# An Evaluation of Approaches to Classification Rule Selection

Frans Coenen and Paul Leng
Department of Computer Science,
The University of Liverpool,
Liverpool, L69 3BX
{frans,phl}@csc.liv.ac.uk

## Abstract

*In this paper a number of Classification Rule evaluation measures are considered. In particular the authors review the use of a variety of selection techniques used to order classification rules contained in a classifier, and a number of mechanisms used to classify unseen data. Four established ordering mechanisms are considered: the standard support/confidence framework, a weighted relative accuracy measure, Laplace accuracy and $\chi^2$ testing. In addition the authors propose a fifth ordering strategy founded on the size of the antecedent so that specific rules are "fired" before more general rules. The techniques to classify unseen data are grouped into three categories: (i) select best first rule (according to ordering), (ii) average best $K$ rules (again according to ordering), and (iii) consider all rules in classifier. The authors demonstrate that rule ordering founded on the size of antecedent works well given certain conditions.*

**Keywords***: Classification rule evaluation measures.*

## 1. Introduction

Classification Association Rule Mining (CARM) is based on the observation that a subset of the Association Rules (ARs) generated by Association Rule Mining (ARM) algorithms can effectively be used for the purpose of classification [2]. ARs [1] are rules of the form $A \Rightarrow C$ where $A$ and $C$ (Antecedent and Consequent) are disjoint subsets of a set of binary valued attributes defined by the input data set. In the case of Classification Rules (CRs) the variable $C$ is a unary subset of a set of classes defined with respect to the input data. Consequently, given a particular data set, the set of possible CRs will be a subset of the set of possible ARs. The notation $r.A$ and $r.C$ will be used to indicate respectively the antecedent and consequent (class) of a rule $r$.

CARM algorithms (broadly) can be categorised into two groups according to the way that the CRs are generated:

1. **Two stage algorithms** where a set of CARs (Classification Association Rules) are produced first (stage 1), which are then pruned and placed into a classifier (stage 2). Examples of this approach include CMAR [10] and CBA [11].
2. **Integrated algorithms** where the classifier is produced in a single processing step. Examples of this approach include Apriori-TFPC [7], and induction systems such as FOIL [12], PRM and CPAR [13].

Once the classifier has been established (usually presented in the form of a list of rules), regardless of the methodology used to generate it, there are a number of proposed mechanisms for using the resulting classifier to classify unseen data. These can be itemised as follows (given a particular case):

1. **Best Rule:** Select the first "best" rule that satisfies the given case according to some ordering imposed on the rule listing. The ordering can be defined according to many different ordering schemes. Four established schemes are:
   (a) **CSA:** Combinations of confidence, support and size of antecedent, with confidence being the most significant factor. Example CARM systems that use CSA ordering include CBA [11] and Apriori-TFPC [7], and (only during the early stages of processing) CMAR [10]).
   (b) **WRA:** The weighted relative accuracy which reflects a number of rule "interestingness" measures as proposed in [9].
   (c) **Laplace Accuracy** Laplace accuracy [4] measures as used in FOIL [12], and PRM and CPAR [13].

(d) $\chi^2$ **Testing:** $\chi^2$ values as used, in part, in CMAR [10].

An alternative to CSA, that has bot been considered in the literature to date is to make use of the size of the antecedent as the most significant factor followed by confidence and support, i.e. **ACS** ordering.

2. **Best** $K$ **rules**: Select the first best K rules that satisfy the given case and then select a rule according to some averaging process. "Best" in this case is defined according to an imposed ordering of the form described in 1. This technique is used, for example, in CPAR [13].

3. **All rules**: Collect all rules in the classifier that satisfy the given case and then evaluate this collection to identify a class. One well known evaluation method in this category is *Weighted* $\chi^2$ (WCS) testing as used in CMAR [10].

Note that a rule $r$ satisfies a case $n$ if $r.A \subset n$, the class to be assigned to $N$ will then be $r.C$.

In this paper we compare the above satisfaction/ordering techniques which are described in some further detail in the following two Sections. The evaluation of the techniques is carried out using a number of sets of CRs, generated using the Apriori-TFPC [7] CARM algorithm which is briefly reviewed in Section 4. The evaluation is discussed in detail in Section 5 and some conclusions offered in Section 6.

## 2. Rule Ordering

As noted in the introduction to this paper five different rule ordering strategies are considered here, four established strategies and one new strategy. Each is described in more detail in the following five sub-sections.

### 2.1. CSA ordering

The confidence-Support framework remains very common amongst current ARM algorithms. Given a CR, $r$, the support for $r$ ($sup(r)$) is the proportion of occurrences of the set $r.A \cup r.C$ in the input data compared to the number of records in the input. The confidence of $r$ ($conf(r)$) is then given as $sup(r)/sup(r.A)$. Using the support and confidence framework algorithms such as Apriori-TFPC [7], CBA [11] and CMAR [10] make use of support and confidence values in their rule ordering mechanism.

CSA (Confidence, Support, size of Antecedent) ordering is defined as follows:

1. **Confidence**: A rule $r_1$ has priority over a rule $r_2$ if $conf(r_1) > conf(r_2)$.

2. **Support**: A rule $r_1$ has priority over a rule $r_2$ if $conf(r_1) = conf(r_2)$ and $sup(r_1) > sup(r_2)$.

3. **Size of antecedent**: A rule $r_1$ has priority over a rule $r_2$ if $conf(r_1) = conf(r_2)$, $sup(r_1) = sup(r_2)$ and $|r_1.A| < |r_2.A >|$.

Given that confidence values are normally calculated as real numbers it is unusual to have CRs with identical confidence values other than in the case of "100% confidence rules". Where 100% rules are found the associated supported value "comes into play". The size of the antecedent is seldom used in CSA ordering.

It should be noted that CARM algorithms that use the support/confidence framework, and this includes ACS and systems founded on WRA and Laplace accuracy as discussed in the following Sub-sections, usually also make use of user defined minimum support/confidence threshold values during the CR generation process. Consequently rules contained in the final classifier will be such that $sup(r) > min\ support\ threshold$ and $conf(r) > min\ conf\ threshold$.

### 2.2. Weighted Relative Accuracy (WRA)

The use of WRA (Weighted Relative Accuracy) was proposed in [9] as a unifying mechanism for determining CR expected accuracy. WRA was specifically designed as a rule ordering mechanism that reflects a number of rule measures. The idea is that the WRA measure is a synthesis of a number of rule "interestingness" measures. The WRA for a rule $r$ is calculated as follows:

$$wra(r) = weighting \times relative\ accuracy$$

Where relative accuracy is equal to

$$confidence(r) - sup(r.C)$$

The term "relative" is used in the sense that the support for a rule is compared with its expected support — a concept not dissimilar to the idea under-pinning $\chi^2$ testing (see below). A negative relative accuracy indicates that the accuracy associated with $r$ is less than the fixed rule $true \Rightarrow C$. So that rules with a low "generality" (i.e. a low $sup(r.A)$ value) are not given a high accuracy measure the support value for the rule antecedent is used to weight the relative accuracy measure. Thus:

$$wra(r) = sup(r.A) \times (confidence(r) - sup(r.C))$$

In [9] it is demonstrated that the WRA measure, as calculated above, is equivalent to a number of alternative rule measures, namely:

1. The weighted relative sensitivity,
2. weighted relative specificity, and
3. weighed relative negative reliability.

WRA also reflects a measure of novelty, where novelty is defined as $sup(r) - (sup(r.A) \times sup(r.C))$ [9].

## 2.3. Laplace Accuracy

The use of the *Laplace expected accuracy estimate* with respect to classification rule generation was first proposed by Clark and Boswell [4]. It has subsequently been used in a number of CARM algorithms to order lists of CRs (for example CPAR [13]). In [13] the measure is defined, given a rule $r$, as follows:

$$Laplace\ accuracy(r) = \frac{sup(r.A \cup r.C) + 1}{sup(r.A) + k)}$$

where $k$ is the number of classes. Note that in this case support is defined as the actual number of records (in the training or test set) that contain $r.A \cup r.C$ or $r.A$.

## 2.4. $\chi^2$ Testing

$\chi^2$ testing is a well known statistical technique (see for example [8]) used to determine whether two variables are independent of one another by comparing a set of *observed values* ($O$) against a set of *expected values* ($E$) — values that would be expected if there were no association between the variables. A $\chi^2$ value is calculate using the identity:

$$\chi^2 = \sum_{1 \leq i \leq n} (0_i - E_i)^2 / E_i$$

where $n$ is the number of observed/expected values (this is always 4 in the case of CARM). If the result is above a given *critical threshold value* then it can be said that a relationship between the variables exists, otherwise there is no relationship. A good description of the use of $\chi^2$ testing with respect to CARM is given in [10]; for completeness an alternative description, in the form of an example, is given below. To determine whether a $\chi^2$ value is significant or not a *critical value* for $\chi^2$ must be known. Such critical values are usually published in tables with *significance level* along the X-axis (expressed as a percentage) and the *degrees of freedom* ($DF$) along the Y-axis. In the case of classification rule mining $DF$ is always going to be 1. For CMAR a significance level of 5% was used giving a threshold value of 3.8415 (this value has also been used in this paper).

### 2.4.1 $\chi^2$ Testing Example

Given a rule $r$ such that: $sup(r) = 6$, $sup(r.A) = 8$, $sup(r.C) = 12$ and $N$ (the number of records in the training set) is 32, a contingency table (referred to by some authors as a confusion matrix) of observed values can be produced as follows:

| | $C$ | $\neg C$ | |
|---|---|---|---|
| $A$ | $sup(r)$ | $sup(r.A) - sup(r)$ | $sup(r.A)$ |
| $\neg A$ | $sup(r.C) - sup(r)$ | $N - sup(r.A) - sup(r.C) - sup(r)$ | $N - sup(r.A)$ |
| | $sup(r.C)$ | $N - sup(r.C)$ | N |

which will, in the above case give:

| | $C$ | $\neg C$ | |
|---|---|---|---|
| $A$ | 6 | 2 | 8 |
| $\neg A$ | 6 | 18 | 24 |
| | 12 | 20 | 32 |

The expected values are then calculated thus:

| | $C$ | $\neg C$ | |
|---|---|---|---|
| $A$ | $(12 * 8)/32 = 3$ | $(20 * 8)/32 = 5$ | 8 |
| $\neg A$ | $(12 * 24)/32 = 9$ | $(20 * 24)/32 = 15$ | 24 |
| | 12 | 20 | 32 |

The Chi-squared value is then:

$$\chi^2 = \sum_{1 \leq i \leq 4} (0_i - E_i)^2 / E_i = 6.4$$

This is above the critical value of 3.8415 and therefore it can be said that, in this example, the rule is significant.

## 2.5. ACS or Specificity ordering

In this paper it is proposed that a good alternative ordering to CSA (as described above) is ACS ordering (size of Antecedent, Confidence and Support) which is defined as follows:

1. **Size of antecedent**: A rule $r_1$ has priority over a rule $r_2$ if $|r_1.A| > |r_2.A >|$.
2. **Confidence**: A rule $r_1$ has priority over a rule $r_2$ if $|r_1.A| = |r_2.A >|$ and $conf(r_1) > conf(r_2)$.
3. **Confidence**: A rule $r_1$ has priority over a rule $r_2$ if $|r_1.A| = |r_2.A > |$, $conf(r_1) = conf(r_2)$ and $sup(r_1) > sup(r_2)$.

The intuition behind this ordering is that more specific rules should be "triggered" before more general rules are attempted. For example we may have a classifier, ordered using CSA, comprising two rules as follows:

| # | Rule | Conf. |
|---|---|---|
| 1 | $\{ab\} \Rightarrow \{y\}$ | 80% |
| 2 | $\{abcd\} \Rightarrow \{x\}$ | 75% |

Given a case $\{abc\}$ this would be classified, using "best first" case satisfaction, as belonging to class $y$ when intuitively class $x$ would be more likely to be the correct class. ACS ordering thus ensures that specific rules have a higher precedence than more general rules so that in the above example the class $x$ would be returned. It will be demonstrated, in Section 5, that ACS ordering works well if a high confidence threshold value is used.

## 3 Classification

In the introduction to this paper three alternative case/record classification mechanisms were identified. Two of these, "best k" and "$\chi^2$ testing" are briefly discussed below so as to provide some necessary further detail ("best first" has the obvious interpretation).

### 3.1. Best $K$ Testing

The intuition behind "best $k$ testing" is that 'one cannot expect that any single rule can perfectly predict the class label for every example satisfying its body' [13]. Given a case $n$ to be classified the best $K$ approach is as follows:

1. Obtain all rules that satisfy $n$.
2. Keep only best $K$ rules for each class, or all rules if there are less than $K$ rules for a particular class
3. For each group determine some average expected value to be maximised (e.g. confidence, size of antecedent, Laplace accuracy, $\chi^2$ value).
4. Select the class associated with the best average.

In [13] a value of 5 was suggested as an appropriate value for $K$.

### 3.2. Weighted $\chi^2$ Testing

Weighted $\chi^2$ Testing is used in a number of CARM algorithms, such as CMAR [10], to classify data by considering entire groups of rules that satisfy a given case. With respect to CMAR, given a case $n$ to be classified, the procedure is as follows:

1. Collect all rules that satisfy $n$, and
   (a) If consequents of all rules are identical, or only one rule is found, classify case according to the consequents.
   (b) Else group rules according to class and determine the combined effect of the rules in each group. The class associated with the "strongest group" is then selected.

The strength of a group is calculate using the WCS (Weighted $\chi$ Squared) value. As reported in [10] this is done by first defining a Maximum $\chi^2$ (MCS) value for each rule $A \Rightarrow C$:

$$MCS = \left( min(\chi^2(A), \chi^2(C)) - \frac{\chi^2(A)\chi^2(C)}{N} \right)^2 \times N \times e$$

where the function $min$ returns the lesser of its two arguments, $N$ is the number of records in the test set, and e is calculate as follows:

$$e = \frac{1}{\chi^2(A)\chi^2(C)} + \frac{1}{\chi^2(A)N - \chi^2(C)} + \frac{1}{N - \chi^2(A)\chi^2(C)} + \frac{1}{(N - \chi^2(A))(N - \chi^2(C)}$$

For each group of rules the Weighted $\chi^2$ (WCS) value is defined as:

$$WCS = \sum_{1 \leq i \leq g} (\chi_i^2 \times \chi_i^2)/(MCS)$$

where $g$ is the number of CRs in the group. The class associated with the group of rules with the highest WCS value is selected as the class to be allocated to the case.

## 4. Apriori-TFPC

To evaluate the above approaches a number of variations of the Apriori-TFPC [7] algorithm were created, each reflecting one of the identified approaches. The Apriori-TFPC classification rule generation algorithm is founded on the Apriori-TFP (Total From Partial) Association Rule Mining algorithm [6] ; which, in turn, is an extension of the Apriori-T (Apriori Total) ARM algorithm [5]. All three algorithms, Apriori-T, Apriori-TFP and Apriori-TFPC, were developed by the authors[1].

[1] Apriori-T, Apriori-TFP and Apriori-TFPC my be obtained from http://www.csc.liv.ac.uk/ frans/KDD/Software.

Apriori-T is an "apriori" style algorithm (such as first proposed by Agrawal and Srikant [1]) designed to process a binary valued input data set so as to identify frequent (large) itemsets and store the resulting frequent itemset information in a "reverse" set enumeration tree called a T-tree (Total support tree). This T-tree can then be processed to identify ARs.

Apriori-TFP proceeds in a similar manner to Apriori-T except that, instead of operating with the raw input data directly, the input data is first preprocessed and placed in a P-tree (Partial support tree). As such Apriori-TFP has all the functionality of Apriori-T with the additional advantage that it operates in a much more efficient manner. Advantages which are particularly significant when operating with data that features many duplicate records and/or records with duplicate leading sub-strings (in this respect attribute ordering is advantageous).

Apriori-TFPC is an extension of Apriori-TFP designed to produce Classification Association Rules (CARs) whereas Apriori-T and Apriori-TFP were designed to generate ARs. Apriori-TFPC is different to many other CARM algorithms in that it does not use the "standard" two stage approach: (1) generate all CARs, (2) prune CARs to produce a classifier. Instead Apriori-TFPC comprises only a single stage where CRs are generated as part of the "frequent set identification process". As the T-tree is developed nodes in branches whose root represents a classifier are tested for their appropriateness as classification rules. To date this has been achieved with respect to a confidence threshold; however, to evaluate the different techniques considered in this paper this can equally well be achieved using a $\chi^2$ threshold, Laplace accuracy or a WRA measure. Once it has been established that a node represents a suitable CR the node is not processed any further. Nodes are also pruned during the generation process according to a user supplied support threshold. The nature of the rules produced is therefore very much dependent on the threshold value used.

A high level view of the Apriori-TFP CR generation algorithm is presented in Table 1 where $N_k$ indicates the set of level $K$ T-tree nodes and $CN_k$ the set of candidate level $K$ T-tree nodes. The constants $minsup$ and $minconf$ indicate the minimum support and confidence thresholds supplied by the user. Remember that although the algorithm compares a calculated confidence value to a threshold value when deciding whether to add a rule to $R$ or not, this could equally well be a Laplace or WRA accuracy value or a $\chi^2$ value.

To prevent the T-tree from growing too large an additional test (not shone in Table 1) is included in the generation process in that if the number of nodes in the T-tree is greater than 80,000; on completion of a level, no further levels are generated.

On completion of the generation process a default rule is also identified; this is the class associated with the last rule in $R$. The last $m$ rules whose class is equivalent to the default class are then removed from $R$ and replaced with a default rule.

**Algorithm: Apriori-TFPC**
**input** The set of classes $C$, and
input data stored in P-tree
**output** A list of rules $R$

$K \leftarrow 1$
Generate $CN_K$
Determine support for $CN_K$ with reference to P-tree
Prune $CN_K$ according to $minsup$ to give $N_K$

$K \leftarrow 2$
**Start loop**
Generate $CN_K$ from $B_{K-1}$ nodes
**if** $CN_K = \emptyset$ **break**
Determine support for $CN_K$ with ref. to P-tree
Prune $CN_K$ according to $minsup$ to give $N_K$
**Start loop**
For each $n$ in $N_K$ where node root is
in $C$, generate a candidate CR $r$
**if** $conf(r) > minconf$
$R \leftarrow R \cup r$
Remove $n$ from T-tree
**End loop**
$K \leftarrow K + 1$
**End loop**

**Table 1** *Apriori-TFPC algorithm*

The different ordering and case satisfaction techniques considered in this paper, and itemised in the foregoing sections can be combined into eleven different variations of Apriori-TFPC as follows:

1. **Best First and CSA**: T-tree pruning using support and confidence thresholds, rule ordering using CSA, case satisfaction using best first.
2. **Best First and ACS**: T-tree pruning using support and confidence thresholds, rule ordering using ACS, case satisfaction using best first.
3. **Best First and WRA**: T-tree pruning using support and WRA thresholds (WRA threshold set to 0), rule ordering using WRA, case satisfaction using best first.
4. **Best First and Laplace**: T-tree pruning using support and Laplace thresholds, rule ordering using Laplace accuracy, case satisfaction using best first.
5. **Best First and $\chi^2$**: T-tree pruning using support and $\chi^2$ thresholds ($\chi^2$ threshold set to 3.8415, assuming a significance level of 5%, and a "degree of freedom" of 1), rule ordering using $\chi^2$, case satisfaction using best first.

6. **Best $K$ and CSA**: T-tree pruning using support and confidence thresholds, rule ordering using CSA, case satisfaction using best $K$.

7. **Best $K$ and ACS**: T-tree pruning using support and confidence thresholds, rule ordering using ACS, case satisfaction using best $K$.

8. **Best $K$ and WRA**: T-tree pruning using support and WRA thresholds (WRA threshold set to 0), rule ordering using WRA, case satisfaction using best $K$.

9. **Best $K$ and Laplace**: T-tree pruning using support and Laplace thresholds, rule ordering using Laplace accuracy, case satisfaction using best $K$.

10. **Best $K$ and $\chi^2$**: T-tree pruning using support and $\chi^2$ thresholds ($\chi^2$ threshold set to 3.8415, assuming a significance level of 5%, and a "degree of freedom" of 1), rule ordering using $\chi^2$, case satisfaction using best $K$.

11. **All and WCS**: T-tree pruning using support and $\chi^2$ thresholds ($\chi^2$ threshold set to 3.8415, assuming a significance level of 5%, and a "degree of freedom" of 1), case statisfaction using all rules that satisfy a given case and WCS (rule ordering irrelevant).

In the case of experiments using "best K" techniques $K$ was set to 5.

## 5. Evaluation

Experiments were conducted using a range of data sets taken from the the UCI Machine Learning Repository [3]. The chosen datasets were discretized using the LUCS-KDD DN software[2], where appropriate continuous attributes were ranged using five sub-ranges. The experiments were run on a 1.2 GHz Intel Celeron CPU with 512 Mbyte of RAM running under Red Hat Linux 7.3.

The first set of evaluations undertaken used a confidence threshold value of 50% and a support threshold value of 1% (as also used in the published evaluations of CMAR [10] and CBA [11]). The results are presented in Table 2 where the best accuracy obtained for each of the data sets is highlighted in bold print. The row labels describe the key characteristics of each data set, in the form which it was discretised. For example, the label $adult.D131.N48842.C2$ denotes the "adult" data set, which includes 48842 records in 2 classes, with attributes that for the experiments described here have been discretised into 131 binary categories.

| Data | *Best first* | | | | | *Best K first* | | | | | *All* |
|---|---|---|---|---|---|---|---|---|---|---|---|
| Set | *CSA* | *ACS* | *WRA* | *Lap.* | $\chi^2$ | *CSA* | *ACS* | *WRA* | *Lap.* | $\chi^2$ | *WCS* |
| adult.D131.N48842.C2 | 76.1 | 76.1 | 65.2 | 76.1 | 32.8 | 76.0 | **76.3** | 57.8 | 76.1 | 33.7 | 76.1 |
| anneal.D106.N798.C6 | **85.5** | **85.5** | 68.4 | 83.7 | 59.1 | 79.4 | 79.5 | 69.2 | 82.2 | 42.3 | 80.7 |
| auto.D142.N205.C7 | 12.7 | 19.6 | **54.9** | 48.0 | 19.6 | 13.7 | 13.7 | 32.4 | 43.1 | 12.7 | 20.6 |
| breast.D47.N699.C2 | **98.0** | **98.0** | 81.1 | 96.6 | 83.1 | 93.4 | 93.4 | 72.2 | 91.7 | 88.5 | **98.0** |
| connect4.D129.N67557.C3 | 65.8 | 65.8 | 37.3 | 65.8 | 61.1 | 65.8 | 65.8 | 62.4 | 65.8 | **65.9** | 65.8 |
| glass.D52.N214.C7 | **49.5** | 38.3 | 34.6 | 46.7 | 13.1 | 42.1 | 42.1 | 26.2 | 43.0 | 13.1 | 13.1 |
| heart.D53.N303.C5 | 49.7 | 54.3 | **57.6** | 55.0 | 55.0 | 53.0 | 53.0 | 56.3 | 55.0 | 55.0 | 55.0 |
| hepatitis.D58.N155.C2 | 71.4 | 79.2 | 67.5 | 79.2 | 20.8 | **80.5** | **80.5** | 70.1 | 79.2 | 28.6 | 61.0 |
| horseColic.D94.D368.C2 | 67.9 | 78.2 | **83.7** | 77.8 | 62.5 | 70.7 | 70.1 | 78.3 | 66.8 | 51.1 | 62.5 |
| ionosphere.D172.N351.C2 | 81.7 | **91.4** | 76.0 | 87.4 | 84.6 | 77.1 | 88.0 | 62.9 | 82.9 | 69.7 | 62.9 |
| iris.D23.N150.C3 | **94.7** | **94.7** | 93.3 | 93.3 | 93.3 | 92.0 | 92.0 | 93.3 | 92.0 | 89.3 | **94.7** |
| led7.D24.N3200.C10 | 67.1 | 57.2 | 32.9 | 66.6 | 26.0 | 66.2 | 66.3 | 31.9 | 66.8 | 19.1 | **73.8** |
| letRecog.D106.N20000.C26 | **44.2** | 34.9 | 16.9 | 42.4 | 28.1 | 41.0 | 41.2 | 8.1 | 41.3 | 25.6 | 38.1 |
| mushroom.D127.N8124.C2 | **96.0** | 89.1 | 47.5 | 46.7 | 88.4 | 71.1 | 69.7 | 66.9 | 69.8 | 85.6 | 53.1 |
| nursery.D32.N12960.C5 | 80.0 | 74.4 | 70.6 | 80.0 | 70.6 | 69.8 | 69.9 | 70.3 | 70.6 | 68.6 | **85.3** |
| pageBlocks.D55.N5473.C5 | **89.8** | **89.8** | 79.6 | **89.8** | 3.4 | **89.8** | **89.8** | 68.5 | **89.8** | 5.4 | 2.0 |
| penDigits.D90.N10992.C10 | **79.5** | 40.6 | 33.1 | 78.1 | 49.9 | 54.4 | 54.8 | 35.2 | 55.0 | 46.8 | 70.8 |
| pimaIndians.D42.N768.C2 | 74.2 | 74.2 | **76.0** | 74.5 | 65.1 | 71.3 | 73.7 | 66.9 | 70.6 | 62.0 | 70.6 |
| ticTacToe.D29.N958.C2 | 66.6 | 66.0 | 65.8 | 66.0 | 65.8 | 64.3 | 51.4 | 54.9 | 64.9 | 65.8 | **78.9** |
| waveform.D108.N5000.C3 | 66.2 | 61.1 | 62.5 | 66.4 | 62.5 | 57.8 | 58.3 | 60.0 | 58.7 | 60.0 | **77.6** |
| wine.D68.N178.C3 | 70.8 | 82.0 | 86.5 | **92.1** | **92.1** | 67.4 | 67.4 | 71.9 | 80.9 | 84.3 | 91.0 |
| zoo.D43.N101.C7 | 88.0 | 80.0 | 54.0 | 62.0 | 74.0 | 70.0 | 74.0 | 58.0 | 62.0 | 62.0 | **92.0** |
| Average | 71.6 | 69.6 | 61.1 | 71.6 | 55.0 | 66.7 | 66.9 | 57.9 | 68.6 | 51.6 | 64.7 |

**Table 2** *Classification accuracy ($minsup = 1\%$, $minconf = 50\%$)*

[2]The LUCS-KDD DN is available at http://www.csc.liv.ac.uk/~frans/KDD/Software/LUCS-KDD-DN/.

It should also ve noted that, in all cases, the input data was divided in half with the first half used as the training set and the second half as the test set. A "better" accuracy figure might have been obtained using Ten-Cross Validation; however, it is the relative accuracy that is of interest here and not the absolute accuracy.

From Table 2 it can be seen that:

- No approach produced a best result for every input data set considered.
- For some data sets a fairly consistent accuracy was produced across the different approaches (for example the $iris$ data set).
- For other data sets the distribution of accuracy values was notable (for example the $pageBlocks$ data set).
- The best average accuracy was produced using "best first" case satisfaction with either CSA or Laplace accuracy ordering.
- In terms of the number of best classifications per input data set "best first" with CSA ordering produces the best result in 8 out of the 22 different data sets considered; with WCS producing the best result in 7 out of the 22 data sets.
- The only approach that did not produce a best result for at least one data set was WRA with "best $K$" case satisfaction.
- $\chi^2$ ordering, with both "best first" and "best $K$" case satisfaction, produced the lowest overall accuracy. This is possibly because $\chi^2$ is a measure of the "unexpectedness" between attributes and classes; and as such expected, but possibly extremely valid, CRs appear at the end of the rule list. However the WCS approach, where the entire rule list was considered, worked reasonably well.
- In general "best $K$" testing produced reasonable results but not as good as "best first". It was conjectured that best $K$ (and WCS) might work better if a lower support threshold were used and therefore more rules were generated, however, further experiments (not shown here) proved this conjecture to be unfounded.
- The proposed ACS ordering, using a confidence threshold of $50\%$, worked reasonably well but not as well as was hoped. It was conjectured that this was probably because many specific rules with relatively low confidence were given a high precedence over higher confidence but more general rules. A further set of experiments was therefore conducted using a higher confidence threshold of $75\%$ (and maintaining the support threshold at 1%). The results are shown in Table 3 (for those approaches that make use of a confidence threshold).

From Table 3 it can be seen that when the confidence threshold is increased to $75\%$ with respect to CSA and ACS ordering, best results are obtained using "best first" and ACS. ACS with "best first" also produces the greatest number of best accuracies with respect to the data sets tested (10 out of 22). Table 3 also illustrates that by reducing the overall number of rules (by increasing the confidence requirement) the "best $K$" approach deteriorates.

With respect to overall execution time there is little to distinguish the methods, WCS on average takes marginally longer, but in each case the data set is processed in a number of seconds.

| Data Set | *Best first* | | *Best K first* | |
|---|---|---|---|---|
| | *CSA* | *ACS* | *CSA* | *ACS* |
| adult | 80.7 | 76.1 | **80**.9 | 76.4 |
| anneal | 88.0 | 85.5 | **89.5** | **89.5** |
| auto | 13.7 | 12.7 | 12.7 | **19.6** |
| breast | **98.0** | **98.0** | **98.0** | 97.1 |
| connect4 | 65.8 | 65.8 | **65.9** | **65.9** |
| glass | 32.7 | 36.4 | **46.7** | 45.8 |
| heart | 43.0 | **54.3** | 29.8 | 29.8 |
| hepatitis | 64.9 | **79.2** | 53.2 | 53.2 |
| horseColic | 60.9 | **77.1** | 47.8 | 58.7 |
| ionosphere | 91.4 | 91.4 | 66.3 | **94.3** |
| iris | 89.3 | **94.7** | 88.0 | 88.0 |
| led7 | 63.4 | **64.6** | 64.1 | 64.1 |
| letRecog | 28.6 | **29.1** | 28.5 | 28.4 |
| mushroom | **96.2** | 89.1 | 82.0 | 81.9 |
| nursery | **89.6** | 87.2 | 87.6 | 88.3 |
| pageBlocks | **89.8** | **89.8** | **89.8** | **89.8** |
| penDigits | **83.0** | 79.3 | 71.0 | 71.2 |
| pimaIndians | 76.0 | **76.8** | 73.7 | 73.7 |
| ticTacToe | **69.1** | 66.2 | 55.9 | 55.7 |
| waveform | **76.4** | 67.8 | 66.0 | 66.8 |
| wine | 70.8 | **83.1** | 27.0 | 28.1 |
| zoo | **86.0** | 78.0 | 66.0 | 66.0 |
| Average | 70.8 | 71.9 | 63.2 | 65.1 |

**Table 3** *Accuracy ($minsup = 1\%$, $minconf = 75\%$)*

## 6. Conclusion

In this paper a number of alternative rule ordering and case satisfaction strategies have been considered. Four established ordering strategies were examined: (i) CSA, (ii) WRA, (iii) Laplace and (iv) $\chi^2$ testing; together with three different rule satisfaction mechanisms: (i) "best first", (ii)

"best $K$ first" and (iii) WCS. In addition the authors proposed a fifth ordering strategy where specificity is the overriding factor when considering rule ordering so that more specific rules are given a higher precedence than less specific rules. Specificity is defined according to the number of attributes contained in the antecedent of a rule.

The principal findings of the evaluation are as follows:

- There is no overall best ordering suited to all the data sets used in the experiments.
- The "best first" case satisfaction mechanism works better than "best $k$" in all the data sets tested.
- ACS ordering produces the best result provided that a relatively high confidence threshold is used (a confidence threshold of $75\%$ is suggested).
- For lower confidence thresholds ($50\%$ to $75\%$) CSA and Laplace ordering coupled with a "best first" case satisfaction produced good results.

The above suggests that ACS ordering coupled with a "best first" strategy and a relatively high confidence threshold will give an optimum result with respect to CARM.

## References


[1] Agrawal, R. and Srikant, R. (1994). *Fast algorithms for mining association rules*. Proc. 20th VLDB Conference, Morgan Kaufman, pp487-499.

[2] Bayardo, R.J. (1997). *Brute-Force Mining of High-Confidence Classification Rules*. Proceedings of 3rd International Conference on Knowledge Discovery and Data Mining, AAAI, pp 123-126.

[3] Blake, C.L. and Merz, C.J. (1998). *UCI Repository of machine learning databases*. http://www.ics.uci.edu/ mlearn/MLRepository.html, Irvine, CA: University of California, Department of Information and Computer Science.

[4] Clark, P. and Boswell. R. (1991). *Rule Induction With CN2: Some Recent Improvements.* Proc. European Working Session on Learning (ESWL'91), Porto, Portugal. pp151-163.

[5] Coenen and Leng (2004). *Data Structures for Association Rule Mining: T-trees and P-trees* To appear in IEEE Transaction in Knowledge and Data Engineering.

[6] Coenen, F., Leng, P., Goulbourne, G. (2004). *Tree Structures for Mining Association Rules.* Journal of Data Mining and Knowledge Discovery, Vol 15 (7), pp391-398.

[7] Coenen, F., Leng, P., Zhang, L. (2004) *Generating optimal classification association rules*. Submitted to PKDD 2004.

[8] Diamond, I and Jefferies J. (2001). *Beginning Statistics: An Introduction for Social Scientists*. Sage Publications, London, pp183-197.

[9] Lavrač, N., Flach, P. and Zupan, B. (1999) *Rule Evaluation Measures: A Unifying View* Proc. Ninth International Workshop on Inductive Logic Programming (ILP'99), Springer-Verlag, pp174–185.

[10] Li W., Han, J. and Pei, J. (2001). *CMAR: Accurate and Efficient Classification Based on Multiple Class-Association Rules.* Proc ICDM 2001, pp369-376.

[11] Liu, B. Hsu, W. and Ma, Y (1998). *Integrating Classification and Association Rule Mining.* Proceedings KDD-98, New York, 27-31 August. AAAI. pp80-86.

[12] Quinlan, J. R. and Cameron-Jones, R. M. (1993). *FOIL: A Midterm Report.* Proc. ECML, Vienna, Austria, pp3-20.

[13] Yin, X. and Han, J. (2003). *CPAR: Classification based on Predictive Association Rules.* Proc. SIAM Int. Conf. on Data Mining (SDM'03), San Francisco, CA, pp. 331-335.