# CorClass: Correlated Association Rule Mining for Classification

Albrecht Zimmermann and Luc De Raedt

Institute of Computer Science, Machine Learning Lab, Albert-Ludwigs-University Freiburg, Georges-Köhler-Allee 79, 79110 Freiburg, Germany
{deraedt,azimmerm}@informatik.uni-freiburg.de

**Abstract.** A novel algorithm, CorClass, that integrates association rule mining with classification, is presented. It first discovers all correlated association rules (adapting a technique by Morishita and Sese) and then applies the discovered rule sets to classify unseen data. The key advantage of CorClass, as compared to other techniques for *associative classification*, is that CorClass directly finds the associations rules for classification by employing a branch-and-bound algorithm. Previous techniques (such as CBA [1] and CMAR [2]) first discover all association rules satisfying a minimum support and confidence threshold and then post-process them to retain the best rules.
CorClass is experimentally evaluated and compared to existing associative classification algorithms such as CBA [1], CMAR [2] and rule induction algorithms such as Ripper [3], PART [4] and C4.5 [5].

## 1 Introduction

Rule discovery is popular in the field of machine learning and data mining. Rule learning techniques have largely focussed on finding compact sets of rules that can be used for classification. On the other hand, approaches within data mining have concentrated on finding all rules that satisfy a set of constraints (typically based on support and confidence). Recently, these two approaches have been integrated in that a number of different research groups have developed tools for classification based on association rule discovery (associative classification). These tools such as CBA [1] and CMAR [2] in a first phase typically discover all association rules and then post-process the resulting rule sets in order to retain only a small number of suitable rules. Several studies [6, 1, 2] have shown that the obtained rule sets often perform better or comparable to those obtained using more traditional rule learning algorithms such as PART[4], Ripper[3] or CN2 [7].

We present a novel approach to associative classification, called CorClass (correlated classification). CorClass directly aims at finding the $k$ best correlated association rules for classification, adapting a technique due to Morishita and Sese[8]. The advantage is that this is not only more efficient (no post-processing is necessary) but also more elegant in that it is a direct approach. To the best of

the authors' knowledge, it is the first time that Morishita and Sese's framework is being applied to discovery of classification rules.

When using association rules for classification, one needs to employ a strategy for combining the predictions of different association rules. In [6] several approaches have been evaluated and compared to existing techniques.

The remainder of the paper is structured as follows. In the next section the basic terminology used throughout the paper is introduced. In section 3 we explain how an extension of Morishita's approach to a single multi-valued target attribute can be derived and explain the algorithm used for mining the rules. Different strategies for combining derived rules following [6] are introduced in section 4. In section 5, the experimental setup and results will be presented and we conclude in section 6.

## 2 Preliminaries

### 2.1 Association Rule Mining

Let $\mathcal{A} = \{A_1, ..., A_d\}$ a set of attributes, $\mathcal{V}[A] = \{v_1, ..., v_j\}$ the domain of attribute $A$, $\mathcal{C} = \{c_1, ..., c_t\}$ a set of possible class values for class attribute $C$. An instance $e$ is then a tuple $\langle v_1, ..., v_d \rangle$ with $v_i \in \mathcal{V}[A_i]$. The multiset of instances $\mathcal{E} = \{e_1, ..., e_n\}$ is called a data set.

**Definition 1** ***(Literal)*** *A* ***literal*** *is an attribute-value-pair of the form $A = v, v \in \mathcal{V}[A]$. An instance $\langle v_1, ..., v_d \rangle$ satisfies a literal $A_i = v$ iff $v_i = v$.*

**Definition 2** ***(Association rule)*** *An* ***association rule*** *$r$ is of the form $b_r \Rightarrow h_r$ with $b_r$ of the form $l_1 \wedge ... \wedge l_i$ called the* ***rule body*** *and $h_r$ of the form $l'_1 \wedge ... \wedge l'_j$, $b_r \cap h_r = \emptyset$ the* ***rule head****. An instance $e$ satisfies $b_r$ iff it satisfies all literals in $b_r$ and it satisfies $r$ iff it satisfies $h_r$ as well.*

**Definition 3** ***(Support and Confidence)*** *For a given rule $r$ of the form $b_r \Rightarrow h_r$,*

$$sup(r) = |\{e \mid e \in \mathcal{E}, e \text{ satisfies } r\}|$$

*is called the* ***support*** *of $r$ on $D$,*

$$conf(r) = \frac{sup(r)}{|\{e \mid e \in \mathcal{E}, e \text{ satisfies } b_r\}|}$$

*the* ***confidence*** *of $r$.*

### 2.2 Association Rules for Classification

When learning in a classification setting, one starts from a databases of instances and a target class attribute. The goal is then to induce a set of rules with the class attribute in the rule head. In this context, only association rules relating the

rule body to a certain class value are of interest. In the literature on associative classification the term *class association rule* has been introduced to distinguish such rules from regular association rules whose head may consist of an arbitrary conjunction of literals.

**Definition 4** ***(Class Association Rule)*** *A* ***class association rule*** $r$ *is of the form* $b_r \rightarrow h_r$ *with* $b_r$ *of the form* $l_1 \wedge ... \wedge l_r$ *called the* ***rule body*** *and* $h_r$ *of the form* $C = c_i$ *the* ***rule head****, where* $C$ *is the class attribute.*

Satisfaction of a rule is treated as above. For an unclassified example satisfying $b_r$ the rule predicts class $c_i$. Support and confidence are defined as above. For convenience we will write $b_r \Rightarrow c$ for $b_r \Rightarrow C = c$ and $sup(c)$ for $sup(C = c)$ from now on. Note that *confidence* is called *accuracy* in the context of rule induction.

Rule sets can then be applied to classify examples. To this aim, one combines the predictions of all rules whose body satisfies the example. If there is only one rule, then the corresponding value in the head is predicted; if there is no rule body satisfying the example, then a default class is predicted; and if there are multiple rule bodies satisfying the example, then their predictions must be combined. Various strategies for realizing this are discussed in section 4.

Many different rule learning algorithms are presented in the literature, cf. [7, 5, 4, 3]. A relatively recent approach to rule learning is *associative classification*, which combines well-developed techniques for association rule mining such as [9, 10] with post-processing techniques. Several such techniques are discussed in the next section.

## 3 Methodology

### 3.1 Current Approaches to Associative Classification

In association rule mining the usual framework calls for the user to specify two parameters, $sup_{min}$ and $conf_{min}$. The parameter $sup_{min}$ defines the minimum number of instances that have to satisfy a rule $r$ for $r$ to be considered interesting. Similarly $conf_{min}$ sets a minimum confidence that a rule has to achieve.

In [1], the first approach to associative classification, rules are mined with $sup_{min} = 1\%$ and $conf_{min} = 50\%$ using *apriori* [9]. Once all class association rules satisfying these constraints are found, the rules are ranked based on confidence, support and generality and an obligatory pruning step based on database coverage is performed. Rules are considered sequentially in descending order and training instances satisfying a rule are marked. If a rule classifies at least one instance correctly, the instances satisfying it are removed. An optional pruning step based on a pessimistic error estimate [5], can also be performed. The resulting set of classification rules is used as an ordered decision list.

In [2], the mining process uses similar support and confidence thresholds and employs *FP-growth* [10]. Pruning is again performed after mining all rules and involves a database coverage scheme, and also removes all rules that do not correlate positively with the class attribute. Classification of an example is

decided by a weighted combination of the values of rules it satisfies. The weight is derived by calculating the $\chi^2$-value of the rules and normalized by the maximum $\chi^2$-value a rule *could* have to penalize rules favoring minority classes. For a more thorough discussion see [2].

In [6], both *apriori* (with similar parameters to the CBA setting) and *predictive apriori* [11] are employed. In *predictive apriori*, the confidence of rules is corrected by support and rules with small support are penalized, since those will probably not generalize well. For a more in-depth explanation we refer the reader to the work of Scheffer [11]. The resulting rule sets are employed both pruned (in the manner used by CBA) and unpruned. Classification is decided using different schemes that are described in section 4

There are a few problems with the support-confidence framework. If the minimal support is too high, highly predictive rules will probably be missed. Is the support threshold set too low, the search space increases vastly, prolonging the mining phase, and the resulting rule set will very likely contain many useless and redundant rules. Deciding on a good value of $sup_{min}$ is therefore not easy. Also, since the quality of rules is assessed by looking at their confidence, highly specialized rules will be preferred. Previous works [11, 2] attempt to correct this by penalizing minority rules during the mining and prediction step respectively. Finally, for the support based techniques it is necessary to mine **all** rules satisfying the support- and confidence-thresholds and extract the set of prediction rules in a post-processing step.

Correlation measures have, as far as we know, so far not been used as search criteria. The reason lies in the fact that measures such as *entropy gain*, $\chi^2$ etc. are neither anti-monotonic nor monotonic, succinct or convertible and therefore do not lend themselves as pruning criteria very well. Morishita et al in [8] introduced a method for calculating upper bounds on the values attainable by specializations of the rule currently considered. This effectively makes convex correlation measures anti-monotonic and therefore pruning based on their values possible.

Correlation measures quantify the difference between the conditional probability of the occurence of a class in a subpopulation and the unconditional occurence in the entire data set. They typically normalize this value with the size of the population considered. This means that predictive rules will be found that are not overly specific. Additionally setting a threshold for pruning rules can be based on significance assessments, which may be more intuitive than support. The upper bound finally allows dynamic raising of the pruning threshold, differing from the fixed minimal support used in existing techniques. This will result in earlier termination of the mining process. Since the quality criterion for rules is used directly for pruning, no post-processing of the discovered rule set is necessary.

In the next section we will briefly sketch Morishita's approach and outline how to extend it to a multi-value class variable. This will then be used to find the rules sets for classification.

**Table 1.** Contingency table for $b_r \Rightarrow c$

| | $c$ | $\neg c$ | |
|---|---|---|---|
| $b_r$ | $sup(r) = y$ | $sup(b_r \Rightarrow \neg c)$ | $sup(b_r) = x$ |
| $\neg b_r$ | $sup(\neg b_r \Rightarrow c)$ | $sup(\neg b_r \Rightarrow \neg c)$ | $sup(\neg b_r)$ |
| | $sup(c) = m$ | $sup(\neg c)$ | $n$ |

### 3.2 Correlation measures and Convexity

Let $n = |\mathcal{E}|$, $m = sup(c)$ for a given class value $c$ and $x = sup(b_r)$, $y = sup(r)$ for given rule $r$ of the form $b_r \Rightarrow c$. A contingency table reflecting these values is shown in table 1. Since virtually all correlation measures quantify the difference between expected and observed distributions, such measures $\sigma$ can, for fixed $\mathcal{E}$, be viewed as functions $f(x, y) : \mathbb{N}^2 \to \mathbb{R}$.

The tuple $\langle x, y\rangle$ characterizing a rule's behavior w.r.t. a given data set is called a *stamp point*.

The set of *actual* future stamp points $S_{act}$ of refinements of $r$ is unknown until these refinements are created and evaluated on the data set. But the current stamp point bounds the set of *possible* future stamp points $S_{poss}$. For the 2-dimensional case they fall inside the parallelogram defined by the vertices $\langle 0, 0\rangle$, $\langle y, y\rangle$, $\langle x - y, 0\rangle$, $\langle x, y\rangle$, cf. [8].

Quite a few correlation functions are convex ($\chi^2$, *information gain, gini index, interclass variance, category utility, weighted relative accuracy* etc.). Convex functions take their extreme values at the points on the convex hull of their domain. So by evaluating $f$ at the convex hull of $S_{poss}$ (the vertices of the parallelogram mentioned above), it is possible to obtain the extreme values bounding the values of $\sigma$ attainable by refinements of the $r$. Furthermore, since $\langle 0, 0\rangle$ characterizes a rule that is not satisfied by a single instance and $\langle x, y\rangle$ a rule of higher complexity that conveys no additional information, the upper bound of $\sigma(r')$ for any specialization $r'$ of $r$ is calculated as $max\{f(x-y, 0), f(y, y)\}$. The first of the two terms evaluates a rule that covers no example of the class $c$ and in a two-class problem therefore perfectly classifies $\neg c$, while the second term evaluates a rule that has 100% accuracy with regard to $c$.

### 3.3 Multi-Valued Target Attribute

A similar contingency table to the one shown in table 1 can be constructed for more than two classes (which cannot be treated as $c$ and $\neg c$ anymore). An example for such a table is shown in table 2.

The body defined by the vertices $\langle 0, 0, 0\rangle$, $\langle x - y_1, 0, y_2\rangle$, $\langle x - y_2, y_1, 0\rangle$, $\langle x - (y_1 + y_2), 0, 0\rangle$, $\langle y_1 + y_2, y_1, y_2\rangle$, $\langle y_2, 0, y_2\rangle$, $\langle y_1, y_1, 0\rangle$, $\langle x, y_1, y_2\rangle$ encloses all stamp points that can be induced by specializations of the current rule. This can of course be extended to a higher number of classes. It has to be kept in mind though that $2^t$ number of points will be induced for $t$ classes.

**Table 2.** Contingency table for three class values

| | $c_1$ | $c_2$ | $c_3$ | |
|---|---|---|---|---|
| $b_r$ | $sup(b_r \Rightarrow c_1) = y_1$ | $sup(b_r \Rightarrow c_2) = y_2$ | $sup(b_r \Rightarrow c_3) = x - (y_1 + y_2)$ | $sup(b_r) = x$ |
| $\neg b_r$ | $sup(\neg b_r \Rightarrow c_1)$ | $sup(\neg b_r \Rightarrow c_2)$ | $sup(\neg b_r \Rightarrow c_3)$ | $sup(\neg b_r) = n - x$ |
| | $sup(c_1) = m_1$ | $sup(c_2) = m_2$ | $sup(c_3) = n - (m_1 + m_2)$ | $n$ |

Similar to the 2-dimensional approach, an upper bound on the future value of class association rules derived from the current rule can be calculated. For the remainder of this paper we will refer to this upper bound for a given rule as $ub_\sigma(r)$. Following the argument for the 2-dimensional case, the tuples $\langle 0, 0, 0\rangle$ and $\langle x, y_1, y_2\rangle$ can be ignored.

### 3.4 The CorClass Algorithm

The upper bound allows for two types of pruning w.r.t. the actual mining process. First, the user can specify a threshold which might be based on e.g. significance for $\chi^2$. It should be easier to find a meaningful threshold than it is when looking for a minimal support that does not lead to the creation of too many rules but at the same time captures many interesting and useful ones. Second, the goal can be to mine for a user specified maximum number $k$ of rules in which case the threshold is raised dynamically. We will only describe the $k$-best algorithm here since deriving the threshold-based algorithm should be straightforward. The algorithm is listed in figure 1.

The algorithm starts from the most general rule body (denoted by $\top$). We use an optimal refinement operator (denoted by $\rho$ in the listing). This refinement operator is defined as follows :

**Definition 5** ***(Optimal Refinement Operator)*** *Let $\mathcal{L}$ a set of literals, $\prec$ a total order on $\mathcal{L}$, $\tau \in \mathbb{R}$.*

$$\rho(r) = \{r \wedge l_i \mid l_i \in \mathcal{L}, ub_\sigma(l_i) \geq \tau, \forall l \in r : l \prec l_i\}$$

*is called an* ***optimal refinement operator****.*

The operator guarantees that all rules can be derived from $\top$ in exactly one possible way. So, no rule is generated more than once. Since only literals are added whose upper bound exceeds the threshold, the value of the refinement has a chance of exceeding or matching the current threshold, if $ub(r) \geq \tau$.

In each iteration of the algorithm, the rule body with the highest upper bound is selected for refinement. The support counts $x$ and $y_i$ for the resulting rules are computed and the score and the upper bound is calculated.

**Fig. 1.** The CorClass Algorithm

Input: Dataset $\mathcal{E}$, $k \in \mathbb{N}$, $C = \{c_1, ..., c_t\}$
Output: The $k$ rules with highest $\sigma$-values on $\mathcal{E}$

$P := \{\top\}$, $S := \emptyset$, $\tau := -\infty$
**while** $|P| \geq 1$
  $p_{mp} = argmax_{p \in P}\ ub_\sigma(p)$
  $P := P \setminus \{p_{mp}\}$
  $\forall b_i \in \rho(p_{mp})$
    **if** $k > |S|$ **then** $S \cup \{b_i\}$
    **elsif** $qual(b_i)$ **then**
      $S := S \setminus \{s_j \mid argmin_{s_j \in S} \sigma(s_j)\} \cup \{b_i\}$
      $\tau = min_{s \in S}\ \sigma(s)$
    **endif**
  $P := \{p \mid p \in P, ub_\sigma(p) \geq \tau\} \cup \{b_i \mid ub_\sigma(b_i) \geq \tau\}$
**return** $S$

Now decisions have to be made about:

1. Raising the threshold $\tau$.
2. Including the current rule in the temporary solution set $S$.
3. Including the current rule in the set of promising rules $P$ i.e. the set of candidates for future refinement.

Decisions 1. and 2. are actually combined since looking up the threshold is implemented by returning the score of the lowest-ranked rule $r_k$ currently in the solution set. The decision on whether to include the current rule $r$ in the temporary solution set is based on three criteria. First, $\sigma(r)$ has to exceed or match $\sigma(r_k)$ of course. Second, if there already is a rule with the same $\sigma$-value the question is, whether it is a generalization of $r$. So, $r$ is only included if it has a different support, since otherwise it includes at least one literal giving no additional information. This decision is denoted by the predicate ***qual*** in the algorithm listing (figure 1). If $r$ is included, $r_k$ is removed and the threshold raised to $\sigma(r_{k-1})$. After processing all refinements the set of promising rules is pruned by removing all rules with $ub_\sigma(r_i) < \sigma(r_k)$. Also, all refinements whose upper bound exceeds the threshold are included in the promising set. In this way, a branch-and-bound algorithm is realized that never traverses parts of the search space known to lead to suboptimal rules.

The algorithm terminates once there is no candidate remaining for refinement. During mining, the rules in the solution set are already ranked by (1) score, (2) generality, and (3) support. The majority class of all instances satisfying a rule is set as the head. Should not all training instances be covered, the majority class of the remaining ones is set as the default class.

## 4 Combination Strategies

When applying a set of discovered rules for classifying an example, it may be necessary to combine the predictions of multiple rules satisfying the example or to resolve the conflicts between them. Various strategies for realizing this have been considered. We discuss the approaches we evaluate below, following [6].

### 4.1 Decision List

The most straightforward approach is to use the list of rules created as a decision list (since rules are ranked by quality according to some criterion) and use the first rule satisfied by an example for classification. This approach is chosen by CBA [1]. To compare the quality of the rules found by our solution to CBA's, we will use this approach as well.

### 4.2 Weighted Combination

The alternative consists of using a weighted combination of all rules being satisfied by an as yet unseen example for classification. The general way to do this is to collect all such rules, assign each one a specific weight and for each $c_i$ predicted by at least one rule sum up the weights of corresponding rules. The class value having the highest value is returned. In the literature on associative classification [6, 2], several approaches for conflict resolution are explored. Among these are:

- Majority Voting [6]: all rules get assigned the weight 1.
- Linear Weight Voting [6]: a rule $r$ is assigned the weight

$$1 - \frac{rank(r)}{rank_{max} + 1},$$

  where $rank(r)$ is the rank a rule has in the list returned by the algorithm and $rank_{max}$ the number of rules returned in total.
- Inverse Weight Voting [6]: a rule $r$ is assigned the weight

$$\frac{1}{rank(r)}$$

- In [2] a heuristic called *weighted-*$\chi^2$ is proposed. Since some improvement with regard to the decision list approach is reported, we use this weight as well in our experiments.

Even though other conflict resolution strategies, e.g. *naive Bayes* [12] and *double induction* [13], have been proposed, we do not employ these strategies in our experiments. The main reason for this is that our focus was on comparing our rule discovery approach to established techniques and we use [6] as reference work.

## 5 Experiments

To compare our approach to existing techniques, we use the results of the experiments by Stefan Mutter, published in [6]. We also compare CorClass to CMAR and for this comparison use the results published in [2]. Our choice of data sets is somewhat limited by the need for comparison with existing work. We used *information gain*, $\chi^2$ and *category utility* [14] as $\sigma$. The columns reporting on the classification accuracy of a metric are headed *Gain*,$\chi^2$ and *CU* respectively.

Predictive accuracies in [6] are estimated using stratified 10-fold cross validation followed by discretization of numerical attributes using Fayyad and Irani's MDL method [15]. To facilitate a fair comparison we choose the same evaluation method for CorClass. In the experiments, $k$ was set to 1000 for CorClass.

### 5.1 Results

The accuracies reported in [6] were obtained using rule sets of size 10–1000. In addition different pruning techniques were evaluated. Due to space restrictions we used the best accuracy *apriori* and *predictive apriori* obtained, regardless of rule set size or pruning method, for comparison. The results for the different combination strategies are shown in tables 3,4,5 and 6. CBA results are considered in the table that reports on the results for decision lists since it uses a (pruned) decision list. The values for CBA are also taken from [6]. Note that the authors report that they have not been able to reproduce the results in [1], neither using the executable obtained from the authors of [1] nor using their own re-implementation.

The tables list the average predictive accuracy and the standard deviation for each combination of correlation measure and conflict resolution strategy. **Bold** values denote the best accuracy for the respective data set and combination strategy.

**Table 3.** Comparison of Obtained Accuracies using the Inverted Weighting Scheme

| Dataset | *Apriori* | $Apriori_{pred}$ | *Gain* | $\chi^2$ | *CU* |
|---|---|---|---|---|---|
| Balance | 71.66±5.85 | 70.55±4.52 | 70.55±5.06 | **74.88±4.71** | 74.56±4.49 |
| Breast-w | 65.47±0.47 | 88.94±8.85 | 94.56±2.84 | **94.99±2.05** | **94.99±2.05** |
| Heart-h | 63.95±1.43 | **83.01±6.25** | 79.95±8.67 | 79.95±8.67 | 79.95±8.67 |
| Iris | 92±6.89 | 92±6.89 | 95.33±5.49 | **96±4.66** | 94±6.62 |
| Labor | 71.67±16.72 | 79.33±19.55 | **82.38±13.70** | 77.61±12.28 | 77.61±12.28 |
| Lenses | 66.67±30.43 | 70±28.11 | 71.67±31.48 | **75±42.49** | 71.67±41.61 |
| Pima | 65.11±0.36 | **74.35±3.95** | 74.21±6.02 | 74.21±6.02 | 74.08±7.02 |
| Tic-Tac-Toe | 65.34±0.43 | **96.67±1.67** | 75.66±5.15 | 75.03±4.61 | 75.03±4.61 |

**Table 4.** Comparison of Obtained Accuracies using the Linear Weighting Scheme

| Dataset | *Apriori* | $Apriori_{pred}$ | *Gain* | $\chi^2$ | *CU* |
|---|---|---|---|---|---|
| Balance | 68.8±6.13 | 73.92±5.2 | 70.55±5.06 | **74.88±4.71** | 74.55±4.49 |
| Breast-w | 65.47±0.47 | 65.47±0.47 | **96.99±2.08** | 96.85±1.89 | 96.85±1.89 |
| Heart-h | 63.95±1.43 | 77.17±5.52 | 81.67±9.58 | **81.97±7.71** | **81.97±7.71** |
| Iris | 64±11.42 | 91.33±6.32 | 94.67±5.26 | **95.33±5.49** | **95.33±5.49** |
| Labor | 64.67±3.22 | 77±15.19 | **82.38±13.70** | **82.38±13.70** | **82.38±13.70** |
| Lenses | 63.33±32.2 | 70±28.11 | **75±32.63** | 71.67±41.61 | 71.67±41.61 |
| Pima | 65.11±0.36 | 66.54±2 | 74.84±4.28 | **75.37±5.43** | **75.37±5.43** |
| Tic-Tac-Toe | 65.34±0.43 | **75.78±1.37** | 74.69±4.22 | 75.13±3.92 | 75.13±3.92 |

**Table 5.** Comparison of Obtained Accuracies using Majority Vote

| Dataset | *Apriori* | $Apriori_{pred}$ | *Gain* | $\chi^2$ | *CU* |
|---|---|---|---|---|---|
| Balance | **76.46±4.81** | 73.92±5.41 | 71.83±7.76 | 75.04±4.66 | 74.72±4.46 |
| Breast-w | 88.55±4.03 | 92.51±9.95 | 97.28±1.58 | **97.42±1.63** | **97.42±1.63** |
| Heart-h | **83.02±6.35** | 82.34±8.54 | 82.69±7.33 | 82.01±7.27 | 82.01±7.27 |
| Iris | 92.67±7.34 | 91.33±9.45 | **96±4.66** | **96±4.66** | **96±4.66** |
| Labor | 76±20.05 | **86.67±15.32** | 84.76±15.26 | 84.29±11.97 | 84.28±12.97 |
| Lenses | **68.33±33.75** | **68.33±33.75** | 65±38.85 | **68.33±43.35** | **68.33±43.35** |
| Pima | 72.14±4.51 | **74.49±6.3** | 74.45±3.48 | 73.93±3.87 | 73.93±3.87 |
| Tic-Tac-Toe | **98.01±1.67** | 92.48±3.09 | 76.18±6.58 | 74.72±4.79 | 74.72 ±4.79 |

**Table 6.** Comparison of Obtained Accuracies using a Decision List

| Dataset | *CBA* | *Apriori* | $Apriori_{pred}$ | *Gain* | $\chi^2$ | *CU* |
|---|---|---|---|---|---|---|
| Balance | 71.5±5.97 | 71.5±5.97 | 71.5±5.97 | 70.55±5.06 | **74.88±4.71** | 74.72±4.71 |
| Breast-w | **95.13±3.03** | 88.83±3.73 | 88.65±8.7 | 94.13±2.08 | 94.56±2.42 | 94.56±2.42 |
| Heart-h | 80.63±7.2 | 82.67±6.43 | **83.36±5.26** | 78.26±9.39 | 77.23±10.97 | 77.23±10.97 |
| Iris | 92.67±6.63 | 91.33±6.32 | 91.33±6.32 | **95.33±5.49** | 84±10.04 | 84±10.04 |
| Labor | 79±19.5 | 79.33±21.07 | 79.33±21.07 | **82.38±13.71** | **82.38±13.71** | **82.38±13.71** |
| Lenses | 66.67±30.43 | 66.67±30.43 | 70±28.11 | **71.67±31.48** | **71.67±41.61** | **71.67±41.61** |
| Pima | 74.1±4.48 | 73.83±4.97 | **74.22±4.67** | 71.99±7.07 | 73.93±3.87 | 73.93±3.87 |
| Tic-Tac-Toe | **99.06±1.25** | 97.39±1.8 | 97.28±1.66 | 76.08±5.76 | 75.24±4.68 | 75.24±4.68 |

As can be seen, CorClass on almost all occasions achieves best accuracy or comes very close. This is most noticeable for the Breast-w and Iris data sets on which the different CorClass versions perform very well for all combination strategies. The only set on which CorClass constantly achieves worse accuracy than the other techniques is the Tic-Tac-Toe set. This is caused by the fact that a single literal already restricts coverage quite severely without having any discriminative power. To reliably predict whether $x$ will win, one needs at least three literals. So, the discriminative rules are penalized by correlation measures because their coverage is low.

Results derived using the *weighted-*$\chi^2$ heuristic are compared to the accuracies reported in [2] (table 7). It is not entirely clear how the accuracy estimates in [2] were obtained.

The only data set among these on which CMAR performs noticeably better than CorClass is again the Tic-Tac-Toe set. It is interesting to see, that using the *weighted-*$\chi^2$ heuristic improves CorClass' performance on the Tic-Tac-Toe data set strongly when compared to the other weighting schemes.

**Table 7.** Comparison of Obtained Accuracies using the *Weighted-*$\chi^2$ Scheme

| Dataset | *CMAR* | *Gain* | $\chi^2$ | *CU* |
|---|---|---|---|---|
| Breast-w | **96.4** | 96.13±2.44 | 96.13±2.44 | 96.13±2.44 |
| Heart-h | 82.2 | 81.33±9.18 | **82.3±7.13** | **82.3±7.13** |
| Iris | 94 | **96±4.66** | 94.67±8.19 | 94.67±8.19 |
| Labor | **89.7** | 84.76±15.26 | 87.14±16.71 | 87.14±16.71 |
| Pima | 75.1 | 75.76±4.88 | **75.89±5.11** | **75.89±5.11** |
| Tic-Tac-Toe | **99.2** | 86.42±4.51 | 88.72±4.4 | 88.72±4.4 |

Finally, in table 8 we compare the best results obtained by CorClass to several standard machine learning techniques, namely *C4.5*, *PART* and *Ripper* as implemented in WEKA [16]. CorClass compares well to those techniques as well. It outperforms them on the Breast-w and Labor data sets, achieves competitive results for the Heart-h, Iris and Pima sets and is outperformed on Balance, Lenses and Tic-Tac-Toe, while still achieving reasonable accuracy.

### 5.2 Discussion

Generally, CorClass performs well on the data sets considered, and achieves for all but the Tic-Tac-Toe data set repeatedly the best accuracy.

It is interesting to see that there is no single correlation measure that clearly outperforms the other two, even though $\chi^2$ performs slightly better. There is also no ideal combination strategy for combining the mined rules for classification. The majority vote performs surprisingly well (giving rise to the best results for CorClass on five of the eight data sets), considering that it is the least complex

**Table 8.** Comparison of CorClass to standard rule learning techniques

| Dataset | *Gain* | $\chi^2$ | *CU* | *C4.5* | *PART* | *JRip* |
|---|---|---|---|---|---|---|
| Balance | 71.83±7.76 | 75.04±4.66 | 74.72±4.46 | 76.65 | **83.54** | 80.80 |
| Breast-w | 97.28±1.58 | **97.42±1.63** | **97.42±1.63** | 94.69 | 94.26 | 94.13 |
| Heart-h | **82.69±7.33** | 82.3±7.13 | 82.3±7.13 | 81.07 | 81.02 | 78.95 |
| Iris | **96±4.66** | **96±4.66** | **96±4.66** | **96** | 94 | 94.67 |
| Labor | 84.76±15.26 | **87.14±16.71** | **87.14±16.71** | 73.67 | 78.67 | 77 |
| Lenses | 75±32.63 | 75±42.49 | 71.67±41.61 | **81.67** | **81.67** | 75 |
| Pima | 74.84±4.28 | **75.89±5.11** | **75.89±5.11** | 73.83 | 75.27 | 75.14 |
| Tic-Tac-Toe | 86.42±4.51 | 88.72±4.4 | 88.72±4.4 | 85.07 | 94.47 | **97.81** |

of the weighting schemes. This indicates that even rules of relatively low rank still convey important information and do not have to be discounted by one of the more elaborate weighting schemes.

All experiments ran in less than 30 seconds on a 2 GHz Pentium desktop PC with 2 GB main memory, running Linux. The number of *candidate rules* considered by CorClass during mining was on average smaller than the number of *discovered* rules (before pruning) reported in [6]. This difference is probably more pronounced for smaller rule sets since *apriori*-like approaches have to mine all rules satisfying the support and confidence constraints and post-process them while CorClass can stop earlier. We plan to investigate this issue more thoroughly in the future.

## 6 Conclusion and Future Work

We have introduced CorClass, a new approach to associative classification. CorClass differs from existing algorithms for this task insofar, that it does not employ the classical association rule setting in which mining is performed under support and confidence constraints. Instead it directly maximizes correlation measures such as *information gain*, $\chi^2$ and *category utility*. This removes the need for post-processing the rules to obtain the set of actual classification rules.

In the experimental comparison, we evaluated the different correlation measures as well as several strategies for combining the mined rule sets for classification purposes. We compare our algorithm to existing techniques and standard rule learning algorithms. In the experiments, CorClass repeatedly achieves best accuracy, validating our approach.

While the results are promising, there is still room for improvement. As the results on the Tic-Tac-Toe data set show, rules with relatively low coverage but high discriminative power tend to be penalized by our approach. For data sets whose instances are divided in more than two or three classes, this problem will be more pronounced since rules separating, e.g., half of the classes from the other half will achieve a high score but have low classification power. A

possible solution to this problem could be to calculate a separate $\sigma$-value and upper bound for each such class for a given rule body, thus coming back to the 2-dimensional setting Morishita introduced. During rule mining the best of these values would determine the inclusion in the solution set and promising set respectively. Such an approach would also allow for the usage of heuristics such as *weighted relative accuracy*, cf. [17]. A second approach could lie in maximizing the *weighted*-$\chi^2$ criterion directly since the experimental results showed that it offsets the selection bias of the correlation measures somewhat. We plan to extend our technique in these directions.

Since CorClass performs well when compared to the traditional heuristic rule learning algorithms such as Ripper and PART, we hope that CorClass may inspire further work on the use of globally optimal methods in rule learning.

## Acknowledgments

This work was partly supported by the EU IST project cInQ (Consortium on Inductive Querying).

This work was strongly inspired by the master's thesis of Stefan Mutter. We sincerely would like to thank him and Mark Hall for allowing us to use their experimental results for comparison purposes. We would also like to thank Andreas Karwath and Kristian Kersting for their helpful suggestions. Finally we thank the anonymous reviewers for their constructive comments on our work.

## References

1. Liu, B., Hsu, W., Ma, Y.: Integrating classification and association rule mining. In Agrawal, R., Stolorz, P.E., Piatetsky-Shapiro, G., eds.: KDD, New York City, New York, USA, AAAI Press (1998) 80–86
2. Li, W., Han, J., Pei, J.: CMAR: Accurate and efficient classification based on multiple class-association rules. In Cercone, N., Lin, T.Y., Wu, X., eds.: Proceedings of the 2001 IEEE International Conference on Data Mining, San José, California, USA, IEEE Computer Society (2001) 369–376
3. Cohen, W.W.: Fast effective rule induction. In Prieditis, A., Russell, S.J., eds.: ICML 1995, Tahoe City, California, USA, Morgan Kaufmann (1995) 115–123
4. Frank, E., Witten, I.H.: Generating accurate rule sets without global optimization. In Shavlik, J.W., ed.: ICML 1998, Madison, Wisconsin, USA, Morgan Kaufmann (1998) 144–151
5. Quinlan, J.R.: C4.5: Programs for Machine Learning. Morgan Kaufmann (1993)
6. Mutter, S.: Classification using association rules. Master's thesis, Albert-Ludwigs-Universität Freiburg/University of Waikato, Freiburg, Germany/Hamilton, New Zealand (2004)
7. Clark, P., Niblett, T.: The CN2 induction algorithm. Machine Learning **3** (1989) 261–283
8. Morishita, S., Sese, J.: Traversing itemset lattices with statistical metric pruning. In: PODS, Dallas, Texas, USA, ACM (2000) 226–236

9. Agrawal, R., Srikant, R.: Fast algorithms for mining association rules in large databases. In: Proceedings of the 20th International Conference on Very Large Databases, Santiago de Chile, Chile, Morgan Kaufmann (1994) 487–499
10. Han, J., Pei, J., Yin, Y.: Mining frequent patterns without candidate generation. In: Proceedings of the 2000 ACM SIGMOD International Conference on Management of Data, Dallas, Texas, USA, ACM (2000) 1–12
11. Scheffer, T.: Finding association rules that trade support optimally against confidence. In De Raedt, L., Siebes, A., eds.: PKDD 2001. Lecture Notes in Computer Science, Freiburg, Germany, Springer (2001) 424–435
12. Boström, H.: Rule Discovery System User Manual. Compumine AB. (2003)
13. Lindgren, T., Boström, H.: Resolving rule conflicts with double induction. In: Proceedings of the 5th International Symposium on Intelligent Data Analysis. Lecture Notes in Computer Science, Berlin, Germany, Springer (2003) 60–67
14. Gluck, M.A., Corter, J.E.: Information, uncertainty, and the utility of categories. In: Proceedings of the 7th Annual Conference of the Cognitive Science Society, Irvine, California, USA, Lawrence Erlbaum Associate (1985) 283–287
15. Fayyad, U.M., Irani, K.B.: Multi-interval discretization of continuous-valued attributes for classification learning. In: Proceedings of the 13th International Joint Conference on Artificial Intelligence, Chambéry, France, Morgan Kaufmann (1993) 1022–1029
16. Frank, E., Witten, I.H.: Data Mining: Practical Machine Learning Tools and Techniques with Java Implementations. Morgan Kaufmann (1999)
17. Todorovski, L., Flach, P.A., Lavrač, N.: Predictive performance of weighted relative accuracy. In Zighed, D.A., Komorowski, H., Zytkow, J.M., eds.: PKDD 2000. Lecture Notes in Computer Science, Lyon, France, Springer (2000) 255–264