *weighted relative accuracy* heuristic used in as a search heuristic in CN2-SD rule construction.

## CLASSIFIATION RULE LEARNING ALGORITHM APRIORI-C

This section presents the APRIORI-C algorithm from which the APRIORI-SD subgroup discovery algorithm was developed.

### Association Rule Learning

Mining of association rules has received a lot of attention in recent years. Compared to other machine learning techniques, its main advantage is a low number of database passes done when searching the hypothesis space, whereas its main disadvantage is the time complexity of association rule learning. One of the best-known association rule learning algorithms is APRIORI (Agrawal et al. 1993; Agrawal and Srikant 1994). This algorithm was extensively studied, adapted to other areas of machine learning and data mining, and successfully applied in many problem domains (Bayardo et al. 1999; Megiddo and Srikant 1998; Ali et al. 1997; Mannila and Toivonen 1996; Agrawal et al. 1998).

An association rule has the form $X \rightarrow Y$, where $X$ and $Y$ are itemsets, which are subsets of $I$, the set of all items in the domain of investigation, consisting of a set of transactions. In the standard machine learning terminology, transactions correspond to training examples (records in a database), an item is a binary feature, and itemsets are conjunctions of features. In association rule learning, a binary feature $A_i = v_{ij}$ is generated for each value $v_{ij}$ of a discrete attribute $A_i$. For numeric attributes, items are formed by attribute discretization. In classification rules, the right-hand side ($Y$) of a rule is single feature denoting the target class, and the left-hand side ($X$) is a conjunction of features. Throughout this paper, items and features will be used as synonyms.

The quality of an association rule is defined by its *confidence* and *support*. *Confidence* of a rule is the conditional probability of $Y$ given $X$: $p(Y|X)$.[1] *Support* of a rule is an estimate of the probability of itemset $X \cup Y$: $p(X \cdot Y)$. Confidence and support are computed by relative frequency estimates of probability as follows:

$$\begin{aligned} Conf(X \rightarrow Y) &= p(Y|X) = \frac{p(X \cdot Y)}{p(X)} \approx \frac{n(X \cdot Y)}{n(X)} \\ Sup(X \rightarrow Y) &= p(X \cdot Y) \approx \frac{n(X \cdot Y)}{N}, \end{aligned} \tag{1}$$

where $n(X)$ is the number of transactions that include itemset (feature) $X$, $n(X \cdot Y)$ the number of transactions that include itemset $X \cup Y$ (conjunction of features $X \wedge Y$), and $N$ is the number of all the transactions (all the records in the data set).

## APRIORI-C

This section presents the APRIORI-C algorithm (Jovanoski and Lavrač 2001), which adapts the APRIORI algorithm to classification purposes. The idea of using association rules for classification has been previously addressed in Liu et al. (1998). The main advantage of APRIORI-C over its predecessors is lower memory consumption, decreased time complexity, and improved understandability of results. The parts of APRIORI-C that are essential for the reader to understand the derived APRIORI-SD algorithm are outlined next.

In APRIORI-C, the association rule learning algorithm APRIORI was adapted to classification purposes by implementing the following steps:

1. Discretize continuous attributes.
2. Binarize all (discrete) attributes.
3. Perform data pre-processing through feature subset selection.
4. Run the optimized APRIORI algorithm by taking in consideration only rules whose right-hand sides consist of a single item, representing the target class value.
5. Post-process the set of induced rules by rule ordering and best rule subset selection.
6. Use these rules to classify unclassified examples.

These steps of the APRIORI-C algorithm, as well as the approaches to feature subset selection, are described in detail in Jovanoski and Lavrač (2001). Here we describe the last three steps, the APRIORI-C optimizations

(step 4), rule post-processing (step 5), and the classification of examples (step 6). Steps 5 and 6 are the main steps we changed to obtain APRIORI-SD.

### Optimizations of the APRIORI-C Algorithm

To better adapt to classification purposes, APRIORI-C includes the following optimizations:

- **Classification rule generation**. Rules with a single target item at the right-hand side can be created during the search. To do so, the algorithm needs to save only the supported itemsets of sizes $k$ and $k+1$. This results in decreased memory consumption (improved by factor 10). Notice, however, that this does not improve the algorithm's time complexity.
- **Prune irrelevant rules**. Classification rule generation can be suppressed if one of the existing generalizations of the rule has support and confidence above the given *minSup* and *minConf* thresholds. To prevent rule generation, the algorithm simply excludes the corresponding itemset from the set of supported itemsets of size $k+1$. Time and space complexity reduction are considerable (improved by factor 10 or more).
- **Prune irrelevant items**. If an item cannot be found in any of the itemsets containing the target item, then it is impossible to create a rule containing this item. Hence, APRIORI-C prunes the search by discarding all itemsets containing this item.

### Post-processing by Rule Subset Selection

By setting low values of parameters *minSup* and *minConf*, the algorithm often induces a large number of rules, which may hinder the understandability of the induced ruleset. Moreover, problems of rule redundancy, incapability of classifying examples and poor accuracy in domains with unbalanced class distribution may also occur. A way to avoid these problems is to select from the set of induced rules a subset of best rules, and add a default rule to the resulting ruleset. APRIORI-C selects the best rules as follows:

- **Use B best rules**. The algorithm first selects the best rule (the rule having the highest support), eliminates all the examples covered by this rule, sorts the remaining rules according to support, and repeats the procedure until $B$ best rules are selected or there are no more rules to select, or there are no uncovered training examples left. The algorithm then stops and returns the classifier in the form of an *if-then-else* rule list.

- **Use B best rules for each class**. The algorithm behaves in a similar way as in the "use *B* best rules" case, but selects *B* best rules for each class (if that many rules exist for each class). By this approach, rules induced for the minority class(es) will also be included into the classifier.

When tested with several values of parameter *B*: 1, 2, 5, 10, 15, and 20, it was shown in Jovanoski and Lavrač (2001) that "use *B* best rules" and "use *B* best rules for each class" do not differ significantly in terms of accuracy, except when there are significant differences in class distributions; in this case, "use *B* best rules for each class" is superior. Next, both algorithms increase their accuracy significantly when using a *default* rule, assigning the majority class to the examples that have not been covered by the best *B* rules; this increase gets smaller with increased value of parameter *B*, but still remains noticeable. Finally, in terms of accuracy and understandability, when the value of parameter *B* reaches 10, ruleset accuracy is comparable to the accuracy of the original ruleset.

Another rule post-processing procedure, "use example weighting to select *B* best rules," was implemented in APRIORI-C, similar to "use *B* best rules." The difference is that covered examples are not eliminated, but instead their weights are decreased; covered examples are eliminated when their weights fall below a given threshold. Due to some implementational deficiencies, this procedure did not perform well in the experiments of APRIORI-C. Improvements and details of the improved weighting scheme are given later when describing APRIORI-SD.

### Classification Schemes

To classify an example with all the rules found by the algorithm, APRIORI-C first sorts the rules according to the support criterion, finds in the list of rules the first rule that covers the example, and classifies the example into the class of the right-hand side of this rule. If no rule covers the example, the example is marked as unclassified.

This initial scheme has been improved by adding a default rule to the set of induced rules, which assigns the majority class to the examples that have not been covered by the induced ruleset.

### APRIORI-SD

The main modifications of the APRIORI-C algorithm, making it appropriate for subgroup discovery, involve the implementation of an example weighting scheme in rule post-processing, a modified rule quality function incorporating example weights into the weighted relative accuracy heuristic,

**TABLE 1** The Pseudo-Code of APRIORI-SD

algorithm ***APRIORI − SD*** (*Examples*, *Classes*, *minSup*, *minConf*, *k*)
*Ruleset* = ***APRIORI − C***(*Examples*, *Classes*, *minSup*, *minConf*) set all example weights of *Examples* to 1
*Majority* = the majority class in *Examples*
*Resultset* = {}
**repeat**
  *BestRule* = rule with the highest weighted relative accuracy value in *Ruleset* (computed using Equation 4)
  *Resultset* = *Resultset* ∪ *BestRule*
  *Ruleset* = *Ruleset* \ *BestRule* decrease the weights of examples covered by *BestRule* (using the example weighting scheme) remove from *Examples* the examples covered more than *k*-times
**until** *Examples* = {} **or** *Ruleset* = {}
**return** *Resultset* = *Resultset* ∪ "*true* → *Majority*"

a probabilistic classification scheme, and the use of the ROC space for improving the evaluation of discovered rules.

Table 1 presents the pseudo-code of the APRIORI-SD algorithm. The input arguments of the algorithm are: *Examples*, *Classes*, *minSup*, *minConf* and *k* *Examples* are the set of training examples, *Classes* are the values of the class attribute, parameter *k* determines the threshold for covered example elimination in rule post-processing ensuring the convergence of the algorithm, and parameters *minSup* and *minConf* denote the APRIORI minimal support and confidence parameters, constraining rule search (Agrawal et al. 1993; Agrawal and Srikant 1994; Jovanoski and Lavrač 2001). The default values of the parameters in APRIORI-SD are *minSup* = 0.03, *minConf* = 0.8 and $k = 5$.

APRIORI-SD generates the initial set of rules by means of function *APRIORI-C*. This function uses the APRIORI-C (Jovanoski and Lavrač 2001) algorithm—without feature subset selection in data pre-processing and without rule post-processing—to find all rules with the class attribute at the right-hand side, satisfying the *minSup* and *minConf* constraints. This ruleset is ordered according to the *weighted relative accuracy* quality function from best to worst. The best rule is selected, covered examples are re-weighted, and the procedure repeats these steps until one of the stopping criteria is satisfied: Either all examples have been covered more than *k* times, or there are no more rules in the ruleset.

### Probabilistic Classification Scheme

In classification rule learning, induced rulesets are treated as "ordered" or "unordered." Ordered rules are interpreted as an *if-then-else* decision list (Rivest 1987) in a straightforward manner: When classifying a new example, the rules are sequentially tried and the first rule that covers the example is used for prediction. APRIORI-C uses this interpretation of rules for example classification.

In the case of unordered rulesets, the class distribution of covered training examples is attached to each rule. Rules of the form:

$$X \rightarrow Y \ [\mathit{ClassDistribution}]$$

are induced, where numbers $n(X \cdot Y_j)$ in the *ClassDistribution* list denote, for each class $Y_j$, the number of training examples of class $Y_j$ covered by the rule. When the ruleset is used as a model for classifying a new example, all rules are tried and those covering the example are collected. If a clash occurs (several rules with different class $Y_j$ 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 frequent class. If no rule fires, a default rule is invoked which predicts the majority class of uncovered training examples.

This voting mechanism is illustrated by an example, taken from a description of the voting mechanism of the CN2 rule learner. Suppose that the task is to classify an animal that is two-legged, feathered, large, non-flying, and has a beak, and the classification is based on a ruleset composed of three rules with the $[\texttt{bird}, \texttt{elephant}]$ class distribution assigned to each rule. Take the following ruleset (for simplicity, the ruleset does not include the default rule):

$$\texttt{legs} = 2 \wedge \texttt{feathers} = \texttt{yes} \rightarrow \texttt{class} = \texttt{bird} \ [13, 0]$$
$$\texttt{beak} = \texttt{yes} \rightarrow \texttt{class} = \texttt{bird} \ [20, 0]$$
$$\texttt{size} = \texttt{large} \wedge \texttt{flies} = \texttt{no} \rightarrow \texttt{class} = \texttt{elephant} \ [2, 10]$$

All rules fire for the animal to be classified, resulting in a $[35, 10]$ class distribution. As a result, the animal is classified into majority class `bird`.

APRIORI-SD uses a different probabilistic classification scheme than the described CN2 voting scheme. To illustrate the APRIORI-SD probabilistic classification scheme, take again the same animal (which is two-legged, feathered, large, non-flying, and has a beak) and its classification based on three probabilistic rules, with a probability distribution assigned to each rule:

$$\texttt{legs} = 2 \wedge \texttt{feathers} = \texttt{yes} \rightarrow \texttt{class} = \texttt{bird} \ [1, 0]$$
$$\texttt{beak} = \texttt{yes} \rightarrow \texttt{class} = \texttt{bird} \ [1, 0]$$
$$\texttt{size} = \texttt{large} \wedge \texttt{flies} = \texttt{no} \rightarrow \texttt{class} = \texttt{elephant} \ [0.17, 0.83]$$

In APRIORI-SD, the animal is classified as a bird by averaging the probabilities, resulting in the final probability distribution $p(\texttt{class} = \texttt{bird}) \approx \frac{1+1+0.17}{3} = 0.72$ and $p(\texttt{class} = \texttt{elefant}) \approx \frac{0+0+0.83}{3=0.28}$. In this probabilistic

classification scheme, subgroups covering a small number of examples are less heavily penalized than in the CN2 voting scheme.

### Example Weighting Scheme

The APRIORI-SD weighting scheme treats the examples in a way that covered positive examples are not deleted when the currently "best" rule is selected in the post-processing step of the algorithm. Instead, each time a rule is selected, the algorithm stores with each example a count $i$ of how many times (with how many rules) the example has been covered so far.

Weights of positive examples covered by the selected rule decrease according to the formula $w(e_j, i) = \frac{1}{i+1}$. In the first iteration all target class examples are assigned the same weight $w(e_j, 0) = 1$, while in the following iterations the contributions of examples are inverse proportional to their coverage by previously selected rules. In this way the examples already covered by one or more selected rules decrease their weights while rules covering many yet uncovered target class examples whose weights have not been decreased will have a greater chance to be covered in the following iterations. Covered examples are completely eliminated when their weights fall below a given threshold (e.g., when an example has been covered more than $k$ times).

### Weighted Relative Accuracy

Weighted relative accuracy (*WRAcc*) is used in subgroup discovery to evaluate the quality of induced rules. We use *WRAcc* instead of support when selecting the "best" rules in post-processing.

We use the following notation. Let $n(X)$ be the number of examples covered by rule $X \rightarrow Y$, $n(Y)$ the number of examples of class $Y$, and $n(X \cdot Y)$ the number of correctly classified examples (true positives). We use $p(X)$, $p(X \cdot Y)$, etc., for the corresponding probabilities. Rule accuracy, or rule confidence in the terminology of association rule learning, is defined as $Acc(X \rightarrow Y) = Conf(X \rightarrow Y) = p(Y|X) = \frac{p(X \cdot Y)}{p(X)}$. Weighted relative accuracy (Lavrač et al. 1999; Todorovski et al. 2000) is defined as follows.

$$WRAcc(X \rightarrow Y) = p(X) \cdot (p(Y|X) - p(Y)) \tag{2}$$

Weighted relative accuracy consists of two components: generality $p(X)$, and relative accuracy $p(Y|X) - p(Y)$. The second term, relative accuracy, is the accuracy gain of rule $X \rightarrow Y$ relative to the fixed rule *true* $\rightarrow Y$, which predicts all instances to be of class $Y$; rule $X \rightarrow Y$ 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 $X$ with a given rule head $Y$. As it is easy to obtain high relative accuracy with highly specific rules, i.e., rules with low generality $p(X)$, generality $p(X)$ is used as a "weight," so that weighted relative accuracy trades off generality of the rule ($p(X)$, i.e., rule coverage) and relative accuracy ($p(Y|X) - p(Y)$). All the probabilities in Equation 2 are estimated by relative frequencies, e.g., $p(X) \approx \frac{n(X)}{N}$, where N is the number of all instances.

***WRAcc with Example Weights***

The rule quality measure *WRAcc* used in APRIORI-SD has been further modified to enable handling example weights, which provide the means to consider different parts of the example space when selecting the best rules.

The modified *WRAcc* measure is defined as follows:

$$wWRAcc(X \rightarrow Y) \approx \frac{n'(X)}{N'} \cdot \left( \frac{n'(X \cdot Y)}{n'(X)} - \frac{n'(Y)}{N'} \right), \tag{3}$$

where $N'$ is the sum of the weights of all examples, $n'(X)$ is the sum of the weights of all covered examples, and $n'(X \cdot Y)$ is the sum of the weights of all correctly covered examples.

***Improved WRAcc with Example Weights***

The third term in the definition of *wWRAcc* in Equation 3, $\frac{n'(Y)}{N'}$, represents the portion of weighted positive examples in the population of weighted examples: When example weights change, the value of this term changes too.

If this term is replaced by the corresponding term from the original definition of *WRAcc* in Equation 2, $\frac{n(Y)}{N}$, an improved *wWRAcc'* definition is obtained:

$$wWRAcc'(X \rightarrow Y) = \frac{n'(X)}{N'} \left( \frac{n'(X \cdot Y)}{n'(X)} - \frac{n(Y)}{N} \right) \tag{4}$$

By this term replacement *wWRAcc'* is forced to reflect the improvement of the rule's (weighted) accuracy with respect to the accuracy of the default rule ($true \rightarrow Y$) in the original population.

The analysis of different *WRAcc* variants, in terms of their behavior in the ROC space, outlined next, indicates that *wWRAcc'* of Equation 4 is preferred, compared to *WRAcc* and *wWRAcc*. Consequently, *wWRAcc'* is the default heuristic used in best rule subset selection post-processing procedure of APRIORI-SD.

**TABLE 2** Confusion Matrix

| | Predicted positive | Predicted negative |
|---|---|---|
| actual positive (*Pos*) | *TP* | *FN* |
| actual negative (*Neg*) | *FP* | *TN* |

## ROC Analysis for Subgroup Discovery

This section introduces the confusion matrix and the ROC (receiver operating characteristics) space (Provost and Fewcett 2001) used for rule and ruleset evaluation.

Take a rule/ruleset acting as a classifier of examples. The confusion matrix shown in Table 2 defines the notions of *TP* (number of true positives), *FP* (number of false positives), *TN* (number of true negatives), and *FN* (number of false negatives), where "actual positive" (negative) are the examples in the training set that are (actually) positive (negative), and "predicted positive" (negative) are the examples that rule $X \rightarrow Y$ predicts as positive (negative).

The ROC space (Provost and Fawcett 2001) is a two-dimensional space that shows classifier (rule/ruleset) performance in terms of its *false positive rate* (also called "false alarm"), $FPr = \frac{FP}{TN+FP} = \frac{FP}{Neg}$ plotted on the $X$-axis, and *true positive rate* (also called "sensitivity") $TPr = \frac{TP}{TP+FN} = \frac{TP}{Pos}$ plotted on the $Y$-axis. Applying the same notation as used to define *confidence* and *support* in Equations 1, $FPr$ and $TPr$ can be expressed as: $FPr = \frac{n(X \cdot \overline{Y})}{Neg}$, $\frac{TPr = n(X \cdot Y)}{Pos}$. Take, for instance, subgroups discovered by subgroup discovery algorithms, shown in Figure 1, where each subgroup is represented by a $(FPr, TPr)$ point in the ROC space.

The ROC space is appropriate for measuring the success of subgroup discovery, since subgroups whose $TPr/FPr$ tradeoff is close to the main diagonal (line connecting the points $(0, 0)$ and $(1, 1)$ in the ROC space) can be discarded as insignificant. The reason is that the rules with $TPr/FPr$ on the main diagonal have the same distribution of covered positives and negatives ($TPr = FPr$) as the distribution in the entire data set.

## ROC ANALYSIS OF *WRAcc* VARIANTS

This section shows the effects of different example weighting schemes on the *WRAcc* quality function, analyzed in the ROC (receiver operating characteristics) space.

### Analysis of *WRAcc*

Following the guidelines from Flach (2003)and Fürnkranz and Flach (2003), we use the isometrics in the ROC space to represent the *WRAcc*

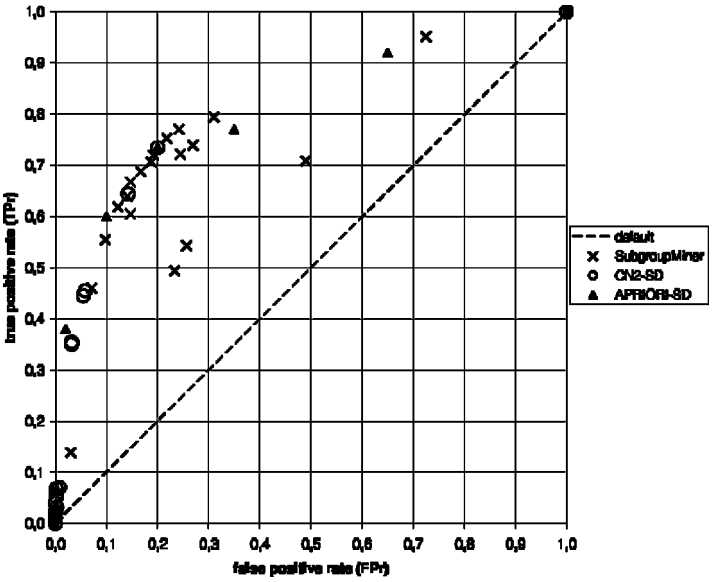


**FIGURE 1** Subgroups, induced by three subgroup discovery algorithms, APRIORI-SD, CN2-SD, and SubgroupMiner, shown as (*FPr*, *TPr*) points in the ROC space.

quality function defined by Equation 2. An ROC isometric is a line in the ROC space connecting points with equal value of the selected quality function. In the case of the *WRAcc* function, the ROC isometrics are parallel to the main diagonal in the ROC space.

The definition of *WRAcc* (Equation 2) can be rewritten in terms of *TPr* and *FPr* as $WRAcc\ (X \rightarrow Y) = p(Y) \cdot (1 - p(Y)) \cdot (TPr - FPr)$ (Lavrač et al. 2004), hence an iso-*WRAcc*-line is defined by

$$TPr = \frac{WRAcc(X \rightarrow Y)}{p(Y) \cdot (1 - p(Y))} + FPr.$$

For a fixed class distribution, $p(Y) \cdot (1 - p(Y))$ is constant. For fixed values of *WRAcc*, *WRAcc* iso-lines have the form $TPr = FPr + a$, which indicates that $WRAcc(X \rightarrow Y)$ is proportional to the vertical distance $a$ of rule $X \rightarrow Y$ to the ROC diagonal.

In Figure 2 the main diagonal (line connecting the points $(0, 0)$ and $(1, 1)$ in the ROC space) is denoted with a thicker line. Points lying on this diagonal, with $TPr = FPr$, represent subgroups with the same distribution of positive and negative examples as in the entire data set. Points on the ROC diagonal have a *WRAcc* value equal to 0, points above the diagonal have a positive value of *WRAcc*, and points below the diagnonal have a negative value. The further away a point is from the main diagonal towards the

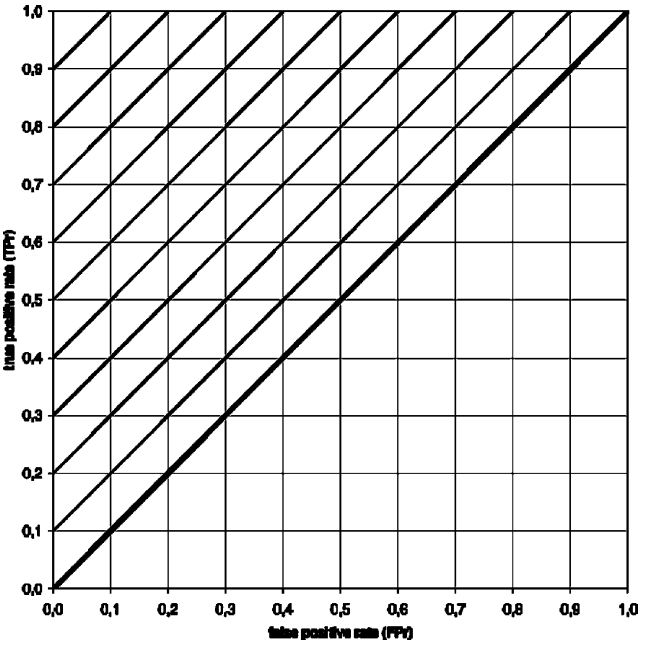


**FIGURE 2** ROC isometrics for the *WRAcc* quality function.

point $(1, 1)$ in the ROC space, the larger is the value of the *WRAcc* function in that point (we do not take into account points with negative *WRAcc* as these points would represent subgroups with the proportion of positives that is smaller than the preoportion of positives in the entire data set).

Using *WRAcc* as its rule quality function, APRIORI-SD tries to select subgroups that are as far as possible from the main diagonal in the ROC space and at the same time as close as possible to the point $(1, 1)$. Point $(1, 1)$ in the ROC space is sometimes referred to as the "ROC heaven" because it represents a subgroup covering all the positives and none of the negatives. It is also the point in which *WRAcc* reaches its maximum value.

### Analysis of *wWRAcc*

By adding example weights to the *WRAcc* function, we obtain the modified *wWRAcc* function defined by Equation 3. All three terms of *wWRAcc*: $\frac{n'(X)}{N'}$, $\frac{n'(X \cdot Y)}{n'(X)}$, and $\frac{n'(Y)}{N'}$ include example weights both in the numerator and denominator of the fraction. In this way, when example weights are decreased, both the values of the numerator and denominator decrease, keeping the value of *wWRAcc* "balanced."

We can illustrate the effect of example weighting by analyzing the *wWRAcc* function after the selection of the first best rule. Equation 5 shows

the right-hand side of Equation 3: The *wWRAcc* value after the weights of examples covered by the first rule have been decreased from 1 to $\frac{1}{2}$ (assuming an unrealistic scenario that no negative examples have been covered by the best rule[2]):

$$\frac{\frac{n(X)}{2}}{N-\frac{n(X)}{2}} \cdot \left( \frac{\frac{n(X \cdot Y)}{2}}{\frac{n(X)}{2}} - \frac{\frac{n(Y)-n(X \cdot Y)}{2}}{N-\frac{n(X)}{2}} \right). \tag{5}$$

Equation 5 shows that the number of examples covered by the rule ($n(X)$) affects the angle of iso-*WRAcc* lines: The more examples covered, the lower the angle, and vice versa. Due to factor $\frac{n'(Y)}{N'}$, which keeps *wWRAcc* balanced when example weights decrease, the ROC isometrics for *wWRAcc* look very much like those for *WRAcc* (see Figure 2).

### Analysis of *wWRAcc′*

In order to push *wWRAcc* out of balance and change its ROC isometrics independently of rule coverage, consider the third term in the definition of *wWRAcc* (Equation 3) $\frac{n'(Y)}{N'}$. When example weights change, the value of this term changes too, keeping the equation balanced. Replacing this term by $\frac{n(Y)}{N}$, which occurs in the original *WRAcc* definition, the new *wWRAcc′* definition in Equation 4 is obtained, reflecting the accuracy improvement of the subgroup with respect to the default rule ($true \rightarrow Y$) on the original population. The *wWRAcc′* measure is unbalanced with respect to example weights, meaning that its ROC isometrics change when the example weights change (independently of rule coverage), as shown in Figure 3.

Figure 3 shows how ROC isometric lines change from solid to dashed to dotted when the weights of (positive[3]) examples decrease.

Thick lines in the figure denote ROC isometrics for value 0 of the *wWRAcc′* function. Solid lines show the behavior of *wWRAcc′* with weights of all examples equal 1. Dashed lines show *wWRAcc′* for the extreme case where all positive examples have weight $w(e_j, 1) = \frac{1}{2}$. Dotted lines represent the same quality function in the case of all positive examples having weight $w(e_j, 2) = \frac{1}{3}$. For the sake of clarity of the figure, only the iso-lines for positive *wWRAcc′* values, and only three iso-lines with example weights 1, $\frac{1}{2}$ and $\frac{1}{3}$ are shown.

### Illustration of Example Weighting

We illustrate the effect of weighting by explaining step-by-step the discovery of subgroups by APRIORI-SD on the example of predicting *Class 0* in the problem of U.K. traffic accident data analysis, described in detail

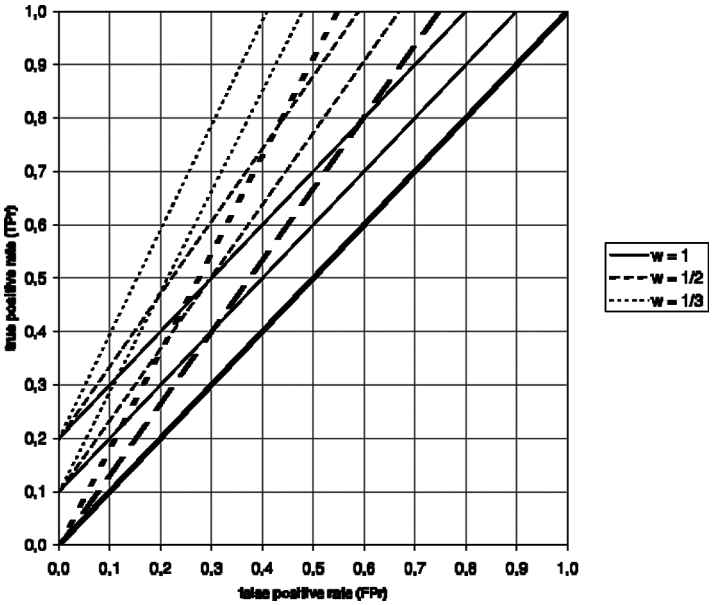


**FIGURE 3** ROC isometrics showing the effects of example weighting on the $wWRAcc'$ quality function used in APRIORI-SD.

later. We explain the selection of the first three subgroups. The procedure used in APRIORI-SD, illustrated in Figure 4 (which equals Figure 1 with subgroups induced by CN2-SD and SubgroupMiner removed, keeping just APRIORI-SD subgroups), goes as follows:

- Initially all the examples have weight $w(e_j, 0) = 1$. APRIORI-SD selects the "best" subgroup, i.e., the subgroup with the maximal value of $wWRAcc'$. In Figure 4 this subgroup is represented by the triangle in point (0.65,0.92). The $wWRAcc'$ value of the subgroup is 0.062 (the maximum value of $wWRAcc'$ being 0.23). The solid line going through the subgroup is the iso-line for $wWRAcc' = 0.062$. The thick solid line represents the iso-line for $wWRAcc' = 0$. The weights of all positive examples covered by the subgroup are now decreased to $w(e_j, 1) = \frac{1}{2}$.
- The value of $wWRAcc'$ is recomputed, taking into account the new weights. Again the "best" subgroup is selected. In the figure this newly selected subgroup is shown as a triangle in point (0.35,0.77). The $wWRAcc'$ value of the subgroup is 0.02 (the maximum value for $wWRAcc'$; now being 0.13). The meaning of the lines is the same as before, only this time the lines are dashed (large dash). The weights of all positive examples covered by the newly selected subgroup are reduced (from 1 to $\frac{1}{2}$ and from $\frac{1}{2}$ to $\frac{1}{3}$).

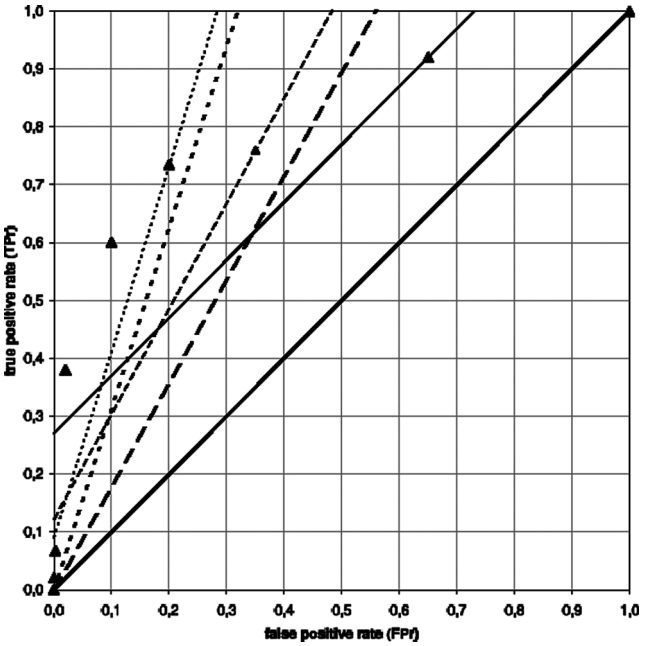


**FIGURE 4** The effect of example weighting used in APRIORI-SD.

- The value of $wWRAcc'$ is again recomputed, taking into account the new weights and again the "best" subgroup is selected. In the figure the new subgroup is shown as a triangle in point (0.20,0.73). The $wWRAcc'$ value of the new subgroup is 0.007 (the maximum value for $wWRAcc'$ being 0.07). Dashed lines (small dash) are used to show the iso-lines. The weights of all positive examples covered by the new subgroup are again reduced (from 1 to $\frac{1}{2}$, from $\frac{1}{2}$ to $\frac{1}{3}$ and from $\frac{1}{3}$ to $\frac{1}{4}$) and the algorithm is run iteratively until all the subgroups are discovered.

## ROC Analysis of Alternative Example Weighting Schemes

The $wWRAcc'$ quality function gives way to defining alternative weighting schemes. In this section, two alternative weighting schemes for APRIORI-SD are presented and analyzed in the ROC space.

### *$wWRAcc'$ by Weighting Just the Covered Positive Examples*

The weighting scheme described in this section is very similar to the one used by the original APRIORI-SD algorithm, with the difference that in this new scheme only the covered positive examples are re-weighted. The original APRIORI-SD's weighting scheme re-weights all the covered examples. The behavior of this new weighting scheme in conjunction with

the $wWRAcc'$ quality function is shown in Figure 3. APRIORI-SD's weighting scheme would behave very similarly to the new scheme if used in conjunction with the $wWRAcc'$ function with the difference that the increase of angle of the ROC isometrics with the decrease of example weights would be less drastic than in the case of the new weighting scheme.

As seen in Figure 3, APRIORI-SD with this weighting scheme would tend to discover more accurate subgroups—ROC isometrics tend to become more and more vertical with the decrease of example weights thus pushing rule selection to an area that contains subgroups with few negative examples.

***wWRAcc' by Weighting Just the Covered Negative Examples***

The behavior of the weighting scheme described here is shown in Figure 5. The figure shows that by decreasing the weights only of the covered negative examples, the angle of ROC isometrics decreases with the decrease of example weights, behaving just the opposite in comparison with the previous weighting schemes.

It is dangerous to use this weighting scheme, as clearly shown in Figure 5. By decreasing the weights of covered negative examples, the lower angle of the ROC isometrics allows the algorithm to find subgroups lying under the main diagonal in the ROC space.

To cope with this problem we have to correct the $wWRAcc'$ function in such a way that subgroups with a positive value of the new quality function will always lie above the main diagonal. To achieve this, we push the ROC isometrics (dashed lines in Figure 5 above the main diagonal by subtracting

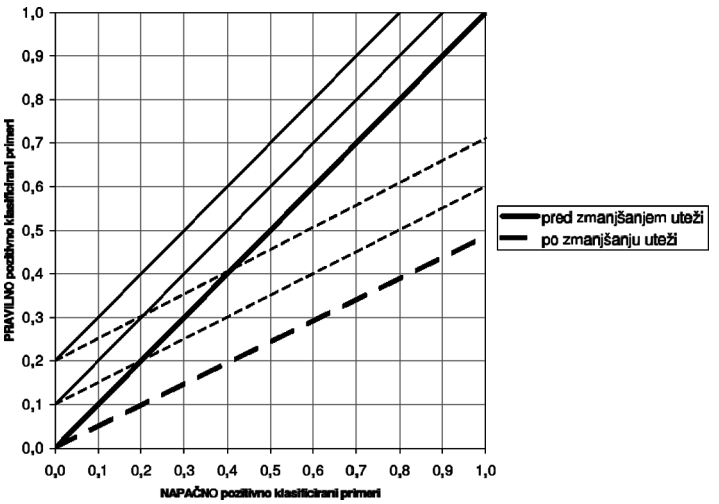


**FIGURE 5** ROC isometrics: the effects of weighting just the negative examples on the $wWRAcc'$ quality function.

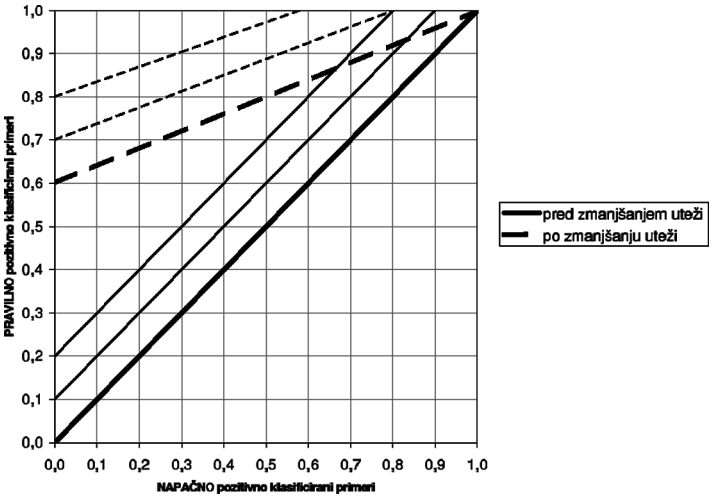


**FIGURE 6** ROC isometrics: the corrected weighting of negatives and its effect on the $wWRAcc'$ quality function.

the value of $wWRAcc'$ of the default rule $(true \rightarrow Y)$ from the $wWRAcc'$ value of the subgroup. The corrected $wWRAcc'$ can be expressed as follows:

$$wWRAcc'(X \rightarrow Y) - wWRAcc'(true \rightarrow Y) \tag{6}$$

Its behavior is shown in Figure 6. As shown by this figure, APRIORI-SD with this weighting scheme will tend to discover larger subgroups: ROC isometrics tend to become more and more horizontal with the decrease of example weights thus pushing rule selection to an area that contains subgroups which cover a large number of examples.

## EXPERIMENTAL EVALUATION

This section provides results of the experimental evaluation of APRIORI-SD aimed at verifying our claims that the mechanisms implemented in the APRIORI-SD algorithm are indeed appropriate for subgroup discovery.

### Evaluation Measures

The evaluation measures used in the experimental evaluations are presented next.

- **Coverage.** The average coverage measures the percentage of examples covered on average by one rule of the induced ruleset. Coverage of a

single rule, $R_i$, is defined as

$$Cov(R_i) = Cov(X_i \rightarrow Y) = p(X_i) \approx \frac{n(X_i)}{N}. \quad (7)$$

The average coverage of a ruleset is computed as

$$COV = \frac{1}{n_B} \sum_{i=1}^{n_B} Cov(R_i), \quad (8)$$

where $n_B$ is the number of induced rules.

- **Support.** For subgroup discovery it is interesting to compute the overall support (the target coverage) as the percentage of target examples (positives) covered by the rules, computed as the true positive rate for the union of subgroups. Support of a rule is defined as the frequency of correctly classified covered examples:

$$Sup(R_i) = Sup(Class \leftarrow Cond_i) = p(Class.Cond_i) = \frac{n(Class.Cond_i)}{N} \quad (9)$$

The overall support of a rule set is computed as

$$SUP = \frac{1}{N} \sum_{Class_j} n(Class_j \cdot \bigvee_{Class_j \leftarrow Cond_i} Cond_i), \quad (10)$$

where the examples covered by several rules are counted only once (hence the disjunction of rule conditions of rules with the same $Class_j$ value in the rule head).

- **Size.** Size is a measure of complexity (the syntactical complexity of induced rules). The ruleset size is computed as the number of rules in the induced ruleset (including the default rule):

$$SIZE = n_B. \quad (11)$$

Size $n_B$ of the induced ruleset equals $B + 1$ ($B$ best rules plus one default rule). In addition to ruleset size used in this paper, complexity could be measured also by the average number of rules/subgroups per class, and the average number of features per rule.

- **Significance.** Average rule significance is computed in terms of the likelihood ratio of a rule; the average is computed over all the rules. Significance (or *evidence*, in the terminology of Klösgen [1996] indicates how significant is a finding, if measured by this statistical criterion. In the CN2 algorithm (Clark and Niblett 1989), significance is measured in

terms of the likelihood ratio statistic of a rule as follows:

$$Sig(R_i) = Sig(X_i \rightarrow Y) = 2 \cdot \sum_j n(X_i \cdot Y_j) \cdot \log \frac{n(Y_j \cdot X_i)}{n(Y_j)\frac{n(X_i)}{N}}, \qquad (12)$$

where for each class value $Y_j$, $n(X_i \cdot Y_j)$ denotes the number of examples with class value $Y_j$ in the set where the rule body $X_i$ holds true, and $n(Y_j)$ is the number of examples with class value $Y_j$ in the data set. Note that although for each generated subgroup description one class value is selected as the target class value, the significance criterion measures the distributional unusualness unbiased to any particular class value. As such, it measures the significance of rule condition only. The average significance of a ruleset is computed as:

$$SIG = \frac{1}{n_B} \sum_{i=1}^{n_B} Sig(R_i). \qquad (13)$$

- **Unusualness.** Average rule unusualness is computed as the average *WRAcc* computed over all the rules:

$$WRACC = \frac{1}{n_B} \sum_{i=1}^{n_B} WRAcc(R_i). \qquad (14)$$

  As discussed previously, *WRAcc* is appropriate for measuring the unusualness of separate subgroups, because it is proportional to the vertical distance from the diagonal in the ROC space (see the underlying reasoning presented previously.
- **Predictive accuracy.** It is important to note the percentage of correctly predicted instances. For a binary classification problem, ruleset accuracy is computed as follows:

$$ACC = \frac{TP + TN}{N}. \qquad (15)$$

  Note that ACC measures the accuracy of the whole ruleset on both positive and negative examples, while rule accuracy or rule confidence (defined as $Acc(X \rightarrow Y) = Conf(X \rightarrow Y) = (TP/TP + FP)$) measures the accuracy of a single rule on positives only.
- **Area under ROC curve.** The method for computing the area under ROC curve (*AUC*) interprets a ruleset as a probabilistic model, given all the different probability thresholds as defined through the probabilistic classification of test instances. AUC can thus be computed by employing

combined probabilistic classifications of all rules/subgroups (Lavrač et al. 2004), as indicated next. 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, if not, negative. The 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 the ROC space). This results in $M + 1$ points in the ROC space, where $M$ is the total number of examples to be classified. Equivalently, we can order all the test examples by decreasing the predicted probability of being positive, and tracing the ROC curve by starting in (0,0), stepping up when the tested example is actually positive, and stepping to the right when it is negative, until we reach (1,1). In the case of ties, we make the appropriate number of steps up and to the right at once, drawing a diagonal line segment. Each point on this curve corresponds to a classifier defined by a possible probability threshold. The ROC curve depicts a set of classifiers, whereas the area under this ROC curve indicates the combined quality of all rules/subgroups (i.e., the quality of the entire ruleset). This method can be used with a test set or in cross-validation, but the resulting curve is not necessarily convex. For details on this method, see (Lavrač et al. 2004). A description of this method applied to decision tree induction can be found in Ferri-Ramirez et al. (2002).

### Evaluation on Selected UCI Data Sets

We experimentally evaluated our approach on 23 data sets from the UCI Repository of Machine Learning Databases (Murphy and Aha 1994). In Table 3, the selected data sets are summarized in terms of the number of attributes (discrete and continuous), number of classes, number of examples, percentage of examples in the majority class, and the maximal value of the *WRAcc* function. All continuous attributes were discretized with a discretization method described in (Witten and Frank 1999) using the WEKA tool Kononenko (1995).

The comparison of APRIORI-SD with three classification rule learners APRIORI-C, RIPPER, and CN2 was performed using the evaluation measures described previously. The area under the ROC curve evaluation was computed only on two-class problems (first 16 data sets in Table 3). The method we used for evaluation was 10-fold stratified cross validation. The parameters used to run the algorithms APRIORI-SD and APRIORI-C were: $minConf = 0.8$ and $minSup = 0.03$, and $k = 5$[4]. We used the version of RIPPER implemented in WEKA (Witten and Frank 1999) with default parameters; Boswell's implementation of CN2 was used (Clark and Boswell 1991).

**TABLE 3** Date Set Characteristics

| | Domena | #Attrib. | #Discr. | #Cont. | #Class. | #Ex. | Maj. Class (%) | Max. WRAcc |
|---|---|---|---|---|---|---|---|---|
| 1 | australian | 14 | 8 | 6 | 2 | 690 | 56 | 0.246 |
| 2 | breast-w | 9 | 9 | 0 | 2 | 699 | 66 | 0.224 |
| 3 | bridges-td | 7 | 4 | 3 | 2 | 102 | 85 | 0.128 |
| 4 | chess | 36 | 36 | 0 | 2 | 3196 | 52 | 0.250 |
| 5 | diabetes | 8 | 0 | 8 | 2 | 768 | 65 | 0.228 |
| 6 | echo | 6 | 1 | 5 | 2 | 131 | 67 | 0.221 |
| 7 | german | 20 | 13 | 7 | 2 | 1000 | 70 | 0.210 |
| 8 | heart | 13 | 6 | 7 | 2 | 270 | 56 | 0.246 |
| 9 | hepatitis | 19 | 13 | 6 | 2 | 155 | 79 | 0.166 |
| 10 | hypothyroid | 25 | 18 | 7 | 2 | 3163 | 95 | 0.048 |
| 11 | ionosphere | 34 | 0 | 34 | 2 | 351 | 64 | 0.230 |
| 12 | iris | 4 | 0 | 4 | 2 | 150 | 66 | 0.221 |
| 13 | mutagen | 59 | 57 | 2 | 2 | 188 | 66 | 0.224 |
| 14 | mutagen-f | 57 | 57 | 0 | 2 | 188 | 66 | 0.224 |
| 15 | tic-tac-toe | 9 | 9 | 0 | 2 | 958 | 65 | 0.228 |
| 16 | vote | 16 | 16 | 0 | 2 | 435 | 61 | 0.238 |
| 17 | balance | 4 | 0 | 4 | 3 | 625 | 46 | 0.248 |
| 18 | car | 6 | 6 | 0 | 4 | 1728 | 70 | 0.210 |
| 19 | glass | 9 | 0 | 9 | 6 | 214 | 36 | 0.230 |
| 20 | image | 19 | 0 | 19 | 7 | 2310 | 14 | 0.120 |
| 21 | soya | 35 | 35 | 0 | 19 | 683 | 13 | 0.113 |
| 22 | waveform | 21 | 0 | 21 | 3 | 5000 | 34 | 0.224 |
| 23 | wine | 13 | 0 | 13 | 3 | 178 | 40 | 0.240 |

Table 4 presents summary results of the comparisons on UCI data sets, while details can be found in the appendix. For each performance measure, the summary table shows the average value over all the data sets, the significance of the results compared to APRIORI-SD ($p$-value), and the WIN/LOSS/DRAW in terms of the number of domains in which the results are better/worse/equal compared to APRIORI-SD. The analysis shows the following:

- In terms of the average coverage per rule, APRIORI-SD produces rules with significantly higher coverage (higher the coverage better the rule) than both APRIORI-C, RIPPER, and CN2.
- APRIORI-SD induces rulesets with lower support than RIPPER and CN2, covering a smaller portion of the target concept, thus leaving more examples unclassified. APRIORI-SD then classifies these examples with the default rule. This fact is also the cause for poorer performance of APRIORI-SD in terms of classification accuracy.
- APRIORI-SD induces rulesets that are significantly smaller than those induced by APRIORI-C, RIPPER, and CN2 (smaller rulesets are more understandable and thus better).
- APRIORI-SD induces significantly better rules in terms of significance measured by the average $\chi^2$ likelihood ratio (rules with higher significance are better) than APRIORI-C, RIPPER, and CN2.

**TABLE 4** Comparison of APRIORI-SD with Different Rule Learning Algorithms. The Best Results are in Bold

| Performance measure | Data sets | APRIORI-SD | APRIORI-C | RIPPER | CN2 | Detailed results |
|---|---|---|---|---|---|---|
| **COV** | 23 | **0.534**±0.26 | 0.363±0.19 | 0.190±0.19 | 0.131 ±0.14 | Table 7 |
| • significance (p value) | | | 0.000 | 0.000 | 0.000 | |
| • win/loss/draw | | | 1/22/0 | 1/22/0 | 1/22/0 | |
| • sig.win/sig.loss | | | 1/17 | 1/21 | 1/21 | |
| **SUP** | 23 | 0.83±0.13 | 0.81±0.12 | 0.84 ± 0.07 | **0.85** ±0.03 | Table 8 |
| • significance (p value) | | | 0.022 | 0.771 | 0.616 | |
| • win/loss/draw | | | 7/16/0 | 13/10/0 | 11/12/0 | |
| • sig.win/sig.loss | | | 1/7 | 9/9 | 9/9 | |
| **SIZE** | 23 | **3.58** ± 1.96 | 5.61 ± 2.84 | 16.12 ± 27.47 | 18.18 ± 21.77 | Table 9 |
| • significance (p value) | | | 0.000 | 0.035 | 0.003 | |
| • win/loss/draw | | | 2/21/0 | 3/20/0 | 1/22/0 | |
| • sig.win/sig.loss | | | 2/19 | 2/19 | 1/21 | |
| **SIG** | 23 | **12.37** ± 7.26 | 2.60 ± 0.55 | 2.36 ± 0.55 | 2.11 ± 0.46 | Table 10 |
| • significance (p value) | | | 0.000 | 0.000 | 0.000 | |
| • win/loss/draw | | | 1/22/0 | 1/22/0 | 1/22/0 | |
| • sig.win/sig.loss | | | 0/22 | 1/21 | 0/22 | |
| **WRACC** | 23 | **0.047** ± 0.03 | 0.042 ± 0.03 | 0.021 ± 0.02 | 0.017 ± 0.02 | Table 11 |
| • significance (p value) | | | 0.000 | 0.001 | 0.000 | |
| • win/loss/draw | | | 0/22/1 | 4/19/0 | 3/20/0 | |
| • sig.win/sig.loss | | | 0/11 | 3/18 | 3/19 | |
| **ACC** | 23 | 79.98±16.67 | 81.02± 16.50 | **83.46** ± 10.24 | 81.61 ± 11.66 | Table 12 |
| • significance (p value) | | | 0.039 | 0.282 | 0.489 | |
| • win/loss/draw | | | 13/10/0 | 10/13/0 | 8/15/0 | |
| • sig.win/sig.loss | | | 7/0 | 8/7 | 6/10 | |
| **AUC** | 16 | **82.80**± 8.70 | 80.92 ± 9.95 | 80.11 ± 10.23 | 82.16 ± 16.81 | Table 13 |
| • significance (p value) | | | 0.190 | 0.027 | 0.871 | |
| • win/loss/draw | | | 6/10/0 | 4/12/0 | 11/5/0 | |
| • sig.win/sig.loss | | | 4/6 | 4/7 | 9/6 | |

- APRIORI-SD induces rulesets with higher unusualness than APRIORI-C, RIPPER, and CN2. Since unusualness (*WRAcc*) is considered the most important measure for estimating the quality of discovered subgroups, we can claim that APRIORI-SD discovers "the best" subgroups.
- In terms of predictive accuracy APRIORI-SD is insignificantly worse than RIPPER and CN2, while being significantly worse than APRIORI-C.
- As the comparisons in terms of the area under the ROC curve (AUC) are restricted to binary class data sets, only the 16 binary data sets were used in this comparison. Notice that while being better than APRIORI-C and RIPPER, APRIORI-SD is comparable to CN2.

### Evaluation on a Real-Life Data Set

In order to compare APRIORI-SD with two other state-of-the-art subgroup discovery algorithms, CN2-SD (Lavrač et al. 2002) and SubgroupMiner

(Klösgen and May 2002), we applied these algorithms to a real-life problem—the U.K. traffic challenge data set. This data set is a sample of a larger and more complete relational data set—the UK traffic data set briefly described next. The results of the comparison are presented in the form of ROC plots.

***The U.K. Traffic Accident Data Set***

The U.K. traffic data set includes the records of all the accidents that happened on the roads of Great Britain between years 1979 and 1999. It is a relational data set consisting of three related sets of data: the ACCIDENT data, the VEHICLE data and the CASUALTY data. The ACCIDENT data consists of the records of all accidents that happened in the given time period; VEHICLE data includes data about all the vehicles involved in these accidents; and CASUALTY data includes the data about all the casualties involved in the accidents. Consider the following example: Two vehicles crashed in a traffic accident and three people were seriously injured in the crash. In terms of the TRAFFIC data set, this is recorded as one record in the ACCIDENT set, two records in the VEHICLE set, and three records in the CASUALTY set. Every separate set is described by approximately 20 attributes and consists of more than 5 million records.

***The U.K. Traffic Challenge***

The task of the challenge was to produce classification models (in our case, subgroup descriptions) to predict skidding and overturning for accidents from the U.K. traffic data set (Mladenić and Lavrač 2003). As the class attribute *Skidding and Overturning* appears in the VEHICLE data table, the data tables ACCIDENT and VEHICLE were merged in order to make this a simple non-relational problem. Furthermore, a sample of 5940 records from this merged data table was selected for learning and another sample of 1585 records was selected for testing. The class attribute *Skidding and Overturning* has six possible values. The meaning of these values and the distribution of the class values in the training and test sets are shown in Table 5.

**TABLE 5** The Meaning and the Distribution of Classes in the U.K. Traffic Challenge Data

| Class | Meaning of class values | Train | Test |
|---|---|---|---|
| 0 | No skidding, jack-knifing or overturning | 64.26 | 64.67 |
| 1 | Skidded | 22.07 | 22.46 |
| 2 | Skidded and overturned | 7.27 | 6.88 |
| 3 | Jack-knifed | 0.20 | 0.06 |
| 4 | Jack-knifed and overturned | 0.19 | 0.44 |
| 5 | Overturned | 6.01 | 5.49 |

***Experimental Results***

We compared subgroup discovery algorithms APRIORI-SD, CN2-SD, and SubgroupMiner by applying them to the U.K. traffic challenge training data to construct subgroups and then test these subgroups on the test data. The results are plotted in the ROC space. Because of the fact that only binary class problems can be plotted in the ROC space, we had to transform the original problem of predicting a class with six values to six binary problems, predicting each class in turn as positive and the remaining classes as negative. All three subgroup discovery algorithms were run with the following parameters (APRIORI-SD with $minConf = 0$, $minSup = 0.01$, and $k = 5$; CN2-SD using the additive weighting scheme, 99% significance threshold and beam size 5; SubgroupMiner with beam size 10, max. length of rules 6, and suppression factor $\alpha = 1$).

We discarded the problems of predicting Class 3 and Class 4 (see Table 5 for the meaning of class codes) because they contained too few test examples (see the distribution in Table 5), and we discarded the ROC plot for the problem of predicting Class 2 because it is very similar to the ROC plot for predicting Class 1.

The results of the comparisons on the remaining three problems of predicting Class 0, Class 1, and Class 5 are shown (plotted in the ROC space) in Figures 7, 8, and 9, respectively. We can describe these problems as the problems of predicting the majority class (Class 0), the minority class (Class 5), and the class that is neither majority nor minority (Class 1).

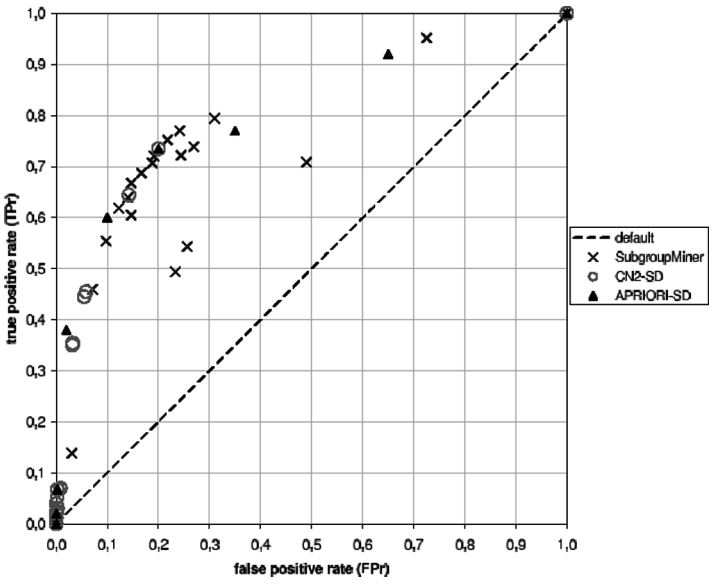


**FIGURE 7** The ROC plot for the problem of predicting Class 0.

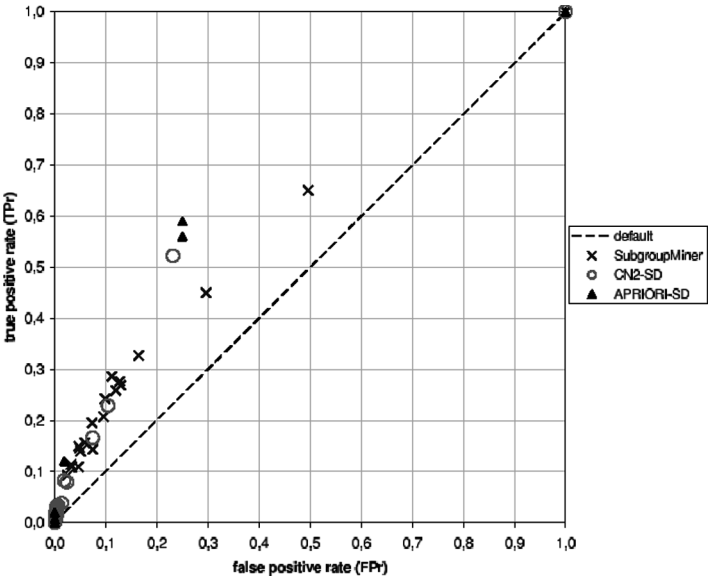


**FIGURE 8** The ROC plot for the problem of predicting Class 1.

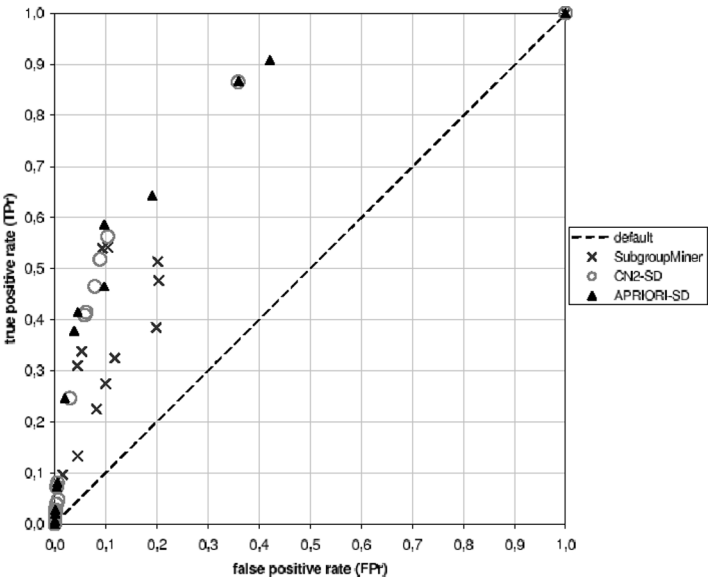


**FIGURE 9** The ROC plot for the problem of predicting Class 5.

The following can be observed by analyzing the results in Figures 7, 8, and 9:

1. Both APRIORI-SD and CN2-SD discovered smaller and more accurate subgroups (points nearer to the point $(0, 0)$ in all the three figures) than SubgroupMiner.
2. SubgroupMiner discovered larger but less accurate subgroups. This is especially true for the problem of predicting the majority class (Class 0 in Figure 7).
3. If the ROC convex hull, connecting the best performing subgroups (subgroups with the best *TPr*/*FPr* tradeoff), we could observe that SubgroupMiner discovered several of subgroups that do not lie on the ROC convex hull and are thus sub-optimal.
4. Both APRIORI-SD and CN2-SD discovered better subgroups (the distance from the ROC diagonal is larger) when dealing with the problem of predicting a minority class (see Figures 8 and 9).
5. APRIORI-SD is "better" than CN2-SD in terms of the average *WRAcc* (the subgroups discovered by APRIORI-SD are on average further away from the main diagonal in the ROC space than those discovered by CN2-SD).

***Interpretation of the Results***

This section explains each of the five findings of the previous section, starting with the last one.

The fifth finding—APRIORI-SD being better than CN2-SD in terms of the average *WRAcc*—can be explained by the fact that CN2-SD is bound to miss some "good" subgroups by using heuristic search, while APRIORI-SD using exhaustive search takes into consideration all "potentially good" subgroups.

The fourth finding—both APRIORI-SD and CN2-SD discovered better subgroups when dealing with the problem of predicting a minority class—can be attributed to the *WRAcc* heuristic used in APRIORI-SD (in rule post-processing) and CN2-SD (used in heuristic beam search of rules). This result experimentally confirms the appropriateness of the *WRAcc* heuristic for subgroup discovery, which aims at finding subgroups maximizing the distance from the ROC diagonal (Lavrač et al. 2004).

The third finding—SubgroupMiner discovered a lot of subgroups that do not lie on the ROC convex hull—can also be attributed to the fact that the algorithms use different heuristics when searching the space of possible rules/subgroups.

To explain the first two findings, we use ROC isometrics described in Flach (2003) and Fürnkranz and Flach (2003). With the help of ROC isometrics we can investigate the behavior of quality functions used in APRIORI-SD, CN2-SD, and SubgroupMiner. APRIORI-SD and CN2-SD

use the same quality function to find subgroups $wWRAcc'$ with example weights described in Equation 4. The behavior of this quality function is shown in Figure 3 in the form of ROC isometrics, where each line represents some value of the quality function (see Flach [2003] and Fürukranz and Flach [2003] for a detailed description of ROC isometrics).

We have analyzed the effects of example weighting through the analysis of Figure 3. We have observed that, in general, while still remaining parallel, the angle of ROC isometrics for $wWRAcc'$ increases with the decrease of the weights of (positive) examples. Here we compare the APRIORI-SD $wWRAcc'$ isometrics of 3 with the behavior of SubgroupMiner's quality function shown in Figure 10.

We can now proceed explaining the first two findings from the results by looking at Figures 3 and 10. Figure 3 shows that the *WRAcc* quality function with example weights used by APRIORI-SD and CN2-SD "tries harder" to discover more accurate subgroups, i.e., lowering the weights on positive examples makes the lines in the figure more vertical. Since there are no large subgroups that are at the same time highly accurate, the effect of weighting in our case results in finding small, highly accurate subgroups (explanation of the first finding). The second finding can be explained

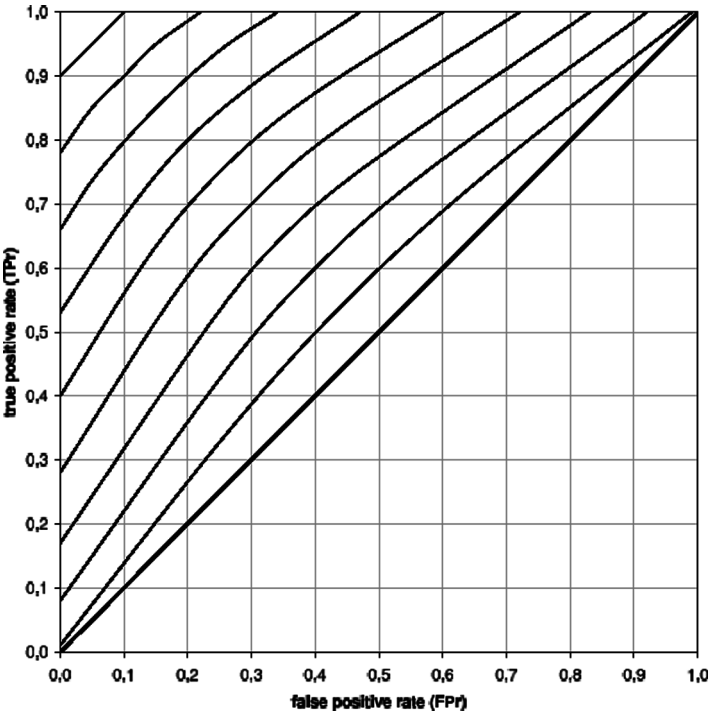


**FIGURE 10** ROC isometrics of the quality function used in SubgroupMiner.

by looking at Figure 10. We can see that SubgroupMiner's quality functions tend to discover small and accurate subgroups and at the same time large and inaccurate ones (note the bending of iso-lines towards the points (0,0) and (1,1)). The latter fact explains the second finding from the results. Why did SubgroupMiner not discover small and accurate subgroups (in such a number as APRIORI-SD and CN2-SD did) can be attributed again to the heuristics used by the algorithms in searching the space of potentially "good" subgroups.

***Evaluating WRAcc Variants on the U.K. Traffic Challenge Data***

This section uses the U.K. traffic challenge data to evaluate the performance of different *WRAcc* variants proposed in this paper.

We applied APRIORI-SD with the following parameters ($minConf = 0$, $minSup = 0.0001$, $k = 5$) and additional constraints ($minWRAcc = 0$, *maximal no. of terms in a subgroup* $= 10$) on the training set of 5940 examples. The algorithm was run 18 times (six times to discover subgroups for each of the class values—one was always set as positive and the other five as negative; three times for each of the weighting schemes).

The following performance measures were used in the comparisons. *SIZE*: the number of discovered subgroups on the training set; *ACC*: the accuracy of a ruleset on the test set (of 1585 examples); and *COV*: the average coverage of a subgroup.

The results are shown in Table 6 and confirm the theoretical findings from earlier. We can see from this table that when using $wWRAcc'$ with weighting just the positive examples, the algorithm finds subgroups that are on the average smaller and more accurate. On the other hand, by using

**TABLE 6** The Results of Applying APRIORI-SD with Different Weighting Schemes on the U.K. Traffic Challenge Data

| | Performance measures | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|
| | ACC | | | COV | | | SIZE | | |
| Class | □ | + | − | □ | + | − | □ | + | − |
| 0 | 0.875 | 0.901 | 0.823 | 0.231 | 0.213 | 0.402 | 112 | 91 | 19 |
| 1 | 0.449 | 0.502 | 0.397 | 0.101 | 0.076 | 0.183 | 83 | 74 | 12 |
| 2 | 0.101 | 0.124 | 0.090 | 0.050 | 0.041 | 0.101 | 20 | 15 | 6 |
| 3 | 0.023 | 0.023 | — | 0.005 | 0.005 | — | 3 | 3 | 0 |
| 4 | 0.035 | 0.040 | 0.028 | 0.011 | 0.007 | 0.019 | 6 | 5 | 2 |
| 5 | 0.203 | 0.251 | 0.183 | 0.088 | 0.076 | 0.205 | 31 | 25 | 8 |

□: the *wWRAcc* weighting scheme used in APRIORI-SD.
+: the *wWRAcc'* by weighting just positive examples.
−: the *wWRAcc'* by weighting just negative examples.

(corrected) *wWRAcc'* and weighting just the negative examples, on the average larger and less accurate subgroups are discovered by the algorithm.

Another thing that can be seen from Table 6 is that by using APRIORI-SD's original weighting scheme, more subgroups are discovered then when one of the two alternative weighting schemes is used.

## CONCLUSIONS

Following the ideas presented in Lavrač (2004), we have adapted the APRIORI-C algorithm to subgroup discovery, resulting in the APRIORI-SD subgroup discovery algorithm. Experimental results on 23 UCI data sets demonstrate that APRIORI-SD produces smaller rulesets, where individual rules have higher coverage, significance, and unusualness compared to rule learners APRIORI-C, RIPPER, and CN2. These factors are important for subgroup discovery: Smaller size enables better understanding, higher coverage means larger subgroups, and higher significance and unusualness mean that rules describe subgroups whose class distribution is significantly different from the entire population. This is achieved by virtually no loss in terms of the area under the ROC curve and accuracy.

We have evaluated the results of APRIORI-SD also in terms of classification accuracy and AUC and, have shown a small increase in terms of the area under the ROC curve compared to APRIORI-C and RIPPER. On the other hand, an insignificant increase in AUC compared to CN2 could be attributed to the use of non-discretized attributes in CN2. APRIORI-SD was insignificantly worse in terms of predictive accuracy than RIPPER and CN2, while being significantly worse than APRIORI-C. Notice however, that subgroup discovery is not intended at maximizing accuracy.

By comparing the APRIORI-SD with two subgroup discovery algorithms, CN2-SD and SubgroupMiner, on real-life U.K. traffic challenge data, we have shown that APRIORI-SD acts very similarly to CN2-SD and is more suitable for predicting the minority classes, while SubgroupMiner found larger and more accurate subgroups when dealing with the majority classes. On the other hand, SubgroupMiner tends to produce larger subgroups. In conclusion, APRIORI-SD was slightly better than CN2-SD in the real-life U.K. traffic challenge problem, using the ROC space as an evaluation tool. While compared to SubgroupMiner, neither of the two algorithms was absolutely better than the other one. They can provide us with different insights of the data. It is then the task of an expert to have a final word on which of the algorithms produces better subgroups in relation to her needs.

In addition, following the ideas presented in Flach (2003) and Fürukranz and Flach (2003), we used ROC analysis to study the behavior of various example weighting schemes. We have presented the arguments for modift-

ing *WRacc* to $wWRAcc'$. In addition, we have provided a theoretical analysis of new weighting schemes pointing out that while the first scheme ($wWRAcc'$ by weighting just the covered positive examples) is more focused, guiding rule selection towards smaller and highly accurate subgroups, the second one ($wWRAcc'$ by weighting just the covered negative examples) guides rule selection toward more general subgroups, which are larger and less accurate.

An important aspect of subgroup discovery performance, which was neglected in our study, is the degree of overlap of the induced subgroups. The challenge of our further research is to propose extensions of the weighted relative accuracy heuristic and the ROC space evaluation metrics that will take into account the overlap of subgroups.

## REFERENCES


Agrawal, R., J. Gehrke, D. Gunopulos, and P. Raghavan. 1998. Automatic subspace clustering of high dimensional data for data mining applications. In *Proceedings of ACM SIGMOD Conference on Management of Data*, pages 94–105.

Agrawal, R., T. Imielinski, and R. Srikant. 1993. Mining association rules between sets of items in large databases. In *Proceedings of ACM SIGMOD Conference on Management of Data*, pages 207–216.

Agrawal, R. and R. Srikant. 1994. Fast algorithms for mining association rules. In *Proceedings of the 20th International Conference on Very Large Databases*, pages 487–499. Santiago, Chile: Morgan Kaufmann Newport Beach, CA: AAAI Press.

Ali, K., S. Manganaris, and R. Srikant. 1997. Partial classification using association rules. In *Proceedings of the 3rd International Conference on Knowledge Discovery and Data Mining (KDD'97)*, pages 115–118.

Bayardo, R. J., R. Agrawal, and D. Gunopulos. 1999. Constraint-based rule mining in large, dense databases. In *Proceedings of the 15th International Conference on Data Engineering*, pages 188–197.

Clark, P. and R. Boswell. 1991. Rule induction with CN2: Some recent improvements. In *Proceeding of the 5th European Working Session on Learning*, pages 151–163.

Clark, P. and T. Niblett. 1989. The CN2 induction algorithm. *Machine Learning* 3(4):261–283.

Cohen, W. W. 1995. Fast effective rule induction. In *Proceedings of the 12th International Conference on Machine Learning*, pages 115–123. Morgan Kaufmann.

Ferri-Ramírez, C., P. A. Flach, and J. Hernandez-Orallo. 2002. Learning decision trees using the area under the ROC curve. In *Proceedings of the 19th International Conference on Machine Learning*, pages 139–146. Morgan Kaufmann.

Flach, P. A. 2003. The geometry of ROC space: Understanding machine learning metrics through ROC isometrics. In *Proceedings of the 20th International Conference on Machine Learning*, pages 194–201. Seattle, WA: ACM Press.

Fürnkranz, J. and P. A. Flach. 2003. An analysis of rule evaluation metrics. In *Proceedings of the 20th International Conference on Machine Learning*, pages 202–209. Washington, DC: ACM Press.

Gamberger, D. and N. Lavrač. 2002. Expert guided subgroup discovery: Methodology and application. *Journal of Artificial Intelligence Research* 17:501–527.

Gebhardt, F. 1991. Choosing among competing generalizations. *Knowledge Aquisition Journal* 3:361–380.

Jovanoski, V. and N. Lavrač. 2001. Classification rule learning with APRIORI-C. In *Progress in Artificial Intelligence: Proceedings of the 10th Portuguese Conference on Artificial Intelligence*, pages 44–51, Springer.

Klösgen, W. 1996. EXPLORA: A multipattern and multistrategy discovery assistant. In *Advance in Knowledge Discovery and Data Mining*, 249–271. Cambridge, MIT Press.

Klösgen, W. 1999. Applications and research problems of subgroup mining. XI International Symposium on Foundations of Intelligent Systems table of contents (ISMIS'99). *Lecture Notes in Computer Science* 1609, Springer–Verlag 1999, Warsaw, Poland, 1–15.

Klösgen, W. 2002. *Handbook of Data Mining and Knowledge Discovery*, chapter Subgroup Discovery, 213–242. New York: Oxford University Press.

Klösgen, W. and M. May. 2002. Spatial subgroup mining integrated in an object-relational spatial database. In *Proceedings of the 6th European Conference on Principles and Practice of Knowledge Discovery in Databases*, pages 275–286. Helsinki, Finland: Springer.

Kononenko, I. 1995. On biases in estimating multi-valued attributes. In *Proceedings of the 14th International Joint Conference on Artificial Intelligence*, pages 1034–1040. Montreal, Canada: Morgan Kaufmann.

Lavrač, N., P. A. Flach, B. Kavšek, and L. Todorovski. 2002. Adapting classification rule induction to subgroup discovery. In *Proceedings of the 2nd IEEE International Conference on Data Mining*, pages 266–273. IEEE Computer Society.

Lavrač, N., P. A. Flach, and B. Zupan. 1999. Rule evaluation measures: A unifying view. In *Proceedings of the 9th International Workshop on Inductive Logic Programming*, pages 74–185. Springer.

Lavrač, N., B. Kavšek, P. A. Flach, and L. Todorovski. 2004. Subgroup discovery with CN2-SD. *Journal of Machine Learning Research* 5:153–188.

Liu, B., W. Hsu, and Y. Ma. 1998. Integrating classification and association rule mining. In *Proceedings of the 4th International Conference on Knowledge Discovery and Data Mining (KDD'98)*, pages 80–86. New York: AAAI Press.

Mannila, H. and H. Toivonen. 1996. Discovering generalized episodes using minimal occurrences. In *Proceedings of the 2nd International Conference on Knowledge Discovery and Data Mining (KDD'96)*, pages 146–151. Portland, Oregon: AAAI Press.

Megiddo, N. and R. Srikant. 1998. Discovering predictive association rules. In *Proceedings of the 4th International Conference on Knowledge Discovery and Data Mining (KDD'98)*, pages 274–278.

Michalski, R. S., I. Mozetič, J. Hong, and N. Lavrač. 1986. The multi–purpose incremental learning system AQ15 and its testing application on three medical domains. In *Proceedings of the 5th National Conference on Artificial Intelligence*, pages 1041–1045. Morgan Kaufmann.

Mladenić, D. and N. Lavrač. 2003. *Data Mining and Decision Support for Business Competitiveness: A European Virtual Enterprise*. Ljubljana: DZS.

Murphy P. M. and D. W. Aha. 1994. *UCI Repository of Machine Learning Databases*. Available electronically at http://www.ics.uci.edu/mlearn/MLRepository.html.

Provost, F. J. and T. Fawcett. 2001. Robust classification for imprecise environments. *Machine Learning* 42(3):203–231.

Rivest, R. L. 1987. Learning decision lists. *Machine Learning* 2(3):229–246.

Todorovski, L., P. A. Flach, and N. Lavrač. 2000. Predictive performance of weighted relative accuracy. In *Proceedings of the 4th European Conference on Principles of Data Mining and Knowledge Discovery*, pages 255–264. Springer.

Witten, I. H. and E. Frank. 1999. *Data Mining: Practical Machine Learning Tools and Techniques with JAVA Implementations*. Morgan Kaufmann.

Wrobel, S. 1997. An algorithm for multi-relational discovery of subgroups. In *Proceedings of the 1st European Conference on Principles of Data Mining and Knowledge Discovery*, pages 78–87. Springer.

Wrobel, S. 2001. Inductive logic programming for knowledge discovery in databases. In S. Dzeroski and N. Lavrac, (eds.) *Relational Data Mining*. Springer Verlag, Berlin.

Železný, F., N. Lavrač, and S. Džeroski. 2003. Constraint-based relational subgroup discovery. In *Proceedings of the 2nd International Workshop on Multi-Relational Data Mining (KDD-2003)*, pages 135–150. Washington, DC: AAAI Press.

## ENDNOTES

1. Confidence $p(Y|X)$ in association rule learning is called rule accuracy in classification rule learning, and precision in information retrieval.
2. Empirically, this assumption does not seriously affect the angle of the iso-*WRAcc* lines.
3. We show that by decreasing the weights of negative examples, the picture changes. However, by decreasing the weights of all covered examples, we mostly decrease the weights of positives, because the algorithm is trying hard to cover as many positives and at the same time as few negatives as possible.
4. We also ran APRIORI-SD with $minConf = 0$ and $minSup = 0.01$ in order to exhaustively search the space of rules, but the results were almost equal to the ones presented in this paper.

## APPENDIX: DETAILED RESULTS OF EXPERIMENTS ON UCI DATA SETS

**TABLE 7** Average Coverage (*COV*) of Rules with Standard Deviations

| # | APRIORI-SD *COV*± *sd* | APRIORI-C *COV*± *sd* | RIPPER *COV*± *sd* | CN2 *COV*± *sd* |
|---|---|---|---|---|
| 1 | 0.550 ± 0.06 | 0.430↓ ± 0.04 | 0.090↓ ± 0.01 | 0.071↓ ± 0.01 |
| 2 | 0.300 ± 0.03 | 0.190↓ ± 0.02 | 0.100↓ ± 0.01 | 0.079↓ ± 0.10 |
| 3 | 0.540 ± 0.05 | **0.600**↑ ± 0.06 | **0.800**↑ ± 0.08 | **0.625**↑ ± 0.03 |
| 4 | 0.530 ± 0.05 | 0.500 ± 0.05 | 0.050 ± 0.00 | 0.048 ± 0.01 |
| 5 | 0.300 ± 0.03 | 0.280 ± 0.03 | 0.090↓ ± 0.01 | 0.057↓ ± 0.08 |
| 6 | 1.000 ± 0.00 | 0.710↓ ± 0.07 | 0.390↓ ± 0.04 | 0.312↓ ± 0.06 |
| 7 | 0.300 ± 0.03 | 0.110↓ ± 0.01 | 0.070↓ ± 0.01 | 0.053↓ ± 0.08 |
| 8 | 0.670 ± 0.07 | 0.280↓ ± 0.03 | 0.160↓ ± 0.02 | 0.107↓ ± 0.09 |
| 9 | 0.850 ± 0.08 | 0.520↓ ± 0.05 | 0.400↓ ± 0.04 | 0.207↓ ± 0.04 |
| 10 | 0.520 ± 0.05 | 0.500 ± 0.05 | 0.100↓ ± 0.01 | 0.093↓ ± 0.00 |
| 11 | 0.240 ± 0.02 | 0.220 ± 0.02 | 0.160↓ ± 0.02 | 0.099↓ ± 0.05 |
| 12 | 0.840 ± 0.08 | 0.520↓ ± 0.05 | 0.520↓ ± 0.05 | 0.378↓ ± 0.01 |
| 13 | 0.910 ± 0.09 | 0.480↓ ± 0.05 | 0.280↓ ± 0.03 | 0.160↓ ± 0.11 |
| 14 | 0.880 ± 0.09 | 0.470↓ ± 0.05 | 0.230↓ ± 0.02 | 0.142↓ ± 0.01 |
| 15 | 0.290 ± 0.03 | 0.130↓ ± 0.01 | 0.040↓ ± 0.00 | 0.030↓ ± 0.01 |
| 16 | 0.710 ± 0.07 | 0.680 ± 0.07 | 0.190↓ ± 0.02 | 0.129↓ ± 0.01 |
| 17 | 0.380 ± 0.04 | 0.230↓ ± 0.02 | 0.040↓ ± 0.00 | 0.021↓ ± 0.00 |
| 18 | 0.260 ± 0.03 | 0.160↓ ± 0.02 | 0.030↓ ± 0.00 | 0.022↓ ± 0.05 |
| 19 | 0.840 ± 0.08 | 0.370↓ ± 0.04 | 0.150↓ ± 0.02 | 0.066↓ ± 0.01 |
| 20 | 0.240 ± 0.03 | 0.140↓ ± 0.01 | 0.040↓ ± 0.00 | 0.039↓ ± 0.11 |
| 21 | 0.280 ± 0.03 | 0.100↓ ± 0.01 | 0.060↓ ± 0.01 | 0.040↓ ± 0.01 |
| 22 | 0.240 ± 0.03 | 0.190↓ ± 0.02 | 0.010↓ ± 0.00 | 0.004↓ ± 0.01 |
| 23 | 0.620 ± 0.06 | 0.550↓ ± 0.06 | 0.380↓ ± 0.04 | 0.231↓ ± 0.01 |
| Avg | **0.534** ± 0.26 | 0.363 ± 0.19 | 0.190 ± 0.19 | 0.131 ± 0.14 |
| • *p* | | 0.000 | 0.000 | 0.000 |
| • w/l/d | | 1/22/0 | 1/22/0 | 1/22/0 |
| • s.w/s.l | | 1/17 | 1/21 | 1/21 |

**TABLE 8** Overall Support (*SUP*) of Rulesets with Standard Deviations

| # | APRIORI-SD *SUP* ± *sd* | APRIORI-C *SUP* ± *sd* | RIPPER *SUP* ± *sd* | CN2 *SUP* ± *sd* |
|---|---|---|---|---|
| 1 | 0.79 ± 0.07 | 0.75 ± 0.07 | **0.86**↑ ± 0.10 | 0.81 ± 0.09 |
| 2 | 0.64 ± 0.03 | 0.63 ± 0.03 | **0.76**↑ ± 0.01 | **0.88**↑ ± 0.01 |
| 3 | 0.82 ± 0.08 | 0.85 ± 0.08 | 0.81 ± 0.06 | **0.87**↑ ± 0.05 |
| 4 | 0.78 ± 0.09 | 0.75 ± 0.08 | **0.86**↑ ± 0.06 | **0.87**↑ ± 0.06 |
| 5 | 0.69 ± 0.05 | 0.65 ± 0.06 | **0.90**↑ ± 0.01 | **0.80**↑ ± 0.01 |
| 6 | 0.98 ± 0.15 | 0.97 ± 0.15 | 0.81↓ ± 0.03 | 0.90↓ ± 0.03 |
| 7 | 0.77 ± 0.06 | 0.76 ± 0.07 | **0.98**↑ ± 0.02 | **0.89**↑ ± 0.03 |
| 8 | 0.67 ± 0.08 | 0.63↓ ± 0.07 | **0.76**↑ ± 0.04 | **0.84**↑ ± 0.03 |
| 9 | 0.85 ± 0.08 | 0.86 ± 0.09 | 0.70↓ ± 0.10 | 0.87 ± 0.10 |
| 10 | 0.52 ± 0.08 | 0.56 ± 0.08 | **0.94**↑ ± 0.01 | **0.84**↑ ± 0.01 |
| 11 | 0.68 ± 0.04 | 0.65↓ ± 0.05 | **0.87**↑ ± 0.03 | **0.83**↑ ± 0.03 |
| 12 | 0.84 ± 0.09 | 0.82 ± 0.10 | 0.89 ± 0.03 | 0.82 ± 0.04 |
| 13 | 0.91 ± 0.12 | 0.87↓ ± 0.12 | 0.91 ± 0.11 | 0.87↓ ± 0.10 |
| 14 | 0.88 ± 0.10 | 0.85 ± 0.09 | 0.79↓ ± 0.04 | 0.84 ± 0.05 |
| 15 | 0.84 ± 0.08 | 0.85 ± 0.09 | 0.88 ± 0.05 | 0.83 ± 0.04 |
| 16 | 0.71 ± 0.08 | 0.73 ± 0.08 | **0.86**↑ ± 0.07 | **0.85**↑ ± 0.07 |
| 17 | 1.00 ± 0.00 | 0.85↓ ± 0.00 | 0.81↓ ± 0.09 | 0.86↓ ± 0.08 |
| 18 | 0.94 ± 0.02 | 0.89↓ ± 0.02 | 0.76↓ ± 0.05 | 0.81↓ ± 0.06 |
| 19 | 0.91 ± 0.03 | **0.96**↑ ± 0.03 | 0.92 ± 0.01 | 0.83↓ ± 0.01 |
| 20 | 0.98 ± 0.01 | 0.98 ± 0.00 | 0.83↓ ± 0.06 | 0.90↓ ± 0.06 |
| 21 | 0.95 ± 0.02 | 0.92 ± 0.02 | 0.83↓ ± 0.05 | 0.81↓ ± 0.05 |
| 22 | 1.00 ± 0.00 | 0.90↓ ± 0.00 | 0.82↓ ± 0.01 | 0.81↓ ± 0.02 |
| 23 | 0.97 ± 0.01 | 0.94↓ ± 0.02 | 0.79↓ ± 0.04 | 0.82↓ ± 0.05 |
| Avg | 0.83 ± 0.13 | 0.81 ± 0.12 | 0.84 ± 0.07 | **0.85** ± 0.03 |
| • *p* | | 0.022 | 0.771 | 0.616 |
| • w/l/d | | 7/16/0 | 13/10/0 | 11/12/0 |
| • s.w/s.l | | 1/7 | 9/9 | 9/9 |

**TABLE 9** Size (*SIZE*) of Rulesets (In Terms of the Number of Rules) with Standard Deviations

| # | APRIORI-SD *SIZE* ± *sd* | APRIORI-C *SIZE* ± *sd* | RIPPER *SIZE* ± *sd* | CN2 *SIZE* ± *sd* |
|---|---|---|---|---|
| 1 | 3.5 ± 0.15 | **2.6** ↑ ± 0.51 | 11.6↓ ± 1.01 | 12.4↓ ± 1.95 |
| 2 | 4.2 ± 0.43 | 8.0↓ ± 0.26 | 10.7↓ ± 0.12 | 12.6↓ ± 1.04 |
| 3 | 2.4 ± 0.51 | 2.7 ± 0.04 | **1.4** ↑ ± 0.15 | **1.8** ↑ ± 0.10 |
| 4 | 1.4 ± 0.20 | 3.2↓ ± 0.38 | 17.5↓ ± 0.83 | 14.6↓ ± 1.81 |
| 5 | 4.4 ± 0.30 | **3.9** ↑ ± 0.35 | 10.2↓ ± 0.32 | 12.8↓ ± 1.56 |
| 6 | 1.0 ± 0.00 | 3.5↓ ± 0.00 | 2.9↓ ± 0.12 | 3.7↓ ± 1.37 |
| 7 | 6.2 ± 0.05 | 9.7↓ ± 0.83 | 11.5↓ ± 1.13 | 15.1↓ ± 1.89 |
| 8 | 1.4 ± 0.14 | 4.4↓ ± 0.10 | 5.2↓ ± 0.04 | 6.4↓ ± 1.53 |
| 9 | 2.8 ± 0.72 | 4.4↓ ± 0.28 | 2.5 ± 0.69 | 3.0 ± 0.29 |
| 10 | 1.4 ± 0.71 | 3.0↓ ± 0.08 | 9.3↓ ± 0.14 | 10.1↓ ± 1.02 |
| 11 | 3.5 ± 0.63 | 5.8↓ ± 0.13 | 6.6↓ ± 0.02 | 7.6↓ ± 1.01 |
| 12 | 2.1 ± 0.49 | 2.5 ± 0.61 | **1.8** ↑ ± 0.75 | 3.8↓ ± 1.24 |
| 13 | 2.8 ± 0.14 | 4.1↓ ± 0.07 | 3.3↓ ± 0.16 | 4.7↓ ± 1.30 |
| 14 | 2.3 ± 0.27 | 4.0↓ ± 0.23 | 2.4 ± 0.05 | 5.2↓ ± 0.90 |
| 15 | 7.1 ± 0.22 | 10.3↓ ± 1.00 | 25.9↓ ± 1.94 | 21.2↓ ± 3.48 |
| 16 | 2.0 ± 0.44 | 4.2↓ ± 0.41 | 5.7↓ ± 0.55 | 7.1↓ ± 1.59 |
| 17 | 4.2 ± 0.18 | 6.2↓ ± 0.55 | 24.0↓ ± 1.60 | 28.7↓ ± 3.89 |
| 18 | 5.8 ± 0.56 | 6.8↓ ± 0.17 | 34.5↓ ± 3.01 | 83.8↓ ± 5.37 |
| 19 | 2.8 ± 0.09 | 5.3↓ ± 0.45 | 5.9↓ ± 0.17 | 12.9↓ ± 1.68 |
| 20 | 5.3 ± 0.20 | 9.7↓ ± 0.94 | 21.7↓ ± 1.34 | 32.8↓ ± 2.64 |
| 21 | 8.2 ± 0.24 | 12.9↓ ± 1.12 | 17.2↓ ± 1.17 | 35.1↓ ± 3.54 |
| 22 | 5.1 ± 0.15 | 7.3↓ ± 0.20 | 135.3↓ ± 12.73 | 77.3↓ ± 4.07 |
| 23 | 2.4 ± 0.21 | 4.5↓ ± 0.36 | 3.4↓ ± 0.20 | 5.5↓ ± 1.26 |
| Avg | **3.58** ± 1.96 | 5.61 ± 2.84 | 16.12 ± 27.47 | 18.18 ± 21.77 |
| • *p* | | 0.000 | 0.035 | 0.003 |
| • w/l/d | | 2/21/0 | 3/20/0 | 1/22/0 |
| • s.w/s.l | | 2/19 | 2/19 | 1/21 |

**TABLE 10** Average Likelihood Ratio (*SIG*) of Rules with Standard Deviations

| # | APRIORI-SD *SIZE* ± *sd* | APRIORI-C *SIZE* ± *sd* | RIPPER *SIZE* ± *sd* | CN2 *SIZE* ± *sd* |
|---|---|---|---|---|
| 1 | 8.4 ± 0.04 | 2.3↓ ± 0.03 | 2.8↓ ± 0.18 | 2.0↓ ± 0.05 |
| 2 | 14.2 ± 0.02 | 3.1↓ ± 0.14 | 1.5↓ ± 0.55 | 2.7↓ ± 0.10 |
| 3 | 8.2 ± 0.02 | 2.8↓ ± 0.05 | 3.3↓ ± 0.12 | 2.1↓ ± 0.01 |
| 4 | 9.8 ± 0.15 | 2.9↓ ± 0.06 | 1.8↓ ± 0.05 | 2.4↓ ± 0.06 |
| 5 | 16.4 ± 0.06 | 2.5↓ ± 0.07 | 2.3↓ ± 0.19 | 2.0↓ ± 0.01 |
| 6 | 10.4 ± 0.03 | 2.4↓ ± 0.04 | 1.8↓ ± 0.75 | 1.9↓ ± 0.03 |
| 7 | 11.0 ± 0.05 | 2.5↓ ± 0.02 | 2.6↓ ± 0.02 | 2.0↓ ± 0.02 |
| 8 | 5.2 ± 0.06 | 2.7↓ ± 0.03 | 1.9↓ ± 0.08 | 1.9↓ ± 0.09 |
| 9 | 10.5 ± 0.12 | 2.8↓ ± 0.06 | 2.8↓ ± 0.60 | 2.7↓ ± 0.03 |
| 10 | 4.2 ± 0.04 | 1.8↓ ± 0.08 | 2.3↓ ± 0.65 | 1.4↓ ± 0.04 |
| 11 | 1.9 ± 0.02 | 2.1 ± 0.03 | 1.8 ± 0.04 | 2.0 ± 0.04 |
| 12 | 7.5 ± 0.03 | 2.9↓ ± 0.06 | 2.4↓ ± 0.04 | 1.9↓ ± 0.03 |
| 13 | 15.3 ± 0.05 | 2.3↓ ± 0.18 | 2.5↓ ± 0.64 | 2.1↓ ± 0.00 |
| 14 | 15.3 ± 0.03 | 3.4↓ ± 0.02 | 2.1↓ ± 0.13 | 2.5↓ ± 0.08 |
| 15 | 15.2 ± 0.17 | 3.2↓ ± 0.04 | 3.3↓ ± 0.48 | 2.5↓ ± 0.05 |
| 16 | 12.0 ± 0.03 | 3.3↓ ± 0.06 | 1.5↓ ± 0.02 | 2.6↓ ± 0.04 |
| 17 | 5.6 ± 0.06 | 3.0↓ ± 0.04 | 3.1↓ ± 0.06 | 2.7↓ ± 0.03 |
| 18 | 25.0 ± 0.07 | 2.3↓ ± 0.05 | 2.1↓ ± 0.33 | 1.5↓ ± 0.00 |
| 19 | 2.5 ± 0.08 | 1.2↓ ± 0.06 | **3.0**↑ ± 0.60 | 1.0↓ ± 0.07 |
| 20 | 29.5 ± 0.19 | 2.2↓ ± 0.12 | 2.7↓ ± 0.66 | 1.5↓ ± 0.00 |
| 21 | 16.8 ± 0.05 | 2.6↓ ± 0.04 | 1.5↓ ± 0.33 | 2.4↓ ± 0.02 |
| 22 | 26.4 ± 0.18 | 3.5↓ ± 0.07 | 2.8↓ ± 0.07 | 2.6↓ ± 0.04 |
| 23 | 13.5 ± 0.04 | 2.1↓ ± 0.02 | 2.5↓ ± 0.05 | 2.0↓ ± 0.07 |
| Avg | **12.37** ± 7.26 | 2.60 ± 0.55 | 2.36 ± 0.55 | 2.11 ± 0.46 |
| • *p* | | 0.000 | 0.000 | 0.000 |
| • w/l/d | | 1/22/0 | 1/22/0 | 1/22/0 |
| • s.w/s.l | | 0/22 | 1/21 | 0/22 |

**TABLE 11** Average Unusualness (*WRACC*) of Rules with Standard Deviations

| # | APRIORI-SD *WRACC* ± *sd* | APRIORI-C *WRACC* ± *sd* | RIPPER *WRACC* ± *sd* | CN2 *WRACC* ± *sd* |
|---|---|---|---|---|
| 1 | 0.045 ± 0.08 | 0.039↓ ± 0.10 | 0.030↓ ± 0.09 | 0.022↓ ± 0.09 |
| 2 | 0.038 ± 0.03 | 0.037 ± 0.03 | 0.036 ± 0.04 | 0.034↓ ± 0.04 |
| 3 | 0.023 ± 0.08 | 0.013↓ ± 0.09 | − 0.014↓ ± 0.09 | − 0.016↓ ± 0.08 |
| 4 | 0.043 ± 0.05 | 0.040 ± 0.03 | 0.021↓ ± 0.04 | 0.020↓ ± 0.04 |
| 5 | 0.043 ± 0.06 | 0.038↓ ± 0.06 | 0.021↓ ± 0.06 | 0.013↓ ± 0.06 |
| 6 | 0.040 ± 0.07 | 0.031↓ ± 0.08 | **0.062**↑ ± 0.08 | **0.058**↑ ± 0.07 |
| 7 | 0.036 ± 0.02 | 0.036 ± 0.02 | 0.019↓ ± 0.02 | 0.012↓ ± 0.02 |
| 8 | 0.048 ± 0.05 | 0.041 ± 0.05 | 0.029↓ ± 0.05 | 0.026↓ ± 0.04 |
| 9 | 0.030 ± 0.09 | 0.025↓ ± 0.07 | 0.012↓ ± 0.08 | 0.004↓ ± 0.07 |
| 10 | 0.008 ± 0.03 | 0.006 ± 0.04 | **0.017**↑ ± 0.04 | **0.013**↑ ± 0.04 |
| 11 | 0.043 ± 0.03 | 0.041 ± 0.02 | 0.043 ± 0.03 | 0.041 ± 0.02 |
| 12 | 0.039 ± 0.04 | 0.032↓ ± 0.04 | 0.027↓ ± 0.03 | 0.024↓ ± 0.04 |
| 13 | 0.044 ± 0.03 | 0.040 ± 0.03 | 0.034↓ ± 0.03 | 0.024↓ ± 0.03 |
| 14 | 0.044 ± 0.10 | 0.040 ± 0.10 | 0.009↓ ± 0.10 | 0.009↓ ± 0.10 |
| 15 | 0.045 ± 0.08 | 0.040 ± 0.08 | 0.017↓ ± 0.08 | 0.015↓ ± 0.07 |
| 16 | 0.046 ± 0.01 | 0.041↓ ± 0.01 | 0.023↓ ± 0.01 | 0.017↓ ± 0.00 |
| 17 | 0.085 ± 0.01 | 0.081 ± 0.02 | 0.014↓ ± 0.02 | 0.005↓ ± 0.03 |
| 18 | 0.043 ± 0.06 | 0.038↓ ± 0.07 | 0.018↓ ± 0.07 | 0.009↓ ± 0.06 |
| 19 | 0.122 ± 0.07 | 0.113↓ ± 0.08 | 0.015↓ ± 0.07 | 0.007↓ ± 0.07 |
| 20 | − 0.022 ± 0.01 | − 0.023 ± 0.01 | − **0.008**↑ ± 0.01 | **0.004**↑ ± 0.01 |
| 21 | 0.045 ± 0.07 | 0.035↓ ± 0.08 | 0.020↓ ± 0.07 | 0.015↓ ± 0.08 |
| 22 | 0.052 ± 0.02 | 0.046↓ ± 0.03 | 0.008↓ ± 0.02 | 0.001↓ ± 0.03 |
| 23 | 0.139 ± 0.00 | 0.137 ± 0.00 | 0.038↓ ± 0.00 | 0.033↓ ± 0.01 |
| Avg | **0.047** ± 0.03 | 0.042 ± 0.03 | 0.021 ± 0.02 | 0.017 ± 0.02 |
| • *p* | | 0.000 | 0.001 | 0.000 |
| • w/l/d | | 0/22/1 | 4/19/0 | 3/20/0 |
| • s.w/s.l | | 0/11 | 3/18 | 3/19 |

**TABLE 12** Accuracy (*ACC*) of rulesets with standard deviations

| # | APRIORI-SD *ACC*±*sd* | APRIORI-C *ACC*±*sd* | RIPPER *ACC*±*sd* | CN2 *ACC*±*sd* |
|---|---|---|---|---|
| 1 | 87.26 ± 7.80 | 89.99 ± 8.29 | 82.03↓ ± 7.40 | 81.62↓ ± 3.55 |
| 2 | 97.48 ± 8.90 | 95.85 ± 9.36 | 94.76↓ ± 8.60 | 92.28↓ ± 1.07 |
| 3 | 86.02 ± 7.97 | 87.17 ± 8.08 | 86.17 ± 7.80 | 82.45↓ ± 3.89 |
| 4 | 96.16 ± 8.62 | 94.52 ± 9.13 | **98.90**↑ ± 8.95 | 94.18 ± 3.71 |
| 5 | 75.00 ± 7.41 | 74.95 ± 7.32 | 71.29↓ ± 6.53 | 72.77↓ ± 9.33 |
| 6 | 67.90 ± 5.84 | **71.66**↑ ± 6.51 | 67.85 ± 6.31 | 68.71 ± 1.79 |
| 7 | 69.52 ± 5.98 | 71.19 ± 6.22 | **72.52**↑ ± 6.62 | **72.40**↑ ± 7.60 |
| 8 | 79.83 ± 7.08 | 79.57 ± 7.22 | 69.88↓ ± 6.02 | 74.10↓ ± 4.15 |
| 9 | 82.30 ± 7.46 | 82.44 ± 7.73 | 81.36 ± 7.82 | 80.74 ± 7.59 |
| 10 | 99.91 ± 9.11 | 99.20 ± 9.82 | 99.16 ± 8.99 | 98.58 ± 0.60 |
| 11 | 88.97 ± 8.70 | **92.44**↑ ± 8.71 | 86.34 ± 8.50 | **91.44**↑ ± 6.62 |
| 12 | 95.19 ± 9.33 | 95.59 ± 8.65 | 96.01 ± 9.60 | 91.33↓ ± 2.04 |
| 13 | 79.40 ± 7.30 | **81.72**↑ ± 7.69 | 76.51↓ ± 6.79 | 80.87 ± 1.32 |
| 14 | 79.17 ± 7.01 | **81.08**↑ ± 7.89 | 74.21↓ ± 7.05 | 72.28↓ ± 2.81 |
| 15 | 75.21 ± 7.21 | **80.15**↑ ± 7.23 | **85.79**↑ ± 8.44 | **98.01**↑ ± 0.60 |
| 16 | 96.47 ± 8.79 | 94.63 ± 9.07 | 93.47↓ ± 9.04 | 94.24 ± 0.39 |
| 17 | 75.13 ± 7.21 | 77.93 ± 6.86 | **79.48**↑ ± 7.87 | 74.71 ± 8.62 |
| 18 | 85.21 ± 7.11 | 84.47 ± 7.64 | **92.73**↑ ± 8.50 | **89.82**↑ ± 5.33 |
| 19 | 66.49 ± 6.23 | 66.09 ± 6.05 | 64.84 ± 5.70 | 60.60↓ ± 1.83 |
| 20 | 19.98 ± 1.89 | 18.25 ± 1.37 | **86.42**↑ ± 7.92 | **58.88**↑ ± 5.70 |
| 21 | 68.21 ± 6.54 | **71.89**↑ ± 7.17 | **89.39**↑ ± 7.96 | **88.73**↑ ± 3.01 |
| 22 | 75.58 ± 6.32 | **81.01**↑ ± 7.47 | **78.85**↑ ± 6.90 | 69.18↓ ± 8.92 |
| 23 | 93.23 ± 9.18 | 91.64 ± 8.29 | 91.51 ± 8.51 | 89.16↓ ± 1.33 |
| Avg | 79.98 ± 16.67 | 81.02 ± 16.50 | **83.46** ± 10.24 | 81.61 ± 11.66 |
| • *p* | | 0.039 | **0.282** | 0.489 |
| • w/l/d | | 13/10/0 | **10/13/0** | 8/15/0 |
| • s.w/s.l | | 7/0 | **8/7** | 6/10 |

**TABLE 13** Area Under the ROC curve (*AUC*) of Rulesets with Standard Deviations

| # | APRIORI-SD *AUC* ± *sd* | APRIORI-C *AUC* ± *sd* | RIPPER *AUC* ± *sd* | CN2 *AUC* ± *sd* |
|---|---|---|---|---|
| 1 | 84.14 ± 2.00 | 82.11↓ ± 2.01 | 83.22 ± 7.80 | 33.39↓ ± 5.61 |
| 2 | 88.99 ± 3.05 | **91.50**↑ ± 3.11 | **90.07**↑ ± 8.02 | **90.74**↑ ± 3.57 |
| 3 | 81.15 ± 2.03 | **85.96**↑ ± 2.03 | **84.14**↑ ± 7.59 | **84.51**↑ ± 0.15 |
| 4 | 90.79 ± 3.02 | 90.97 ± 3.00 | 88.94↓ ± 7.91 | **96.22**↑ ± 2.55 |
| 5 | 76.94 ± 4.06 | 76.25 ± 4.09 | 76.34 ± 7.41 | 71.33↓ ± 7.86 |
| 6 | 66.48 ± 1.05 | 67.18 ± 1.08 | 63.27↓ ± 5.69 | **70.53**↑ ± 5.99 |
| 7 | 74.25 ± 4.25 | 70.98↓ ± 4.08 | 66.95↓ ± 6.66 | 71.99↓ ± 5.76 |
| 8 | 85.13 ± 2.07 | 75.47↓ ± 2.09 | 72.70↓ ± 6.84 | 74.17↓ ± 5.35 |
| 9 | 84.08 ± 3.06 | 78.86↓ ± 3.03 | 79.58↓ ± 7.82 | 78.81↓ ± 4.64 |
| 10 | 93.16 ± 4.00 | **97.29**↑ ± 4.00 | **96.36**↑ ± 9.48 | **96.22**↑ ± 2.31 |
| 11 | 90.09 ± 2.08 | 75.58↓ ± 2.04 | 88.52 ± 8.28 | **94.46**↑ ± 1.52 |
| 12 | 90.82 ± 2.00 | 89.83 ± 2.01 | 90.20 ± 8.99 | **99.17**↑ ± 0.23 |
| 13 | 78.84 ± 3.10 | 77.50 ± 3.12 | **75.31**↑ ± 7.25 | 83.20↓ ± 8.68 |
| 14 | 72.32 ± 3.08 | **77.98**↑ ± 3.00 | 74.30 ± 6.98 | **75.06**↑ ± 6.13 |
| 15 | 71.69 ± 3.68 | 62.90↓ ± 4.02 | 63.42↓ ± 6.29 | **97.90**↑ ± 0.36 |
| 16 | 96.00 ± 1.06 | 94.38 ± 1.05 | 88.42↓ ± 8.43 | 96.88 ± 1.67 |
| Avg | **82.80** ± 8.70 | 80.92 ± 9.95 | 80.11 ± 10.23 | 82.16 ± 16.81 |
| • *p* | | 0.190 | 0.027 | 0.871 |
| • w/l/d | | 6/10/0 | 4/12/0 | 11/5/0 |
| • s.w/s.l | | 4/6 | 4/7 | 9/6 |